JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

10:648.40: On t. convergence Thm for QA. I was concerned that it was very weak!
Reason: Σ error related only to past, known, data and we couldn't tell about t. future because t. list of Q's was not known.

Well, if ∃ a "true" / "modal" that was generating t answers to Q's then ALP will have Σ error² (past+future) that is close to that of t. "true" / "modal" ("citizen") if Any ☺.

I'm not yet sure this is true.

Lecture.
A "stop loss" if I can't get t. "Report" done on time!

I could (perhaps should – for my own use) write a paper on "Convergence Theorems Revisited" Discuss Gács measure, Emphasize final Thm. Explain how the K-L distance convergence is a stronger result than Σ error². —
Also using not K cost but Σ(codes of true model) can be significantly better.
Then proof of Bay data than proofs for QA data.
Perhaps discuss General "Kinds of Errors": Σ error v.s. choice of / Modal error.

Finals Thm also can be used to find t. difference in convergence rates betw. using 2^{-K} v.s. ALP. Say 2^{-K} is a factor 2^b worse than ALP (which = 2^{-K+b})
Then Σ error² for ALP is K for 2^{-K} is K-b for ALP.

For SM yield, hvr. ALP is 2^b times that of 2^{-K}. This SM yield is very easy to show.
At any time we bet but current fortune × p_i on i=0 and on i=1. If PC is prob of
correct so we started w. #1 fortune we end up w #PC · 2^N for N bits: Each bit we
double our money unless we err. to be all when we lose! So we make $2 \cdot p_i^r$ on rth bit:
where p_i^r is pc of t. wrong bit.
If there are K different bits (radix K) we win $p_i^r \cdot K$ on rth bit so total yield
is #pc · K^N. for total corpus.

A very interesting theorem on SM: If you used a "Blind" method to obtain yield over past N years,
Then this is same as if you proposed t. strategy 1/N yrs 2pc & tried it for N yrs.
Since ALP is a form "Blind", it is a r.e. method of Estimating Future yield. (has discovered this is t. rule.)
Maybe not! Say prob of 1 is $\frac{1}{2} + \frac{1}{N!}$ (actual) & one
Odds are always even. So yield per bet w. future, the error in yearly estimate → 0.
We "know" our yield will be. Maybe not! Say that is t. true rule: Worse ALP & better
rule that approaches this, but we don't know that! We have a Complex rule for PC
calculation that is $\frac{1}{2} + \frac{1}{n}$, but we don't know that.

Or: T. random regression rule gives pc of 1 as a result of past bits occurs in SM
If could give $\frac{1}{2} + \frac{1}{n}$ w.o. our knowing it. This sort of thing is likely to occur in SM

(.28) is very interestingly Relevant to Actual Alpha yield problems.

→ But all it really says is that a time series need not be "Stationary"!
Hvr., Note that ALP is by no means restricted to stationary time series.

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : 64940 : A possl. way to do Expo (=Report):

Descrb pure QATM; ~~[redacted]~~ Give fairamt of detail on how it works. Then discuss various diffys & ~~[redacted]~~ how we can deal w. each: Also some further benefits & diffys of the "items".

So we start w. QATM in which Updating occurs before each problem is solvd -

We begin w. d-IND probs, indeed as such, so TM knows what kind of Induction Model to use.

At first we use a standard apprch on d-funcs.

08 For pure D-IND, the goal is to find a Max PC, D-funct ~~[redacted]~~ that generates the corpus (= ans's all Q's correctly)

09 — [Is it possl. to have (D-recognition functions) to divide the corpus into parts, & have d-functs (w. possibly diffrnt statistics) for the diffrnt "kinds" of Q's recognized? (see .26-.31)]

0 — I was thinking of this Recogn. funct. being part of GPD. Tho the recogn. is (a D-funct), GPD can have D-components! If so, for deciding what kind of PST should be used on a problem, GPD can use D-functs w. impunity (perhaps) since some PST's are "Universal" — & they can solve any solvable problem (eventually). — But otherwise, it would seem wrong for GPD to assign a single PST to a problem — giving it PC = 1.

Hwr, at the beginng, we only have a small no. of PST's & it may be that none of them will solve the assgnd problem. Hwr, this may be corrctd — its just that the "right" PST's take ~~[redacted]~~ to solve certain probs! We still get S-info from that "soln".

10 There is a mix-up here betw PC's in GPD (to choose PST's) & PC's used on "external" induction problems.

To start, there is only one PST for D-IND probs". — it is AZ141, w. perhaps

22 some hacks for finding common sub-functions, & other things like that.

23 Then, there is the possy of (.08-.10)*. I think this gives a new class of functional forms.

26 — I could start by indexing the QA's so that TM could ~~[redacted]~~ diviss several appropriate D-predicates (Q→A). Then, later, we leave out the indices & TM has to invent a d-funct to "recognize" what the indices "told" it: essentially a D-IND prdctn on the indexed info.

b

31 — O.K., so AZ141, w. its trim of .08-.10; .26-.31 may give some reasonable ~~[redacted]~~ D-induction capability.

34 : 64740 Q.K.: Now, can I have TM simultaneously work on the S-IND problem of modifying the PD that Guides TM's search of D-IND prdctn, D-functions?:: But we already have AZ141 (.22-.23)* w. some extra hours.

The PD induced by (.22-.23) viz AZ141 + hacks, is ~~[redacted]~~ a fixed approx.

414.18 tries to Max it. Perhaps the S-funct involved is one that ~~[redacted]~~ → 651.00

EP

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 650.40 invents/notices regularities in t. D-induction process. T. problem is one of speeding up t.. Moxzn of 414.18. This could involve context dependence of pc's of concs in t. D-Induction functions. — T. idea of how O^{n+1} is a 5-funct of O^n, Qns, Ans, & all previous data & "traces" of TM Activity.

.03

.04 [SN] On "Cure Cancer": "Cure cancer" is a BIG problem. If we had some Criterion for Cure, TM could try to satisfy this criterion "as soon as possl.". Could we formulate this as a regular Optzn problem for TM, w.o. "Look ahead" explicitly spelled out — so that TM would automatically, by itself, discover "L.A."? Actually, this is t. sort of thing that I've been worrying about — that a TM with a Big problem, is, essentially a RTM w. large h. — th. BAD kind! It has "Ego", you can't turn it off, etc. It does global self improvement, etc.

12

13 So 2 Interesting developments! (1) 650.34-651.04: Perhaps clarifies meaning of Context dependent pc's of concs, & perhaps generalizes t. idea & tells how We evaluate t. needed pc's. This is t. idea of "internal problems" being "one kind of problem in TM's corpus! (648.40-649.40) ← is on this: I want to clarify it.! → 652.40

.19 (2) (651.04-.12): That ∴ may not need to teach TM "Look Ahead"!

20

Re: (2)(.20): It may be that improving t. GPD (in t. narrow sense of improving t. Guiding PD for Lsrch) may be inadequate for (2) to occur. It may be necy that TM is able to "design" & "assign" arbly good PST's. (which is part of GPD improvement, but it is probly possl. to restrict it so that TM wouldn't get "out of hand"): Just how "Brite" such a restricted TM could be, is unclear!

Try to work out t. details of t. discovery of (2) & see just what features of t. System are actually necy in order to get ≈ L.A.

It seems clear that a suitable PST could do L.A. and find solns to problems faster/better.

30 Probably TM would have to do much "Reasoning" to devize such a PST.

Simply improving t. Guiding PD would seem maximally greedy & ∴ couldn't do L.A.

Would Lrng during Lsrch be better in this respect? Superficially, No.

For "Curing Cancer" TM would have to do expts. in RW. Hvr, this is not signifiably dfrnt. from doing "internal" Expts

→ [Woops!] Any OZ problem: A finite time limit, T. ∴ TM will use L.A. to solve such problems! Note that OZ problems are an impt subclass of Probs involving PST optimization. Small T values don't give very interesting results, but we gradually ↑ T in t. TSQ!

→ Normal IND probs (t. commonest type of problem TM does) are OZ probs of this kind!

6/10/02 ID

11894 ···· 205 : fan but ~ 6/5/02 ! "Niot explorer extension. see f 678 : beginning to Go

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : (651.19) spec On (1) 651.13–.19 ! 650.34–651.04 ; (646.40–647.40) & → (647.12–.20) is main idea

.02 : 651.40 ! Note also that INV probs solvd by Lsrch, can also use LA ! They try to find a fast Mapping funct from prob descrn to soln. — T. fastest such mapping functs could use L.A., before they generate their "final output".

.05 So it looks like LA, & (as a special case) Lrng during Lsrch, are commonly occurring things. Hvr., t. way I have been doing "Lrng during Lsrch" is difrnt from .05. In .05 we use normal Lsrch w. known bound on CJS for known sdr.. My "Lrng during Lsrch" converts it into "non-Lsrch" — so we can't (easily) estimate CJS.

So .05 is prob'ly much better, less ..., than what I had been proposing for "Lrng during Lsrch".

.02 (& what preceded it) seems to be a radical discovery & perhaps GREAT Simplifn. of TM !!
— We use straight Lsrch all t. time for all problems !

Apparently Not ! see ... !

For D-IND probs : for early work, these can be solved as INV probs : we look for Q→A functs in pc order, in Lsrch, so this automatically gives us (somewhat) short codes.

.06 Hvr, INV Lsrch doesn't (ordinarily) do .05 : It looks at t. [Q_i, A_i] set & tries to find a function that maps t. set into a $Q_i \to A_i$ function for all i. It is not nearly a "short" function, but it is fast. Would TLU do t. trick? (remember : for D-IND Dvctin, there is no ambiguity for t. $Q_i \to A_i$: for each Q_i, only 1 A_i is possbl). — It would seem that TLU would be a valid, fast soln. to t. ... D-IND problem, viewed as a INV problem —

.20 So it would be of no value : TLU gives no extra-polation.

Anyway, t. moral is : Regular Lsrch (w.o. ‖ lrng) can do L.A. & prob'ly other v.g. things: including Lrng during search. — For INV & IND & OZ & Opt probs

Hvr, while straight Lsrch can be v.g., because of scaling problems, we need Adaptive Lsrch. Whether it's best to have Lrng during Lsrch is unclear. T. advantages of Lrng during Lsrch is that (a) it speeds up adaptation (b) It seems to deal w. t. difty of many conds being "about t. same" — which make normal Lsrch very slow. (c) There may have been other advantages of Lrng during Lsrch. (d) another is that one could dispense w. Lsrch completely (& this was supposed to be v.g. because Lsrch is not optimum under usual info conditions)

.31 — by working on a cond until it was (by ‖ evolution process) clear that that cond was no longer optimum / that switching choice would ↓ expected time to soln. ← (this last in spirit

.33 of my "New soln" to Hutter's search problem).

This suggests that "Blind Srch" could eventually find a use or by clever Heurs!
Well, it could, but at enormous Lcost ! W.o. ... Lrng, Lsrch is very Limited !

As I see it, there are 2 extreme versions of TM : (1) Pure Lsrch w. lrng. betw. problems, only.
(2) No Lsrch at all, but ‖ evolution as (.31–.33).
Versions any distance betw. these 2 are possl.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20: 652.40 : I want to write up clearly the old version of TM, w. reasons for various parts, before switching to the New version (pursuit).

But first: 651.13-.19; (652.00).

O.K.: Consider first INV prob: (a) One (not exactly Lsrch) way is to do trials for direct soln. in PC order. All trials take about same time so Lsrch is not very relevant. This is usually not a v.g. way to solve INV. probs.

(b) Rayter Ind Lsrch: Trys various funct's to map probdcrn. into soln. Note 652.02: This method has all the power of a general PST, & can do Look Ahead. It can make tentative (or even permanent) trials & Lrn. from those trials. — These trials are within each PST used.

I guess that early TM INV. solving routines will first try to find simple, general funct's that maps all probdcrns into solns. Next, it tries to categorize problems, so it can use a different function for each problem type (to map probdcrn into soln).

Machine Measurements ("OBS") on the problem & deciding what to do on the basis of these obs is "ob-OP Algebra" can then be used. Further decision elements may wait for TM to learn how to "reason".

The improvement of the "Guiding" P.D. for INV Lsrch: This is its "updating" problem.

The updating problem for INV Lsrch: This is a Q A type induction problem.

The Q for each A is and its cc is a set of "data pairs": The first part is the soln to a INV prob. (& soln" in the sense of mapping from prob to soln) & second part is the info the M had at that time: This will be all previous prob, soln pairs & all traces" involved in their solns, & the times taken for these solns.

The data pair derived is assoc. w. each INV prob. solved by TM. The completed Q at time to will be all such pairs for problems given so far & to + Δto.

What we want as output, is a funct that is able to look at the Q & all previous updates in TM's TSQ; For any PST as co-argument, it is able to give a PD for cc needed by that PST to solve that problem. More generally, it will look at the data (including latest prob dcrn) & output a set of PST's w. assoc. pd's on cc needed to solve the problem.

28 [redacted] How to Select a PST or give a guiding PD to

29 the list of [redacted] PST's, is not clear! One possy: For each cc value T, there

30 will be a PC that PST$_i$ will solve it for that time. Its like a horse race!

So one gets the probty that a given PSM$_i$ will solve it further than any of the others:

To obtain this pc is assist via MC. Carlo. From each PST$_i$ obtain a MC carlo time of soln. (it can be ∞), then compare values for all PST$_i$'s & pick the w. min.

Do this many times to get D.f. for each PST "winning".

37 The [redacted] justific. for using the forego D.f. as a Lsrch "guiding PD" is not obvious & may be [redacted] it is not a pretty good way to assign pc's to PST's for a given problem — in fact, if we knew the info we have on the cc d.f.'s for each (prob'crn, PST) pair, Lsrch is rarely the best way to go!

ED
JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

:0: 653.40 : One "justifn." of L scen is that it exercises" t. various PST's so that one can better chose betw. them for a given Problem.

T. uncertainty of 653.28 ff is something I decided to "live with" for t present. [WON does seem to offer a better soln.]

Indepy of t. Q of 653.28 ff, we would like to create t. prob d.f. in to 653.29–.30 as t. output of an inductive inference process — or some other kind of data that could be directly used to Guide problem Solving.

T. update model types discussed above is one important type. Another kind of update!

.09 We are trying to Build a problem → soln function (t. "prob type" can be any type!, '0 INV, IND, OZ, opt.) We want an induction scheme that looks at all previous trials of Functions like that in .09 & induces good trials. On t. basis of all such trials in t. past, & their degree of success or unsuccess, this should induce a pd on all possl functions, F, for any particular problem (?) — But wait, F is a funct that's supposed to work for all problems & all problem types. The "past data" should include t. cc of t. function in obtaining a soln. [SINCE t. F. funct F could involve search, or recursive operations that are very time-consuming.

I want to decrib. what I had in mind when I wanted t pc of a conc. to depend on "Context". (i.e. .23–.29) (see 655.28–.40) (.23–.29) deals w. Generation of PST's — One way to Generate them.

20 **2 aspects of Updating** ① wrt PST's : Generation & Modifn of PST's &

21 reassignments of pc's to them (conditional on problems).

.22 .23 ② Generation / modifn. of PST function by constructing them from "concept functions" — assigning Concs pc's that are functions of "context" of PST; so pc depends on present "state" of partially constructed function being created, as well as on t. problem being solvd. — Thus we obtain both a PST & an assignment of pc to a particular problem (or to actually any problem!).

.29 so ② fits into ① as part of (.21)!

'0 **VERY GOOD!** .21 & .23ff make much of t update procedure clearer by integrating 2 different aspects of it.

31 Is it clear that .23–.29 is generating a PST? Is it trying to do O^{new} = some funct (O^{old}, Q_{new}, A_{new}, ...)?

It does seem to be "Updating". First, be clear on t. legitimacy of what's being done.

Well, O^{old} is part of GPD, so it would appear that it is "UPDATING"; so "Is it legit — & where are Details?" → 655.31 is "details"

We have a O^{old} w. assoc "frequencies of use" of various Concs, ... & various "Definitions".

Consider Updating O^{old} in view of Q_{new}, A_{new}: First attempts: ① use O^{old} as O_{new} directly, see how bad $p(A_{new} | Q_{new})$ is ② In view of this, modify continuous parms of O^{old}; ③ make new definitions ④ recompose O^{new} (This last is tricky! I'm not sure about its legitimacy; I probly work for English text ... but that may be "special".

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20:650.90 : (3) If the pc of Ant1 still looks small, try to find a "recognizer" for Ant1 & make a special function for it to → Ant1. We can use the data in the pc's in the old O9, since it's "all one function" (including the recogn. part!)

I don't know how well (.00) fits into my previous work on "Recognizers & Recir updating"....
As in "CPM": Nov 8,01. Section on "Updating". (.00 may be ≡ ibid, CPM N8.01 Pt §2; 2.2)

.07 Note: In CPM N.8.01, I was using Hard (d-funcs) recogn. funcs. I may want
.08 to use Soft (s-funct) recogn. funcs. → .28

T. UPDATING Algms. in CPM N8.01 (& others that I may develop) are meant to help TM "Get off the Ground". However, I want to make the Algm. "simple", so (a) I & other humans can understand & ; pgm, & debug it (b) TM can understand, modify & use parts of it for other problems, other parts of the System

The 3 "scaling" difficulties:
1) using Non-adaptive search works for small problems only — but are within the vocabulary of the Reference Machine
.05 2) If cps of concs are problem independent, then, as no. of concs grow w. corpus size, the pc's of the concs tend to ↓. I'm not sure this is Nacy, but it did seem to occure in the TSQ I did for ANL in Saarb.....

.19 3) As corpus & instrs, it becomes very expensive to test entire corpus on trial Predicates or existing PST's. We can use statistical testing, & finding of "debit problems" to reduce amt. of testing. Also, Partitioning Corpus: so that only a small part of corpus need be tested on a PST — because that PST is applicable to only a small part of the corpus. This has to do w. "Recogn. Funcs" (appropriate)

As in .07, there is a "Hard" v.s. "Soft" Recogn. funcs. Hard are usually much better, if one can make them. — in .08, the applica to .19 as well as in .07 (≡ CPM N8.01 §2.

.28 In .15, in the Saarb ANL TSQ the "solution": I felt that assigning just Barn Seq. pc's to Concs wasn't "specific" enuf. — which is perhaps how my idea of

"the importance of "Context" evolved

.31 Well, one way to automatically get context is to look at funcs "around" the funct to be measured. This is like T.S. pronn. using previous nbrs as "conditionals". This perhaps equiv. to invoking by inventing "Macros" or "combination functions". Q.k. But I wanted to invoke context on a broader more "Global" level, as well — so the pc of a conc should depend not only on local context (within the function being created), but context like previous history, historical braces, the nature of the problem(s) being solved; previous solns to problems.....

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20:655.40 : Now Contrast 654.31 w. 655.31 : They are above similar pairs ; 654.2[illegible] -.22 is lowest of a contrast ; 654.21 v.s. 654.23.

.02 "Updating" is the response of the system to a new (Qnt+1, Ant+1) : (problem, soln. pair).
It tries to Make (Qnt+1, Ant+1) "fit into the system".
It does this by adjusting the functions used (654.23 -.29; 655.31-.40) as well as the assignment of adjusted functions" to new Q's. pc's of

Also, it can invent new functions (PST's) as well as the pc's that associate new Q's w. the new PST's.

"Updating" is any thing the system does to prepare itself for new problems: Usually this involves Interacting / Integrating the latest Q+A into the existing "System" (or equivalently, modifying the existing System so that the new data fits as well as possible (in the available time)).

The concept of "Updating" is changed about if it is done before each problem — specifically for that problem. See 657.00-.03

.09

.10 Sounds Good for Expositive : That TM works by assigning Q to (probabilistically) 1 or more PST's. Each solves the problem.

.11 Updating is .02 — .09

So .10 could be a v.g. introduction (or even abstract) ...

After prob is solved (or after Soln is found / known / Gotten) (TM need not solve any problems!)
We "Monday Morning Quarterback" it & try to see how TM should have acted (given prob's soln)
by knowing prob's soln & asking how to run system so it could have solved it "best" (fastest, or best for available CB).

The system itself is .10-.11; .02-.09, but I'm still not quite sure of the Details of Operation.

.20 Say we given an INV problem. GPD assigns it to a PST: Say it
Does Search using GPD the "Guiding PD": to solve it.

To do Q+A probs: GPD assigns the new Qnt to a PST — which is itself:
It then gives a GPD(A | Qnt) — a d.f. as output.

N.B. The "factor of 4" of 586 should be "factor of 2" only. Due to the uncertainty of Direct P.S. to Updating':
In advanced systems, this distinction may be necessary!
Here, Even the system would not be doing Search (because of (why)) is not relevant,

To Update the part of the pd devoted to INV probs. At first, it does the update part that assigns the INV prob to the PST that does Search. It does here, update the part of GPD that is Guiding PD for Search. To define that update, we have to first define the corpus to be used. This will be the [Problem, Pdj, Fi, Ti] triplets of the past: Pdj is [illegible] down of the INV problem; Fi is the function successfully solving it.

.30 Ti is cei of that soln. We also may have Pdj, Fj & Tj' info for problems in which,
.31 after [illegible] cc = Tj'; the problem had not yet been solved.

.35 The pd for updating has to make/find/use a function, Z that takes Pdj & FK(i) as input & gives d.d. on the set of soln. for output. We want a Z that maximizes 414.18.
This will be fine for the existing Fk, Pdj, Tj corpus of .3p-.31

??

Here, we also want to invent new Fk's.
This can be done by making a Grammar for the Fk's : here, we [illegible] how to make & remember (Fk, Pdj, Tk)'s over elements! I don't know how to do this!

HVR Notes 657.2-1 !

6/16/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 656.40 : From a "pure induction" pt. of view; Given a new PSr, to find/invent a F_k ∋ it's likely
to have a small ?. (say E? is small) — This amounts to updating befor working on t. problem —
which I felt/was in t. past better because ① I could update "selectively" for t. new problem — i decide
how much time to spend on it. (this mite get rid of factor of 2" in "optimality ratio" ②).

03 As an "example" (not actually INV problem): consider ANL. We want a "short" function
from "3+7" to "10" This is an IND problem: it has a particular "Lsrch soln" that can work:
i.e. make f(Q) → A, trials in pc order (or least order) : We need a Guiding P.D. for t. Lsrch.
The "Goodness" of a f(.) is given by its pc — and perhaps also, t. cc needed to execute it
for t. entire TSQ (perhaps?).

0 T. "Updating" that Juergen did for "symbolic regression": He used as TSQ, pairs of reals.
He had a simple a priori pd on cand. functions given by a few pc's. After solving
many problems t. pc's migrated to ≈ φ ≈ ≃ 1. He used a reenforcement mechanism to modify
t. pc's. My impression was that he should have used ≃ Laplace's Rule rather than reenforcement.
(he had to try various levels for t. reenforcement parameter). T. details of his reenforcement mechanism are impt. Certain kinds of mechanisms can be easily xfmd to "Laplace's rule". I think one uses a
His method seems close to feed forward Neural nets.
"to be chosen by trainer" "adjusted" reenforcement parameter.
Anyway: He used a particularly simple space of functions & PD on those functs, so
29 finding a soln. by simple hill climbing worked fine. (.33)

10 What I'm trying to do now, is get examples of various kinds of Updating.

21 SN on Second Rot in 658.35, F should be problem Independent! In fact, 658.35–.40
Is prob'ly ALL WRONG! There are difrnt amounts of nonelism in TM Models!
Prob dcrn say include soln. CB.
Th. most General! Input = Problem: Output = soln.
Next: Input problem, outputs pd on soln: then use Lsrch.
Next: Input problem, outputs pd on PST's. Each PD could differ further conforming to other PST's
(or (usually) probably Mixes of these).
t. problem or work on it directly level. Even t. top 3 primary input 3 level.

Perhaps list some kinds of problem solns. — then show how t. system gets to them.

29 Or: this last conjecture of all levels!
So when we insert problem into TM, it may try to solve it directly t/o probably a locate
30 ... show to other PST's.
..29 is V.g., but perhaps draw up simple, prelimy System, then show how various
improvements can be made. (663.10)
.32
33 (.29) J. had a difnt pd on all instructions for each position in t. pgm. So, w. 7
opcodes a. pgms of length 10, he had 7^{10} pc params. Hwr., most never occured. → 658.21
.34
On t. other hand, I had been making pc's of cards indep of where they occurd in
t. pgm. Invoking "context" as relevant to pc was a way of looking at previous
parts of a pgm to induce pds on t. "next one":
Mutating (in a very general sense — xfmg) old, successful pgms, is a way of inducing
a pd on future, dependent on t. past. → (Spec 658.00)

ID

.00: 657.40 : In line w. this: Say we have a Sample of hy Gore Pgms: we can then make a probabilistic study of various features (concs) in that sample & induce a pd on pgms from this.

Normally for a BAG of pgms, we have a case count of each pgm. type. We extend this to concs & conc environments, but t. case counts could be various: Say for each instance of a pgm, we use as "case count", some ↑ monotonic funct of its Gore. In studying t. c's of various concs in such a sample, we can use these wierd "case counts" to get something like probability of pgms: They will be biasd toward hy G pgms if we do t. assignment of case counts to c's of concs properly .... ©.

Perhaps if we did it rite we could get [redacted] an output D.f. on expected values of Gores

.10 of Pgms. Since we are using a Bernd.f. on concs, (w. indep concs: tho we combine concs for interaction)

.11 for each conc., we have a set of G values; [redacted] t. Qis: do we want to simply combine them into a scalar "case count" (perhaps + size & Vare),

For from [redacted] (.11L) — what's t. best way to make a d.f. on (funct, G)? → [Fi, Gi]

.15 A better approach than (.11L): List t. various functs & their assoc. G values: How can we Best code this data? Make a lot of defns of concs (t. best of these defns is as yet unclear!) → somehow do it as max pc for t. corpus. For each conc, have a pc def. as a funct. of G. We then express t. corpus in terms of these cons: t. pc of each one, depending on G. We want to do t. formalization w. [redacted].

.20 min rcost. → (.23)

.21 : 657.34 A possl. modifn/improvement: try Juergen's 657.33-.34: A diffnt pd of concs for each instruction address in t. Pgm. : Possl: Partial "Pooling": Most pts in space will have null data. → 669.30

.23 : (.20) Re: .15-.20 : We'd like a 2 or 3 param way to express PC of a conc. as a funct. of G. Think of it this way: for each G value, we have a Grammar. Each Grammar has its own concs, but data. costs of concs are shared because they are all part of t. derivn of t. ...

Single Corpus.

As in t. General Case, values of real params need not be known exactly: t. pc of t. corpus is t. integral over all params space of t. pc's of each portion. This integral is usually easy to approximate.

.30 Param. space: This Integral is usually easy to approximate. As a funct of G (or a "G surrogate")

Expressing PC as a [redacted] conc. as a funct of G (or a "G surrogate")

$pc = e^{KG+a}$ i.e. $\ln p$ is linear in G. (2 params)

More complex: $\ln p$ is quadratic in G (3 params)

.35 We may want to modify G a lot w. a monotonic function before doing these xfmns. → 661.12

.36 SN on AZ141: We look at t. corpus & we see concs that might be indep. We look for dependencies: If any apparent dependency was strong enuf, we can make definitions of new concs that are indep; T. resulting expressn. should have fewer dependent concs. If strong enuf dependencys exist, we again try to define them out of existence [Hur, Note ("objection") of 659.32-.40] → See 659.00

→ 661.20

SCALING: For justifn of Context of Concs: W.O. Context, P.c.'s are too low. Context effectively narrows t. D.f. of possl concs at a pt. & i.e. t. pc of t. "correct" choice.

The "OMO Scam" Out of Money Way OMO Way out of money WOMO Scam.

This is t. principal of AZ141

6/17/02

Z 141: .22–.31 The problem of dealing symbols of [illegible] unexpectedly low frequency (But not .32–.40! NOTE)

659

ID
JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : 658.40 : Could 658.36 : 40 be used to deal w. unusually low frequs of cases? One trouble w. Rami's Rule realizing that they occure, doesn't ↑ pc of corpus much! Could ↓ in pc's of completely concs. be to only/best way of dealing w. this?

.02

.03 From a practical pt of view, for analysis of failure of systems; (.22) unusually low asymmetry pc's, is the main problem. — estimating upper bnds on very small pc's.

Usual way of doing .03: Lists various failure modes. For each of necessary conds/params failure; coincidence of events necy for each failure mode. These events are rare & (perhaps) uncorrelated: If we can empirically show each necy cond has pc $< 10^{-3}$ & if are necy, Risk $\to 10^{-12}$.

We can estimate pc of elements empirically. That they are uncorrelated cannot be seen empirically: Coincidences too rare: So we must use "reasoning" to estimate correlation (this is very uncertain reasoning!)

Other big diffy is listing all failure modes.

.00–.02 is ok. for not so small pc's but can't get pc < say, .01 : it would seem that frequs of .01 or .001 could be observed, but their effect on total pc of corpus would be quite small! May be not so! If sss is 10, knowing pc of event is .01 risks ↑ pc of corpus by $\left(\frac{1}{100}\right)^{10}$ error of .1 in 1 power factor or e in pc. —

.18 T. probly this pc is really .02 & it occurred twice as often & should has a very small likelihood: its $< \frac{1}{2^{10}} \left(= \frac{1}{1000}\right)$.

20

22 : (.02) In .00–.02 it would seem that .13–.21 would be relevant. If we think a pc = a (of a symbol) and it occurs X times out of a possible n & $\frac{X}{n} \neq a$ by an adequate amount then we would do well to realize that pc is not = a, but = $\frac{X}{n}$. This is true if $\frac{X}{n} < a$ just as much as if $\frac{X}{n} > a$(!).

.24 e.g. A & B symbols: we note that AB has a freq $<< f_A \cdot f_B$. We certainly should be able to ↓ corpus pc by defining C ≡ AB (!), because it harder to determine pc of AB occurring would cost pc $\left(P_A \cdot P_B\right)^n \left(1 - f_A \cdot f_B\right)^m$ ("m+n "symbols" in corpus)

30 After defn. it would cost pc = $\left(\frac{n}{n+m}\right)^n \cdot \left(1 - \frac{n}{n+m}\right)^m$; which is far larger than .25 since

31 [illegible] $(x)^n (1-x)^m$ has peak when $X = \frac{n}{m+n}$,

32 Hur, the foregoing argt (as well as 658.36–.40) would seem very "Simplistic" in view of my more complex models for Z 141, involving "alternative parsings" — The idea that not all cases of "A·B" should be parsed as "C". While we try to get a max pc parse, we realize that this may not be the only "best" parse — that (many) others will be as good or almost as good. T. truly best "parses are those that (a) Get high pc in Ramsellas (b) enable new decns that ↑ pc further. This suggests that much Backtracking may be (acceptable/necy).

34 Normally the proper frequency of C will be considerably less than that obtained if all A·B's were parsed as "C". There will be usually a very large no. of parses w. some pc whose total pc is > Rst

6/17/02

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

30: 659.40: Hvr., wrt 659.32 –.40, There does seem to be a paradox! If we code t. corpus (using 2 pranteses to get Laplace's rule) by using pc of each symbol ≡ rel freq of that symbol (in equiv. context) in past corpus, we get a legit coding — but it doesn't take all parses into account!

[SN] A possl. explann. of t $\frac{1}{e}$ factor in Z141 analysis: If t observed freq. of A·B is α, then usually, if AB is a legit. defn., w frequency β, then A·B will usually occur w. frequence $e\cdot\beta$: A fraction $\frac{1}{e}$ of these (i.e. β of them) are to be parsed as AB ($\equiv C$). T. rest are legitimate random juxtapositions of A & B. Actually, it's not so simple, because when $C = A\cdot B$ is defined, t. freqs of A & B ↓.

0 [If t. former were true, then, it [illegible] $F_{AB} = e f_A \cdot f_B$ (for [illegible] ssz→∞)

then · [illegible] replacing any of t. AB's by C would [illegible] ↓ pc of sequence.

Hvr, this seems wrong! ·AB will occur w. freq. $F_A \cdot f_B$ if A & B are indep.

for ssz→∞: say we define C w. f_C: ($C \equiv AB$) so

15 • ~~now pc of co~~ Pc of corpus was $F_A^{F_A n} F_B^{F_B n}$ $\sqrt[n]{x^n \cdot y^n} = xy$

5 now pc of corpus is $(F_A - F_C)^{(F_A - F_C)n} (F_B - F_C)^{(F_B - F_C)n} F_C^{F_C \cdot n}$.

under what conds will .16 be > .15? Write their ratio (n is not of note) [illegible] n's of n

0 $$\frac{(F_A - F_B)^{F_A - F_C}}{F_A^{F_A}} \qquad \left(\frac{F_A - F_C}{F_A}\right)^{F_A} \cdot (F_A - F_C)^{-F_C} \left(\frac{F_B - F_C}{F_B}\right)^{F_B} \cdot (F_B - F_C)^{-F_C}$$

[I forgot rest of symbols!] we actually ↓ no. of symbols, so their pcs ↑.

This factor entering in both .15 & .16. Could I adequately represent it by t single symbol "D" & its F_D?

$F_D + F_A + F_B = 1$

$F_D' + F_A - F_C + F_B + F_C = 1$

$F^{D\,F_D n} \rightarrow (F_D + F_C)^{F_D n}$

$F_D' - 2F_C + F_A + F_B = 1$ | → (.32)

28 $F_D' - F_C - F_D = 0$ $F_D' = F_D + F_C$

· [SN] Make some kind of INDEX of impt works

How to reference them is unclear: Try to standardize on names of various effects.

0 Start out w. various possl. synonyms: followed by standard index names. P.662 is a start

---

32 (.28) Check this n=∞ stuff w. t. Algebra I did w finite n — use those formulae w. large n, & see if I get ≈ results: This could easily point out disparities (if any) & Bugs in Reasoning! In particular, check [illegible] to make sure I use rite no. of each symbol. Because simple formula for pc.

(Q): Could I incorporate t. alternative parsing into a kind of codeon which t didn't invent new symbols, but just that if A occurs, B has a special frequency to follow. I could use a code ~ to Z141. This would give a conditional pc for certain symbols.

6/17/02

00:662.40 [SM] Re: Recent Hutter "fastest & shortest" paper:

If we view "Lsrch" as going from a [defn] ["specification"] of a function, to the function, then this covers (perhaps all of) what Hutter's method does. In the Mat Mult. or inversion problems involving "proofs"; The verifn of the machine's result by the "user", say, could take much time.

More general, the input to TM could deny a problem: If TM "understood" the problem, it could try to solve it. It is of interest, that a pure GA machine may not be able to "understand" or "not be able to be taught to understand" what a Opt or OZ problem is ... the idea of Marrying seems to be ... a Gore, is/not teachable to it.

Hmm! What about Man's deny that one can get fire by rubbing sticks together!

So: Chimps never seem to have devd "Fire" & their language seems to be "primitive" (tho they may have an elaborate "Body Lang", that we are unaware of).

(Actually, where Chimps live, "fire" would not be of so much value, ∴ much less likely to be devd)

.12: 658.35 (GA): The grammar for each level of G, will look a bit different from a normal CFG — Say we had large SSZ for a particular (small) range of G: As a CFG, we ("a deriv "Grammar"), short strings would have hy pc's — but in fact, it may well be that unless string is/at least certain length, G will be quite low. If we use a non-[derived] grammars, perhaps we can arrange so that the pc's have lengths ≤ 5 (say) (or having pc > .01).

IMPT → I unconsciously was thinking of [derived] grammars in my thinking about Grammars for GA!! I have to re think all of my work in that Area!

658.36 – .40 is really not Bad. We successively find "correlns" that are largely because

20 : 658.40: ~AZ141: "~" ... & the correlation then redefines a new set of supposedly

.21 their dem pc + "pc of correln" ↓ pc of corpus. & loop to α "Correlations" are a kind of "Context" analysis.

.22 indep descriptors of the corpus, & we look for correlns: loop to α

The [illegible] looks like all simple correln: ∴ only linear analysis? | always defines new cone so that the freqs of all ones are indep (if the correlation one found was true). ?

Is it true that we can do the loop of .21 – .22 & always defines new cone so that the freqs of all ones ... If so, [.20 – .22 looks v.G] — even if it's "only linear".

It looks a lot simpler than Z141 or AZ141!

It is related to my old scheme of ...

.27 A slight change of subject: In 658.36 – .40 & .20 – .22: Is/my old scheme of predicting each symbol of a seq. based on freqs & pc's of all ngrams (in this corpus) terminating in that symbol.?

.28 · We use all postfixes of corpus that have occured 2 or more times. The definitions are "nested", so this writes & rewrites of defining them.

A simpler (≈ L-Z) way, is to code each ngram by giving its length & how long ago it occured. — If it occured > 1 time, this multiplies pc by 11 codes. — (But this doesn't use nested codes).

How can I use nested codes? Also, when I code just 1 ngram, I guess I assume freqs of all other symbols are indep. It's certainly easier to do the coding w.o. them. —

Re: Nesting definitions: This may not work.

We start out w. the precorpus (≡ the alphabet) followed by the defn. of the ngram, followed by the code for the corpus.

To apply .27 – .28 to AZ141; for each inserted "ngm" (≡ function sub-tree), we have a pc assoc with the previous occurrences of that function tree in the corpus. This may be identical to the function pc evaln. of CPM Sept 8, 01

{So till Nov 8, 01 was 2 months: Endelay Chaos of own sevell, ff!}

— Except that in .27 – .28 I try all poss. post fixes that have ssz ≥ 2. → (663.00) 662 ... [illegible]

6/17/02 INDEX GM is STACK 611

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

1) Z141 : 658.35 ff (A new way to look at the problem)
639.32
660.15 – .40 : A simple way to check older Z141 formula. (see if $\frac{1}{\sigma}$ factor is Logit)
VERY IMPT → 661.12 on non-derived grammars
∃ an earlier (~ within 6/17/02 ...) on Z141 : ... : $\frac{1}{e}$ & parallel passages : Note 659.32 – .40

2) GA : 658.00 – .35 : Ideas on a P.P. for [Cand_i, G_i] — a "language" : Promising!

6/20/02 ID (662 is Endplex) 663

Solving, UPDATING INV .10 (v.g.)

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00: 661.00: CPM of Sept 8,01 does not consider alternative parsings (but it does consider [redacted] the impt one kind of ll coding)

Both (A) CPM 9/8/01 & (B) 661.27–.28 have their deficits:
(A) Has deficy of (OD) (B) Has deficy of not considering codes for things other than t. postfix ngram.
Neither considers (much) alternative parsings. — Deflt ngm delay, deflt function delays
"Base t. postfix ngm" or "t. last ngm defined"
(B) (A)

Perhaps t. fact that Az141 has to take many diffrnt parsings into account is a serious criticism of its use in many applicns (like English — perhaps t. applicns of Az14 for appl of functions is inappropriate!

.00: 657.32 Dropping t. targt for a while: On Solving INV probs (657.21–.32): An example would be t. kind of problems GPS solved. Or formal solns of eqns, or even numerical solns.
T. "regular Lsrch" would List all possible ways of solving such problems. Consider various A.I. techniques developd for INV problems (Math proofs, symb. integration ...). Make a Universal lang. in which these methods are AS's & use this for Lsrch.
So say we did this & used it to solve a bunch of problems — (maybe not all [illegible] solvd!)
How would we update t. pd? Well, it we really tried variants of these PST's on many of t. problems, we would get a corpus that (we hope) would update t. params of t. grammar & relate t. gramm params to t. params of t. curve of probty of time at soln. (This is usually a 3 param curve, (1) probty of any soln. (Ps) (2) if solved, how long (m) (3) σ² of previous m). — it could be that t. curve may be of a diffrnt form, hvr.
We can start by not having σ² for each PST, but doing Pool of All PST's: Then perhaps make "local pooling" 661.12, .35 → (see 658.00 — on how to make [strings, Gore] grammars.) to get σ²; Also similar attempts to pool for (1) & (2) (Ps) (m).
The universality of t. Grammar should enable us to create new PST's & [strings, Gore]
assign (Ps, m, σ²) to each of them, as part of t. updating process.

.26 [So some Q about whether t. Mxt corpus]
I guess, .10 – .26 is O.K. in spirit. — [To some Q about whether t. PSTS & their params] — I'm leaving this then really gives rite way to update t. PSTs & their params — (see 661.10 Stoch op. vs. "moment") problem for t. [I will want to write Samplns & Elzns of it,] to start out, Next, lets Go to t. QA induction & its updating. — → (see 664.00)

.31: (SN) T. Az141 of CPM 9.8.02 is a [redacted] "HMM" in which t. no. of states can ↑ (by 1 at most?) w. each emitted symbol. T. "states" are t. legal Gncs.
Actually, this "HMM" of .31 is more complex (& more interesting) than CPM 9.9.02: since it can include context as determiner of "pc of next symbol".

6/21/02 ID

UPDATING INDUCTION : co #(V.G.) see 663.10 for updating INV

664

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : 663.30 spec : INDuction & its Updating: ~~We want to~~ t. problem is to take t. [QA] corpus & find a good O" (414.18). Since it is an OZ problem, t. "normal" Lsrch soln. is to Lsrch over possl. mappings from [Q,A] to O". (using a Least "stop" criterion.)

.03 → One elementary ~~ma~~ mapping is to simply try O" cands in PC order! The PC is ~~derived~~ itself looks corpus → cand [QA] to O". like Lsrch, it is out. Its just another possl. mapping method to go from [QA] to O". — Say look for "regys" in corpus.

.06 There are certainly smart ways to Go from corpus to cand. — Say look for regys in corpus. There are certainly smart ways to Go from corpus to cand. — In general, regy types are probabilistically related. New regys (regy classes) are suspected from known regys. — In general, regy types are probabilistically related. New regys / are suspected from known regys. — to suggest a new class of regys to try on t. GPD can contain info of that sort. — Also GPD can suggest Obs to use that in turn GPD uses to suggest regys to look for basis of regys found thus far.

.10 One might devise a "grammar of regys": ~~#~~ any Universal mechanism can essentially "Lsrch all possl. regys." I want fast tests for them, & of course the probabilistic relations betw. them as (.06–.10). → .20

.14 In .03 ~~&~~ "updating" wants to improve .03: This can be done (one way) ... improve the ~~...~~ PST by improving that P.D. (Whenever a P.D. is used in a PST, we can usually improve t. PST by ...

.15 is to improve t. Guiding P.D. — (we start w. an uncondl. p.d., then we learn to make it conditional on t. corpus & on ~~#~~ O' values for earlier corpi — & any other (standard technique) "contextual clues" we may be able to find. (.36)

NB: T. pd on O" spaces is initially via t. index pc's of conces. Also defining new "conces", Then pc's of conces can depend on context! Also NOTE 658.36 ON Finding Regys in a Corpus. → .36 looks v.g. — but is it relevant here? It is on AZ141, which would seem to make it Relevant here → .36

.19 We found one of them, then "Addative". One good type of regy: This type is sort of "Addative". We found one of them, then

.20 : (.14) → find another ∃ indep of t. first, & find new ones, w.o. having to examine t. whole corpus. We can make various Obs on t. corpus that correlate (.22.R)

.22 Many kinds of regys are testable w. smaller. We can make various Obs on t. corpus that correlate w. various regys. Some ~~#~~ Sets of ops are v.g. because as a vector, these ops

.24 "narrow down" & ↑ pc of certain regys very much.

So: .15–.19 tells how to "improve" (≡ "update") t. PST that I'd been thinking about in .03 ; .20–.24 tells how to "improve" (≡ update) ~~...~~ a somewhat general class of regys (to be used for predn).

Note: "Updating" need not ~~...~~ (take advantage of / use) "latest data": It just does whatever it can to improve t. GPD, ~~...~~ with available CB. Ideally, it should be done

.30 before a particular problem, so TM knows what part of t. GPD" to "improve".

.31 (664.00 ff) & (663.10 –.30) looks fine! Quite General, yet I do know of specific PST's (But maybe not completely General?) that I can start w., & t. techniques I had for updating those PST's ~~...~~ Do fit into "t. General plan" — particularly if I consider that updating can just improve old & create new PST's (among its other subgoals).

.36 : (.22.R, .19) : N.B. In .03, .15: Improving t. Guiding PD is an elem. of t. more General problem of improving t. Update Algm. Presumably, t. Non-El version of Updating might be dealt w. by a ~~...~~ "later, more Mature" TM. So at t. beginning, TM really is not up to working on non-el. update problem. T. Q is, can TM advance enuf, (using t. El. update) to usefully work on Non-el. update?

spec spec 665.20 spec →

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 664.10 ON T. "50% Soln.": It would seem very el. to conceive of t. ratio of time spent on updating to t time spent solving "external" problems, to be a useful param. to adjust. That in general, t. system should decide itself when to update, & how much cc activity do spend on it. On t. other hand, generally improvement t. system (i.e. updating) can really be done indiply almost, of when external problems are worked on. This (our range "S.I." activity) just needs a certain amt. of cc — & t. "proof" of 50% soln. allows for cc relatively indiply of when external probs are worked on, & t. relative "phasing" of S.I. & external prob. soln. Doing these activities simultly. (time share) (in 11) would seem to be fine!

Expo on "50% soln": Say we have 2 (same) machines each w. same clock speed working on problem: M_1 has title resp. wt internal to external time share.
M_2 " 50" " " " " " "
If we slow down M_1 by factor of 2; M_1 will then be $\leq M_2$
at least as much time on both internal & external problems. M_2 will be at least as good. — it will spend
It is not to say that normal M_1 is not > twice as fast as M_2 in solving problems — but I think it's "unlikely"! — p. 673.11
M_2 could be waay better than M_2 —

.19 Spec. 664.90 SOS T. System: A problem comes in: It is assigned (probabilistically) to one or more
PST's, by t. GPD. They work on it (possibly by recursively breaking back sub-problems to PS.
Total system } 2/o to other PST's until t. probs are solved, or until cc is exhausted (& the system has
done as well as it could w. that cc).
Parallel (time share 2/o (& w.) to Pb's, t. system is working on
.26 t. problem of self-improvement. (S.I.). S.I. is same as "updating". There are
Various elements of t. system that can be improved — based (partly) on info obtained
from problems solved! by t. system, or by subst. inserted by trainer (different degrees of soln. can be inserted)
Some such elements: (1) Assignment of resources pc's to
PSTs as function of probdesn. (2) Improvement of PST's (3) invention of
new PST's. (4) Improvements of PD's used by PST's (these are all "END" type problems)
.31 (5) Improvement of t. [QA] problem function. [Better induction: Represent t. S.I. problem can be expressed as 1 or more QA's]
The problem of S.I. is "just another problem" for t. system that resolves in 11
w. its other "External" problems. Most of t. S.I. prob. sub-problems
.35 can be expressed as part of t. [QA] corpus.

.38 So: in addition to .20 — .35, what is needed is descn of (a) types of probs solved
(b) various PST's wd to solve these probs (c) just how S.I. can be usefully formulated
usefully
25 of a prob. & subprobs that are normally solvable by t. system.
(normal)
T. Rpgs. stuff does breakdown t. descn of t. system into parts that can be worked on/described

6/22/02 TSQ's .16ff, .20ff How to write them
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:665.40 separately! So this looks like a GREAT BREAKTHRU in both exposition & in design of
01 TM itself!

In exposition: Explaining what GPD is, will prob'ly be quite diffl. Comparing it to Google can help a lot. It is more problem oriented than Google. How does it differ from Maple or Mathematica? I can go into the differences — Perhaps Moravec's "talk" given in the [redacted] paper/report.

Re: Google: it is constantly being updated — as new info appears & as new topics of interest develop; & as interest (output desired / profile) of old topics change.

---

0 List Things that have to be done for Report/System Desn.
11
12 1) (a) List & explain types of problems: (b) Tell which types will be [redacted] in initial system. 663.10ff of INV 664.000 on INV
13 2) List some PST's assoc w. various prob types. Spend much time on QA probs.
15 3) Look at various aspects of updating. 663.10ff / 663.40 ff on updating INV. / 664.00 ff on updating IND. ← These were very general, good, items for updating.
16 4) T.TSQ: I have at various times, discussed various general methods of TSQ writing. Perhaps try to list new impt. ideas. (see (.20)).

20 : Re. TSQs: some ways (1) Top down (SBS) start w. goal problem; devise conc. net for the goal, devise probs for each conc. (or set of concs).
(2) Start w. Human Alg. Text, see if can devise [redacted] conc net a/o TSQ to solve it.
(3) Look at Solns to old AI problems. Try to devise conc. net for those probs — "Classic AI probs": Then TSQ to realize that conc. net. Symbolic Integration; Proving Theorems; Solving alg eqns, differential eqns: Probs. solved by GPS. Probs solved by Humans in [redacted] Newell, Simon's "Human Prob. Solving".
(4) Idea of Getting TM to be familiar w. say, Algebra, so it could learn English that talked about Algebra.
30 (5) writing TSQ's for various Levels of TM knowledge: e.g. To solve linear quad, cubic equ. say, using one (invertable substitution) heuristic.
(6) Try closer — TSQ's for learning to do certain symbolic Integrals.
(7) Try to get TM to Lrn to do various things that Maple does. (Reverse Engrng Maple)
(8) What about analog lrng? Learning to reach certain control goals. The broom balancing problem, etc.
(9) Look at Various problems Solved by G.A. Since I conject TM to solve them Much faster.
(10) Once I get the details of the lrng system (setup/decided upon): Try a not very carefully constructed TSQ in Algebra, say, & see where the errors are CJS's are. — Then insert

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

60: 666.40 needed concs (via proper problems) (like a virus inserting a Gene or DNA snippet).
One way to get TM to lrn a lot of Algebra: Just list impt ideas I want it to lrn in "epistemologic order" — look at CJS's betw. them, & determine needed concs betw. them & design need problems/examples/HINTS or other ways to BRIDGE the successive CJS's.
.04 E.g. I had some trouble w. ANL: so assume TM got over that hurdle & continue!

The TSQ suggestions of 666.20 – 667.04 seem fine — (particularly #10).
It may be that I have the problem "under control": Try to list the weakest parts: see if I have refs to solns of those problems. The idea here, is not to forget the solns.!
666.10–.15 is a good start.
.09 T. INV approach of 663.10–.30 + 658.00–.35, 661.12 ff! seems fine for an advanced TM!
10 I need examples of how to work simple problems, hvr. T. appr. is based on discovery a grammar of PST's that are for INV probs (PST's used in early A.I. work on INV probs).
Are there some simpler, more direct ways to solve some inv. probs using Lsrch?
Perhaps look into those early A.I. probs. Tower of Hanoi, Missionary Cannibles, etc in that Book on GPS.
Actually, the methods of (.09) are not bad for elementary INV probs as well — & so only very simple INV we just have a far simpler grammar for PST construction! — GPS has a kind of Grammar of sub-techniques: Backward probs can be solved. Construction of a Vector Corc. Deriving OPs & a forward chaining; ... $\|\ \|^2$ of vector corr.
.40 Obs & an ob-op algebra to
Hvr, I may want to not discuss "elementary problems" in "my early TM discussions" — Mainly work on QA Induction. & the methods of (.09) as a kind of "aside" & write discuss the advanced INV soln methods of (.09) as a kind of "aside" remark in the report (or a remark for me).

.26 T. most impt kind of problems, of course: QA & its updating. (664.00–.40)
I DO want to clarify the context dependence of Concs in construction of O^n conds. Just how the context (& extended context) works, & just what part of the system the pc of the definitions of the contexts are "changed to"
.30 Also look at that stuff in CPM:(N,8,01) on Recognition funcs & their "Updating", ⇒ (668.00 spec)
.31
SN INV probs can be solved by QATM in what looks like close to normal INV Lsrch: We present QATM w. a (large?) set of INV prob. decns & their correct answers (that is "short answers"), QATM must find a pobliztic (or exact) relation from those Q to those A's. This "relation" could be the same "universal INV" function that INV Lsrch looks for. T. problistic aspect of it can be then used to find the soln. by straight Lsrch!
SN t. PSLQ ("Partial Sums" of squares vector & LQ Lower-diagonal-orthogonal) Matrix factorization.) Algm.
For Integer relation detection! Given [x_i] real or complex: to find z_i (integers)
∋ $\vec{x} \cdot \vec{z} = \phi$ ($\vec{z} \neq 0$). This is a poly time Algm. (look in Google under "PSLQ") I think it might be used to find short exps (= integers) — but this is clearly not normally a poly time task!

6/23/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

(see 667.31) : T. main Discussn. will be IND & its updating: 664.00 .31
664.03-.04; .15-.19 are on t. "simple" way to do IND. (QA)! It (supposedly) deals w. "corpus" of cases.
How does this treatment of "simple" QA, fit in w.t. approach of CPM · N.8.01 — involving, say "Recognition functions" & MoirHeus?

In t. forgy. discn. of IND (QA) probs: I have been mostly (& unintentionally) discussing D-functs. rather than S-functs. To what extent can t. discn. be extended to S-functs?

.06 S-functs would seem to be a "larger space" than d-functs — But one can search over that space just as one searches over d-funct space. In addition, t. S-funct space is "soft", so we can start out w. "Broad" s functs, that fit everything to some extent — then narrow t. D.f. by "Hill climbing": It may end up being faster than working for d-functs! —

0 The shape of t. hill (Gorc funct) may be smoother for S-functs than D functs, so sometimes easier to climb. (Hvr see .35 Z)

11 Hvr. A big problem that I haven't worked on much! For each S-IND problem, one has to chose at least one S-funct representation to use. There are several such representations. Each PST for a problem may have a difnt 1 or more s-funct representations. At some pt., TM has to interconvert them, so it can make comparisons.

14 What I want to do, is give more detail on how QA is done.
One method is to find a single d-funct (O^n) for a set of QA's. This is uncondl. probability.
Trainer can give TM several such sets, identified by "indices" on Q's.
0 TM learns. sets w. separate O^n functs: Cones can be shared betw O^n's.
The recognition problem is very simple because of t. indices.
Later, TM learns to "index" unindexed Q's into various sets of indices.
Later, TM invents new index types. T. problems needing new index
25 types are not solved by "present" functs, so they are categorized together.
In .25 TM devises "Recogn. functs" In t. update process, S.I. consists of
a corpus p.c. This can involve changing recogn. functs.

30 When we "graduate" to S functs, we can start w. d-Recognition functions
then do S-recogn., because its "softer" & easier to hill climb.
We will always try to get all functs as close to 0 & 1 as possl. (to a pc of corpus)
So we will want them to evolve towards d functs.

37: (.11) But note that S-functs can be as "jagged" as d functs — (even more jagged) —
In t. report, I can just descb. t. forgy. stuff on d-functs, S functs: & ($\frac{d}{s}$) recogn. functs.
Having much detail on just how to do t. updating is not absly necessary: I do want to write a lot of (comments/hints on how) in my copy of t. report

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

668.40: Next, I want to disc. how concs feat pc's & how they may depend on context.
For d-facets, in which we have d-recogn. funcs, we can shave overhead of defns. by having distinct "branches" of the Recogn. funcs. .15

NB In CPM N.802 I had a set of R^* funcs that partitioned the corpus.
This is only 1 way of doing the partitioning. This Partitioning can be done in many ways.
have many possible forms, the one in (CPM-N) was simple-looking, but not nearly
Best, or even very good!

Hyp. Partitionings (usually d-funcs, but often s-funcs) is very impt.
When TM is first given a problem, the GPD partitions corpus (usually s-function)
by assigning 1 or more PST's to it.

.14 The partition Question is "What "kind" of (Problem or whatever) is this?". → 670.12
One very BIG, IMPT function of Partitioning: To ↑pc of soln; To make search "Easier".

15: .02 → For the "context" problem: I want to define ① the goal of the process: What's to be Maxed? What's Core?
② the corpus being used. ③ the existence of CB < Do is impt. Just what the CB is may not be so impt. (But I'm not sure of this).

19 The "top problem": Given corpus [Qi,Ai] to find O''s of high pc for it:
Subclass of methods: Buildup O'' from sub functions (= concs). This is guided by pc's obtained by looking at previous problems "like this one". The pc of a conc will depend on the "situation".
"Situation" ≡ "Context".

Note .20: This is a subclass of methods to solve .19. { Juergen's method .30 may be a method in class. If it is by hand, misses the only one point! }

.19 can be regarded as one of the problems in [Qi,Ai] corpus.

.26 .15–.26 does seem to clarify the "meanings of "Context": I think I have a similar, earlier writing that does about the same thing.

Note .15–.26 discusses a particular form of function Generalization — specifically, AZ141.

Using Juergen's method (657.33, 658.21...) the learning (S.I.) would seem to be quite difficult.
His probably params for each instruction are different for each position in the code string.
His codes are strings; Mine are more like trees.
A seeming Advantage of Tree structure is that it facilitates defns. of Common Subtrees — Corresponding to Common Concepts (Sub-funcs). Whether this is easy to do with "linear" String format, is unclear: I don't immediately see how to do it.
Maybe "common substrings" (≡ ngrams)?
{ The 2 methods seem to do quite different things. J's method seems to approach a d-funct at the end of the search. AZ141 doesn't seem to do this at all! It remains a very "S" function & throughout the search.
It would seem like a good idea to study just how the 2 methods differ. We could mix

6/25/02

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:669.40 t. 2 methods in a STEINish way (i.e. partial pooling of J's pc's 658.21) — Hvr, I'd like to mix t. 2 methods better so J could have "Strns" or "equivs": Benefit of a conc. that is used many times. (maybe put this Q "on t. Stack" → 611.) [In J: pc of conc. is funct only of nature of conc; & position in grmr (maybe). In AZ11: " " " " " of nature of conc., position from neighboring concs, nature of problem, previous probs, etc. → 673.3]

.02 [SN] In the Kullback-Liebler k-L formula ST8T3: t. logs of t. true & t. approxn pc's → 0. This is very impt.! It means that in t. case of very small pc's that t. Z will still converge (?) Maybe not so simple!

I.e. $p\ln(p/p') + (1-p)\ln\left(\frac{1-p'}{1-p}\right)$ that → 0. (p is actual, p' approxn)

for small p, p': $\ln(1-p) \approx -p$ so → $\approx p\ln\frac{p'}{p} + (1-p)(p-p')$; so $p\ln\frac{p'}{p}$ is t. largest term by far ($(1-p) \approx 1$)

10 & $p\ln\frac{p'}{p} \to 0$: well, p is very small, t. Q is how big is $\frac{p'}{p}$? — can be large if p is small. $p\ln\frac{p_1}{p}$ is usually $< p_1$ i.e. $p\left(\frac{p_1}{p}\right) > p\ln\left(\frac{p_1}{p}\right)$ | ϕ or p (B)

so again $\left[\, p\ln\frac{p'}{p} + p - p' \,\right] \to \phi$ — could partly cancel mainly cancel other term. So it's not clear as to what's happening: $\Sigma\, p\ln\frac{ep'}{p} - p' \to$ const.?? so ≈ $|p_1|$ average no big deal

2 669.14 T. partitioning problem: Changes/improvements ① Add new partition, ② Modify old partition ③ Merge 2 or more partitions ④ Delete a partition (part of ② or may be result of ① &/or ③)

In ④ we can sometimes delete a partition if its function is now done by one or more other partition functs. If we have S— (soft, stochastic) partitions, we may delete one if all of its P's's are very small & other partitions do have hi pc's in comparison (for t. same args).)

o [GPM N8.01] ← Discusses updating of d-Partitions.

Within a partition, searching for good P (produ) functions is somewhat easier. Context for concs will include t. i.d. of t. partition, perhaps part of t. partitioning function that are closely assoc. w. this partition. [Partitions ≡ "Categories" = "Types" = "Kinds" = "properties"]

Often, a new problem will have several "properties": Some will be more useful for partitioning than others.

T. entire system as I see it now, is rather easy to describ: A problem comes in, Q

GPD (PST$_i$ | Q, 1) obtains a Pd on PST's: T. index, 1 means that this is what we want from t. Q. (selection function)

(we could, alternatively write GPD$_1$(PST, Q) — where GPD$_1$ is a different element of t. set of PD's that constitute GPD; But then we would not have "sharing of info" betw. t. sections (transfer of lrng)

b → ~~after prob is solved (or during soln) GPD is "improved/updated". The criterion for improvement is~~

Next, the PST's will work (usually in ||T.S.) on t. problem, Q.

3A → They will obtain a PD over possib. answers, A_i. Is this pd "part of" GPD? —

35 → If so, what about t. PST's that worked on/solved t. problem?

In case of QA problems, t. answer will be simply obtained by direct applicn. of GPO — updating acts directly on GPD so 414.18 is unex. Updating can also improve algms to maxz. 414.18.

In case of INV problems:

Clearly a better way to update is for t. PST to be able to call & update algms & decide

6/26/02 ID 9:20 P 671

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 670.40 What aspect of updating it should emphasize. T. PST may decide that for this particular problem & t. amount of new info since last update, that it would be better to spend all problem time on t. problem directly. So: updating can be done before, during, or after or "ad libitum" ← chosen by PST.

It is perfectly O.K. for a PST to simply give a problem to GPD (say a QA problem).

See 670.34-.35: When a PST finishes a problem, it just stops! It can finish due to CB, or for INV probs, when it finds a soln. We might have an operation mode for optzn problems (including IND probs), in which T.M periodically outputs t. value of Gorc obtained thus far. Trainer can stop t. problem at any Gorc level desired!

.10 Well, say I can get it to work problems: Now, how does it (update/selfimprove)? cf (.19)

[SN] A [Qi,Ai] set can be integrated w. any other such sets, or have its own "O" max 414.18
If 2 such sub-corpi "are indep" how do they share defn. costs? (i.e. transfer learning).
Well, "M dependence" means that we have a recogn. (d) function that can reliably distinguish between t. 2 sub-corpi, & use different P's for them. Then t. system as a whole, has a single set of QA's & 414.18 is applied directly. The 2 P's for t. 2 subcorpi can share defn. costs, in a common "definition module". — Actually, there is only one O" function for both subcorpi, but this O" is realized by having recogn. funcs within itself.

.19 (.10) I had that that each PST could be updated in its own particular way: but I think that The update methods have a lot in common. 10^{12} By c 1000$ ≈ 1000 hours.

20

21 T. main update problems [redacted] for IND problems (INV probs can be formulated as IND probs in a way so that t. solns. are actually ≡ identical Lsrch. See 667.26-.31, 668.00ff 668.14ff, on IND & Update

.24R Maximum INV problem, soln pairs as a special kind of QA corpus. It ends up doing t. usual method of Lsrch soln. for INV. — Does this help at all w. t. problem of updating t. Guiding pd? → (.28)

So, list problem types & assoc. PSM's. Then dvb

25 How each PSM is updated & How t. system as a whole is updated (i.e. SI). → (672.00)

26

28 Re. (21-.25): T. IND [redacted] treatment of INV probs is not exactly t. same as T. direct [redacted] INV. method.

In t. IND [redacted] treatment, we want a funct that maps from problem defn to soln.

30 We want a type (short) funct. Whether cc is larger or not is irrelevant but most methods of soln (say using Lsrch) must also give smaller. (the Lsrch gives max (pc/cc))

Th. soln. by IND may not be "too Bad": which suggests that we may be able to solve optzn problems w.o. t. 2 stage construction of t. needed Guiding PD's:

.33 This 2 step construction could be very time consuming — tho t. first stage involves optimizing a 3 param (say) pd on cc for solns. then integration to get D.F. for "best" cand.). — DMAA
• The IND approach, while not exactly tite, may be O.K. — & it may actually be Better because so much cc is saved!

Could we use t. forgg. approx. for general optzn. problems? Certainly not obvious! obvious! → Note 672.16

6/27/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.0 (671.26 spec.) : 663.10 ff is time for INV probs & their updating
664.00 ff " " IND " " " " various good, general ideas.
665.36 – 666.01 : Fine general way to understand (explain) this TM model.) ← This is a v.g. summary of the System.

So: Write reviews of updating for INV & IND: 663.10 ff & 664.00 ff are two, but spell it out in more detail & look for other methods!

Also, Make sure I know just how TM reacts to new problems: I think it was trivial — but may be not!

SN Context Sensitive Grammars (as opposed to CFG's) may be able to implement an adequate set of String - rewrite rules, to simulate an UMC. So CSG's may be (possibly) universal.

.0 Is there any relation to context dependent pc's of concs in ≈ AZ141?
If so, this would suggest that even a small amt of such context dependence
Could increase t. expressability of functs in AZ 141 a great deal! (linear Bounded Automata)
Hvr, universal rewrite rules can get rid of a part of a (partially developed) string. My Context dependence in AZ 141 doesn't seem able to do this! Context sensitive Grammars are recognizable by Linear Bdd Automata. Many need bad or input string length — so they are not universal. problem

.16 (spec 671.40) ON t. 2 stage process of: Getting PD that a given PST will be best for a given CB for a given Problem.
In most cases, we assume t. PD is relatively insensitive to Variation of CB.
For Larger CB's, hvr, t. nature of t. problem changes, since "LookAhead" becomes possl.
.20 So we use dynamic pgmg ideas. Self Improvement?
.21 Does "S.I." normally involve Solu. of t. "Action Algm Evaln" problem?
.22 How are PST's improved"? Well we have this data set [PST_{ij}, G_{ij}, $Prob_j$]
We "pass a S-Grammar" thru this Corpus & obtain a general [illegible]
.25 t. PD (G | PST_i, $Prob_j$) distribution. This extrapolates, modifies, deletes (to some extent)
This process improves both the GCPD and t. [PST_i] set in a unified way
.27 Maximizing Apriori (GPD) · $\prod_{ij}$ GPD(G_{ij} | PST_{ij}, $Problem_j$) is a ≈ Non EI solu of t. "Update Problem".
.30 .27 is a "Max likelyhood" modified by t. Apriori (GPD) factor.

A constraint on solu. of .27 is that th. GPD function be in a form so that we can "integrate" it so that for a given new problem, we can list t. PST's in order of proby that they would give best score w. Given CB. → Note 674.00–.07: t. Monte Carlo soln. is not "cc intensive". } Does my Monte Carlo method deal w. both of these pts.?

Another constraint: The factor under $\prod_{ij}$ is a bit diffrnt for INV than for Optzn. probs

Also note (.16–.20)↑ is relevant when CB's are "large".

T. forgg, plus t. Max Corpus Thing gives a relatively "simple" formal Soln. to t. "Learning to Solve Problems" problem.

Note that PST's can "call" one another, or "Call" t. main system procedure.

6/30/02 673

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

:20: SN In Barron's 1993 paper (~ Neural nets – nonlinear approximations) he seemed to get faster convergence than ALP does. This may be because he imposes constraints on the function being approximated, that are different from the constraints I used in ALP analysis. He assumes that the hy w components of the true hypersurface ↓ at a ≥ a certain rate for ω > ω₀, or something like that.

The constraints I used were that the true system was describable by k continuous params — if not non-linear, was locally linear.

Note: This is "Curve fitting" not Time Series Regression.

SN Solovay showed that ALP will eventually make preds w. arbtly large pc for "0" (undefined) — in Li Vit book (Reflections)

Can we also show it will eventually give arbtly large error(s) in conditional pc of 1 a/o 0?

:11 :665.15 SN On "50% solution": Say we have 2 machines w. same cpu's: (A) operates w. optimum ratio of internal to external problem Time share, (B) has 50% Time share.

If we double cpu clock rate of (B), it will be at least as fast as (A) since it will spend at least as much time on internal & on external problems.

Yet I had the idea that A might work problems much faster than 2 × speed of (B).

Thinking about it now, it seems that speeding up (B) by factor of 2 should only double its prob. solving speed exactly.

How did I get idea that A could be much faster than B?

Say A spent 90% of time on S.I. first, then 10% on external probs: which it then solved 200 times as fast.

S.I. time is exponential in effect on external probs, so

Woops! I assume time share, so A can't wait until S.I. is done before working external problems.

Also the most A can be faster than B is by $2^2 = 4$ (if S.I. brings Exponential speedup.).

.23 Maybe not: exponential yield can be ≫ squaring by doubling S.I. time! Is this true? → The System will take

[right margin: No More Than twice as long as optimum system, to get as good a score as optimum system]

.24 :611.27 Back on track: How can Logical Reasoning (as a way to solve problems) fit into 672.27?

I have some recent good discn. of this problem. (I scanned 570.00 — 673.23, didn't find any reference! I'm sure there is a ref. in those pp. (>1 ref!).)

[right margin: on "J's method" vs. AZ141 669.30; 654.23ff / 654.23 contains dep. of pc of rules]

Anyway, I think the idea was that TM would lrn "logical reasoning" as part of its TSQ: It would eventually lrn to solve problems ~ to those needed for "Updating" — i.e.

:20 :670.02 SN On "J's method" vs. AZ141! In J's method: It is easy for the system to narrow down on a particular pgm & give it a pc ≈ 1. In AZ141; If Rules have pc's; we were only allowed macros. The only way to get hy pc for a pgm. is for that pgm to be a Macro. My impression is that it would have to be used > 1 time (which may be impossl!).

Alternatively, AZ141 could be allowed to have much context dependence of rules — Enough of this "Context dependence" can fix it so that ≈ only 1 pgm can be generated having pc > .5, say (or > .9 or .99!)

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

: when one picks t. peak of a ~~bunchwise~~ individual sample, the peak is spuriously hy, but it is ≈ t. largest expected individual (??)

Well, I suspected "No": here's a sample case: The population is in 2 parts.
(Set 1 One) has mean ϕ & $\sigma = 1$; (Set 2) mean .5 & $\sigma = .1$.

One would usually pick a peak from Set 1. which is bad!

So it mite be best to pick t. peak of $(x_i - k\sigma_i)$, where k depends on parameters of t. ensemble.

If ~~σ~~ σ's of individuals are unk. one can't do this. Tho one can, if t. individuals are in subsets of known σ, it mite be possible to ~~[illegible]~~ do.so

Does there exist a function $f(x_i, \sigma_i) = \alpha_i$ ∋ picking i w. peak α_i has expected error of zero.? (or some such criterion)

For SSZ betw. ~ 2 or 3 & 100 SOY doesn't change much; it's, say ~ .76 or .80

So picking i w max $x_i - .76$, say, would not be bad in ~~[illegible]~~ for many samples.

I want f(x) to be ∋ picking peak then subtracting proper SOY, gives least expected value.

$f(x, \sigma) = x$ is t. default.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.01 611.10,.17 [SN] Out GPD₁, GPD₂ (Quasi-hill problem): Since t. precision needed for GPD_2 (t. $Q \to PST_i$) isn't large, we can do integrations rapidly, or do t. Mt. Carlo method w. only 2 or 3 "pts" for each PST_i. So probably t. process wouldn't take much cc.

In fact, it may be possl to estimate optimum precision to use! It will be when t. expected G loss by using less precision is offset by t. fact we have more cc to ↑ G in other ways.

.07

.08 Re: Desirata: ① pc's sum to 1 or ≤1 eg. Lsrch CJS estimate is meaningful.

② if $G \to F(G)$ (F is monotone ↑ w. unique inverse ... at least monotonic, or $\dot{F} > 0$) then result is invariant. (Not true of GA).

.10

.11 Bad features: Interaction of pc's of PST's is not captured. — unless we recalculate t. pc's after we've run t. Best one for a while: As is, several PST's may be very

.12

.13 similar (say 10 of them). They all get ≈ wt. of ~ .1 (and one works on them all (effcy = .1 (!)) — But note .34

It would be better to work on these PC one, for a while and do simultaneous eval a, or sequencial alternation evaluations, then note that others have had pc's reduced — also first PST has been worked on, so switching to next as finely (This is WON approach which is certainly better than Lsrch)

.20 So, t. Best way is WON: Using the Prob vs. (cc, G) curves for each (problem, PST_i) pair.

Note this is Won for OZ probs — which is different from Won for INV probs.

Won for OZ only uses OR. — But interactive pc's make it very diffic.?

Another Bad feature of WON approach: It gives no CJS estimate.

.26 A possl. way out is to consider fewer PST's per problem (Humans do this). When we do use several, they are ≈ indep or interact in simple way.

→ PST's that use trials that can be also used by other PST's do pose an impt. kind of problem: It is desirable to take advantage of this kind of interaction, but I haven't begun work on problem! There is certainly a vast no. of optzn methods that use consecutive trials (may be most such methods!) } t. trial sharing problem

.30 The way we do this would depend (partly) on whether trials were cheap or expensive

.34 In .12–.13 time spent on ≈ PST's is not entirely wasted, since it gives useful info for induction corpus. to help evaluate t. set of PST's.

Consider a System to Solve QA's only: We have t. "2" PST's of 664.03 and 664.06

[N.B.] In general, when "Passing a grammar thru t. data" to update PST's and their pc's! T. Grammars I've considered do not allow problistic dependence of PST's. This may be a very Normal Characteristic of "Grammatical Induction"! — that t. pc's at set of ass's are indiv-

Consider "Grammar" of ...

7/2/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00: 674.40: [SN] I just found some versions of a [4-17-01] report that I wrote: looks very close to what I have in mind now! Just why did I discard it?

.02 Unclear! What were some of the diffys that became apparent after 4.17.01? (.31)

I write write up biggest diffys in present system (1) 674.10 ff seems like a big one!
(2) Maybe other "items on the STACK" of 611.00ff
(3) Development of WON: (I should put this on STACK of 611!).

[SN] I ~~dont~~ couldn't find ref. to ~~latest Metalet t. System~~: I. latest breakthru was realization that t. update i'beast of 665.26-.35 could be expressed as simply finding a good stochastic PST Grammar.
672.22-.40 is t. present model, & seems ≡ the 4.17.01 report (.00).
My recent idea, not clearly expressed in 672.22-.40, was that in fitting a model to the data on how good various PST's did w. various Q's (≡ problems), I was simultaneously solving all of t. update sub-problems of 665.20-.31. This is still t. "2 stage" assignment of pc's to PST_j, Q_j pairs & has t. diffys of 674.00-.40. (comparing better t. PST's)

.17: 674.40 A poss. soln. to t. diffy of 674.00-.40 ↑ Mite be to select t. best PST (t. one most likely to give max G for the present Q_j) ≡ "PST_o", work on it until it no longer looks like t. best bet — then switch to t. better looking PST. We do have to take into account that we've already worked some on "PST_o". This is not Lsrch (But note (.26)).

Of course t. individual PST's can be Lsrch (or Adaptive Lsrch) — In which case the different PST's mite have difent "Guiding PD's" — switching between guiding PDs is not so hard! I did consider modifying t. PD during Lsrch — which amounts to t. same thing.

.26: (.21) Well! The original model in which we time shared w. in t. ac pc of PST_i, was not Lsrch anyway! (It may have been inspired by Lsrch, but it has few of t. details of Lsrch.)

.29 → Is there a way we can do Lsrch on PST's? ← → 678.10 on Corrals
678.10

.31: (.02) I looked at ID 140.00 ff (4.21.01) This was how state of TM research at that time was more advanced than at $89: main pts.
1) I now understood Bussef Z 141 ... "Repressing" "market" a particular path
2) My understanding of updating during Lsrch "trees" is much better:
3) I appreciate more of "corrals" of cands for Lsrch (related to 2) → Also Note 674.11 ff on "corrals" A Major present unsolved "problem"
4) For induction: I didn't understand about t. "Encyc" matter of dealing w. large (mainly irrelevant) corpi.
5) I didn't appreciate that all heurs could be put into Lsrch form, since we always Lsrch over PST's — & any heur can be put into a PST.
6) I hadn't yet discovered t. various parts of Mxt corpus theory. (a) Soln of QA & OZ, & time series as part of same corpus (also, note "Encyc"). — The 2 stage GPD.
T. actual soln. of QATM (414.18)
7) I hadn't discovered proper criterion of convergence for induction on BAGS & [Q,A] exprs: i.e. "conjugacy of tht"

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

ro: 675.40 8) T. idea that cands could be PST's - they could be any pgm. — including things better than Lsrch, & things to replace t. "Top Pgm".

9) T. scaling problems: a) Lsrch → adaptive Lsrch. b) t c of cands hsould depend on context (or other ways to deal w. this scaling) c) Slowdown due to time spent testing latest models of "system function". (we now deal w. this by breaking system funct into parts, & by "statistical testing" also (finding / selecting) particular problems that are particularly good for clobbering a "System function"

(0) T. new idea of Recogn. Algms. helps much w. 9)

0 Other Pgm (10) — what new ideas since 4.21.01 (14 months ago). >

143.35 ~~A =~~, 144.00ff has an interesting possl. formal method of [redacted] General prob. Solvg. The Update Criterion is not obvious hvr.

13 11) → Also, t. idea of how to devise S-functs: T. 3 in put one was one that I worked out, → Note relevance! but actually, TM will be using many dif'nt methods to represent S-functs — The exact form will depend on t. problem: Hvr. all methods should be interconvertable so I can compare results, for updating.

12) Applicn of System to Hutter-type Probs:
A kind of "Gauss" of this method: Say n is size of problem; Solve random problem w. small n, using "scalable" techniques (= techniques scalable to larger n), then apply method to larger problem. If "small probs" are large enuf & we they are randomly constructed, t. chances of ~~them~~ t. successful methods working for larger probs, is quite hy!
Actually, this method is often close to one used by humans: We study how to solve small probs, but we only use trial methods that are scalable to large problems. We may occasionally obtain methods that do not scale (we have no proof that they will scale) but this will occur rarely. To descrb. t. method, mention use of hypothetical diagrams to "prove" thrms in Geometry.

[SN] "New" [Iverson, APL, "J"] programming lang: J is t. result of many yrs. work in looking at use, dialects of APL. I have several papers on this! Perhaps best is Iverson's history of it — 1991. See my files on "J-langcg." ~~or~~ — "folder in "SS2". I'll have to be more familiar w. APL & J before I can get much from Iverson's "History of J" — Tho it gives motivations for various features. I may disagree (much) w. him on his conscious reasons: But he may have better reasons he's unaware of!

→ Anyway, I want to use J/APL in TM, a/o [have TM able to invent it —]

since TM is always moves toward "terse notation".

Iverson's latest idea is "Tacit Pgmg" — this is functional pgmg, in which "Tacit" t. args. are not explicitly stated.

The 80 p. "Introduc to J" by H Dorres is brief, good. I think he claims he can manipulate formulas as Arguments (like LISP).

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

:676.90: There are ~ 200 "primitive functs" in J : Those are "wired in" functs, Probly there are only a few "primitives" in the usual sense. T. Q is : how many of J's "prims" would TM be able to discover? Also, would it ever some others? Essentially, all those "prims" are are "Macros" — wasteful functions that it is "useful" to define, because they compress corpus. I want to make sure that TM has adequate learning for discovering any/all of the 200 "prims"

.06 So this is Interesting! — That I have this "Grand Loop" = TM that should be able to acquire any ° of skill, if it can "get off the ground". A I have the 200 "prims" That are a kind of Measurement of TM's heuristic power. A Q is: "How are those a better Measurement" than other skills: (e.g. ability to discover (linear/mult.) regression in a real Time Series')? I guess it's because the "200 prims" are closer to "primitive" — "near the bottom" — Very "fundamental". We could, of course, "wire in" those 200 prims" — i.e. give TM "J" to start. Would it then have an adequate set of "Macros"?

The basic Idea of the "TM Loop", is That whenever TM solves a problem, the "present" update Alg makes some of the new info flow into the "next update Alg".

IMPLICIT
Iverson feels That the J's ability to do "implicit prog" — i.e. define functs w/o using Dummy args — is very impt. This amounts to being able to build manipulate strings That can later be "told" to execute them as representing a function (see in LISP! Execute & Quote)

IN GENERAL Re the "TM Loop" (.06). 3 impt components: (1) The set of PST's, (2) The PST used for "updating" (3) The TSQ. During TM's Development, I Monitor (Keep watch on) 1, & 2. If 2 isn't growing properly, I have to put stuff in TSQ to guide it a/o hand write in improvements to PST's in General, & the update PST in particular.

The "Circularity" is perhaps via the Duality of the pgms That TM can & "cannot" write (& "cannot" means maybe "can write", but KB is quite too far out).

As is, I'm starting w. Lsrch (w. various forms) for the PST of the Update Alg. — Also using (as) some PST for induction & probly INV probs (Prog). I could add other Induction methods of 663.06 ff ("regularities"). Also various Partitioning Methods ("Recogn functs").

.35 Dud Then, There was the flaw that I had in Mind in § 589: Simple Adaptive Lsrch (w/o Trng during Lsrch). If the CJS was too large, I put stuff in the TSQ to give "Bridging Concepts" (Bridging Converges). I Guess what I was worried about in .35 was That certain Macros were not accessable this way, i.e. certain Macros involved "learning" from Sequenced trials.

6-5 1811 Pan
6-5 45 Harm
45/18 = 5/2 = 2.5
ln 6/5 = .182
18/.182 ≈ 10
.182 18"
1 100"
∴ 10M ℓ

7/8/02 1D

Now f. data explorer bk. | Old ones starting page! Bot-v6/3/02 p P699 ~ 30/1 40 B. to Ray sw written. — Say 40pp to 2 — 3:20 → 4:20 P. — 678 — 5+/1

00: 677.40 : 677.23 → .40 are a summary of ways to deal w. expected diffys in TM.
Perhaps write this up in more detail. It is a kind of summary of "where I am" in t. TM project.
There is also t. very General Non-cl. Update soln. of [672.27]
677.23 – .40 are variations within t. Genl. Soln. of 672.27

→ (672.27, (t. Gen soln.) does seem to have t. diffy of "Corralns": 674.11. see discns. of possl. fixes.
"WON" is probly t. best way to deal w. Corralns. (See discussion after 674.11) but I haven't much
worked it out. Hvr. Go over t. possl. "solns" to t. problem on 674.11 – .40, ff. 678.10 → looks v.g..

10: [SN] 675.29: (On Corralns) A joint pd on [ ] × [probj] would have an Enormous Amt. of info in it! } Not Bad! ≶
for a given problem, hvr, we need only t. joint pd on [PST_i]. still exponential in no of PST's.
What we need at most is t. cross corralns of t. PST's. (wrt problemj): for n PST's this is
only $\frac{n^2-n}{2}$ params, We mite break up [PST_j] into ≈ "equiv" sub-sets.
(hy cross corr within sets, low cross corr betw sets). Then if there are k sub-sets – its like
k indip PST's. — so do them in // by Time share w. pc wts.
We mite obtain very approxt. "equivt. classes" by Clustering.

19 So perhaps t. big problem is not how to Use t. data re: its output of GPD, }
20 — but how to get t. corralns within GPD, in t. first place! Perhaps, if there
are just, say 10 or 20 PST's that mite be relevant to a problem, one finds
t. cross corralns. just before working t. problem — because they'd be difnt. for
24 each problem. → (679.35)
— In fact Much of GPD, will have to be done "just before each problem", since
we wouldn't otherwise know which aspect of GPD, we really need to
"Update". So, while "theoretically" a perhaps to simplify Explnn.,
30 One mite say the Updating can be done after each problem ---- in fact,
(& "in practice") most (if not all) updating will be done just before each
problem.

One aspect of updating that is done just after new data comes in
corresponds to "looking for an "Explann" for t. new data" — Then,
perhaps after a few explanns are found, to try to think of what domains
those explanns/regularities are relevant to. This is certainly what a (679.40)

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

:678.40: "Good Scientist" does.

So it may be that there is a normal "logic" division of the updating process between "after new data arrives" & "just before each problem is solved".

However, the update after new data arrives, may be an artifact of human memory. It seems to have to do w. (where / in what form / whether) the data is stored. ("whether" is perhaps of most import) — since humans probably do not store all input; though a machine could. Even w. machines, the Q is "At what level of accessability (cost) the data (& its partial compression) should be stored" — this is common to Humans & to Machines

10 At first, I was thinking that GPD_1 would always be "completely" updated after each (Problem soln / new data input) — This is clearly incorrect. "Completely" is meaningless, since $CR < \infty$.

For Humans perhaps much of the "post-data update" involving "looking for an Explanation" has to do w. expected future utility of such data/explanations. Also, at for post-data time, one has time to do the update, & it seems that one will probably have to do it evenually, so one uses that time for updating. This would seem to be

20 marginally relevant to machines: A machine should be using all of its time usefully. If the machine has some apparent "free time", it should be using that time for updating. However, what part/aspect of the GPD_1 (or "GPD_2") should be updated, is unclear.

It would seem that raw data coming in would be a candidate for updates. Also (perhaps of most import) — one must compress the input data anyway to know how much CC should be spent updating it! There is, in general, this uncertainty about how much CC to spend updating something because one doesn't know if it's going to be useful. — So risk is involved:

30 One may find the update gave very little of expected value..... so one has to guess or make decision about whether & how much to compress (& update) data ---- that such decision is being continuously made as the data is being (processed / updated).

35: 679.24: In fact instead of fitting the crossover, or clustering model to the complete ("exact") GPD_1 We will very probably try to express the GPD_1 in the first place as a crossover or clustering or sparse classes model.

7/9/02

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:679.40 this (particularly the "set of equiv classes" model) could vastly simplify (construction/computation/formulation) of the GPD$_1$.

{ So 678.19, 679.35 680.00 : looks like an adequate problem
.04 Soln. to the "correln" problem. → 684.31 }

05 : Another problem of perhaps much import is J's method v.s. AZ141 — Mainly it has to do w. adequacy of AZ 141.

Perhaps a related Q: How adequate is 677.35-.40? [Conc net, "Bridge" concs . . .]

Note: in 677.35-.40 (conc net . . .) Once one has used a conc to solve a problem, its pc will ↑, but it may not ↑ enuff to warrant a "definition" . . . even when it is used 1 more time.

In general, I'm worried about the "Scaling" problem that occurred w. ANL at Saarb. — That for n = no. of concs, the pc of each $\propto \frac{1}{n}$. So solns w. k concs have pc $\propto n^{-k} = \frac{1}{n^k}$: Main ways I thot of to ↓ this effect: (1) Breaking Corpus into α parts, so $n \to \frac{n}{\alpha}$ (2) have pc of a conc dependent context [(1) might be considered a "special case" of this. — That "Type" of problem was a kind of "context".]

Say a (soln.) problem involved 3 mem (not so hy pc) concs & some "connecting" concs — for human to discover such a soln, hvr, would be very fast: A good part of this speed is due to seeing analogys w. other problem solns & other Heuristic tricks. I really have to find more Search heurs for problem solving. The Conc. net itself should tell mainly how to solve the problem, but other heurs are like Catalysts to enable the solns at "reasonable pcs" →. . . .

So what I may need to do is first make the Conc net, but see what other "Catalytic" Heurs are used by Humans.

I was going to say that Lsrch uses additional techniques not used by Humans & their "Conc net": Not so! Lsrch gets solns w. cc $\propto \frac{\text{cc of soln}}{\text{pc of soln}}$. We prob. can't do much better!

Except that Lsrch is definitely not optimum: WON can often give much better results! WON takes advantage of More info (facts) in the GPD$_2$

[Perhaps Impt. weakness of Conc. net analysis: Often, in working on a problem, Various unsuccessful (cands) trials will suggest much better cands. Learning during (srch) Lsrch]

This amounts to Gross changes of pc dt. of cands by learning from 1 or 2 cand trials.

— which is "like": choose a hy pc cand. α: try (it): from (what happened) (Learning). decide on one or more new cands to try: go to α

7/9/02 681

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 680.40: So: 680.05—.40 points up t. need of a hy level study of just how probs are solvd — i or conc. nets i assoc. "Heurs" exact.

: Conclusion: T. best ways of deciding on which PST to use,

1) In all cases do GPD, first to get Pd of G as a funct of (Probj, PST_i, cc)

2) Do 678.19, 679.35, 680.00 as a soln. to "corveln." problem

3) Or if df's for t. PST_i's are indip, I can use a WON soln. (Req needs be developd, but old ideas at Sarb, i more recent works would be useful for review)

.09 4) if assuming indipc. of t. PST_i d.f.'s is possl. T. classic PST_2 approach, using

.10 M.T. Carlo to make choices, is not bad. Hvr, see recent note: → .10R

.09-.10: It may be possl, to find (or 2) params of t. GPD, df of a PST_i that characterize it well enough so that c(t,cs) (param) ordering (in order param; If 1 param, partially order i use tables or more time consuming methods for deciding order.

Probably t. most diff't part of TM design is 680.05 ff — T. design of TSQ's: # Conc.nets — "Heuristic factorization" (finding set of adequate heurs (which may not be so easily expressed in t. conc. net!)

[Tho perhaps By definition, all needed Heurs are in t. conc. net]

So, if a heur is a conc in t. net, then it would seem usually essential that these concs (Heurs) be context dependant.

.20

21 **SN** I have this idea of "Heurs" (as components of t. conc. net) i also t. idea of PST's — which are not part of a conc. net. Certain of the PST's (maybe large only) use t such i conc. nets. [in t. conc. net] — Other PSTS can use any derivable methods of solving problems. (Conceivably a "Heur" could "call" a PST!

It (pretty is) may be necy to allow "Heurs" to be PST's or parts of PST's, if we want all Heurs to be expressable in t. system (since a "Heur" could be practically anything, i Heurs in t. conc. net seem to be rather restricted in what they can do: Modifying pd on order to t such trials (?) ← unclear as to whether t. Heurs in t. conc net can do (approximate) all poss reorderings of trials) ← A even reordering

.30
.31 of trials doesn't cover "Quick abort" type of heurs.

What Has to be done for TSQ analysis i design: To take various Disparate kinds of Heurs i see if I can fit them into t. models of .21-.31. If not, fix t. models. What I really need is a better way of categorizing ("Typing") heurs.

2/10/02
FD

682

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 681.40: [~SN] ~~[illegible]~~ My ~~hot~~ about hours is "Like", I have solved many (diff'nt) problems: t. solns. have certain features in common — I want to emphasize these features in trials for soln of new **problems**. "Induction" seemed to be more like ≈ t. same problem — or, essentially only one problem w. no antecedents ("Max" (e.g.) 419.18")

04 In fact "Max 419.18" has a "history", (≡ corpus) of $\sum [Q, A]$ corpus + O^n ≢ Q_{n+1} → ~~[illegible]~~ (O^{n+1})
Its not exactly t. rite ~~[illegible]~~ way to look at t. problem, hvr. T. ~~tendency~~ demand ≈ implied is that we use a system ~~to that~~ ~~[illegible]~~ usd successfully in t. past to obtain a new O^{n+1}: (.04 is not quite rite! T. corpus of pairs is
$\left[[Q_i, A_i]_{i=1}^{k}, \; O^k \right]_{k=1 \ldots n}$, The implied O^{n+1} dfi for input $[Q_i, A_i]_{i=1}^{n+1}$ (i.e. t.
12 "A_{n+1}" corresponding to "Q_{n+1}" = $\sum [Q_i, A_i]_{i=1}^{n+1}$ (diffn't modifs of Q's, A's.) Then acts on Q_{n+1} to give t. desired A_{n+1} d t. → .27

T. problem (for me) is: I don't see any "Item seq." to suggest modules! — Little repetition of concs, — Nothing like t. repeated use of various hours or various ⟨concs ≡ sub-functions ≡ sub ideas⟩.

-18 Perhaps we'd get it, for ~~[illegible]~~ t. [QA] problem, we had a recogn. functn for each Q type & a separate $F_i(Q) \ni A_i$ d-f for each Q_i. We could then share
.20 concs betw f_i's.
.21 Later, we try to "integrate" t. f_i's & modify t. Recogn. functs so as to ↑ fit corpus
.22 Earlier, I had ~~[illegible]~~ that of .18 – .21 as a "Not so good, but is all else fails, try it" type of soln. — but it may not be so bad as t. usual general way & "first try" ~~to~~ type of soln. We may want to get very many $f_i(Q_i) \to A_i$ functs so we can ① all these pc's ② Have alternative ways to do t. "integration" of .21.

27 (.12) → Another Big trouble w. (.04 – .12). Its ~~[illegible]~~ just extrapolating t. corpus; its not (ordinarily) trying to maximize ≅ 419.18. It just looks for (f(Q) → A)'s that are similar to t. successful ones of t. past. — This is not a bad way of doing it, but it is (8)
30 **Certainly not t. Best** (.305) (→ 683.10) (Tho could TM realize that "Performing Optimization" was t. Best, ~~[illegible]~~ we could find
31 ~~Unostive~~ Highest **Pc** model for t. past data!?!?) → 683.10
32 [SN] re: (.305 → .31) This may be fine! An "Optimization routine" is a pgm. It will, indeed, get close to t. desired corpus w. ~~[illegible]~~ hi pc (w. CB.) (if optzn pgms have hi pc — even just t. optzn idea — perhaps have optzn idea, be a primitive). TM doesn't have to ~~[illegible]~~ understand what optzn "means", in any ~~[illegible]~~ Human way — yet it could use it as a way to (write / derb) pgms & t. a way to assign pc's to pgms.

7/10/02 ID [illegible] +8 = 5/8 = 54/p. | Line 21: Switch to pen 7/8/02 #1 | (for stack 1.08) 611,000... 683

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

=0.68 2.40 : Still not clear in my mind as to whether (682.305 - .40) [illegible] is correct a/o [illegible] mechanics of how it would work.

TM could "lrn" t. conc. of "optimization" after being given a seq. of examples of it. Ordinarily, it take much. cc to verify that something mite be an example of optzn w.a certain CB. Theoretically even tho t. "optzn conc" mite have an initial (apri) long descr — w.r.t SSZ, it could be as good ("explain"/model) of a corpus. Its not clear tho, that having TM lrn. what "optzn" means is much better any. Better than having it be a "wired in" conc.

08 (≡ 682.305-.40) Hvr. (.00 ref) would seem to be not very critical at! Potential stack.

09 Note. that a pure Q.A.TM mite lrn "titive way" to do INV problems in a n way!

10 (682.30 / 682.31) Hvr. 682.305-.40 has a more serious implication: Unless I show it works fine, I have trouble of 682.27-.30 — that TM isn't lrng (optzn/updating IND) properly!

A way to (somewhat) deal w. this: Consider a few (or one) method(s) of finding good "414.18" maxzns: T. finding of "regys" in t. QA corpus. The idea here is that t. "Corpus" for this problem is an "ordered" sequence of $[Q_i A_i]_k O^k$ pairs $k=1|n$.

The "set of keys" for O^{i+1}_k almost always includes those for O^i_k; & usually some new ones. In this sense, t. (code/descr) of O^{k+1} is a/usually a smallish modifn. of t. (code/descr) of O^k.

20 of O^k. A smallish modifn could be regarded as a "Mutation". So, t. Q is: "What's (t. best/a.v.g.) way to do this? (A (extended) context-dependent Mutation)."

Most Generally: One has various possl. "Update Algms" (≡ PST's). To find Better ones: T. [illegible] Core Being: "How much ↑ in 414.18 for a given CB". (for comparison of some Algms, CB doesn't have much effect — for others, it is very impt: for very large CB, one can do "Look ahead" & do "experiments" to get info.).

Hvr, rite now, I want to simply Decrb T. current update/induction algm. — (not nec. looking into how to improve it!). I guess this is to show how "Context of [illegible] peaks enters [illegible] —

31 how it differs from t. "Default" (i.e. AZ(4)). — 684.03

"Back tracking" in this Model: To make trials for creation of O^{n+1}: we start in successful $O^{n+1-\alpha}$ models (α is "t. amount of Back track") & we make trial "mutations"/additions.

Trouble w. this: [illegible]

If $O^{n+1-\alpha+1}$ was made by a modifn. of $O^{n+1-\alpha-1}$ (which is quite possl.) then modifying $O^{n+1-\alpha}$ will not catch t. particular soln. that $O^{n+1-\alpha+1}$ has to its corpus.

7/11/02 ID 68~~4~~ 684

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : 682.40 ! So; for each O^{n+1}, we will have a backtrack index bt_m, that tells how far back it backtracked to get its soln. Normally ~~to~~ to discover ~~...~~. O^{n+1}, we will ~~...~~

.02 To Be continued.......

.03 : 683.31 [7/13/02] One way to modify O^n : (Simply Backtracking by 1, would merely I'd have to take at least as long as it took to find O^n plus more, because I have an additional constraint: i.e. ~~...~~ $O^{n+1}(A_{n+1}, Q_{n+1})$ must be large.) : One ~~...~~ "small modifn"!

TM "recognizes" Q_{n+1} as different from all previous Q's; It uses O^n for all other Q's & devises a special O for Q_{n+1}. This combined algm $\equiv O^{n+1}$. ~~...~~

We do not need t. additional recoh ~~determining~~ O^n in this "soln". We may then try to improve this "O^{n+1}"

10 by ~~...~~ (generalizing / simplifying) (Class constraint in) t. recogn. algm, so it recognizes some of t. past Q's — yet gives O.K. results — also, perhaps modifying t. assoc Q $\overset{S}{\to}$ A algm.

So .03 ff. above gives one meaning of "small change in O^n".

Another possl. meaning is, perhaps, obtained by logical analysis of O^n — that

17 modifying it in certain ways will only change its response to certain ~~...~~ Q's.

(Sidenote, right margin:) for Baker for Dr for work. (Even when she has to Corpo. Even however so... bay (C that will heat day) (Milston round my neck for life) I now pronounce you Man and wife.

20 The most impt unanderstood Q about TM seems to be 682.00 — 684.17 : Essentially : "Just how do we do IND updating"? Perhaps More to t. pt.: Just how does one or more of t. Relevant PST's work?)

681.00ff is a review of how TM decides which PST to use.

Updating involves (least el) Finding a good grammar s relating Q's to PST's & ~~...~~ df's of Gorc ~~...~~ as a funct of CB. At first we ~~...~~ this into Getting a good Set of PST's; & Getting GPD to relate them to incoming Q's.

.21 is on how to design a Good PST for QA problems. — what I was Mainly Concerned w. was a Rationall for context dependent pc'oh concs : (also relationship to J's method.)

30

.1 : 680.04 [SN] It will probably be possl. to ~~...~~ design a (approx soln.) to t. "correln" d.f. ⇒ it will be very Easy to tell when to drop a particular PST & which other to jump to. T. approxn. form of t. joint D.f. can't have more than a few params, because ~~...~~ there isn't much data, & t. models I would be using, usually aren't v.p.! (☺) (— so I couldn't afford many params)

So this whole bussiness of Sequencing t PST's could be "very easy" — "very smooth"!

36 A ~~...~~ Reasonable approach to PST for induction: Make a grammar of ~~...~~ S - Functs that have been used ~~...~~ as ~~...~~ O^n trials: ~~...~~ use G (≡ corpus pc) as part of corpus, (See stuff I (relatively) recently wrote on GA for how to do this). [568.00 - .20] From "G corpus," I can get Good Trials in a Mt Carlo way, or, because of t. Simplicity of t. G corpus,

I think this is correct.

7/18/02

TM in "STABLE STATE": 06

685

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

50:684.40 I may be able to (apparently) integrate from already here: Do go on to Pty Deeper: it looks quite Reasonable!

T. idea of 684.36 is simply my realizn. that INP problems should be solvd some way as any other problem (INV or OZ). This seems to solve one of last outstanding TM. probs (other than TSQ!).

There certainly are other problems (e.g. see 611.00 ff re "stock"), but at present, T. system seems "stable" enough to write rather than fatal "report". (But do note 667.26 is innocent beginning of discussn. of updates of IND funcns. — It also follows thru to 685.05 ff.

T. statn. of soln. of the IND problem is a recursive defn. The "Boundary Cond" is induction using trials of θ^{th} in pc order w. Lcost cutoff: Note that this is not "Lsrch". Lsrch always has possl. of only good soln. This "Boundary Cond." does not have that possy.

It might be best for Exposition, to defn. & pursue induction (QA probs on the) TM in detail — Giving some reasonable (initial) "Boundary values" for various types of IND problems of interest. It should be possl. to write TSQ's thereafter by machine. INP & OZ probs capabilities could be added later — but the "recursive IND Engine" is what I mean "driving force" in the system.

NB: in 684.36 ff, we realize that a {GPD1, GPD2} approach is necy for IND probs, & that it probly relevant to the evaln. of pc. of Conds wrt. Contexts — But the exact details of this "soln." to the "context evaln" problem has yet to be defined.!

(7/21/02) Ah! Re: Recent work on "correlations" betw. Lsrch trials ("conds") (Granta House Thrm.)
First GHT says if trials are indep (& consequently on cc's of trials), then $\frac{pc_i}{cc_i}$ order of best. If trials are not indep, then joint PD is known, the best path may be diffl to compute (NP hard?). If we could do the tricks I discussed about "correlling", this could serve "lowy during Lsrch" & ↑ speed of soln. considerably: Maybe discuss w. Levin.
∃ 2 ways we can have "Low" during Lsrch: (1) T. Joint GPD over conds remains constant during Lsrch (Pc's have also "corrects" that we can take advantage of.
(2) T. GPD is actually modified as the during Lsrch, as a result of what happens during Certain of the trials. : It may be possl. to assume the cond's pc's are indep, & do (2) & this may simulate (1) to some extent (?)

(3) An approx. GHT (.22) for correlated Conds: Pick cond w. max ("incremental") $\frac{pc}{cc}$.

The (Revised) GHT: we have the Joint PD on conds.

Woops! In the original GHT, there was only one prize. Is this a nasty cond. for

7/25/02
IP

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:685.40 Max $\frac{pc_i}{cc_i}$ to be "Best choice" soln.? I would guess not. I which case a joint PD on cands is possl. & Meaningful. (means 686-1 = 685.37-.38.) So consider -1.37-.38! As soon as we make a trial & it fails, we have a new (Marginal) P.D. We can pick candn. max $\frac{pc}{cc}$. (we know cc_i's in advance)

This is a "Max Greed" Approach. A less Greedy approach looks for trials into future,

This is a Dynamic Pgmg. Problem.

Consider $k=2$: We chose our first cand c_i: prob of win is p_i; $cc = cc_i$;

So pc of win with $cc_2 = cc_i$ is pc_i: If c_i loses; its ($pc = (1 - pc_i)$) following least cond. from p_{ij}, c_{ij} formula.

~~So chang~~ Then pc of win is $(1-pc_i)\, p_{ij}$ at total cc of $cc_i + cc_{ij}$

I think we chose cand of max $\frac{p_{ij}}{c_{ij}}$.

So pc_i at cc_i win cost | $(1-pc_i)\, p_{ij} = pc$ (cand of max $\frac{p_{ij}}{cc_{ij}}$) at total win cost $cc_i + cc_{ij}$.

I have 2 trials! At the end of them, I have certain pc's of win; certain pc's of total cc.

— which trials to do?

We have choices betw. "Objects": Each consists of several possl. outcomes of various pc's:

Say

p_1	cc_1,	win	v.s.	p_1'	cc_1',	win
p_2	cc_2,	win.		p_2'	cc_2',	win
p_3	cc_3,	no win.		p_3'	cc_3',	no win.
$(= 1-p_1-p_2)$	$(= cc_1 + cc_2)$			$(= 1-p_1'-p_2')$	$(= cc_1' + cc_2')$	

Which is better?

maybe (p_0, cc_0, win is a third possl choice (?)
($(1-p_0)$ cc_0 no win.

This Problem Seems Very Impt! I really don't have to Solve it just Now! → 687.31

Put it on the Stack: 611! Also Note: It is not exactly what TM needs to solve: The exact, needed, problem May be easier, or HARDER! But solving present problem is a useful step prob'm: Shows how they have useful approx. soln.

→ 689.11 may be a formal soln!

(Right margin note: If: Cands are divided up able into several "equiv classes" of correlated Genc, Ppon, A Good strat would be to pick the best poss from each equiv class. Each time it fails, we get rid of a lot of trials of low Genc. This is a "practical" approach in Real TM as well as a poss...)

An outline of the Paper:

Introdn: Gives dems of types of problems solved. INV; OZ probs:

Derb. END prob (QA): ~~show how it is special Case " of OZ problem.~~

Say that it is a special case of OZ, but don't explain how, yet! Give ref. to Section where it is discussed.

A nice way to Organize the paper would be to follow the Abstract:

Derb. INV probs. in detail; OZ probs! Give lots of examples. { Give diffnt kinds of OZ probs: { Lang & short TM w. & w.o. noise; Time Limiting -- ect.

Then discuss QA probs as impt. kind of OZ prob: Give 414.18, w. explanation but no many detail. I.e. don't tell how to Get approp of OZ. → 687.00 →

Perhaps/Just Go thru material of Abstract, in a little more detail.

~~Thereaw~~ Then, in intro, I list contents of following sections (other than Appendices).

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:(Spec 686.36): Derb. GCPD: A very General cond P.D : Give Google as an example.
Explain how it contains ~ all "knowledge" in TM. (if we include PST's as part of GCPD).
Perhaps discuss "index nos." that (sometimes) tell it what "kind" of response is wanted.
Perhaps be a bit vague at first about how t. GCPD figures out what we expect of it —
Then introduce "index nos".

Unclear as to how much to put in Intro. It should include info from Abstract, but give
Some more detail — also Give section no. where more detail is given.

.04 Intro: Start w. history of project (86, 89, 94): Give general goal of a very useful
.0 ~~[illegible]~~ "Scientist's Apprentice" The convergence theorem of Alg. Prob'y
suggests that ~~[illegible]~~ if there were no limitations on computational resources, we could ~~[illegible]~~ easily design
~~[illegible]~~ a machine that would be extremely good at prediction —
~~[illegible]~~ at least as good as any existant practical algorithm. ~~[illegible]~~
One of the main goals of the present project is to ~~find how good we can do
with limited computational resources~~. determine the theoretical and practical
effects of limitations in computing resources. It should be noted that this work
0 is only distantly related to computational learning theory, which is concerned with the system's
~~[illegible]~~ assymtotic ~~[illegible]~~ behavior for ~~[illegible]~~ large problems. Most of the solutions
we find for problems ~~[illegible]~~ have computational complexity that is exponential in
problem size. The system works by modifying its vocabulary ~~[illegible]~~ so that the
problems to be solved are small. It is our impression that ~~people usually solve problems~~
~~[illegible]~~ this is the way that people usually solve problems — by reformulation,
rather than ~~[illegible]~~ extended search (maybe reference to Minsky
"Society of Mind"). ⟶ ⟶ (68920)

0
.1 : 686.25 SN Toward a criterion for cond soln: say ~~[illegible]~~ "r" won; want to win in k.b. trials, we
want to order all k.b. cond. sequences ~~[illegible]~~ ending in "r", ∋ expected
cc is min. (Q: for original problem: I think > 1 soln. (≡ winning cond)
is ~~[illegible]~~ legal. — does this apply in original GHTI proof? — does it apply in
present case?)

A possl. soln. to t. problem: Say we decide on a sequence of conds to try,
w understanding that we quit as soon as we find a winner".
-389 There are M conds: we have a list, C_i; $C_i = 1/M$. ~~[illegible]~~ $\leq C_i \leq M$.
C_i is t. "name" of t. i^{th} cond in t. list ("Names" = 1/M.

7/28/02 ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

DD : Say $c_1, c_2 \ldots c_M$ is a possible list.

For that list let pc_i & cc_i be the pc & cc assoc. w. card c_i.

Then, for each adjacent pair of cards, we could interchange them to obtain a new list. pc_i, cc_i ; pc_{i+1}, cc_{i+1} are the original pc, cc pairs.

pc_i', cc_{i+1} ; pc_{i+1}', cc_i are the ~~interchanged~~ pairs after interchange.

I suspect that interchange would ~~be~~ give a lower "expected" total cc factorism,

10 if $\frac{pc_i'}{cc_{i+1}} \cdot \frac{pc_{i+1}'}{cc_i}$ was $> \frac{pc_i}{cc_i} \cdot \frac{pc_{i+1}}{cc_{i+1}}$ i.e. $pc_i' \cdot pc_{i+1} > pc_i \cdot pc_{i+1}$.

But first look at the ~~[illegible]~~ Genc.

For the original C_i vector of 687.39 expected cc before soln. is:

.14 $$\sum_{j=1}^{M} \left(pc_j \left(\prod_{i=1}^{j-1} (1-pc_i) \right) \left(\sum_{i=1}^{j} cc_i \right) \right)$$ } we want to Minimize this.

pc of success on C_j; prob'ty of failure before C_j (this is not the product of indiv pc: it's the product of condl. pc's); Σ cc of success on C_j

I think I used this proof method for the original GHTI, but ~~&~~ was the Genc as complicated looking? There are only a few terms in the Σ of .14 that change

20 when we do interchange of ~~[illegible]~~ c_i & c_{i+1}. for $\sum_j$ ~~[illegible]~~ I think ~~[illegible]~~ its ~~[illegible]~~, j = i & j = i+1 that are critical. $c_j \; c_{j+1}$

For the original order, we have

.25 $$pc_j \cdot \prod_{i=1}^{j-1} (1-pc_i) \left(\sum_{i=1}^{j} cc_i \right) + pc_{j+1} \prod_{i=1}^{j} (1-pc_i) \left(\sum_{i=1}^{j+1} cc_i \right).$$

For the permuted order.

.28 $$pc_j' \cdot \underbrace{\prod_{i=1}^{j-1} (1-pc_i)}_{A} \cdot \left(\underbrace{\left(\sum_{i=1}^{j-1} cc_i \right)}_{B} + cc_{j+1} \right) + pc_{j+1}' \left(\prod_{i=1}^{j-1} (1-pc_i) \right) \cdot (1-pc_j') \left(\sum_{i=1}^{j+1} cc_i \right)$$

α: $\sum_{i=1}^{j+1} cc_i = B + c_j + c_{j+1}$

30 rewriting .25! $$pc_j \left(\prod_{i=1}^{j-1} (1-pc_i) \right) \left(\left(\sum_{i=1}^{j-1} cc_i \right) + cc_j \right) + pc_{j+1} \left(\prod_{i=1}^{j-1} (1-pc_i) (1-pc_j) \cdot \sum_{i=1}^{j+1} cc_i \right)$$

quite similar!

.28 → $pc_j' \cdot A \cdot (B + cc_{j+1}) + pc_{j+1}' \cdot (1-pc_j') \cdot A \cdot (B + cc_j + cc_{j+1})$

.30 → $pc_j \cdot A \cdot (B + cc_j) + pc_{j+1} \; (1-pc_j) \cdot A \cdot (B + cc_j + cc_{j+1})$

.30 → $pc_j \cdot A \cdot B + pc_j \cdot cc_j A + pc_{j+1} A \cdot (B + \overbrace{cc_j + cc_{j+1}}^{\Delta}) - pc_{j+1} \cdot pc_j \; A \cdot (B + \Delta)$

While .14 is pretty nice, I'm not sure about .20 ff! Are the Σ's $j+1$th terms the only difference & are they just .25 & .28?

In the original GHT proof $pc_j = pc_j'$; $pc_{j+1} = pc_{j+1}'$: In which case (.28 = .30) — so something's wrong! Also note, that ~~about~~ my comparison of

7/28/82 689

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University 8:00P
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : 688.40 & 588.14 for t. 2 cases was wrong!. For $J < J_0$ (say C_{J_0} & C_{J_0+1} were interchanged)
T. sums of 588.14 up to $J = J_0 - 1$ are t. same, but for $J > J_0+1$ the $\prod_{i=J_0+1}^{M}(1-pc_i)$ products
differ by $\frac{(1-pc_i)(1-pc_{i+1})}{(1-pc_i')(1-pc_{i+1}')}$.

04 Hvr, in spite of t. 688 20ff errors, -1.14 does seem to be correct for both the older GMT & for t. new (— in t. new situation in which Pc's are dependent, we still have to chose a C_i $(i=1/M)$ sequence a priori. -1.14 is t. Func we want to Min. (Tho as written, it is poss!. that there is no case which wins. — This may be a decent Genzn!) So Minzing -1.14 looks like t. desired formal

10 Soln.

11 Hvr, (NB!): in t. C_i sequence (vector), each pc_j is, in general, a function of all C_i's that have come before it: i.e. we know that all of them failed. So in this General case, t. problem seems much more diff't., tho minzing -1.14 still seems like t. formal soln. We can assume each pc_i depends only on C_{i-1}, but this is a very definitely simplified approxn.... → 695.30

17

20 : 687.30 ! Re: t. "introduction", I had considered giving more motivational background — like saying that this system is meant to be toward a realization of "STRONG A.I." Hvr this would be best put into t. Talk rather than t. introduction.

23 (This could be more chance 2 g? sites.) Perhaps General idea of 687.09-.30: The ALP is known to be v.g. w. CB=∞. What is best strategy for CB<∞? Or can we make v.g. (not nec'ly optimum) strategys for CB<∞? → Discuss 586, 89 & t. "factor of 4" question.
For my self: I may want to discuss in some detail, t. "factor of 4" of &
just why it wasn't exactly right (i.e. not all bcors could be expressed as static

30 Modifn. of t. GCPD). But many could: Some hours involve changing Pd during srch! .. are not Csrch. (No CJS estimate is (easy) possl.). But any hour can be expressed as a PST.
Then explain factor of 2 in "50% solution"...
→ Hvr, If ALP is so good w. CB=∞, there are easier ways to test finite CB, than t. TM system I've devised!

So perhaps emphasize This system as an attempt to follow t. path of Human lrng.

35 Or: Just Desb. Sol 86-89-94: Then say this is Contin of that work: Correction of errors, Development of parts that were inadequately defined.

7/30/02 T. G(x)·f(t) problem. 690

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

The present paper will be is a continuation of that work, ~~bringing the system~~ in certain cases, replacing conjecture by proof, and bringing the system much closer to practical realization.

In the form of a simplification and formalization [redacted]

The system is meant to ~~learn~~ operate much as humans do. When the infant system is first presented with a problem,

~~The data structures~~ The information in the system (knowledge) is contained in two data structures: The General Conditional probability distribution (GCPD) and the set of Problem Solving Techniques (PST's)

The GCPD is able to take as input a great variety of strings, numbers or ~~other~~ other data types. Its output is a probability distribution on ~~the~~ strings numbers or other ~~data~~ data types. ~~Web search Engine~~ It may be thought of as a very general web search engine — like Google. ~~While~~ Search Engines usually have as output a list of web pages in order of relevance to query. The GCPD will have as/ output a list of possible responses to its input: each with an associated probability.

The set of PST's at the beginning of system life are inserted by the system trainer.

→ Though the system is meant to be a General Problem Solver (GPS), ~~the~~ the kinds of problems it solves and the techniques it uses are well beyond those described in Newell and Simon's pioneering work (give reference ~~to~~ to GPS). It also differs from ~~the~~ GPS in that few problem solving techniques are built into the system. It ~~acquires them~~ augments them by discovering new methods through experience in prob solving. ~~[illegible]~~ The problems ~~it solves~~ it solves fall into two ~~[illegible]~~ classes: Inversion problems and time limited Optimization problems. ~~While all problems~~ Most ~~[illegible]~~, but not all, problems fall into these categories. We feel that the system is sufficiently general so that it can learn to solve other kinds of problems as well.

One approach: Make intro ≡ abstract, but simply add material. perhaps start out w. (.00)$ or (−1.38) 1.38

~~Do~~ explain 1 of "CF all infeasible pd" "F.N. of '89" and why they are "Not Bad", but wrong. i.e. slightly good

33 on T. ~~or~~ ~~time~~ varying ~~O~~ O₂ problem Gov = G(x)·f(t) : One has (usually) a finite set of PST's to ~~[illegible]~~ max G(x). GGPD, gives us a pd on G values for the problem as a funct of CB=t, for each PST. We could just select the PST w. best "expected value" of G(x) f(t). Using this cor ~~[illegible]~~ (criterion) we choose a PST, and work on it for time To. We then compute as to whether we expect G(x) f(t) to ↑ by further work on this PST. — If not, we stop. We may present our "best x 692.00

7/30/02

ID

Bibliography

691

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

CACM Aug 2002: 2 arts. of int.:

1) Data Mining: pp 28–54 7 papers: Of much interest; Explain various Compression, Pred'n schems.

2) "Swarm Intell." p62: Explains how it works; Various forms:
"Ant Colony" ect.

Both have lots of References.

7/31/02

TP

805 — 692

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:64 0.90 Thus far" or go to another PST if it has an expected ↑ in present $G(x)f(t)$.

(This all assumes that these decreasing calculations to be zero times.)

I suspect that this may also solve the "Chess" problem. In fact, it may be the same as my most recent idea about "Chess".

It is likely that any problem solvable by a human can be solved by this system — If we can get that human's methods into a "PST" in the system. I assume that this is always possible — but the requisite TSQ may be diff. to derive.

Mainly, I feel that after TM has done a lot of problem solving at the upper levels of Trainer "improving the" GCPD, [PST]" in view of past occurrences, it should be able (I.e. basic ones) to discover PSTs that I don't understand, & PSTs that know about. don't yet

---

[JN] about the "Convergence Terms" for Bayes & QA:

First do the KL form of them for them: From this, we easily get ① the err² form on bits.

② Pro KL form on (elements of Bayes) or on "A"s from ① (last) → ③ err² form on Bayes elements or on A's: This last err² form is not as good as bit form of ① but it is easier to see.

② Is actually of interest: $E\left(\sum P_{true} \cdot \ln\left(\frac{\text{estimate}}{\text{true}}\right)\right) <$ constant.

[JN] I was concerned w. deciding which of the non-optimum codes or models to store — expecting future "Backtracking" needs.

Remarks ① would this save a max of X2 in cc?

② A possibly good way to do this: to "cluster" the codes into classes that are "similar." — (There will perhaps be a "distance" function betw. classes.)

For problem, we store the best class usually (maybe add in some others if they leave enough room) for Backtracking, try classes different from the (perhaps "Distant") formerly Best class.

We want a class that has family by PC, yet is most "distant" from the formerly Best class that has been rejected by the new data.

These "Classes" can be obtained by "Clustering" Algms.

8/1/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:692. : The ~~system~~ research ~~we~~ ~~will~~ described is a continuation of earlier work on machine learning (S86, 89, 94). While the present system is ~~...~~ in general ~~quite~~ similar to the earlier ones, we have worked out many ~~...~~ critical aspects ~~...~~ of implementation — resulting in a simpler, more general device.

We will describe in some ~~much~~ detail, aspects of operation that were not ~~altogether~~ clear in the earlier papers.

The general operation of the system hasn't changed much, but our understanding of its operation has. What we ~~will present~~ have is a ~~simpler, more general system that~~ clearer presentation of a system that is at once simpler and more general — with a much better understanding of the ~~...~~ extent to which the system is optimal.

~~One thing that has changed much, vacuity ... , is ~~our~~ understanding of the extent that system is optimum.~~

Q: How is present system different from S86, 89, 94? Several ~~...~~ aspects of the earlier work was not clearly ~~...~~ defined, ~~...~~ and much of the recent work has been ~~it has been possible to work out the details of operation in a clearer, simpler way~~ ↳ been more toward more exact definition of the system, and an understanding of its limitations

Much of the recent work has been in

Maybe not bother w. this: few people will be even mildly acquainted w S86-94.

2-2 SO: I have indicated 13 ¶ nos. in x. abstract (L-AB)

Modifns. for introduction: ¶2 : Give refs (S86, 89, 94 √?). remove last line, insert it after ¶4! (¶4.5)

~~Suggest those limitations.~~

Than say: "The generality of the system makes it likely that it can ~~be adapted ... beyond these limitations~~ we can ~~modify it to~~ bring it beyond these limitations"

¶3 INV: Give formal Model.

¶4 OZ: Give formal Model. [In § — we will show an important kind of induction problem can be transformed into ~~a~~ a time limited optimization problem. ~~Almost~~ Most ~~Many~~ of the computational resources of the present system are devoted to solving induction problems of this kind.

¶ 5.5 All of the "knowledge" in the ~~...~~ system is contained in two data structures: ~~...~~ A set of Problem solving Techniques (PST's) and a General Conditional Probability distribution (GCPD). A PST is a subroutine which, when given a problem description, will attempt to solve that problem. The GCPD ~~...~~ of a mature system, may be thought of as a very large very general probabilistic information retrieval system, ~~web search engines,~~ is something like the GCPD — ~~except~~ some Google, ~~one of best known~~ but while Google presents results (web pages) in order of probability, it doesn't give ~~...~~ probability values.

8/1/02 ID

694

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

8p #P9 say Lsrch will be discussed in section — .
L search uses a probability distribution to guide search for a solution to a problem.
Initially, this probability distribution will be the GCPD. However, simple Lsrch ...
:
As the system ~~matures~~ matures, it modifys Lsearch and eventually replaces it with better PST's. ~~section~~ This is discussed in Section — (on "How Good is the System") ~~discuss this~~

P10 add last line : perhaps in abstract as well as introdn.

10 P11 follow w. discn. of Shrunken Lrng (small cjs). — Discuss large CJS as system can afford. Gives better Genzn. capabilities.

.12 P12 section ... discusses the differences ~~in~~ in the 2 approaches

(in the 2 approaches?)

Successive Problems
Lrng , Successive Rounds
S " trials.

Sections
1) Lsrch for ~~...~~ INV i for OZ. How they differ. perhaps ~~...~~ By successive Lsrch; Lrng during Lsrch.
2) T. QA problem; formal soln. show how its an OZ problem
Give convergence thm: for both big "A's.
3) Perhaps section on PST's : subsection of Lsrch.
b 4) Maybe first show how IND's are OZ's. (?2)
clear

SN In General OPtzn problem, we want to Maximize some function of CC & G. — we need a linear ordering of CC, G pairs. Using first CB is one way of ordering In Anytime problems, I'm not sure as to what t. criterion is! Another possy: Minz time to achieve a certain G-level.

21 SN Is there a ~~clear~~ Gorc for t. General update of (GCPD, Σ PST$_i$) — ? whats Gorc for a PST? ← 696.13

AH! This seems to be a BUG: Only GCPD is directly updated. This part of t. system doesn't care if PST's are noting — all it is interested in is predicting how good they will be w. a particular problem. → See 695.00-07 for likely soln

(a) is it an OZ problem, it is an induction problem.

1) QA: how they are OZ probs Give conv. thms, etc.
2) PST's: (a) Lsrch & variations (b) other methods.
Maintains GCPD Much
3) Updating.

((What would motivate t. updating process to devise v.g. PST's? — Also, what is t. Gorc of a PST?

30 we ~~don't~~ really need ~~...~~ comparison of PST's:
T. idea is that we want to keep a new PST if it is better (for at least one problem) than any other PST (Not exactly, we also have see how well GCPD assigns problems to it).
(So a new PST may not be "better" than any other, but it may be easier to tell when that PST is appropriate for a problem → When new PST's are first introduced/invented, they are usually only known to be applicable to only a few probs — later we expand their utilities. We may ask "why" are they good? & — "when" are they good?
So maybe we start out w. a particular problem, & invent PST's for it.

Normally, I have no over-all Gorc for t. system. For INV probs, I can always tell if a PST is any good on a particular problem.

→ 695.00

8/13/02 Traitoring, Moling, Informers, Stoolies, Defection, Betrayal, Molering [BLACKMAIL, BRIBERY, BURGLERY] Betrayal, Buggary

ID 695

JOINT·CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street·Cambridge, Massachusetts 02138 (617) 495-7908

.00:694.40: Consider a stoch grammar that has for AS's all PST's i all problems that, & all possl. G's & cc's. From this P.D., I could ask which PST was most likely to give the best G for a particular problem for the particular CB (Part of prob. defn). (.03R) Note: I want a joint PD for all PST's, so I can Learn from each trial.

.03

.04 (There is some elite Modifn of the forgo. for INV probs). [So 694.2113 not an optzn problem — It is an induction problem.]

So if I update before the problem, I can do .00–.04 for the problem.

.00–.04 should (in theory) Generate a PST that is Best for the particular even

.07 OZ (or INV) problem. Using (.03R) for "learning during search" is/more diff'lt, but theoretically possl.

.0 Is .00–.07 really a formal Soln. to the problem Difty of 694.30? Looks like it!

Hvr, the practical solns/approxns are normally quite distant from that "formal soln".

The mechanics of approxn. are unclear; What I may do is write various approx ways to solve 694.30 & use .00–.07 to refine/optimize these Methods.

.15 [SN] In early life, the pc of a PST for induction (or even OZ), could simply be obtained by looking at the final PST for various problems — the "one" that "finally solved" the problem. For corpus, we have {Problem_i (w.o. CB given!) PST_i} pairs that were successful. From this we can get P(PST_i | Prob_new) for all PST_i's — even newly invented ones.

.20 Of course, leaving out the CB clearly makes it all very inexact. Also in the soln. to OZ probs, There may have been many trials, initiated by PST's other than the final one: The Final one may have done little else but make the final few trials based on info collected by earlier ("unsuccessful" PST's).

So in general, the nature of the G'GPD & its method of being Updated, will vary during the TM's Life.

.27 (NB) hvr: in forgo, the fact that a given PST was "Best" in a given problem is much biased by how much wt. that PST got from GCPD for that problem! If it got much wt., then much cc was spent on it. } see 761.00–.16 for near "final soln" for that problem.

.29

.30: 688.17 [SN] Re: [G.H.Thrm!] (GHT — Gambling House Thrm) In the original Thrm: One & only 1 bet wins: The strategy is to minimize expected loss before winning. If we Genlz so that >1 bet can win & its possl that none will win, I'm not so sure of the Gorc! In case of >1 winner possl & possy of no win!

Consider all orderings of all trials: for each ordering, one has an expected cc before winning. I think in all orderings, the probty of none even win (total loss) = product of probys that all will lose. (win) is the same. — "–1" since Σ=1. Yes

I guess that w. n cands, there are 2ⁿ probys. to describe system. Are they indep params? I think so

.37 — The proby that all of the cands will fail, is one of the in the Joint D.F. It is indep of order of trials.

So: .07 implies that ∃ an ordering w. min expected cc to win — based on trying them all.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.01 : Look at "CPM" stuff (both of them) to see how parts fit in.
~~Also, may be use of some of their exposition stuff~~

.03 SN on I 414.18: I was going to use S78T3 cor to prove this:
Now I'm not so certain about S78T3 cor! It says nothing about the aprp of the approximating p.D.
It could be purely ad-hock
On t. other hand, if $\prod_{i=1}^{n} p'(x_i|x_{1\ldots i-1}) > e^{-k} \prod_{i=1}^{n} p(x_i|x_{1 \ldots i-1})$ ← approximating function.
then E (error) $< \frac{k}{2}$. This is true, indep of p''s derivation.

10 I guess t. badness of an "A.H." P.D. is that it is unlikely to retain this small k in t. future.
If I put in t. aprp factor, t. [redacted] pd, p' amounts to a coding on t. reference UMC, and we can compare it directly to t. code assoc. w. t. "generating" pd. [redacted], P(·).
If we want to compare 2 cand. pd's w. t. true pd. p(·), it may be nec'y to put in t. aprp constants.
Likely that t. soln. is what I talked about in "T. Discovery of ALP"; That S78T3 gave t. v.g. soln., perhaps "Best" that one could get, using stated aprpd. — It was "sum of all codes";
That any finite cc approxn. was < that value but t. larger it was, t. closer it was to t. correct value.
While this is true for t. closeness to t. "rite" Semi measure, we are interested in t. ratio of two Semi-measures.
(This is familiar territory "... I've been there before ... what was conclusion (if any)?),
Hvr, taking logs: If A' is closer to A than A'' is; } then A'−B' need not be closer to A−B than A''−B'' is.
If B' is " " B " B'' is

20 On t. other hand: That statement in "The Discovery of ALP" did not invoke S78T3. | It was for semi measure only
≡ T bigger t. better
If we do use t. "convergence thm" we get a "Bound on error"

Actually, t. conditions for t. corollary are more stringent than I thot:
.23 $p'(x_i(n)) > e^{-k(n)} p(x_i(n))$ $i = 1/2^n$ $x_i(n)$ is t. i^{th} string of length n.
~~to all string~~ so .23 is true for all strings of length n. Superficially, it would seem that we don't need to be that general! — That .23 only has to be true for a small fraction of t. strings of length n.
On second thot: t. corollary is about Expected Value so it prob'ly has to cover all " " " "

Say $\mu(x)$ is t. "true" generating pd. → $P_M(x)$ is t. universal pd. : so $P_M(x) > e^{-k}\mu(x)$ for all x.
w. finite dern. wrt. M
30
31 Say $P^o(x)$ is a pd that one has found. It has dern pc $= \alpha$ & it assigns pc $= \beta$ to a certain corpus.
$P_M(x)$ has dern pc $= 1$, and it assigns β_M to that corpus.
.33 I expect that $\beta \cdot \alpha < \beta_M \leq \mu$(corpus)
$\beta \cdot \alpha \lesssim P_M(\text{corpus}) \lesssim \mu(\text{corpus})$. Actually μ(corpus) can be any thing ≤ 1 & > 0, but for a long corpus, it will have a certain "entropy" — (mean ln p per symbol) (usually).
∃ $p^o(x)$ will always have < all of codes of $P_M(x)$.
Any code for t. corpus can be regarded as (a dern of pd) + dern of corpus in terms of pd (2 part codes)
So any pd and corpus (like p^o (.31)), will be a subset of t. codes of $P_M(\cdot)$ —

8/7/02 Texas Tokyo | GM Toyota 698
NYC Hiroshima
IRAQ N. Korea

20 : ∴ So it will love to assign a lower $\beta\cdot\alpha$ than β_M (−1.33), i.e. $\beta\cdot\alpha \lessapprox \beta_m$.

I If $p'(x)$ has dern $p_c = \alpha_1$ & assigns β_1 to corpus.
& $\alpha_1\cdot\beta_1 > \overline{\alpha\beta}$ (−1.31), then $p'(\cdot)$ would seem to be better than $p^o(\cdot)$ in predicting t. corpus.

The big idea here is that $\alpha\cdot\beta$ is t. amt. of wt. given by the "para" $p^o(\cdot)$ to t. corpus.
For this corpus, at least, larger $\alpha\cdot\beta$ means that t. wt. is closer to that of $P_M(\cdot)$, t. universal d.f.

[But say $\alpha_1\cdot\beta_1 < \alpha\cdot\beta$ but $\beta_1 > \beta$. So, for t. corpus alone $p'(\cdot)$ looks better than $p^o(\cdot)$.

10 → AH! For pred'ns we should use wt. $\alpha\cdot\beta$ for $p^o(\cdot)$ & wt. $\alpha_1\cdot\beta_1$ for $p'(\cdot)$. p^o & p' combine linearly w. wts α & α_0 resp. — for corpus of length ϕ. As we move thru t. corpus, t. wts change at t. end of corpus and they are like (.10)

However, [redacted] no matter how good $\alpha\beta$ or β is, p^o could assign a zero to something that $M(\ )$ would assign >0 to, in which case $\frac{p^o}{\mu}$ would be $\phi \le k$ ($m\,e^{-k}$) would be ∞ ! so expected $\sum$ ratio 2 would be $\le \infty$ (∅).
This trouble occurs whenever p^o is not universal (tho that's not exactly it: If p^o were universal, then it assigns ϕ to no possl. seq's hvr. p^o could still assign >0 to every possl. seq. & yet not be universal. (e.g. say p^o is t. uniform d.f. — or any Bernoulli seq))

20 So: any d.f. like p^o plus t. uniform d.f. would never assign ϕ to any corpus.
T. uniform DF would get very little wt. if p^o were even marginally useful! (uniform D.F. ≡ "Random")

In general, getting data that would lead one to believe that $p^o(\ ) \gtrsim e^{-k}\mu(\ )$ for all x. seems unlikely! Showing that it is true for just known corpus is also unlikely.

So t. main idea then, is that P_M is of ≈ known goodness (e.g. Thr 3 $78)
And the larger $\alpha\cdot\beta$ is, t. closer p^o is to "P_M" in t. sense of more codes.
So, since P_M is optimally close to μ; p^o is perhaps close to μ.
I guess analogous arguments hold for 414.18

Read notes in $78 folder; It may have good ideas on forgoing! (It may, indeed, but those notes are not "easy to read"!)

30 → T. problem is, hvr if $\alpha\cdot\beta > \alpha_1\cdot\beta_1$, then is p^o_s more likely to be close to μ than p'_s is?

Perhaps P_M is as close as we could ever get to μ (using M's a priori info).

If we could find out where the code equivalents of p^o & p' overlap, we could simply add p^o + p' and cut out t. overlap! — tho p^o & p' are often normed, they still both consist of a set of codes. Unfortly, this expressn of p^o & p' as sets of codes

.37 can probably be done in ≥ 1 way. — Tho if we use Arith coding, there will be a unique single code for p' & for p^o. — In which case it would seem that we could do a simple wtd mean of the 2 forms. [Hvr., see +1.09 !]

8/10/02

1:55A
β

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : Well, if we use a wtd ~~sum~~ mean of P^0 & P^1 for predn, & wts are $\alpha\beta$; $\alpha_1\beta_1$, resp.; then clearly one wants Pams of hyest possl. $\alpha_i\beta_i$.
And the goal is to ~~get~~ find a set of Pams ∋ $\sum_i \alpha_i\beta_i$ is ~~max~~ as large as possl: best approx. to P_M. In general, we don't ask whether P^1 is "better" than P^0: we just use wtd. mean w. wts $\alpha \cdot \beta$; $\alpha_1\beta_1$.

This sort of "solves" the problem of justifying by looking for Pams of large $\alpha_i\beta_i$ — they contribute most to the predn. (most wt.).

T. forgg. Idea is not new: I just forgot it!

.09 There is some uncertainty about this $\sum\alpha_i\beta_i$, hvr: I'm not sure that there can't be "overlap" in the codes for P^0 & P^1. The arith coding avg of −1.37 is assuring, but I don't have complete confidence in it!

.10 If "overlap" betw. Pems ~~...~~ is impt, then finding a single Pam of Max $\alpha \cdot \beta$ becomes the Main Goal. [See .32] ← This depends on "No overlap" of codes betw. P^0 & P^1.

Ⓐ Another Q that I probly worked out long ago: for the next predn, I can use P^0 & P^1 w. ~~...~~ wts $\alpha \cdot \beta$; $\alpha_1 \cdot \beta_1$ resp. or Ⓑ I can devise a new Pam that predicts every bit of the corpus, using a wtd mean of P^0 & P^1, but the wts change with each bit, depending on success thus far of each individual Pem.
Would Ⓐ & Ⓑ end up identical? — its a "simple" algebraic Q.

.20 On second thot A & B seem the same: A just considers the predn of the $n+1^{th}$ bit (for a n bit corpus), But Ⓑ predicts for each of the ~~...~~ bits.

Perhaps write up section of Report" that this is about NOW! So I don't forget it.!
Give ~~...~~ discussn in some depth! This is mainly for ME: If necy, I can trim it from the "report".

.26 : 696.20 : Outline of Section on the QA problem: (T. problem ~~has already been defined in introduction~~ No! just mentioned as a kind of OZ problem).

In introduction, perhaps mention "Ganz OZ problem" of (700.00 .16) — or ~~...~~ leave it out of Report but have F.N. in my version of the report. F.N. can refer to notes, or have ~~...~~ contents of Notes.

.30

.31 Do, derb Ganz soln w. $CB = \emptyset$: This does justify the approach I'm using ~~...~~

.32 : on overlap of P^0, P^1: say they overlap: so $P^0 = \alpha_0 + C$; $P^1 = \alpha_1 + C$
C is common overlap. α_0, C, & α_1 are 3 nonoverlapping sets of codes,
T. ideal way to predict would be $(\alpha_0 \cdot \text{predn of } \alpha_0 + \alpha_1 \cdot \text{predn of } \alpha_1 + C \cdot \text{predn of } C) / (\alpha_0 + \alpha_1 + C)$.
If we just mix P^0 & P^1 without considering overlap, we may get predns of P_0, P_1 & C w. rel wts P_0, P_1 & 2C.

I'm assuming that if my ~~Pam~~ Pam has larger $\alpha \cdot \beta$ & is larger than $\alpha_0 \cdot \beta_0$ then it is "closer" to M, the ~~...~~ "generator". of P^1 Larger $\alpha \cdot \beta$ means its closer to P_M: whether it has to be closer to M, is unclear. → 701.00 SPEC

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 699.40! [SN] On OZ problems: In general, we have to maxz G & minz cc.
There are many ways to resolve this (often) conflict. All "soln." are partially ordered at least.
① Per ways to order: ① fix cc & CB: Get Max G: This is usual form of OZ problem
② fix G: minz cc: This is a INV problem form. ! The idea is to "Satisfy a criterion" in min cc.
(③ Any conversion of partial to linear order: Say some (Monotone) function of G & cc to be
.06 Max'd. → see (205.00)! This becomes t. time-varying [G(x,T)] problem!
④ "Anytime problems" I don't know just how they define t. soln. One way!
Do regular OZ w. fixd CB but periodicly CB ← k·CB   k can be 2 (or close to 2).
10 Using large k makes it possl. to change character of soln.
Another way is CB ← CB+Δ whenever we reach CB. The
degree of Greadyness depends on size of Δ or k   2 200k
1.5 100k.

Whatever t. form of OZ problem, we can use t. 2-step Update
of GCPD to Solve it: T. ~~second ste~~ first step is always
.16 t. same, but t. second step depends on t. exact form of OZ problem definition (705.00)

Expo

[SN] After introdn: reader should have clear idea of t. "parts" of t. system
① GCPD ② [PST] ③ Just what t. QA problem is ④ How Updating GCPD, [PST] is done.
How a QA problem is defined: Apnd, corpus.
20 ⑤ How TSQ's are written.

[SN] On sequencal predn. as a QA problem. Normally a code for an "A"
is a S.d. code (discrete universal pd): If prefix codes are allowed — so we want codes for
any string of which t. known A is a prefix, then we can do Sequencial prediction.
In general, t. "Q" of a problem has to tell which kind of problem it is — whether t.
discrete or Continuous p.d. is to be used. This makes the s.d. code & prefix codes give perhaps t. same P.D. for predictions. T. S.D. code gives Cover's "Extension Complexity" for Sequencial Predn.
Say we want to extrapolate t. seq. "A₀"; we insert Q₀, A₀ as part of t. corpus,
30 & ask the algm O^n that for any Q₀, Gives a P.d. on A₀ & its extensions; finite or infinite.

Also explain how to use ‖ time series as Qᵢ, Aᵢ's (did I mean 2 or more T.S.) in different streams w. possl. interaction? — Proby)
Also explain how to use Encyclopedia as infor
" " about how to use a large corpus (say SM or Encyc) as data — that cost of
code include cost of refering to data in t. corpus.
This expands report a lot! Perhaps leave it out or put "footnotes" in my version, that refer
to my notes.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

:00: (spec 699.40) : It may be possl. to show closeness to P_M implies (in this case) closeness to μ.

T. fact that $\alpha \cdot \beta$ is large makes it likely to be close to as t. generator.

Consider: $\sum_{\mu} \mu(\text{corpus})(\ln \mu(\text{corpus}) - \ln P_M(\text{corpus}))$, ≥ 0. Troubles: these do not need to consider t. aprip of P_M or of P^0, —

Consid[redacted]: $\sum_{\mu} \mu(\text{corpus})(\ln \mu(\text{corpus}) - \ln P^0(\text{corpus}))$ Perhaps they have to because we are only

considering P^0 & P' as approxns of t. component codes of P_M, — so t. aprips, α_0, α_1 have to be included.

.09 Given a corpus: C [redacted] M is apsl of $\mu(C) \cdot \text{aprip}(\mu)$: finding a μ ∋ this is Max, is what we were trying to do. So in this sense Maximizing it gives us our best approxn to μ.

.10 Given CB=∞, we can find t. Max of .09 (or .11): it is P_M.

.11 So in finding a funct. f, such that [redacted] f(corpus) · aprip(f) is max, we can be trying to approximate μ rather than P_M.

We know that t. best f that [redacted] we could find w. CB=∞ is P_M & it is v.g. for predn. (small error).

But, as I noted before, no matter how good our f (of .11) is we can't be sure that it is always within a constant factor of μ.

Picking a single f ∋ .11 is max among all of all kinds, once has picked t. most likely funct. (containing μ ...)

So it's a Max likelyhood choice w. careful consideration of a aprip of f.

.19 This approxn. of f is v.g. for CB=∞.

---

20 Introdn: Explains INV probs, OZ probs: (Also 700.00-.16 on OZ & INV & ...)

How fixed CB is not t. only way to define OZ probs.

23 Section in IND: Give normal QA problem. → it's only nec'y that Q & A be finite strings that can be uniquely interpreted as ... (does not define structures). Explain that QA's can be strings of nos — Invariants, vects, or other data structures. Give some good examples (Also Ency case). Also explain how seq induction & Time series can be covered by this formalism also (ency case).

Give 414.18 & explain how it is a OZ prob. & we can have any relation between G & Cs.

(P)

Explain that any Ind prob. is defined by Corpus, & aprip d. & in practical cases, by t. PST' [redacted] to be used & t. CB of (G, relation).

Discuss CB=∞ for 414.18 & some of its properties. Explain about continuation of Q sequence; (KL & error properties. Why Σ error of A's is ... (interesting)) That t. Σ err or KL dist is bounded, — what this exactly means — if Q's are like earlier Q's ... v.s. if they are quite different from each part of corpus.

8/14/02
±D

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

: Expo Computing Resources (≡cc):

When I speak of "Time to solve a problem using a particular PSI", I usually mean a generalization of time: i.e. Computation cost. In addition to time, this can include Memory size, cost of CPU(s), Weight, Electrical energy (watts times time) Cost of cooling (watts times time), Real estate (both on Chip and area of workspace in human Community) .... In general, the relevant combination of costs will be different in each computing environment. Whatever does cost, we can summarize as a scalar "money" "Time" — one of the costs we want to minimize.

Section on induction: In this particular paper, we will mainly discuss a certain limited form of induction, since the updating of the problem system can be put into this form. More general forms of induction can be implemented with minor modifications of this limited form.

We are presented with a sequence of Q_i, A_i pairs ... $i=1\ldots n$ (the "corpus") The Q's and A's can be strings and/or numbers. More generally, they can be any data structure that is uniquely encodable as a finite string of bits.

We are then given a Q_{n+1} and the problem is to obtain a probability distribution over all possible strings, A_{n+1}. The ordering of the

The Q's and A's may correspond to questions and answers in some formal or natural language; They may be the inputs and corresponding outputs of a unknown (or partially known) mechanism — They can be reals, integers, or strings. Many induction problems can be put into this form. With very little modification of our formalism, we can deal with the extrapolation of an unordered sequence of strings (language induction) or reals, (curve fitting). We can deal with sequential prediction — in which our corpus is an ordered list of strings and or numbers,

We can extrapolate using data from a large digital text such as an encyclopedia or dictionary.

A formal solution to the problem is to find (in the available time) → 703.30

---

30 a conditional probability operator O^n such that

31 $$\alpha_n \triangleq P(O^n)\cdot\prod_{i=1}^{n} O^n(A_i|Q_i)$$

is as large as possible.

$O^n(A|Q)$ is the probability of "A" in view of "Q".

$P(O^n)$ is the a priori probability of the conditional probability distribution, O^n. If η is the number of bits in a short program that implements O^n, then $P(O^n) \cong 2^{-\eta}$. Other, more exact forms can be used.

8/14/92

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

$O^n(A_i^j, Q_i)$

00: : ~~Distribution~~ The distribution O^n may have any of several forms. The form may vary with the nature of Q_i.

Some possible forms of O^n ~~[struck out]~~ (the program), with input Q_i:

1. The output is a sequence of A_i^j pairs in approximate order of probabilities — most likely are first ~~[struck out]~~
~~$O^n(A_i^j, Q_i)$ represents the probability of the j th being A_i^j as a response to Q_i~~.

2.a The ~~output is a~~ output is a Monte Carlo distribution, with the probability ~~of a string the string being the output, being~~ of ~~some~~ output string A, being $O^n(A, Q_i)$

2.b

0 3. The algorithm O^n has 2 inputs; A and Q; its output is $P(A|Q)$.

→ One useful ~~[struck out]~~ way to realize this Monte Carlo device: We have a deterministic algorithm ~~[struck out]~~ with two inputs: One is Q, the other is ~~[struck out]~~ a semi infinite binary string. The output ~~[struck out]~~ has [redacted] probability $O^n(A|Q)$ of being A. → O^n_k A real number or Real continuous. ← Actually, we are considering continuous k numbers. continuous j's in (.34)

18 4. If both Q and A are real, $P(A|Q)$ might be represented by ~~an analytical expre~~ continuous equation. Since we are usually interested in peaks of the A distribution, [redacted] equations of this sort can often be approximated by a multivariate distribution about a peak.

21

SN .18 is in T. direction of Hessians as part of soln — so [redacted] a broad, not so high, peak, can be ~~[struck out]~~ "Bigger" than a high narrow peak. [I may have to modify 702.31

25 So 702.31 → $\alpha_n \triangleq \sum_j P(O_j^n) \prod_{i=1}^{n} O_j^n(A_i|Q_i)$:

The summation over j will be replaced by an integral if the index, j, is a continuous quantity. Often the integrals will be multidimensional. Sometimes re writing 702.30 ff

30 a set of conditional probability operators O_j^n such that

34 $$\alpha_n \triangleq \sum_j P(O_j^n) \prod_{i=1}^{n} O_j^n(A_i|Q_i)$$

is as large as possible. We then [redacted] use ← New line $\sum_j P(O_j^n)\, O_j^n(A_{n+1}|Q_{n+1})$ ← New line as an estimate for the (probability distribution of A_n)

$O_j^n(A|Q)$ is the probability of A ~~[struck out]~~ for condition Q, as given by O_j^n

$P(O_j^n)$ is the apriori probability of the conditional probability distribution, O_j^n.

If n is the number of bits in a short program that implements O_j^n, then $P(O_j^n)$ can be approximated by 2^{-n}. Better approximations can be obtained by summing over several short programs for O_j^n.

8/16/02 705

ID

(space 700.16 / 700.06 in particular) SN HAL T. Generalized OZ problem in which t. thing to be Maxd is a funct of G & T, seems to be identical w. t. general ordering function on G & T! We have a partial ordering of G, T pairs viz G' > G a/o T' < T. for a linear ordering we need a funct of G & T — which is t. same as

$G \equiv G(X, T)$.

How to solve this is still unclear but normal fixed CB OZ probs & INV probs are "special cases".

One approach is via $GCPD_1 \to GCPD_2$: In GCPD, we have t. DF's for

10 each [illegible] (PST, G, cc) from t. known ~~functional~~ form that orders G, cc, we can select a PST that for t. given problem is most likely to have Max G(X, T). We can obtain this Monte-Carlo wise since we really don't need an exact soln.

it necy.

SO This Seems to Solve t. classic time varying OZ problem

.12 Max G(X) · F(t) in t. More General form G(X, T).

20 : 704.40 we will then describe improvements in t. system that will overcome these limitations →

21 these improvements may be introduced into the system in either or both of two ways: They can be programmed into the system by the trainer, and/or be

.23 acquired by the system via its training sequence. While it is much better for the system to acquire knowledge via training, some of the needed improvements

.25 may require excessive computation time (take too long) for the system to learn ~~via this way~~ and they have to be, regretfully,

~~the normal training sequence~~ in which case ~~they will~~

.27 programmed in by the trainer.

Maybe say this w.o. saying .21 – .23. — Much shorter. → while almost

30 all of these improvements can be acquired by t. system thru its normal training sequence, some of them ~~t. needed improvements~~ → (.25 – .27)

Do describe t. different kinds of OZ problems: Until recently ~~varieties~~ ~~[illegible]~~ the

this time varying optimization was thot to be unsolvable by Lsrch. We have, however, discovered that our/complex updating technique can be readily adapted to this

[illegible] class of problem

8-16-02
20

[redacted]
706

00: So try Paz: After Introductn:
1) Soln to Induction problem. } "preliminarys"
2) How Lsrch works
3) The evolution of the system, 704.35 → 705.30 gives mitral Expo.

996.13 PX gives a timi of out line of the paper.

$\frac{\frac{1}{2}K}{1-\frac{1}{K}}$ = min
$\frac{K^2}{K-1}$
$\frac{2K}{K^2} - \frac{1}{1-K} = 0$
$\frac{2}{K} = \frac{1}{1-K}$
$K = 2-2K$
$3K = 2$
$K = \frac{2}{3}$
$\frac{\frac{1}{2}K}{1-\frac{1}{K}} = \frac{\frac{1}{3}}{-\frac{1}{2}} = -\frac{2}{3}$!

A "not bad" approach. First do "prelimys" IND & Lsrch.

Then we give "bare bones" (prelimy) system: We do INV probs & modify & P.D. in accord w. "what has worked" in past (indep of cc's of "winning Cands").

Can this be done for INV/OZ probs at all? I guess it can; we would simply modify the P.d. so it would give hy pc's to ~~successful~~ PST's [redacted] that have in the past, been Successful (indep of cc's involved).

For simple Lsrch, for both INV & OZ problems There [redacted] is only one d.f. on PST's for all problems (possibly 1 for INV, 1 for OZ), so its easy to "pool" data.

The GCPD is not being used to select PST's (There [redacted] is only 1 PST being used — i.e. Lsrch)

19

→ 718.22

5X [redacted] I've forgotten details of the "Conv. thm for Bags". I must write this up! i.e. I don't remember details of Thm. (let alone details of proof!) use ALP! Sort of 2 part codes: sum them in ||!

98TM 106–108; 110, 111, 115
best motivation is 111.01–.17 is good.
108.01–.02
110.01–.40
111.01–.37
the "proof": 110.08–.34; 111.03–.17

$\alpha_n \equiv$ ~~[crossed out]~~ $\sum_{j} \alpha_0^j \prod_{i=1}^{n} O^j(A_i)$

A_i is the i^{th} object in bag. probably use $(O^j(A_i))^{m_i}$ if A_i occurs m_i times.

O^j is j^{th} pcm
α_0^j is apriori of j^{th} pcm.

More recent stuff
ID 378.00 ff!
380.23 –.40
381.00 –.08

If $\mu(A)$ is the "true" pcm generating the $\{A\}$

$\alpha_n > \alpha_0^{\mu} \prod_{i=1}^{n} \mu(A_i)$ | α_0^{μ} is apriori of $\mu(\cdot)$.

α_n is indep of order of $\{A\}$

(Pm≡) α_n assigns to the corpus, a pc $> \alpha_0^{\mu}$ times that assigned by μ to the corpus. — INDEP of Corpus.

μ assigns pc's to each of A_i: Does Pm? It can assign a "partial" pc to A_n — "Given the other n−1 A_i's"

8/19/02 (or relative entropy) Kullback–Leibler distance form KL form .30 to 708.10 — 707

708.18 is a kind of BUG

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00:706.40 : I was concerned that the "easy" approach to proof, using the T3 corollary, assumed the probys for P' (based on P_M) were sequentially indep. It seems that they are not at all indep.

Fact. "Bag"/convergence Theorem.

The pc predn. of the k^{th} element in the sequence is:

.06
$$\sum_j \alpha_0^j \prod_{i=1}^{k} O^j(A_i) \Big/ \underbrace{\sum_j \alpha_0^j \prod_{i=1}^{k-1} O^j(A_i)}_{\text{normzn factor}}$$

If this is so, then there is clear cancellation of all (or all but one?) normzn factor. Maybe not! The normzn constant here is wrong. Not exactly! It

0 is correct for [redacted] P_M = semimeasure. The $\frac{P'}{P_M}$ > "α" inequality is true for semimeasure, but a fortiori for measure, which is always ≥ semimeasure. pcs of.

I guess, in the similar way we have an eq. like .06 for the individual bits of the $\{A_i\}$ sequence. This seq. of pc's of bits would enable us to apply T3 corr. directly, in one step, to get desired result. So $\frac{P'}{P}$ > "α" is true for each of the bits of the corpus — so **T3 corr. holds** (is applicable).

Re: the Applicn. of (5.78 T3 corr.:) Does this mean that I must be able to associate

0 a pc w. **any** finite string of bits? (But some of these strings may have pc = 0 is o.k.)

Actually I may need only a pc, (P') assoc. w. a bit string w. "n" tokens before the A_i's.

2nd a M that gives pc's for that same kind of string.

But M is concerned w. "Expected values" i.e. we sum over all poss. (n) sequences of A_i's. These will in general, have different bit lengths.

A perhaps simpler way to prove the K-L part of 5.78.73:

$KL(\vec{P_1}, \vec{P_2}) \equiv \sum p_1^i \ln(\frac{p_1^i}{p_2^i})$. $\vec{P_1} \times \vec{P_2}$ is the cartesian product of $\vec{P_1}$ & $\vec{P_2}$: if P_1 has α elements, P_2 " β ". Then $\vec{P_1} \times \vec{P_2}$ has $\alpha\beta$ elements w. pc's of each being product of component pc's.

0 → problem #1.00-709

Theorem [redacted]: $KL(\vec{P_1}, \vec{P_2}) + KL(\vec{P_3}, \vec{P_4})$ (n vector, m vector)

.35
$$= KL(\vec{P_1} \times \vec{P_3}, \vec{P_2} \times \vec{P_4}) \quad : \text{line .30 has } n+m \text{ terms}$$

line .35 has n×m terms. Any way that almost all of the m×m terms combine? — Yes! → 708.00

.33 .37
$$\sum_{i=1}^{n} \sum_{j=1}^{m} p_1^i p_3^j (\ln p_1^i + \ln p_3^j - \ln p_2^i - \ln p_4^j) \quad 4\,mn \text{ terms.}$$

.30 .30
$$\sum_{i=1}^{n} p_1^i (\ln p_1^i - \ln p_2^i) + \sum_{j=1}^{m} p_3^j (\ln p_3^j - \ln p_4^j) \quad 2n + 2m \text{ terms}$$

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

! Recall, $\sum_{k=1}^{K} p_r^k = 1$ for all r. So, looking at the various terms of -1.35!

$$\sum_i \sum_j p_1^{ij} p_2^{ij} \ln p_1^{ij} = \sum_i p_1^{ii} \ln p_1^{ii}$$

more generally $\sum_i \sum_j p_1^{ij} p_2^{ij} (\ln p_1^{ij} - \ln p_2^{ij}) = \sum_i p_1^{ii} (\ln p_1^{ii} - p_2^{ii})$

$$\sum_i \sum_j p_1^{ij} p_3^{ij} (\ln p_3^{ij} - \ln p_4^{ij}) = \sum p_3^{ij} (\ln p_3^{ij} - \ln p_4^{ij})$$

So $(-1.36) = (-1.35)$ & the Thm is proved! $\sum p_i = 1$ necessary, so we would have to have the "U" symbol as well.

So the KL's additive property & Thm seems to prove S78T3 conv. (Real K-L part)

I looked at Cover & Thomas: Several theorems on KL distance, but none like this one.

Hmm, may give 2 refs Csiszar [78] 1967 (Hungarian Journal) and also Amari [10] ref Book Springer Verlag 1985

The application of the Thm depends on at least 2 things

1) $\sum p_i = 1$; so all probs are normed (possibly by fixing a pt to "U").

2) y. 2 d.p.'s to be combined must be indep so prob of combination = product of probs.

If the p's are not indep (1.35) becomes $\sum_{i=1}^{n} \sum_{j=1}^{m} p_{1,3}^{ij} \left(\ln p_{1,2}^{ij} - \ln p_{2,4}^{ij} \right)$

In the case of S78T3; the p's are not indep. So it would seem that the Thm doesn't Apply!

On the other hand, if this proof is correct, it could seem very likely that the theorem would apply!

The Theorem would apply to successive A_i's, but not to bits within an A_i, as has to S78T3

On the other hand maybe $\sum_{n=1}^{n} p_{1,3}^{ij} = p_3^{ij}$

In (1.36): $KL(p_3, p_4)$ is not really meaningful if p_1 & p_3 are not indep & (p_3 & p_4) are not indep.

There may be a way to derive this so that the Thm is true!

At present its not clear in my mind as to the meaning of KL distance if the second pair of p's is dependent on the first — But the (KL dist / KLD) is derived for the entire 2 step system. (55)

= KL Distance

Another approach way: The method of getting (KLD) for individual A_i's (which are indep)

Or more exactly, realizing each A_i as an indep binary seq. to which S78T3 cor. applies,

Perhaps the Thm applies to successive A_i's, since they are "indep". It is indep, but the wts of the PM components are not.

These are the same.

For the first part of .29: Either the first step, the system is in a [illegible] p.d. over a set of states.

KLD: measure dist betw. 2 p.d.'s over same set of objects

Peter Gacs works this out in his proof of the conv. Thm: Levit thesis 1993 pp 285–288; Levit Second edition 1997; pp 328–331

The theorems he proves about KLD are a simpler, more general way of doing this

8/21/02

F.D

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

TO: : doing my proof of [S78T3], but I suspect it's pretty much the same. (There may be a serious bug in the [redacted] [redacted] IEEE printed version; the expression for A_{n+1} (following Eq(22): should have $\sum^{2^{n+1}}$, $\sum^{2^{n+1}}$ I think. Also, I don't understand the relevance etc. last comment on p 426 of ↑!

Anyway: Re: Gacs' proof; p 329 Claim 5.2.1:

$$\sum_{xy} \mu(x)\mu_2(y|x)\left(\ln\frac{\mu(x)}{\rho(x)} + \ln\frac{\mu_2(y|x)}{\rho_2(y|x)}\right)$$: This is O.K. order of summation is irrelevant for finite sums

The critical definitions are $\mu_2(y|x) \equiv \mu(x,y)/\mu(x)$

and $D_2(\mu\|\rho) \equiv \sum_x \mu_1(x)\, D(\mu_2(y|x)\|\rho_2(y|x))$ ← vary y.

$$\equiv \sum_y \mu_2 [redacted] (y|x) \ln \frac{\mu_2(y|x)}{\rho_2(y|x)}.$$

$\mu_n(\overset{1}{y}|\overset{n-1}{x}) \equiv \mu(\overset{n-1}{x},\overset{1}{y})/\mu(\overset{n-1}{x})$ with $\ell(x)=n-1$, $\ell(y)=1$. $\mu(x,y)=\mu(\overset{n}{x})$ [redacted] $\ell(\overset{n}{x})=n$.

$D_n(\mu\|\rho) \equiv \sum_{\overset{n-1}{x}} \mu(\overset{n-1}{x})\; D(\mu_2(\overset{1}{y}|\overset{n-1}{x})\|\rho_2(\overset{1}{y}|\overset{n-1}{x}))$

Most impt idea.

$$= \sum_y \frac{\mu(\overset{n}{x})}{\mu(\overset{n-1}{x})} \ln\left(\frac{\mu(\overset{n}{x})/\mu(\overset{n-1}{x})}{\rho(\overset{n}{x})/\rho(\overset{n-1}{x})}\right) = \sum_y \frac{\mu(\overset{n-1}{x},y)}{\mu(\overset{n-1}{x})} \ln\left[\frac{\mu(\overset{n-1}{x},y)/\mu(\overset{n-1}{x})}{\rho(\overset{n-1}{x},y)/\rho(\overset{n-1}{x})}\right]$$ indep of y,

$$= \sum_{\overset{n-1}{x}} \mu(\overset{n-1}{x}) \cdots =$$

$\equiv D(\mu(\overset{n}{x})\|\rho(\overset{n}{x}))$

[redacted] $D^2(\mu(\overset{n+1}{x})\|\rho(\overset{n+1}{x})) = D_1(\mu(\overset{n}{x})\|\rho(\overset{n}{x})) + D_{n+1}(\mu\|\rho)$ [redacted]

So list the conditions under which G's proof (of Thm/corollary) are true. | or $m_2(\hat{x})$ can be a (Pty) "Normalized" semicomputable PD. ← As in S78T3

1) any pair of semi measures $m_1, m_2 \ni$ [redacted] $\frac{m_2(\overset{n}{x})}{m_1(\overset{n}{x})} > e^{-f(n)}$.
2) The Bay Thm
3) The QA Thrm. } in both cases we want the general case in which $f(\ )$ is a func of n (or may be constant).

Consider [redacted] n Q_i's: Consider all poss. sequences of n A_i's, for those Q_i's: (T1)

μ will assign a pc to those A's | For every such sequence, the total pc assigned by
P_M " " " " " " " " ... | $\frac{P_M}{\mu}$ will be $> e^{-k}$ (for μ having finite descr, of min cost, k)

NB: P_M is a fixed P.d., i.e. i.e. it assigns to a A_i is indep of ordering of the Q_i's. NO!

[redacted] So, for all seqs of n A_i's; total 'Expected value of total $\Sigma(P_M-\mu)^2$ will be $\leq k$. [redacted] This is for each of the bits in the seq. of A's.

Suppose P_M & μ describe seqs of A's having different lengths?] This is illegal: we only compare

8/22/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20: 709:40 : P_M & μ for identical seqns of A_i's

01 For set of all poss seqns of n A's: T. E of $\left(\sum \text{KLD between corresp bits}\right) < \frac{k}{2}$.

"E" assigns pc to each bit M knew of μ's values for that bit — in view of previous bits in that A. To get KLD, P_M assigns pc's to each bit depending on

04 All previous A's, as well as previous bits in t. A containing bit in question.

Under cond .00 -.04, Gac's theorems about KLD [in Livi+ Vitányi 1997 pp 328-334] apply

And we get: $2 \sum_i D(\mu \| P_M) \leq K$

Here, to get meaningful "E", in .01 = we have to take E over t. i^th bit of the ensemble.

Some members of t. ensemble may not have i bits: but that's o.k., as P_M's "E" then doesn't include them.

So we have, for n "QA's" an ensemble of strings — not all of same length. Both μ & P_M assign pc's to each bit of each string, & we also have a "running product" of pc's for both μ & P_M.

Perhaps try to explain it as in "a paper"!

In t. sequence of bits; we do need "end of A" markers: The positions of the markers profoundly influence pc's of bits that follow them.

One possible way to deal w. this is to use STBTJC on each of the A_i's as a separate corpus. P_M is distinct for each A_i, here. We may need an extra symbol: "stop". [Hopefully, Hutter has published proof that .99 works for 2 symbols.]

Then $\sum \left(P_M(1) - \mu(1)\right)^2 + \left(P_M(0) - \mu(0)\right)^2 + \left(P_M(S) - \mu(S)\right)^2 \leq k$

More Generally ~~$\sum D(\mu(0), P_M(0)) + D(\mu(1), P_M(1))$~~

22 $\sum D(\mu(\cdot) \| P_M(\cdot)) \leq K$; $D(\mu(\cdot) \| P_M(\cdot)) = \sum_{y \in 0,1,S} \left(\mu(y) \ln \frac{\mu(y)}{P_M(y)}\right)$

? or "Y": Pick a dummy variable.

So, t. ensemble is t. set of all/ binary seqns w. n "S's" in them.

27 or, t. ordered set of all finite binary sequences (w. possible repetitions).

28 • For each A_i: $\sum_{\text{finite}} D(\mu, P_M) < E_\mu \ln \frac{\mu(A_i)}{P_M(A_i)} = E \ln \frac{\mu}{P_M}$ $\Big|$ $E \sum \ln \frac{\mu}{P_M} < K$

"E" here?

• But A_i's will vary in length for each "i". — ~~Why do we do this~~

• & we have to do .28 for all possible finite binary strings A_i: 0, 1, 00, 01, 10, 11 ... - each to ∞.

If we include t. "S" symbol, all A_i can be of same length

34 ① Or, use a "Blank" symbol, w. understanding that once a blank occurs then all following must be blank. What this does is make $\mu = P_M$ for all blanks after t. first. In this case, if we consider all A_i of length m (which also includes A_i of length < m) Blank.

• .28 holds w/ M .22 summed over $y = 0, 1, B$

38 & .28 consider A_1 & A_2 is $E \ln \frac{\mu}{P_M}\Big|_{A_1} + E \ln \frac{\mu}{P_M}\Big|_{A_2} = E \ln \frac{\mu}{P_M}\Big|_{A_1 + A_2} > ?$ Yes 711.00-.10 Previously

I need this to be true, if .28 is to be useful

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

re: 710.40 :

$$\sum_{A_1} \mu \ln \frac{\mu}{P_M} \Big|_{A_1} + \sum_{A_2} \mu \ln \frac{\mu}{P_M} \Big|_{A_2} = \sum_{A_1, A_2} \mu \ln \frac{\mu}{P_M} \Big|_{A_1, A_2}$$

$$\cdot \sum \mu A_1 \ln \frac{\mu A_1}{P_M A_1} + \sum \mu A_2 \ln \frac{\mu A_2}{P_M A_2} \quad \alpha$$

$$\sum \mu(A_1 \cdot A_2) \ln \frac{\mu A_1, A_2}{P_M A_1, A_2} \quad \gamma$$

$$\sum_{A_1 A_2} \mu(A_1) \cdot \mu A_2 \left(\ln \frac{\mu A_1}{P_M A_1} + \frac{\ln(\mu A_2}{P_M A_2} \right) \quad \beta$$

.02 .03 .05

$$\sum_{A_1} \sum_{A_2} \mu(A_1)\mu(A_2) \ln \frac{\mu A_1}{P_M A_1} = \sum_{A_1} \mu(A_1) \ln \frac{\mu A_1}{P_M A_1}$$

tr. $\sum_{A_2} \mu(A_2) = 1$ so it looks like $\alpha(.00) = \beta(.03) = \gamma(.02)$

0 — Which main $(00 L) = (.02 L)$ as desired. So -1.36 is true.

Which shows -1.28 can be sounded. (tr. $\sum$ "A_2")

12 So in ≥ -1.28 ! Summing over all A_i :

$$\sum_{i=1}^{N} \left[\sum_{k=1}^{\infty} D(\mu, P_M) \leq \sum \mu \ln \frac{\mu}{P_M} \Big|_{A_i} \right]$$

k = bits of A_i.

So Σ = total bit divergence

was -1.28 (incorrect?) do I need (E) here? Yes!

0 K_0 — K_0 is some large no.

21 $$\sum_{i=1}^{N} \sum_{k=1}^{K_0} \mu(A_i) D(\mu(A_i) \| P_M(A_i)) \leq \sum_{i} \mu \ln \frac{\mu}{P_M} \Big|_{A_i} \leq K$$

22 I think I can do it w. only 2 symbols: Just consider all possible lengths of A_i.

T. expected $E \sum$ term is still $\leq E \ln \frac{\mu}{P_M} \Big|_{A_i}$.

24 Not entirely clear how to do this, hrr. → Note 712.00, hrr (objection!)

Another Q! Re: proof of .00 – .03 Did I assume $P_M(A_1)$ & $P_M(A_2)$ were indep?

If may be that they are not — in which case I'd have to look at new Gacs deal w. conditional situation.

Look at .05 $\sum_{A_1} \sum_{A_2} \mu(A_1 A_2) \ln \frac{\mu A_1}{P_M(A_1)}$: : .05 may be o.k.

$\mu A_1 A_2 = \mu A_1 \cdot \mu A_2$: But, μA_2 may be a function of A_1! Ah, $P_M(A_2)$ may depend on A_1, but $\mu(A_2)$ is indep of A_1

lets look at $\ln \frac{\mu A_1, A_2}{P_M A_1, A_2}$: t. numerator factors o.k. t. Denom doesn't. $P_M A_1$ & $P_M A_2$ are not indep

I'm not sure this gives trouble, hrr. But I do have to look into it!

is $\sum_{A_1} \sum_{A_2} \mu(A_1)\mu(A_2) \left(\ln \frac{\mu A_1}{P_M A_1} + \ln \frac{\mu A_2}{P_M(A_2)} \right)$ | $\sum_{A_1} \sum_{A_2} \ln \frac{\mu(A_2)}{P_M(A_2)} \mu(A_1) \cdot \mu(A_2)$

→ 712.02 effac

$\sum_{A_1} \sum_{A_2}$ ~~$\mu(A_1)\mu(A_2) P_M(A_2)$~~ A_2 also.

8/23/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 (711.34 Spec) (SN) I Think t. "B's" of 710.34 (or equivt) may be necy. We do need info about when t. A_i end, start.

.02 (711.40 Spec) Look at $\sum \mu^{(A_1 A_2)} \ln \frac{\mu(A_1 A_2)}{P_M(A_1 A_2)}$ more carefully

It $= \sum\sum_{A_1 A_2} \mu(A_1)\cdot\mu(A_2)\left(\ln \mu(A_1) + \ln \mu(A_2) - \ln P_M(A_1) - P_M(A_2|A_1)\right)$

$= \sum\sum_{A_1 A_2} \mu(A_2)\cdot\mu(A_1)(\ln \mu(A_1) - \ln P_M A_1)$ $+ \sum\sum_{A_1 A_2} \mu(A_1)\mu(A_2) \ln \frac{\mu(A_2)}{P_M(A_2|A_1)}$ actually should be

$= \sum_{A_1} \mu(A_1) \ln\left(\frac{\mu A_1}{P_M(A_1)}\right)$

$\sum\sum \mu(A_1)\mu(A_2) \ln \mu A_2 \cdot - \ln \frac{P_M(A_1 A_2)}{P_M(A_1)}$

$= \sum_{A_2} \mu(A_2) \ln \mu A_2 + \sum_{A_1} \mu A_1 \ln P_M(A_1) - \sum\sum \mu(A_1)\mu(A_2) \ln P_M(A_1 A_2)$

.10 $\sum_{A_1} \mu A_1 (\ln \mu A_1 - \ln P_M A_1)$ is ok. for t. first term", but t. second term should be

$\sum\sum_{A_1 A_2} \frac{\mu(A_1 A_2)}{\mu A_1} \cdot \ln\left(\frac{\mu(A_1 A_2)}{\mu(A_1)} \Big/ \frac{P_M(A_1 A_2)}{P_M(A_1)}\right)$ which = so T. terms do add up to give desired equality & inequality

→ So I may have proved it! The A_2 term in both the $\le D(\|)$ and $\sum \mu \ln \frac{\mu}{P_M}$ parts are a bit complicated, but they "fit together" properly. — if we assume it works for radix 3.

.18 If, as seems necy, The B (Blank) symbol is needed, then we can make t. corpus be
.19 a simple ternary sequence, which is dealt w. in 578+3. (we have additional
.20 info that $\mu(A_i)$ & $\mu(A_j)$ are indep if $i \ne j$ (The $P_M(A_i)$ & $P_M(A_j)$ are usually (if not always) not independent.

It is my impression that Gacs' proof that $\sum_{i=1}^{n} \mu\, D(\mu \| P_M) < K$ works for any radix. That $D(\mu\|P_M) \ge 2(\mu - P_M)^2$: he only proves for radix 2 — which is easy:

.27 To show it for radix > 2 is not so easy: (is Almost Certainly true): Hutter says he has proved it.

One possl. way: start w. radix 2, then 3: for radix 2: $D(x\|y) = 0$ for $x=y$; $(x-y)^2 = 0$ for $x=y$.
Look at line $x=y$: (show) $\frac{\partial^2 D}{\partial y^2} > \frac{\partial^2 D}{\partial x^2} > 1$ for x, y on unit square.

Can this be shown for radix 3 & higher?

.30 for radix 2: $f = \sum \mu(x)(\ln \mu(x) - \ln p(x)) + (1-\mu(x)) \ln (1-\mu(x)) \ln(1-p(x))$

$\frac{\partial f}{\partial x} = \mu'(x) \ln \mu(x) + 1 - \mu'(x) \ln p(x) - \frac{\mu x}{p(x)} +$

~~for first deriv = 0,~~ Try this: for radix $r \ge 2$: Use Lag. Mults.

.35 We constraint $\sum x_i = 1$. We want $F = \sum_{i=1}^{r} \left(x_i(\ln x_i - \ln y_i) - (x_i - y_i)^2\right) = \min$ p(ohal) i.e. F is always ≥ 0.
W. Constraint $\sum_{i=1}^{r} x_i = 1$.

I'd guess that $\sum y_i \le 1$ is not necy.

2r eqns; 2r+1 unknowns! So we end up w.

a 1 parameter set of minimum F — fine! Its $x_i = y_i$ ($i = 1 \ldots r$)

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

50 : 7R40: No! $X_i = Y_i$ $i = 1|r$ is not a line: It's in $2r+1$ dim space and has r constraints so it's a $r+1$ dim manifold!
$X_i = Y_i$ $(i = 1|r)$ is only part of the solution. Rest are other constraints. — So it's ≥ 1 dim. objects.

$\Sigma y_i = 1$ is not used as a constraint

only $\Sigma x_i = 1$.

03 .. $F = -1.35! \ldots \frac{\partial F}{\partial x_i} = \ln x_i + 1 - \ln y_i - 2(x_i - y_i) + \lambda = 0$ [$+\lambda \frac{\partial}{\partial x_i}$]

05 . . . . . $\frac{\partial f}{\partial y_i} = -\frac{x_i}{y_i} + 2(x_i - y_i) = \phi$??

If $x_i = y_i$ then $\lambda = 1$ (from .03)

" $x_i = y_i$ gives no soln for .05 !

.09 If we add constraint $\Sigma y_i = 1$, we get a λ_2 in

10 $\frac{\partial f}{\partial y_i} + \lambda_2 = -\frac{x_i}{y_i} + 2(x_i - y_i) + \lambda_2 = 0$. so $\lambda_2 =$

From $x_i = y_i$ is a soln. and $\lambda = \lambda_2 = 1$. But Reza says only $r+2$ constraints.

using (.03) & (.05) as is: $\ln \frac{x_i}{y_i} + 1 + \lambda \frac{x_i}{y_i} = 0$ / solving for $\frac{x_i}{y_i}$ gives $\frac{x_i}{y_i} = 1$ if $\lambda = 1$

$= -2(x_i - y_i)$ from .03

but if $\frac{x_i}{y_i} \equiv z$ $\ln z - z = -1 - \lambda$ so z can have any value $\rightarrow \phi$ so we can find an associated λ. for $z < 0$, λ is complex. — but $\frac{x_i}{y_i}$ has to be z same for all i.

However, say $\frac{x_i}{y_i} = \alpha$ $x_i = \alpha y_i$ by .05 $-\alpha + 2(\alpha y_i - y_i) = \phi$ $-\alpha + y_i 2(\alpha - 1) = 0$

20 $-\alpha + 2(1-\alpha) y_i = 0$ $y_i = \frac{\alpha}{2(1-\alpha)}$ $x_i = \alpha y_i = \frac{\alpha^2}{2(1-\alpha)}$

so this is a 1 param manifold: If we choose $\alpha = 2$. then $y_i = \frac{\alpha}{2(1-\alpha)}$ and $x_i = \alpha y_i$ for all i.

$\beta = \frac{\alpha}{2(1-\alpha)}$ $\frac{1}{2\beta} = \frac{2(1-\alpha)}{\alpha} = \frac{1}{\alpha} - 1$ $\frac{1}{\alpha} = \frac{1}{2\beta} + 1$

$\alpha = \frac{1}{\frac{1}{2\beta} + 1} = \frac{2\beta}{1 + 2\beta}$

y_i ; $x_i = \frac{2 y_i^2}{1 + 2 y_i^2}$ However, if $x_i = \alpha y_i$ & $\alpha \neq 1$ then y_i can't be normalized

Conversely, y_i never is normal.

But in my 5758 T3, y_i was normalized! and the relation below held & $(x \cdot y)^2$ was 0.4.

Also $x_i = y_i \big|_{i=1|r}$ is a min point, y of .05 gives $\frac{\partial f}{\partial y_i} = -1$ at $x_i = y_i$

30 Try $r=2$: $x_1(\ln x_1 - \ln y_1) + x_2(\ln x_2 - \ln y_2) - (x_1 - y_1)^2 + (x_2 - y_2)^2 + \lambda = 0$

$\frac{\partial}{\partial x_1} = 1 + \ln x_1 - \ln y_1 - 2(x_1 - y_1) + \lambda = 0$

$\frac{\partial}{\partial y_1} = -\frac{x_1}{y_1} + 2(x_1 - y_1)$

8/25/02 714

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

: Geci' discussion of top of 1329, Li Vit 1997

[scribbled out] $f(Q(0)) = P(0)(\ln P(0) - \ln Q(0)) + P(1)(\ln P(1) - \ln Q(1)) - 2(P(0)-Q(0))^2$

$f(P(0)) = \text{''} \quad \text{''} \quad P(0)$ $= 0$ if $P(1) = Q(1)$.

$$\frac{\partial f}{\partial Q(0)} = \frac{P(0)}{Q(0)} - \frac{P(1)}{Q(1)} + 4(P(0)-Q(0))$$

look at old 578 Notes, early versions.

Ignore:

$$F(Q_0) = \sum_{i=0,1} \left(P_i \ln\left(\frac{P_i}{Q_i}\right) - (P_i - Q_i)^2 \right) \qquad \frac{\partial f(Q_0)}{\partial Q_0} = \frac{P_0}{Q_0} - 2(P_0 - Q_0)$$

Title of Festschrift paper: Convergence theorems revisited: complexity based induction

In the original unrevised 578 paper, I had the proof of KLD $< 2(\epsilon\ln 2)^2$.

How, I only use $\frac{\partial f}{\partial \mu}$ & $\frac{\partial^2 f}{\partial \mu^2}$! How both μ & P_M were normed measures.

So, from (-1.05) $\frac{\partial f}{\partial y_i} = -1$ for unnormed $\vec{y}$ at $\vec{x} = \vec{y}$.

If we normalize as (-1.09), we can get $\frac{\partial(f + \lambda \Sigma y_i)}{\partial y_i} = 0$ at $\vec{x} = \vec{y}$.

$$\frac{\partial^2 (f + \lambda_2 \Sigma y_i)}{\partial y_i} = \frac{x_i}{y_i^2} - 2 = \frac{1}{y_i^2} - 2 \quad \text{at } \vec{x} = \vec{y}.$$

at $y_i = 1$, this is -1! at $y_i > \sqrt{\tfrac{1}{2}} = .707$ it is < 0!!

If, in 712.35 we let $f \equiv \sum_{i=1}^{n} x_i (\ln x_i - \ln y_i) - \frac{1}{2}(x_i - y_i)^2) \left(+ \lambda \Sigma x_i + \lambda_2 \Sigma y_i \right)$

It works out o.k. for the normalized case.

write a Basic pgm to test with examples! My impression was that 712.35 as it was, worked o.k. for $r = 3, 4$!

X = .5 [KLD-SQ.BAS] ← QB Directory
Y = .5 + (1-x) log(((1-x)/(1-y))

Print x * log(x/y) - 2(x-y)^2

x, y,

I got sign of KLD wrong in 712.35! Should be $\sum x_i (\ln x_i - \ln y_i) \ldots$ — **NO! Sign was o.k.**

$-1.05 \rightarrow \frac{\partial f}{\partial x_i} = -\ln x_i - 1 + \ln y_i - 2(x_i - y_i) + \lambda = 0$ for $\vec{x} = \vec{y}$, ∴ $\lambda = +1$ gives ϕ for lag. mults.

$-1.03 \rightarrow \frac{\partial f}{\partial y_i} = \frac{x_i}{y_i} + 2(x_i - y_i) \quad (+\lambda_2)$? ... $\lambda_2 = -1$ gives [scribbled out] ϕ for Lag. Mult.

$\frac{\partial^2 f}{\partial x_i^2} = -\frac{1}{x_i} - 2$: always < 0 ⌒

$\frac{\partial^2 f}{\partial y_i^2} = \frac{-x_i}{y_i^2} - y_i$: always < 0. ⌒

So it works o.k. if $\vec{y}$ is normed, for arby radix.

[KLD-SQ2.BAS] $\ln(1-y) + \ln(y_2)$

same, but for x, y, y_2 $(x-y)^2 + (x-y_2)^2$

say $y_1 + y_2 < 1$.

8/25/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

30:714.40: So using normal $\vec{y}$ (Σ λ=2):

$\frac{\partial\, 713.05}{\partial x_i} = \frac{1}{x_i} - 2$ is > 0 if $x_i < \frac{1}{2}$ $x_i = y_i = .8$

$\frac{\partial\, 713.05}{\partial y_i} = \frac{x_i}{y_i^2} - 2$ is > 0 if $\frac{x_i}{y_i^2} < \frac{1}{2}$ $x_i^2 < \frac{1}{2} y_i^2$

Empirically for r=2; both $\vec{x}$ & $\vec{y}$ normed;

$\frac{\partial f + (\,)}{\partial y_i} > 0$ at x=.8, y=.8 ; at x=.5 y=.5 $\frac{\partial f}{\partial y}$ seem to be slightly < 0 for a small excursion.

(F(.5, .5±.05) → −2.0478 E−08 (exactly same for both)

I tried various values very close to y=.503 f(x=.5, y=.503) < 0 but as y varies

around that the behavior is erratic: .505 is only 3 decimal places, Basic should have no problem) — But differences between small quantities! — try double precision!

Seems to be o.k. in double precision; (I guess it does do ln( ) in double, too.)

30.12 try $f = x \ln \frac{x}{y} + (1-x)\ln\frac{1-x}{1-y} + 2(x-y)^2$ $x \ln x \to$ $-x1$

$\frac{df}{dx} = \ln x + 1 - \ln y$ | $\frac{d}{dx}(1-x)(\ln(1-x) - \ln(1-y))$ + $2(x-y)$

$\ln(1-x) - x\ln(1-x) + x\ln(1-y)$

$\ln x + 1 + \frac{1}{1-x} + \frac{x}{1-x} - \ln(1-x) + \ln(1-y) + 2(x-y)$

20 $\ln x + \ln\left(\frac{1-y}{1-x}\right) + 2(x-y)$ $= \ln x + \ln(1-y) - \ln(1-x) + 2(x-y)$

$(x \ln \frac{x}{y}) = x\ln x - x\ln y \to 1 + \ln x - \ln y$ omitted!

$(1-x)(\ln(1-x) - \ln(1-y)) = \ln(1-x) - \ln(1-y) - x\ln(1-x) + x\ln(1-y) \to -\frac{1}{1-x} + \frac{x}{1-x} - \ln(1-x) + \ln(1-y)$

$2(x-y)^2 \to 2(x-y)$

$1 + \ln x \left(+\frac{x}{1-x} - \frac{1}{1-x}\right) + \ln\frac{(1-y)}{(1-x)} + 2(x-y) = \ln \frac{x(1-y)}{(1-x)} + 2(x-y)$

$= \ln x - \ln(1-x) + \ln y + 2(x-y)$ | If x=y; This is $\ln x$

30 — Maple got $\ln\frac{x}{y} - \ln\left(\frac{1-x}{1-y}\right) + 2x - 2y$ which is correct, I guess (forgot)

$\frac{d\, 30}{dx} = \frac{1}{x} + \frac{1}{1-x} + 2$ which is > 0 for 0<x<1

$\frac{d(12)}{dy} = -\frac{x}{y} + \frac{1-x}{1-y} - 2x + 2y$ = 0 if x=y

$\frac{d}{dy} = +\frac{x}{y^2} + \frac{1-x}{(1-y)^2} + 2 > 0$ for 0<y<1,

$x_1 \ln(x_1/y_1) + x_2 \ln(x_2/y_2) + (1-x_1-x_2)\ln(1-x_1-x_2)/(1-y_1-y_2)$

8/25/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 using maple I did $x_1, x_2, (1-x_1-x_2)$: $y_1, y_2, (1-y_1-y_2)$

$f \equiv D(x||y)$; then $\frac{df}{dx_1}$! $=0$ if $\vec{x}=\vec{y}$

presumably similar for x_2

dry I took dir wrt x_2 again ; got $\frac{1}{x_1} + \frac{1}{1-x_1-x_2}$ +4 which is >0 for unit sq,

try y_1 $\frac{\partial f}{\partial y} = 0$ for $\vec{x}=\vec{y}$.

second dir wrt y_1 is $\frac{x_1}{y_1^2} + \frac{1-x_1-x_2}{(1-y_1-y_2)^2}$ +4 which is >0 on unit sq,

So, for any legal $\vec{x}, \vec{y}$ pt, show that ∃ a $\vec{x}_0, \vec{y}_0$ pt. with $\vec{x}_0 = \vec{y}_0$ ∋

$\vec{x}$; $\vec{y}$

0 On second thot: if all second derivs are positive or zero,

The rtm is in Maple file: KLDIST.MS for (00) one can easily get any first or

second derivative

$x^2 + y^2 + z^3$: (a) ← not colon,

diff(", Y);

put cursor at a, then "Enter", then cursor past "after" ; . Then enter — will get derivative w.r.t. y of the top (";") expression. doing ";" again will dif wrt y of that result.

Anyway, for n=3,

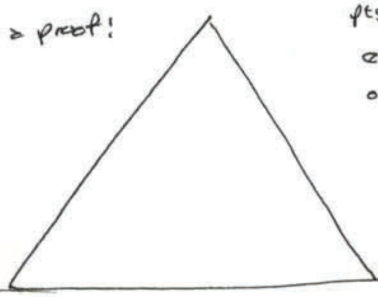
all derivs are 0 & value 0 on $\vec{x}=\vec{y}$ surface

All second derivs are >0 on unit cube for all variables

30 From this info, it may be possible to prove that at all pts, one must go "uphill" form

a $\vec{x}=\vec{y}$ point.

Maybe a proof!



pts. in an = lat Δ represent vectors $\vec{x}$: $\sum x_i = 1$
$\vec{y}$: $\sum y_i = 1$.
each character component = t. distant to one of the sides of the Δ.

30 New tack: we want to show $D(\vec{x}||\vec{y}) \geq 0$ for all $\vec{x}, \vec{y}$:

Take a certain $\vec{x}_0, \vec{y}_0$: we know $D(\vec{x}_0||\vec{x}_0)=0$. Consider a path $\vec{x}_0$, $\vec{x}_0 + t(\vec{y}_0 - \vec{x}_0)$ $\Big|_{t=0}^{t=1}$.

for t=0, D=0, each component of y is changing monotonically along the path, so since

.33 $\frac{\partial^2 D}{\partial y_i^2} \geq 0$ along the path, D must ↑ (unless $\vec{y}_0 = \vec{x}_0$). (for $z = 1/t$)

D is a nice, continuous function, w. no singularities in t. region — except, for edges, where components of $\vec{x}$ or $\vec{y}$ = 0. So inside t. region, we have fine, smooth functions,

∴ $\delta D = \sum_i \frac{dD}{dx_i} \Delta x_i + \frac{dy}{dy_i} \Delta y_i$.

Q. if we have $g = x^2 + y^2 - 2xy$ $\frac{\partial^2 g}{\partial x^2} = \frac{\partial^2 g}{\partial y^2} > 1$ but $\frac{\partial^2 g}{\partial x \partial y} = -1$ what does g( ) look like?

8/26/02
IP

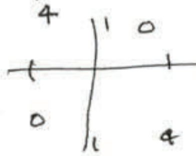
BUG? — $\frac{\partial^2 D}{\partial x_2^2}$ should be symmetric in x_1 & x_3 → ~~$\frac{\partial^2 D}{\partial x_2^2} = \frac{1}{x_2} + \frac{1}{x_3} + 4$~~ 217

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

$\frac{\partial D}{\partial y_1^2} = \frac{x_1}{y_1^2} + \frac{x_3}{y_3^2} + 4$ ~~$\frac{\partial^2 D}{\partial x_1^2} = \frac{1}{x_1} + \frac{1}{x_3} + 4$~~ all wrong

See 717 1/2

:0 : $g(0,\pm 1) = f(\pm 1, \phi) = 1$ ϕ <



$g_{10} = x^2 + y^2 - 10xy$ would be more extreme

So, this is second derivs of g_{10} wrt x & y are both > 0,

T. function can still ↓ much.

So look at $\frac{\partial^2 D}{\partial x_1 \partial x_2}$ & $\frac{\partial^2 D}{\partial y_1 \partial y_2}$ $\frac{\partial^2 D}{\partial x_1 \partial x_2} = \frac{1}{x_3} + 2 > 0$

$\frac{\partial^2 D}{\partial y_1 \partial x_2} = -\frac{1}{y_3} - 2$ is always < 0, but

$\frac{\partial^2 D}{\partial y_1 \partial y_2} = \frac{x_3}{(y_3)^2} + 2$: also always > 0.

.07 This is not relevant to t. proof.

Woops! if $\frac{\partial D}{\partial x_1 \partial x_2} > 0$ we have "like" $x^2 + y^2 + 10xy$: $f(+1,-1) = 1 + 1 - 10 = -9$.

So if t. cross derivs is small, its OK.

$ax^2 + by^2 + cxy$ what is relation of a b c so t. function is always > 0? well if Say a, b known. when is $ax^2 + by^2 > cxy$ or $\frac{ax^2 + by^2}{xy} > c$ $a\frac{x}{y} + b\frac{y}{x} > c$ (Absolute Constant + $0 < \frac{x}{y} < \infty$)

$z = \frac{x}{y}$ min $az + \frac{b}{z}$: $a - \frac{b}{z^2} = 0$

$a = \frac{b}{z^2}$ $z^2 = \frac{b}{a}$ $z = \sqrt{\frac{b}{a}}$

So c must be ~~$\left(\sqrt{\frac{b}{a}}\right)^2 \cdot a + \left(\sqrt{\frac{a}{b}}\right)^2 \cdot b = \min$~~
~~$= a + b$ so $|c|$ must be $< a+b$~~

But I have to go over this "proof" more carefully

no $a \cdot z + \frac{b}{z} > c$ is what we need

$a\sqrt{\frac{b}{a}} + b\sqrt{\frac{a}{b}} > c$ $2\sqrt{ab} > c$ $4ab > c^2$ is nec'y for t. $\frac{\partial^2}{\partial \cdot\cdot^2}$ to be > 0 in all directions.

from this & t.

it is clear that t. cross derivs have to be considered — but they have to be smaller by a certain amount than t. second partial derivs. We necessarily concern ourselves with $\frac{\partial^2}{\partial x^2}$ & $\frac{\partial^2}{\partial y^2}$ & $\frac{\partial^2}{\partial x \partial y}$. t. x & y are both from R or both from Pm; either would work out.

This is because of t. argument of -1.30 -.33!

We can make t. "path" all in t. ($\vec{x}$, say) or all in Pm ($\vec{y}$, say). In (.00 -.07)R, t. x_1, x_2 are from R

.21 & y_1, y_2 from Pm.

So: conclusion of 712.27 — 717.21 : That t. conv. thrm. for Radix 3 can prob'y be shown by ① using Gac's proof for the D(. || .) ≠ ! I think this is for any radix. 712.27 — 717.21 tries to show that " implies $\Sigma(\text{error})^2$: I.e. That

$(x-y)^2 < D(x \| y)$ for radix 3: The idea would involve showing that cross derivatives in (.00 -.21)^9 are smaller than t. "uncrossed" second derivatives (see (.10 -.20)R. I haven't shown this; if true, it would be easy, using Maple to get derivatives. If false, then this proof as is, is No good.

Hrr. Hutter says he proved t. conv. thrm. for any radix: So use his claim: Its almost certainly true, indip of whether his proof is correct (Note that both seqs. compared must be "Measures")

What I want to do now is write up t. conv. thrm. discussion for "T. report/paper(s)".

I'm starting by reading t. discussion in "2 kinds" (S99): The model of induction I'm using is a mixture of t. 2 models discussed in Sections 3.1, 3.2 & 3.3

A difference May be this! (& this argument may apply to t. problem of S99 as well):

8/27/02

Maple : (ID).

Maple file KLDIST.MS

717½

ID

```
>
```

This is "corrected" version!

```
> x*ln(x/y)+(1-x)*ln((1-x)/(1-y))+(x-y)^2:
```

← not used.

Maple Printout

D :=

```
> x1*ln(x1/y1)+x2*ln(x2/y2)+(1-x1-x2)*ln((1-x1-x2)/(1-y1-y2))-(x1-y1)^2-(x2-y2)^2-(x1+x2-y1-
> y2)^2:

> diff(" , x1);
```

$$\ln\left(\frac{x1}{y1}\right)-\ln\left(\frac{1-x1-x2}{1-y1-y2}\right)-4\,x1+4\,y1-2\,x2+2\,y2$$

```
>

>
```

This should be symmetric in $\begin{pmatrix} y_2 \\ x_2 \end{pmatrix}$ and $\begin{pmatrix} y_3 \\ x_3 \end{pmatrix}$!!

$$\underset{1}{(\ln x_1 - 4x_1)} \cdot \underset{1}{-(\ln y_1 + 4y_1)} \Big| \underset{3}{-\ln x_3 + \ln y_3} \Big| \underset{2}{-2x_2 + 2y_2}$$

$-4(x_1-y_1) - 2(x_2-y_2)$ ≟ $-(1-x_1-x_2) - (1-y_1-y_2)$

$-2(x_1-y_1) - 2(x_3-y_3)$

$\frac{\partial^2 D}{\partial (x_2)^2} = \frac{1}{x_2} + \frac{1}{x_3} - 4$ ≤ −2 — it's always ≤ −2 — but it should be

Sym in "1" & "2" indices (how, x_1 & x_3 ; not x_2 & x_3)

---

try $E = (x_1-y_1)^2 + (x_2-y_2)^2 + (x_3-y_3)^2$ $\quad -x_1-x_2+y_2+y_2$

$\frac{dE}{\partial x_1}$? $\quad (4(x_1-y_1) + 2(x_2-y_2)) = -2((x_1-y_1) + (x_3-y_3))$

$E = (x_1-y_1)^2 + (x_2-y_2)^2 + (x_1+x_2-y_1-y_2)^2$

$\frac{dE}{\partial x_1} = 2(x_1-y_1) + 2(x_1+x_2-y_1-y_2) = +2(x_3-y_3)$

So Prob seems to have mechanics of symmetry! — x_1 is involved w. x_1 & x_3, but not x_2!

But even so: $D = 0$ for $x = y$:

but $\frac{\partial^2 D}{\partial x_1^2} = \frac{1}{x_1} + \frac{1}{x_3} - 4$

$\frac{\partial^2 D}{\partial x_1 \partial x_2} = \frac{1}{x_3} - 2$

$\frac{\partial^2 D}{\partial x_2^2} = \frac{1}{x_2} + \frac{1}{x_3} - 4$

$x_1 + x_3 \le 1$ $\quad \frac{1}{x_1} + \frac{1}{x_3}$ is min at $x_1 = x_3 = \frac{1}{2}$

So $\frac{D^2}{\partial x_1^2}$ is ≥ 0.

← get min of this expression wrt x_1 & x_2 by Prob! (easy)

So is $4\left(\frac{1}{x_2} + \frac{1}{x_3} - 4\right)\left(\frac{1}{x_1} + \frac{1}{x_3} - 4\right) > \left(\frac{1}{x_3} - 2\right)^2$?

Let $z = \frac{1}{x_3}$ is $4\left(\frac{1}{x_2} + z\right)\left(\frac{1}{x_1} + z\right) > (z-2)^2$

What is min of $\left(\frac{1}{x_2} + z\right)\left(\frac{1}{x_1} + z\right)$ as a function of x_3 ? (func of z)

$x_3 = 1 - x_1 - x_2$

$z^2 + \left(\frac{1}{x_1} + \frac{1}{x_2}\right)z + \frac{1}{x_1 x_2}$

Drop this for the present! → 717.6.00

8/27/02

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

50: 717$\frac{1}{2}$.90 I took first partial wrt x_1 of $\left(\frac{1}{x_2}+\frac{1}{x_3}-4\right)\left(\frac{1}{x_1}+\frac{1}{x_3}-4\right)$ (Maple: KLdist2.mws)

$$\rightarrow \left(-\frac{1}{x_1^2}+\frac{1}{x_3^2}\right)\left(\frac{1}{x_2}+\frac{1}{x_3}-4\right)+\frac{\frac{1}{x_1}+\frac{1}{x_3}-4}{x_3^2}$$

03 It $=0$ if $\frac{1}{x_1}+\frac{1}{x_3}=4$ and $\frac{1}{x_2}+\frac{1}{x_3}=4$ ← solving this would also give $\frac{\partial}{\partial x_2}=0$

$\therefore x_1 = x_2$ $\quad \frac{1}{x_1}+\frac{1}{1-2x_1}=4 \quad \frac{1-x_1}{x_1\cdot(1-2x_1)}=4$

$4x_1 - 8x_1^2 = 1-x_1 \quad 8x_1^2-5x_1+1=0 \quad 8x^2-5x+1=0$

looks bad! $\quad 25-32<0$ so no real soln! (on the other hand, I know from symmetry considerations that [illegible]

10 I might try to find way to get Maple to try to find num. soln. for $\frac{\partial}{\partial x_1}=\frac{\partial}{\partial x_2}=0$.

or, just plot the function

(margin: soln on line $x_1=x_2$ — at (.25) it is; $x_1=x_2=.297$ ↑ But this is not by "guess" of (.03))

Maybe plot. 717$\frac{1}{2}$.00) $\quad$ for mat. plot3d(expr, x=-0.1..0.1, y=-0.1..0.1);

Looks like ∞ at $x_3=0$ and $x_1=x_2=0$ (or $x_1+x_2=1$)

Look at $x_1=x_2$: $\left(\frac{1}{x}+\frac{1}{1-2x}-4\right)^2$ from $x=0$ to $x=.5$

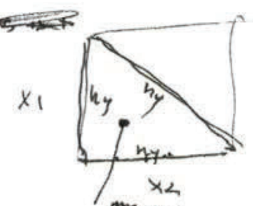
20 $\frac{d}{dx}\left(\frac{1}{x}+\frac{1}{1-2x}\right) = -\frac{1}{x^2}+\frac{2}{(1-2x)^2}=0 \quad x^2=\frac{(1-2x)^2}{2} \quad x=\frac{1-2x}{\sqrt{2}} \quad (\sqrt{2}+2)x=1$

$x=\frac{1}{2+\sqrt{2}} \quad \left(2+2\sqrt{2}+\frac{1}{1-\frac{2}{2+\sqrt{2}}}-4\right)^2 \approx 6.43$

4.83 $\quad$ 2.537

Because of symmetry wrt x_1, x_2, .00 will have min on the line $x_1=x_2$.

.305 $\left(\frac{1}{x}+\frac{1}{1-2x}-4\right)^2-\left(\frac{1}{1-2x}-2\right)^2$ has min at $x\approx .297$; w. value 13.18 and so is > ϕ.



I plotted 4($\frac{1}{x_1}+\frac{1}{x_3}-4$)($\frac{1}{x_2}+\frac{1}{x_3}-4$) − ($\frac{1}{x_3}-2$)^2

$x_1=.1$ to $.4$
$x_2=.1$ to $.4$ $\quad$ It looks like the min really is at $x_1=x_2=.297$

and its value is 13.18: So the result is that

30 the second partial deriv of $D(11)-$ [illegible] is positive in all direction

Hm., I shouldn't spend any more time on this! — that the $E\sum D(11)<k$ is really adequate, & can be easily shown for all radix using Gac's method (I think!)

But just assume Hellor's proof is O.K.

8/27/02

±D

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : Using Notation of S99: § 3.1 --- 3.3: $P(D_n)$ corresponds to μ (the "true" d.f.)

. operating on D_n (which corresponds to A

. Maybe not use that Notation: It will cause confusion: Say $P(A_i|Q_j)$ is the "true"

.03 . generator of the set of A's (w.r.t. the [Q] set). Then $\prod_{i=1}^{n} P(A_i|Q_i)$ is indep of order of the Q's.

Also our approximation functions, $O_j^n(A_i|Q_i)$ will be indep of order of the Q_i.

.07 As a result $\sum_j a_j^0 \prod_{i=1}^{n} O_j^n(A_i|Q_i)$ will be indep of the ordering of the Q_i.

The conditional p.c. obtained for a A_i part way thru the set, will, hvr, depend on the Q_j's that

have preceded it. Both $P(\cdot|\cdot)$ (the "true" generator) &

10 Hvr, for a particular ordering of the Q's, we will obtain a pc for each A_i & an induced pc for each

the wtd set $O_j^n(\cdot|\cdot)$ will obtain a pc for each A_i &

bit of every A_i — & a pc for the termination of each A_i &

So we can use S78T3C: (w. radix 3) on the pair of proby sequences for the bits of $\{A_i\}$ — correctly; the set $O_j^n(\cdot|\cdot)$

the 2 seqs are those obtained by .07 & those obtained by .03 $(P(\cdot|\cdot))$.

From the argument of S99 p 253 (col II), we get this lower bound w. constant factor $2^{...}$

So we get the final convergence thm viz S78T3C.

I must mention in the paper that the probys of the stop symbols S are an impt. part of

20 the QA system — It is important to know if a particular A_i candidate ends

at a particular pt. or not, so one knows whether to start on the next Question.

22: 706.19 → I will use much of the Exposition of CPM Nov. 8, 01 to desrb. the QA soln (for cc = ∞) & just why we are looking for O_j^{n}'s w. large $a_j^0 \prod_{i=1}^{n} O_j^n(A_i|Q_i)$ → $O^j(A_i|Q_i)$ — The "CPM" Notation is

In — There is a second eq. that gives wts of the O_j^n & we want O_j^n's w. much wt.

In S99 § 3.1, I considered all possl. orderings of the Bzs elements. If the $P_M(\cdot)$ is order independent (& it would be well to make it so, because the "Target" P.D. is order indep.) Then we may be able to use just one ordering & not sum over all $\approx n!$ of them

The individual O_j^n's try to simulate μ, the target P.D. & so they are order independent. So the wtd sum of O_j^n's must be Order independent

---

30 Looking at CPM N8, 01: on P2 2nd last ¶ I say "the distribn 414.18 is very accurate" I mite refer to Appendix, which shows this"

It would be well to give an idea of the result of the appendix.

In Appendix ... we show that: (I'll have to make this approximate, — necessary! NO!

since my statement here, will not include pc's of "stop" symbols).

I can Actually Say that total error is $\le k$ because adding in "s" errors makes inequality stronger!

1908

8/28/02

FD

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.30 Re: the Maple calcns: I did a bad mis-typing of formula!

now results $\frac{d}{dx_1}D = \ln\frac{x_1}{y_1} - \ln\frac{x_3}{y_3} + 4(y_1-x_1) + 2(y_2-x_2)$ This should be sym in $\binom{y_2}{x_2}\binom{y_3}{x_3}$

it does = 0 when $\vec{x}=\vec{y}$. (see 717.5 & 717.6 for more recent results!)

.04: 718.40 | Another tack for .Mis.) One can show D to be > 0 at many lattice pts & since its derivative has known bounds in nearby regions, it can't change sign.

We have to show the derivative is bounded by a certain amount, then we can do a check at lattice pts close enough together. The main problem is to get limits on derivatives for each region.

08: 718.40 AH! For CPMN8,01 [?]: "T. Distrib of 414.18 is known ..."

Instead: How accurately the distribution of 414.18? Altho the distribution itself gives the probability of each A_i for each Q_i, we can use it to get probabilities of the individual bits of each A_i: In Appendix B, we show that the total of the error in probability between ~~...~~ over all bits in all of the A_i's, of the error in probability between over all bits in all of the Ai's

In Appendix B, we assume that there is some probabilistic question answering program μ that has a finite description of length k bits in length. We show that if we sum the expected difference between the probabilities given by eq 414.18 and those given by μ, the value will be less than k.

19 How Accurate is the distribn. of eq (1)? Suppose we as: Let us assume that the data sequence $A_1 \ldots A_n$ was generated by some probability distribution μ, conditional on the Q_i's, and that μ can be described with as little as k bits. (did I explain what this means?) ?

How much difference is there in the bit probabilities given by μ v.s. those given by eq (1) Appendix B shows that the expected value of the sum of the squares of the probability differences between the 2 models, is less than $\frac{1}{2}\ln 2 \cdot k$. This result corresponds to the accuracy that we expect in sequencial predictions, using probability models based on universal machines. For a large corpus of data, the square of the probability differences must go to zero more rapidly than $1/n$ — n being, > see 720.00 .03 the total number of bits in the sequence of correct answers, not the number of questions, but the

T. Q A's are encoded. So error < const/n

T. "old" formula may be rite.

The significance of this result is discussed in Appendix B. ??

.33 Significance

34 SN In discussn of "Signific." of this $\frac{1}{2}\ln 2 \cdot k$ result: Perhaps give order of magnitude of k & compare w. no of bits in a corpus.

Discuss case in which kind of Q's changes abruptly: (In which k ↑ a lot.) since the dscrn length of μ must now include new info.

8/29/02 "Appendix B" → This is not exactly App D. App D was inspired by this seeking to be true – this → sec 720

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : [SN] Because the QA's are unordered, whenever a new QA is added to the corpus, all of the p.d.'s for all of the A_i's change. Usually the err² for every A_i is about the same as all other

.02 A_i's. Thus mean err² is $\leq \frac{(\frac{1}{2}\ln 2)k}{n}$. (It is true: .00–.02 doesn't contradict it.

03 Actually, I'm not at all sure of this. I original $\Sigma(\text{err}^2) < k(\ln 2)$ now be rite.
Perhaps when I'm in the process of writing the proof of the Conv. Thrm. for QA's, I can return to this Q. → (80720)

.04 Suppose that there exists an intelligent algorithm M, that can look at any question Q_i and give a good

.06 probability distribution $M(A_i|Q_i)$ on all possible answers A_i.

.07

10 "Appendix B" Convergence Theorems for QA induction.

Suppose that we have an algorithm M, that looks at questions, Q_i and for each such question gives a probability distribution $M(A_i|Q_i)$ on all possible answers A_i, to the question, Q_i. (replace with .06–.07)

(omit) We also have the function defined by eq (1) that also gives a probability distribution for each A_i in response to an ordered sequence of Q_i.

16 → (insert .31–.34) Given any ordered sequence of Q_i's ~~(i=1...n)~~, can we give a measure of how close ~~eq 41418~~ the probability distributions given by ~~eq 41418~~ are to those of the generating function, M?

20 The Kullback-Leibler distance, is one such measure that we will use. (Give reference to Cover-Joy on KL dist). If $M(x_i)$ is a probability distribution on objects x_i ($i=1\ldots r$) and $P(x_i)$ is regarded as an estimate of $M(x_i)$ then (used by Cover)

In the present case the KL distance $D(M\|P)$ is defined to be: ((unambiguity of $M(\;), P(\;)$)

$$\sum_{i=1}^{r} M(x_i)\ln\left(M(x_i)/P(x_i)\right).$$ (May be confusing)

It is the expected value (with respect to M) of $\ln(M(x_i)/P(x_i))$.

Willard Gibbs showed that $D(M\|P)$ is always ≥ 0.

30

31 Let us define $P(A_i|Q_i)$ to be the probability distribution for A_i as given by eq. (1) for a particular sequence of Q_i's. → (.16)

Given this Q_i sequence, what is a good measure of the difference between the probabilities assigned to the A_i's by M and those assigned by P?

.34 Using the technique described by Peter Gacs (Li V 1997 pp; Li V 1993 pp) It is possible to show that

~~$E\sum_{i=1}^{n} D(M(A_i|Q_i)\|P(A_i|Q_i))$~~ =

.39 $$E\sum_{i=1}^{n}\sum_{j} M(A_i^j|Q_i)\ln\left(M(A_i^j|Q_i)/P(A_i^j|Q_i)\right) \leq \left(\tfrac{1}{2}\ln 2\right)k$$ (since I used ln in LHS, I can eliminate ln 2 factor) (Maybe $A_i^j \to A^j$.)

Here the expected value is with respect to M, and the summation, j, is over all possible responses, A_i^j, to Q_i. k is the number of bits in the description of M ... "natural bits".

8/30/02 "Appendix BB" 722

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : Equation (3) ~~721.25~~ gives a very simple measure of the accuracy of equation (1) ~~414.18~~.

There are no "order of one" constant factors or additive terms. A necessary uncertainty is in the value of K. We cannot ever be certain that we know its value. If the generator of the data has a long and complex description, we are not surprised that we should need more data to make good predictions — which is just what eq. (3) ~~721.25~~ specifies.

The value of the constant, k, depends critically on just what universal reference machine is being used to assign a priori probability to the O_j^n and to M. Any a priori information that a researcher may have can be expressed as a modification of the reference machine — by inserting low cost definitions of concepts that are believed to be useful in the needed induction — usually resulting in a shorter code for M (a smaller K) and lesser... We believe that if all of the needed a priori information is put into the reference machine, then eq (1) ~~414.18~~ is likely to be the best probability estimate possible. → .30

.14 (with no ~~additive or multiplicative constant~~) ← ? small.

15 Since we cannot expand the infinite resources needed to evaluate eq 414.18, (are never available). [last ¶ of CPM N8,01 Page 2]

I put this in 6/17/02 version

.17 [SN] Discuss how this is an approach to "STRONG AI": That "common sense" is not needed in ~~doing~~ long Math, And once a system has devised good Ing Algms, it can acquire common Sense by Ing, rather than "Brain Surgery" (= programming). — It's a much better way to acquire "Common Sense". We want to use as little "Brain Surgery" as possl. Explain just Why.

.25 → [T. report so far]: 696.13 — .40 Introdn: More or less done. (+ Abstract w. additions) additions. (on why getting this eq.(1) is good.)

Section on QA problem. [Use CPM N8,01] with Modifs/additions of 704.13–.99 Also addition (722.16–.17)

Need and ~~ad~~ 719.19 → .331 Prizdiscusses Accuracy of the "eqs" : refers to "Appendix B": Done in 6/17/02 version

"Appendix B" is 720.10 — 722.15

.28

.30 : (.14) At first glance, this result may seem unreasonable! Suppose we ask the system many questions about Algebra, until its mean errors are quite small — Then we suddenly begin asking questions about Linguistics — certainly we would not expect the small errors to continue! What happens when we switch domains suddenly, is that k suddenly increases. A M that can answer questions on both Algebra and Linguistics has a much longer description than one familiar with Algebra only. This sudden increase in k accomodates the expected large errors in a new domain in which only a few questions have been asked.

→ ~~809~~ → 808.00 for more recent version of Appendix B

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

If it take time T to generate and test the function G' that solves the problem, then $\sum T_0 \cdot P(G') \leq T$ and $T_0 \leq T/P(G')$. The total time spent on search is $T_0 + T_0/2 + T_0/4 \cdots \simeq 2T_0$. or $2T/P(G')$

May be best to copy section on L-search from 0.5.5.5, 5.8.6 or 5.8.9!

SN For t. talk, & maybe for paper! Explain that the fact that a single probability distribution is able to examine

First state just what 4.4.18 is able to do! We use a same p.d a_j^0 for all kinds of problems: induction problems, & no matter what t. p.d. is that generated t. A's from t. Q's, the search technique will eventually discover that p.d.

This is not amazing in itself. We just have to be sure that our a_j^0 assigned to t. p.d. that generated t. A's, is not zero. This can be done by any Universal Turing machine or any Universal Algorithm.

That this system works is contingent on its getting an acceptably large z_j^0 for t. p.d. that generated the Q's. If that z_j^0 is very small, the errors in predictions will be very large for any reasonably sized data set.

The Amazing part of the system is that about t. system is that by using a Universal algorithm which describes p.d.'s using function definitions that have been useful for describing past successful solutions to problems, we are able to get acceptably high z_j^0's for unknown future problem solutions. and hence in response to questions

Therefore 2 aspects of good induction (1) large values of 4.4.18 (2) good a_j^0 values (i.e. good UTM). → .37

Getting t. best possible 4.4.18 will not give us good induction unless a_j^0 of soln is large enough. Bob Note 734.10!

It would be nice if I could say .99 – .24 in a very clear, concise way!

There are 2 aspects of t. paper! (1) T. logical correctness of t. ideas.

(2) Writing it in a readable, understandable way.

So back just write. The practice of writing will gradually improve my exposition, if I think about it. Also, maybe Get Alex to make criticism of intelligibility, etc.

The 5.8.9 paper wasn't written badly!

SN T. idea that Search may be ≃ optimum if "all info is in P.D": T. "PD" includes t. set of PST's & Pair assoc. PC's. In this case any Turing machine is acceptable in such a system — so we can (in theory) "put all info into t. PD" "& t. system is potentially optimum"

T. optimization of $(G_{CPD}, [T_{PST}])$ is discussed in 724.00 → 728.10

Start out by writing just as if I were explaining it to a person!

SN Re: .29R : we get (1) large $q_{4.18}$ by lay direct (each time) (2) large a_j^0 by large updates cc.
T. Q of best allocation is "solved" by "50% soln". (I could put this in talk & give "proof" that "50% soln" was not Bad." (But note 738.10!)

727.20

9/6/02

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : [SN] ≈ "Political Remarks":

1) One of the media lab's AI students says that Lenat's CYC does not have English lang. input. One has to translate inputs & outputs from CYC's lang ⇔ English.
For this reason One Guy is not using CYC to get info into his Mach: He's using English & Getting people on web to put info (in English) into it.
It has some Induction Capabilities: The Q is how little induction can it have & still be able to be useful?
One problem: "Donkey" was interpreted by many users to mean "Democrat".
(This is the Party symbol.)

10 2) Another Guy said Selfridge's earliest paper on "Pandemonium" had little specification of how it works. He later made a revised paper that gave much more detail.
Hrr, almost all ~~references~~ citations by others are to the earlier paper!
Does this mean that People were mainly interested in the General idea — not the details?
They had their own ideas on how to implement it or modifications of it.
Or, they may never have read either of the papers, & simply ~~copied~~ cascaded citations.

20 (726.40): (−1.09−.40) Looks rather good! −1.32−.34 Looks like a good beginning for a More exact statement of the "Tentative system being optimum within a factor of 2"

My impression: Updating should always be done before or during the "external problem" (never after). "During" uses info created during problem soln.
"Before", is unable to use info created during problem soln. — Perhaps most Common.
→ Also possibly some update Before, & the rest during the problem

.28 " ~~A tentative Claim~~ The possibility that the system is optimum, within a factor of Two" We discuss the meaning of this claim and why it might be valid. (This involves (−1) 726.4, 32−.34) → (728.19)

.29
30 ↳ Put in Abstract and ~~also~~ in Intro. "In an earlier paper, we claimed that under certain conditions the system was ~~optimum~~ within a factor of two of optimum. We ~~discuss~~ investigate ~~in some detail~~, the meaning of the claim and the extent to which it is valid.
(Maybe Stronger Statement!)

"opponent" is perhaps not rite word!

A "LookAhead" [LA] problem is like a "Game" (usually?) in which the [opponent] is not trying to win!
Occasionally, the opponent will be trying to (lose / win) (in an ≈ zero sum Game)
Most Benign situation: Opponent is trying to lose, and is very smart! (e.g. a Trainer, or Parent) → (728.40)

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00: (Spec 727.40) : Re: LA & Cure Cancer type problems: In "cure cancer", the usual way it is to find a good set of "subgoals" that lead to the top goal. Each subgoal is good if it depends on good or good subgoals. The value of a subgoal is if there are many ways to achieve it — if ~~[illegible]~~ a pontrachy to goal, there are many many PC paths to top goal.

In a Dynamic (Feng) General/exact soln of Core Cancer: There are too many paths to evaluate, search. The no. can be reduced by putting many "outcomes" & "responses" in equiv. class; so one only has a few possys at each "step" of the "LA" problem.

In general: to try to solve "cure cancer" consider how a human would go about doing/planning, such a project: Initial epidem. studies to get ideas as to what's relevant.... (to start off).

At some pt. one must decide if problem is or is not within capabilities of present science (e.g. transmutation (Pb→Au) in Middle Ages) vs. doing it now.

.10 (727.29 726.35) Back to 726.32-.34: To what extent can all hours be discovered by the system and (acquired/used) by the system? (By "System" I mean 726.32-.34 —

The idea of the optim. of (GCPD[PST]) being able to acquire any hour.

I had idea that any hour can be expressed as a PST. Not true!! Most hours are Modifns of PSTs. As such, they should become part of the Grammar of (GCPD[PST]) ≤ (A stochastic/analytic) — i.e. a hour can be an "element" of a "stochastic" "grammar".

.20 The "Big Problem" is how they relates to 729.00-.05 — which is How the Grammar is optimized. How do "good hours" help make the "T. System Grammar" (& ∴ the system), better?

.28 may be the most important possl. Q about the system! ← GCPD is a PD & it has the usual PD Gen.

Can hours be used to improve the "sage" "GCPD"? ← Well, hours are noted in observing PST's at work. They do help predict what "is core" a PST will [illegible] in various problems. Why should a hour that tells which PST's are v.g. in a particular situation, be "better" (more sought after) than a "hour" that tells which PST's are particularly BAD!

Perhaps it's a Basic Problem in the all-over-feedback flow of the entire system! The last stage of GCPD1 → GCPD2 is "Open loop".

(Spec 729.00)

9/7/02

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: (SPac >28.40): "Stage₁" of ~~ydda~~ only trys to Get good (PD's predns) of / ~~Gorcs for various~~ cc's of various PST's for various problems
[illegible] — It is not particularly oriented toward hy Gorcs.

In stage₁, TM constructs this (stoch) Grammar which contains (perhaps) all poss PST's. A heur will modify this Stoch Grammar & ↑ its PC [illegible]. A heur that worries about t. low Gorc region of t. Stoch language space can, non-R. less be useful in helping to reject low Gorc cands. in stage 2.

Perhaps a heur could be [illegible] an entire "New" PST, rather than ways to Modify old ones. Well, any thing that modifies t. stage₁ GCPD would be equivalent

10 to a "Heur" — & giving reasonably hy-pc's to certain PST's w/ low (formerly) PC's — is a modifn. of t. Stoch Grammar.

12 [**T. Q is:** Say TM (discovers / recognizes) a heur in a set of problem solns. Show in detail

13 how this modifies GCPD₁ & GCPD₂ in a way that uses t. heur. as we would (Like) expect it to.]

Perhaps regard a Heur as one kind of Modifn. of GPD₁.

Logical Reasoning Can be part of a Heur. Does this Modify t. Q of .12–.13 in any serious way?

At present, my idea of a heur is rather foggy! Needs [illegible] Reviewing.

20 [Well, as simple example of heur: We have originally a P.D. on PSTᵢ, cc G_{ij}, G_{ij}
We find (heuristic) that if PSTᵢ is restricted in a certain simple way, we get much hyer G_{ij} for ~~[illegible]~~ small C_{ij}. This modifn. of GCPD₁ will produce a more "productive" GCPD₂.

Hvr, ~~[illegible]~~ modifns of GCPD₁ of this sort are made (optimized) w. a diffrnt goal than getting a good GCPD₂, so it would seem that t. optimization of

(Def: note GCPD ≠ GPD) GPD₁ would (lacking this knowledge of what it is to be used for) would not be able to work as efficiently as it would if this knowledge were somehow available in constructing GPD₁. One pt. is that w.o. this (bias in such knowledge) t. construction of a PD, is legitimate.

30 One conceives of / ans typ of Meta heuristic unavailable to this system: It seems likely that ~~that~~ there are certain improvements (∆ (given PC of corpus)) of GPD₁ that would produce very little (but some) betterment of GPD₂, than others —

.33 [So one should concentrate ~~[illegible]~~ ~~[illegible]~~ (in improving GPD₁), not merely on ~~[illegible]~~ max ↑ of pc of corpus per cc expended, but on ↑ its pc in a way that ↑ "utility"

.35 of GPD₂.]

→ There is a danger in .33–.35 of biasing t. search for codes in an illegitimate way!

(230.00 SPac)

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20: 729.40 : Hvr., it would seem that there would have to be Logic ways to work on "parts" of GPD, w.o. introducing undesirable "bias". E.g. say one wanted to expand t. part about "Physics" more than part on "Math". (Bc these 2 are nearly related---perhaps all parts of t. GPD, are (potentially) related. — By at almost any algm. that causes one to work on certain parts of GPD, more than others will necessarily give undesired bias.

There's a More General Q: In making induction codes for any corpus at all: what kinds of models of it are such (essentially Lerch) can one tolerate w.o. excessively harmful biases introduced?

— Consider regression: Say we start w. linear to order 5. Next try quadratic w. up to 5 coeffs. Then cubic to 5 coeffs.... t. choice of "orders" biases search: t. chance of which quad & which cubic terms to use, biases search.

Bias: 1) pr sequence MC probs

A big Q (maybe main Q) is Significance of t. biases: their "Payoffs" — Positive or Negative. IN SM, bias toward use of linear Models is counterproductive because computers use linear Models. IN ANN, except for very elementary models, t. variety of model types ↑ rapidly, so that it becomes less likely that t. "comparison" is usefully so. Similarity w. quadratic, cubic, etc, models.

In using pseudo-Random nos. for Monte Carlo integration, say, we are concerned not nearly w. randomness, but w. "orthogonality" of t. p.r. functs to t. funct. be integrated (whatever that means!) (I think we want Zero cross correlation) In some cases, use of very simple, rather regular "p.r. functs" is adequate.

So, in doing GPD$_1$'s search for regs's" to "max z to 14.18," What kinds of biases can we introduce, that will (a) ↑ GPD$_2$ yield, (b) give errors that ultimately ↓ "yield" of TM,

.35 **Consider** cc=∞ for GPD$_1$: This gets rid of 1 kind of Bias. For this case, all hours have been discovered, & t. best PST's are known. (All discoverable hours (limited by Content of corpus (≪ 5222)) have been discovered).

.38 This would seem to be t. best one could do. So approximately ≤ cc=∞, by getting t. equiv. of "14.18 = Max" would seem to be one way

.40

Actually, t. situation is less informative: All solutions to all solvable problems have been found. Hvr, t. PD case w. a known Sol'n will still be of interest & have accuracy limited by t. convergence criterion

732.23

spec 731.00

9/8/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

See .00: 730.40 to approach this. ~~The fact that~~ in view of the desire for a good GPD_2, we may want to bias the srch. (We may have to bias it: i.e. perhaps all finite CB srchs. must be biased.

In Maxzing 414.18 / w. finite CB or in finite cc time series regressn one also does do biasing. Does this biasing help?

08 Just as in predicting via 414.18: there are 2 aspects of optzn: (726.24, 37)
- (1) Maximizing 414.18 (2) Improving the "reference" Pd. — trying to make a_j^0 (the pc of the "true soln") as hi as possl. ≡ "updating"

superficially, it would seem .08:10 is unreasonable! The "reference P.D." is a priori — fixd! Well consider a seq. of INV. problems: $I_1, I_2 \ldots$.
If we had CB=∞, the problem is trivial: we solve I_1 using the best possl. PST_j; we know what this is, because w. CB=∞, we can actually try all possl. PST's on I_1 and we know how long it takes for each.
However, more realistically: say our CB is limited: we only have a small space of PST's that we consider, and the pd's over that space (as a fnctn. of I_i) are also a smallish space. After solving I_1, we update the pd. on PST's. Even if we
20 had CB = ∞, the P.D. wouldn't be perfect because of our limited PST space and our limited set of PD's considered.

22 Consider induction: Given a finite seq. of Q_iA_i's: We have a priori a p.d. of functions that maps problems (≡ sets of $Q_i A_i$ pairs) into functions O_j that probabilistically map Q's into A's. The problem is Maxzing 414.18 (An OZ problem). We have a set of PST's that look at the problem ~~of~~ (from the apriopd and the set $[Q_i, A_i]$) ~~It is~~ We have a PD on PST's to guide the srch. and try to find solutions.
30 { After each problem is solved, we modify this latter P.D.

There is some confusion in my mind between the apriopd and the pd that I'm "updating". In .22 —.30 it seems clear! The update P.D. is ~~quite~~ on quite difrnt arguments than the apriopd. that is the (part of definition) basis of the induction problem being solved. On the other hand I think of "GPD" as "one big pd. for all induction problems (?)
So this last is confusing. I guess the apriopd on all functions is outside "GPD".
It's a special PD that is used to define induction problems — in fact, if as a conditional input to GPD, we ask for the unconditional apriopd on all functions, we can get the desired apriopd as output! This wouldn't change as TM "Matured".

232.00

9/9/02

~~[illegible]~~ 732

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

731.46
:0: ~~[illegible]~~ : 728.19 ~~[illegible]~~ ~~[illegible]~~ 731.40 is very important stuff, but lots of confusion and Repetition.

728.19 —.30 (along w. 726.32—.35) defines t. problem. ——

730.35 considers CB=∞ & begins to see soln. ~~[illegible]~~ —

.03 730.35 —.40 gives (tentative) soln. ——> SEE 734.00—.09: A Great Simplifn.!

This last way be t. best we can do, but t. manner of our search (w. finite CB) for an approxn. of t. CB=∞ case, will introduce Biases — ~~[illegible]~~ apparently, this is unavoidable so we want to do it as intelligently as possl. — aware of what biases we may be getting & aware of any advantages to final output that we may get.

T. system of ~~[illegible]~~ 726.32—.34 is limited to cands that are ~~independent~~ statistically independent.

For a more advanced system, there is a joint pmpd on all cands. Since this space is of too hy dimensionality, we approximate in various ways.

1) first approxn. is indep cands 2) Next subsets of equivalence classes of cands.

3) ~~[illegible]~~ every pair of cands has a cross correln. (for N cands $\frac{N \times N-1}{2}$ components —

[cross matrix is symmetric

4) There are probly many other ways to express dependence! I worked on this

— within last month or #2 — (try to find.)

.23 : (730.24—.38) RS: We have a PD on PST's. Knowing t. Best pst for each problem type! well, CB=∞ would "know" t. soln to each problem & would be able to assign pst to each problem that would be a.h. for that problem & immediately get ~~[illegible]~~ t. ~~[illegible]~~ best soln.

We usually assume that TM can't do such things.

With finite CB, we usually assume that TM might eventually solve a problem & assign PST's on that basis.

For large ~~[illegible]~~ (but <∞) CB, TM might ~~[illegible]~~ solve as much as possl. in t. "Update time", then leave

So — with available CB: TM may solve as much as possl. in t. "Update time", then leave t. rest of t. work for one or more PST's that work in "prob solving time".

- - - - - - - - - - - - - - - -

In General, t. cl. models that we use, ~~[illegible]~~ none of t. fancy things will/can occur, & we will just get rough models. The CB=∞ model will, hvr, get some v.g. "solutions" at v.g. PST's.

Actually, w. finite CB, we don't want to be trying to approximate CB=∞ behavior, because usually CB=∞ behavior is quite difrnt from that obtained using t. CB's we will have.

.38 We can "take advantage of" t. biasing (of finite CB), by making t. search oriented toward hours. (This is only one kind of regularity in t. corpus). Just how much

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : 732.40 ~~t~~ search should be oriented ("biased") in various directions, must be somehow decided on by "Designer" (?)

03 T. man outstanding Q about TM was 728.19-.30 (& 728.32-.35) (Also see 724.00-.08)

The recent ~~[redacted]~~ work (reviewed on 732.00-03) seems to more or less "solve" it?

Note: 734.00-.09 A Great Simplifn!

Note 732.38 on just how to ~~[redacted]~~ discover/implement "heuristics".

.06 : 724.26 (SW) on Lsrch: In the time share Method 724.17, in which the Guiding PD is changing during the Lsrch. We may have to use the "Random" method of Lsrch of 724.12, because of the very little info about GPD_2 is obtained. If so, we could just continue our Bayesian Random Lsrch as "normal" — even tho the GPD_2 is changing during it such. This is not as "efficient" as 724.17-.26 because we waste much time working on Cands that already have too much work done on them. Just how inefficient it is .... I don't know. I may be able to ~~[redacted]~~ estimate, hvr.

A (Guess) stabe in the dark): This may be no worse than 2 times as slow as doing it the "correct" way.

An Approach to Analysis: Say we have a certain ⇐ (state) p.d. for 1 second & we do Random Lsrch for 1 second. (or deterministic Lsrch for 1 sec.) Then the PD suddenly changes (for 1 second). We can either just continue random or Lsrch w/ the new Pd. Or use the "better" method of 724.17-.26

How much "better" is the better method?

If the PC's change very slowly, I suspect that there is little difference. The "wastage" occurs when we work on Cands that already have now have too much work done on them --- whose PC's have recently fr.

.24 (perhaps) better way to do the random Lsrch! When a Cand is chosen, we look at how long ago it was chosen last. (This gives our estimate of present PC) We can next see the total cc spent thus far on that Cand. We also know total cc spent on all Cands thus far so we can estimate whether this Cand has "too much cc" thus far, & skip it if it's "too far"

[We also may note in general recent times at which this Cand was "hit" — to perhaps get a better PC estimate. Say last 4 or 5 accesses: (or fewer when we've just started on a problem.)

.24ff could be better than "normal" random Lsrch. We can adjust the amount of cc we use to "bring the Cand up to standard" — so sometimes, it may have less variance than the "normal" method, even if the GPD is static! At any rate, .24 looks like a v.g. way to do Lsrch!

It would seem that impt. stuff dropped since 728.00:

724.35-726.05: Exp. on Lsrch → Probly for Paper (at last, MX version).
726.06-.36 on the Table — Theory: Why Lsrch may be "optimum": also 727.25; 728.19 : 732.00-.03 Bibliographic summary of Short Stuff and 733.03
726.37 : Discussion of optimality of 414.18 w.r.t. maximum of a_j^0.
Actually: 2 aspects of optimality of 414.18
N.B 734.00-.09!

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: : All Yes | In Sol 86, 89; I feel that some hours (but not all --- Not Quite about say, or Irrp. during Lsrch) could be implemented by reordering trials.

More recently, the idea that modifying, inventing PST's could implement any hour, occurred.

Latest idea: That modifying the d.f. over PST's (first) automatically includes all possible PST's:

So any "invention" of a PST simply consists of assigning a reasonably low pc to that "invented" PST. This means that modifying the P.D. can do 2 things

1) Modify orderings on old PSTs 2) Introduce ("invent") new PST's.

So my old idea, that all hours were expressible as modifications of the P.D. was literally correct!

ag 726.24
PD: 726.24R
726.37

SN Comments re "option of 4.4.18". In 4.4.18, a_j^o is fixed, for a given problem. When we change problems for 4.4.18, the "j" of a_j^o may change, but all of the a_j^o remain the same: This is the Apriori.

On the other hand, consider the P.D. assoc w. Lsrch. for a PST for a given problem. It w. vary w. TM's "Experience." So how does the P.D. for 4.4.18 differ from the PD used in Lsrch? Well: They are quite distinct PD's: The one used for 4.4.18 is ~~another~~ an apriori Part of GPD1. The one used for 4.4.18 induction is GPD2

There is a way for us to change the "apriori" for 4.4.18, however — That is via

20 The "Summary Machine." — In which the new "apriori" is a summary of the corpus up to now. Strictly speaking, we can't change the Summary "apriori", The only way to change it is to "backtrack" — which is equivalent to not really reconsidering the summary machine as a complete Summary. Equiv. to some degree of "Backtracking" would be to retain in Memory, a fair no. of non-optimum codes for the corpus, that are "Significantly distant" from the Apparently "Best" codes. 40y

30 One trouble w. "Summary Machines" [redacted] is that they are (usually) not Universal d.f.s. i.e. we can construct them by fitting a small space of functions to the past Data. This makes "Backtracking" messy and Expensive.

SN I discovered (Twice) that 11 Lsrch is much more than 2 thousand times faster than T<2TLsrch. For decision length n, it is ~ 2n times faster (I Think). The inefficiency of T<2T is that In decoding 0 strings a0, a1, it investigates a0 & a1 indiply. 11 Lsrch decodes a, then continues w "0" & also with "1". Both items share the coding of "a". (How this would work out using random 11 Lsrch, is unclear.)

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : An approach to writing this paper: [Cover "Outline" of 696.15–.40 : Make progressively more detailed outlines. Indicate what parts have been written (or discussed) and where.
One advantage of this: I'll be more certain to keep the whole system in my mind.
So I don't forget. [Probly D:\RAY.

So from 696.15 : Introdn: — I have versions of this in "London intro" R-AY directory

.07 §I QA Problems: [CPM Nov 8, 2001 : The part thru dern as a OZ problem
Also, I have Appendix w. Conv. Param.] → see 722.16–.17 for the modifn. of last part of CPM N801
[701.23 suggests that I explain how Numeric T.S., bag induction, Ency. can be covered
10 by this formalism]
719.19 – .33 should be added to → maybe side 719.34–.40

Appendix B: 720.10–722.15

(T. forg. should be: ≡ 722.25–.28 So that roughly covers the QA problem.

"Introductory Section" PSTs in general — dern. of various sub-sections
.15 § II PST's : Lovin & Morrone outline
a) Derb Lsrch : Perhaps ~ to Sol 86 : Give 2 or 3 GHT's and TEKT T<2T method.
Perhaps Mention "T. First Scaling" problem — or perhaps have section on "Scaling".
Discuss Optimality "If all info is in P.D.". 2 aspects of how P.D. can express hours!
i: reordering of trials, ii: creation of new (PST's). by giving more pcs to PST's
that had ≈ 0 pc before. How this sort of implies Lsrch could be optimum.
Discussing of Zound. srch Algms.
20 Discuss Updating after each dealing. → spec 736.00)
21 (How) Can we put "all info in PD"?

.22 Some ideas to write Expo. papers about:

(1) Relation of my system to "CYC".
My system does "CYC" as one mode of prob. solving.

(2) How having a "single universal" d.f. is not desirable — that it would be of no value.
(The historical evolution of t. bias/g ...) Apnpd
— That one normally uses "summary machines" not "simple" umcs.
Also, that it does make lots of difference as to what reference Umc you
30 use. This fact was not noticed by Math community because they were interested
in potentially infinite ssz. I'm interested in small ssz, (0, 1, 2, 3).

(3) No free Lunch Theorems: That the "Princ. of Indif." "Uniform d.f." is usually meaningless, since
deciding on what a single string should represent has to be decided by user.
I use "uniformity" over umc. input for my "Prize of merit". There is still
choice of how to map output strings to R.N.

(4) How to improve GA (1) continual improvement of mut/crossover (2) realization that mut, cross are just 1 & 2 ssz cases of induction. (3) Ability to process (part of (1)) input problem — to suggest initial population & appropriate mut/cross.

9/16/02

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:(735 Spec, 21): At this point we write discuss the ≈ optimality of 50% soln. (Certain factor of 2" : no "proof")

Maybe also discuss alternative update invocation times. —— But they would probably not be much better than a certain fixed factor (of update cc.) for each problem

---

[SN] : "Proof" that ‖ Lsrch is n times as fast as T←2T, for soln. of length n:

For ‖ srch: no of bits processed: $2^0+2^1+2^2 \cdots 2^n = 2 \cdot 2^n$

For 2←2T " " " " " $= 2 \cdot 2^n \cdot n$ (2^n strings: each of length n)

Then discuss ‖ Lsrch & Mt. Carlo Lsrch: Discuss speed of ‖ Lsrch (but more than 10 or 20 times as fast as T←2T

Then advantage of MC method. Tho: just how is it done? — when there

Well, I was thinking of using it for McCarlo srch over PST's. Each PST has a pc (in Mt Carlo form) of being "the best" one.

Discuss Lsrch for OZ problems: Rem. perhaps discuss Random (McC) Lsrch.

(Tho it occurs in soln of INV probs as well: so may be introduce it here

---

Discuss updating continuously during Lsrch: several ways: 733△6 ff; 733:24 ff in particular

This last one for Mt. Carlo Lsrch.

[N.B. the latter part of this outline has to be "reordered" — its a bit messy.]

[SN] On the time varying OZ problem: G(x)·F(T) = max:

.25 Say A(G(·), F(·)) → x_0, T_0 is a soln. Then Lsrch for A will yield a soln

.26 for $G(x) \cdot 2^{\ell} F(T)$ that is as good as A's (where ℓ is the length of A's dcm).

This soln is not, of course, of much use; but the usual Lsrch solns aren't, either!

As with the "usual" Lsrch solns for regular OZ, INV probs, they are useful for Adaptive Lsrch. — Tho, considering the present problem, ℓ = φ so the soln at .26 is not "acceptable". My most recent soln. of .25 using the general form of time varying Lsrch (which works identically for INV as well) does seem to obtain a kind of "acceptable soln.

~~Soln. is far~~

Other methods of search (724.28 – .34) — This doesn't work: Work on Best cand. w. updating in ‖ until "best cand" changes, then jump to new "Best cand". [Always work on task w. "Min Estimated Remaining Time to Completion". — Min ETC / ERTC] ETC is "like ETA of airlines: Arrival times & varies during flight.

!? — Discuss Non-optimality of Lsrch.

(2) The WON soln. (OR nets).

(3) Methods discovered thru the "general GPD "Grammar": A update that includes discovery of "new" PSTS as well as of modifying pc's of "older PST's".

→ 737.00

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : 736.40 : § III GCPD : 704.28-.34; 723.31

These discussions are rather brief. I might add in some ways to realize

Dad a C.P.D. (cond). (1) I. 3 input CPD (This is easily made into a (1.5) MCCarlo CPD)

.Dad (2) & output is list of (Pc, A) pairs in ~ Pc order.

---

10 Intro to talk : (Maybe ~~intro to~~ start of intro of paper & Abstract)

. I will am going to talk about a system that I've been working on for latest

. developments on a system that I've been working on for ~~quite~~ some

13 time (Sol 86-88, Sol 89) [It is ~~meant to be~~ By. meant to be an approach to strong A.I. —

.135

a system that will surpass man in many areas of science ~~and~~ mathematics, by .285 to outset .30

. 15 . and perhaps art. It differs from ~~most~~ previous work in this area, in that

16 I am not ~~particularly~~ so much interested in knowledge itself, as in how

17 it is acquired — how machines may learn. .176 To start off, the system will

learn to solve two very general kinds of problems. Most, but possibly not all,

20 problems in Science and Mathematics can be expressed in these forms.

:

:

The second kind . . . Be Devising a theory to fit data, surface reconstruction,

in doctrine inference of all kinds, and a few curve fitting . . .

Perhaps don't get into "Strong A.I." — This disturbs many people.

28 ^ Scientists assistant." / Re start at .135 : " It is meant to be a "Scientists

Assistant" of great power and versatility in many areas of Science and Mathematics.

30 It differs from much other ambitious work in this area, in that" (Then start at bugging at .16)

→ perhaps have more in this direction in intro.

.176 .... " To start off the system has very little knowledge, other than algorithms for learning, and a very compact probabilistic model of the world.

"To start off, the system has a very simple probabilistic model of the world, and a very general algorithms for learning. It then learns to solve a stream of sequence of two very general types of problems. Most but possibly not all,

9/16/02 ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : [SN] Unclear in my mind as to exactly how system works. Un all → this may do it:
Say a problem comes in: It is either a INV or OZ problem, (if it's a IND problem, it has to be first converted to OZ). GPD_2 immediately gives it a PD on PST's, and we do eventually Lsrch.; then update GPD. As TM matures we modify Lsrch by updating during Lsrch & by actually not doing Lsrch any more — the updating is done
A Better way: Prob. comes in: we update relevant part of GPD, & then we use & assoc GPD_2
Lsrch to solve the problem. → (31)

.04

09 [SN] On the proof of conv. then for Bags/QA! Perhaps I don't need as much
0 "new stuff" — Gac's proof for conv. of semi measures might be enuf!

.11 Say I consider $P(U|K_1)$ to include not only "U defined" but stopping.
Well, this may not help: we do need the stop symbol — even if the machine does "U"
at end of a "A_i", this is 0, 1, but we need to know its not 0, or 1.
If we have the string $A_1 A_2 \cdots A_n$ concatenated w.o. spaces between them, then there
is spurious pc of a machine producing, say "$A_1 A_2$" as output. The space at the
end of an A_j is an essential part of the output.

20 O.K. .... So use Gac's' system w. (.11) a "U" comes up in the code string "occasionally"
We still have $E \leq \sum$ (prob y 0, error^2 + (prob y later$)^2$ $< k$.
We use Gacs' proof for 0 & 1 only: we don't have to add in errors for U

24 (A trouble w. my "proof" is that just as I showed pc^2 of $M \to 0$ rapidly,
This would show that pc^2 of (U ~~and~~) $\to 0$; which is false!
There may be a way to get around .24 — having "and" occur many times in the past
may modify the conclusion in the conv. theorem!
But don't spend time on this now. The present proof is adequate for time being.

30

31: (.08) I should get this clear in my Mind! (I.E. [Just How TM Operates])
One way of thinking about it: As soon as problem comes in, TM notes if its a INV or OZ problem.
(if its a IND problem, it is converted to OZ by an Input pgm.). Perhaps at this point a GPD_2
distribution is created from GPD_1. [redacted]

36 → Another way: I just do ~ Lsrch every time! We update first (both GPD_1 & GPD_2) -
The nature of update depends on progress in that area of prob solving & in TM in general.
① first no update ② second, update now w. Simple corpus of successful searches (GPD_1 only) ← ? Not quite! Its a special corpus that used by GPD_1
③ GPD_1, GPD_2 but no correlation ④ $GPD_{1,2}$ at end of $T \leftarrow 2T$, look for correln.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00: 738.40 : (5) GPD$_1$, GPD$_2$ more frequently — perhaps during Lsrch (timeshar) (6) Update is watching current trial & so uses "correlations" to decide on next trial. Use of random Lsrch — possibly w. "tricks" of 733.24ff
(7) No Lsrch, update is always simult. w. trials. Jump to new trial when "expected remaining completion time" (ERCT) of new found trial is less than ERCT of present trial (seconds ≡ (6)!)
.03

— 1.36 this time, but I need more of a detaild idea of what "PST's" are being searched. In (1) T. [PST] sets t. apriori given by t. Lispish language. In (2) we have PST's use induction there to begin induction on them to modify "Guiding PD". In (3) Our induction facility must be good enough to get reasonable P.D.'s of cc as a function of problem during PST$_j$ (so do this before we start each problem — since we have to know what t. problem is (3 !).
10 In (4) thru (7) we need progressively better update facilities — our induction on past problems must be progressively better. → see 767.36ff for how to learn new PST's & how to deal w. (.18-.21)

Re: situations 1,2;...,7: ✗(—1.36 - 739.03) Could I devise examples of problem types & of skills that TM has aquired?
.17

18 At one time in t. past, I considered adding (ad hoc), a bunch of PSTS that I felt were v.g. — [struck] put them in "factored" form, a/o have TM
20 "factor" them & devise a stoch lang. using them as corpus.
Would this fit into larger schema — if so, how? → 741.00 spec.
.21

22 Another idea that I'd like to fit into t. large schema — t. use of "context" in constructing trial induction functions. Certain kinds of "context" are "normal" regularitys in a corpus — its a way of defining a type of "Bern. seq." ...
If something is in several difnt contexts, we can use that trick of mixing non-predictors.
wt. of each context = dom(cost) × (SSZ)**. T. pc of dom may perhaps be done by t. no. of legal contexts at that pt. that have been defined thus far.
30 My original motivation for "Context" was that w.o. it, t. pc's of concs become very small, due to t. number of them (as corpus → ∞). "Context" implies small section of corpus being used for pred. It means small SSZ, but perhaps pc's closer to ϕ's, since t. contexts are defined only when they ↑ pc of entire corpus.
→ For IND problems, this "context" will amount to a modifn. of how t. apriori is defined — It is something that can be "added in" later on, as TM develops. Normally t. pc of a conc. depends only on its past frequency. T. ability to define contexts gives a difnt apriori, just as t. ability to define new functions modified t. apriori.

→ 741.00

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

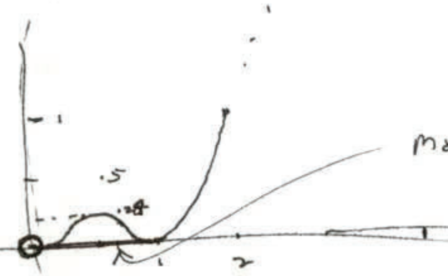
.00 : For t. aprpd ~~of~~ of vals from $-\infty$ to $+\infty$: If one is considering values of coefts for linear (or n.l.) regressn. — ~~T. approp. posits~~

We expect ⇓ from 0 as x↑, T. first "corner" is an impt param: say it = 1 for coefs. since we expect ~~coefts~~ linear coefs to be ~1. (Non-linear may be difrnt — may depend on rms value of signal/coefs).

The part from 1 to +∞ can be like $1/(x(\ln x)^2)$.

$\sqrt{x}\cdot\ln x \to 0$ as $x \to 0$

x	$x(\ln^2 x)$ ↓
0	0
.5	.24
1	0
2	.96
4	7.7
8	34.6



so $\left(1+x\cdot(\ln x)^2\right)^{-1}$ would not be bad.

Maybe may it ϕ for $0<x<1$

Another Q is rate of ⇓ for large x. This determines values for $-1 \lesssim x \lesssim +1$

so int $(0, \infty, x\cdot(\ln(x))^\wedge 2) = ?$

int $((x(\ln x)^2)^{-1}) = -\frac{1}{\ln(x)}$ $\int_0^\infty (x(\ln x)^2)^{-1} dx = -\frac{1}{\ln x}\Big|_0^\infty = 0$ unlikely! integrand is always ≥0.

oh! Integrand has poles at 0, 1, ∞. ← cancels out. ← This prob'ly

int $\int_0^\infty (1+x(\ln x)^2)^{-1} = 2.804281080$ so $(\ )^{-1}$ is t. normz cons ($\approx \frac{1}{e}$) ≈ .3566

So $\int_0^1$ norm. funct. : ≈ .3566 ; $\int_1^\infty$ normal funct ≈ .6434 } took Maple awhile to do this $\int$.

$\cdot f(x) = \frac{1}{x\ln x(\ln\ln x)^2}$ $f' = \frac{1}{(x\ln x(\ln\ln x)^2)^2}\cdot\left(\frac{1}{f}\right)'$.

$\int_0^\infty \frac{dx}{1+x\ln x(\ln\ln x)^2} =$ [imaginary for 0<x<1] [pole at $x=1,0$] [pole at ϕ ← but its negative!] ~~Error~~ It didn't finish this calcn! ~~Better~~ try a $\sum_0^{10000}$ example. does $x\ln x(\ln\ln x)$ ever $= -1$? For $0<x<1$ $\ln\ln x$ is imaginary!

It is my impression that "Riss" ~~universal~~ $2^{\log^* x}$ has lots of wt. in hy x values — so much less on [0,1] interval.

For non-linear (poly) regress'n: t. coeff of x^2 should be ~~multiply~~ ~~...~~ divided by t. ~~...~~ rms value of x. coeff of x^n divided by $(\text{rms of } x)^{n-1}$. After division this way use a prior of .00 ff ($\propto \frac{1}{1+x\ln x(\ln\ln x)^2}$).

→ 764.00
765.00

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: (Spar 739.21): IT should be possl. to "add in" PST's. There seem to be 2 models for [PST] set:

(1) the set of all possl. PST's, i.e. a universal def. on them given by a universal algm.

(2) A finite set of PST's, initially inserted by designer/trainer.

We can get start w. (1) and "add in" a fixed set of PST's: The wts of the new PSTs will have to be estimated relative to the PSTS in (1). The Σ of all apriors need not be 1 — we automatically renormalize w. random or deterministic (LS) such.

How the "update" would work, is unclear. One way would be to keep the 2 sets of PST's separate & update them separately. — This is certainly OK in the spirit of "Universal Transfer Learning".

The relative wts of the 2 sets of pst's could be adjusted so as to maximize pc of corpus. (?)

On the other hand, we have 2 kinds of objects: The O_i^j of 4.19.18, & the PST's. The O_i^j's are supposed to be for the entire corpus. The PST's, are for specific, single problems.

It is possl. that we are looking for a single PST that is same for all problems Env, OZ & IND — like ENV probs had a single search for PST.

In 739.18ff, the set of PST + the factors → a (Universal) stoch grammar. . . .

In the idea of GPD giving a PDon the PST's: There is a redundancy here, since we will "eventually" need only 1 pst, — but we've elected the problem by breaking it down into 2 parts — i.e. GPD & [PST].

{ We can frame the present system either way: as looking for one PST for all problems or a GPD that assigns a difnt PST to each new problem.

We can do some kinds of problems (say IND) one way & 2 (ENV & other OZ probs) the other way: Look into this.

Probably it would be good to look at each "pure" system. I've perhaps the most of 2 GPD on (problems, PST) rather than on PST index (or average over all) problems.

A PST chosen w.o. consideration of problem, must itself look at the problem & categorize the problem to decide what method to use to solve it.

In the original system, the GPD looks at problem descr, so the PST has to more narrowly recategorize the problem for solution. There is less intellectual "work" that need be done by the PST.

9/19/02 [TSQ's].00 continues to 745.23
ps

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

741.40
90: [redacted] : Consider algebra: simple MTM induction: Maybe of 414.18.
T. way we do it is just look for standard "regularity" in t. corpus.
At one run in ANL, I had "3+7=10" Q=(3+7=); A=(10).
.03 3 & 7 are known to be numbers. There are "sum, sub, mul, div" as binary operators. "neg" is unary.
[illegible] "+ & =" are "unknown" symbols. 10 is a numeric output, so we first look for unary
funcs of 3 & 7: none work: Binary funct of 3 & 7: "sum" works: Sum 3,7 & Sum 7,3.
.06, So those solve it. Hvr, there is "reasoning" being used, & TM doesn't know how
to do "that sort of thing". Until TM learns to do that kind of reasoning, it is
"Forbidden".

0 Compute PC of Sum 3,7 (7,3)! precorpus 3, +, 7, =, sum, sub, mul, div, neg, 1, 0.
11 symbols: Sum 3,7 has PC = $\frac{1}{11} \cdot \frac{1}{12} \cdot \frac{1}{13}$ = $\frac{1}{1716}$
since there are 2 ways, total PC of both is $\frac{2}{1716} = \frac{1}{858}$

But It has to do "8+5=13" also:

well, we can let 3, +, 7, = be t. first, second ... fourth symbols in t. Q.
In which case it works for any a+b = eval(a+b), but t. result is critically dependent on just
where symbols are placed. It wouldn't be able to gauge [illegible].
i.e. Q = [redacted] "blank 3+7=" : 5 symbols in Q.

20 Hvr, t. idea of translational invariance would have to be lrnd. No way around it —
unless we build it into TM.

[9/20/02] Another way to think about this: T. general heurs implemented by Lenat in "AM".
Th. Q was, how well they fit together in t. "learnability" of new cncs / heurs.
A possl. way to do this: to "Give" TM t. idea of translational invariance (or fix it so that
xlatl-invariance is not essential (?). — tho recognizing a "a+b" configuration does
involve t. "xlatl invariance" concept.
Also give it "reasoning" cncs needed for .03 – .06 ; .03 – .06 is a "working Backward" —
a common Heur. [we need numerical result, "10" & resl. 10: to get it we need to use funct that has (10 num) as output: list possys. Try simplest possl. inputs to start. [if they don't work,
40 try another (deeper) level of functions.]

31 My Idea is that "all info can be in GPD" — so that if Lsrch can't find a
Soln. in reasonable cc, it's because there was import. info missing from
t. GPD. T. main simple ways of putting info into GPD:
1) Defns. of functs.
2) Defns. of "Contexts" for functions
3) Some I've not thot of.

Very imp. idea re! TSQ Design!

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20: 742.40 ! One way to implement TSQ's using Y. (Nacy?) ideas of 742.31-40!
Establish various "state pts" along a tsq. toward some goal. This is ≈ like the idea of conc. mat., but instead, we write more ordered sort of state pts. — Then we try to bridge ("concatenate") betw. them.

If it proves difft. to implement, e.g., then the language expressing GPD has to (be modified/fixed/replaced).

SN An impt kind of PD that TM does is Bam Seq: defined by a "alphabet" of objects, a no. of times it has occurred, & no. of opportunities for its occurrence — Bayesian.

A list of "state pts"

1) doing 3+5; Then 8-7 ... Finding obvious problem (Bam Seq) model

2) Then 7×8, 4÷16 " " " " .

3) Identifying "relevant context" so that "+" is recognized as → sum, act.

4) Being able to do parsen-unstead expressions.

5) Maybe being able to do Alg expressions w. some parens missing — using hierarchy of functions

1.5 For 1→2 it might be useful to teach Bam first — say to learn to predict freq. of a, v.s. b; then a v.s. b, v.s. c, act. We might make this a "builtin" conc. — So TM would automatically look for duplications of symbol seqs, so it could define Bam Seqs.

How to do (2) (Bam seq. of 3+7, act) → (3) realizn of ...? (earrolns?).

We have essentially a new seq of QA's:

Q	A
3+7=	sum(R1, R3). / 8+9=: sum(,)
9×3=:	mul(R1, R3)

From this, it should be not hard to get "+" → sum; "×" → mul, but the "details" have to be worked out.

29 From (3) $3\left(\frac{x}{+}\right)4$ to (4) (nested expressions): is not apparent.

30 ∝ # A possl. way: it first learns 3+5=8; then 8÷2=4 then (3+5)÷=4.

32 B a/o say it learns a=3+7; then a=10 then a+2=12
(3?) i.e. T. concept of "Equality".

36 ..... from (29): an earlier approach! teach Meaning of "eval": eval(3+7): 10

37 Then eval((3+4)*7) = eval(eval(3+4)*7)

SN Another way to deal w. ANL: Have it be "builtin": i.e. it's just part of our way of communicating w. T.M.! Nvr., I wanted "ANL" as an example of a simple Learning problem. Perhaps it's not a good example! Actually I could delay the decisions as to whether to start w. ANL, by working at higher level. More advanced Arb Intg. Problems.

9/4/[redacted]
ID

GA.00 Theory of how mut/cross dra SSZ=1,2.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: : [redacted] SN GA: "Mutation" ≡ induction w. SSZ=1 ;". How to "do it rite" : from [redacted] cand, X_0, G_0 to be mutated: Construct P.D. on X_i, G_i X_i is cand derivd, G_i is its possl. G.value. chose $x_i \ni G_i$ is Max.

For crossover: Given (X_0, G_0), (X_1, G_1) again construct D.F. $[X_i, G_i, P_i]$ ($\overline{\Xi} \, P_c$)

Chose $x_j \ni E(P_j)$ is max.

T. above choices are for Max Greed. Modifyable if t. D.F.s take "correlns" into account — i.e. is [redacted] Joint PD of all possl. contiguous t. t. set of $[X_i, G_i]$

10 (Spec 743.40): In General, I should be able to state each heur I use/need for a particular problem soln., Then (express those heurs as Modifs. of GPD and/or as Tsq's that would bring about that GPD modifn)

T. clear statement of t. Heur, is perhaps t. hardest part.

15 in ~~[illegible]~~ t. problem of −1.29 (≈ substitution): Substitution is a rather complex ~~[illegible]~~ conc!

16 It may be one of t. primitive functs that Kleene ("Math Mathematics") uses to define a universal Algm.

⊕ Perhaps I can teach $3+(9\times5) \to 3 + \text{eval}(9\times5)$, & similar examples..

A [redacted] (Risky) [redacted] way to teach TM: Have it lrn rather ~~[illegible]~~ to solve rather diff't

20 probs., [redacted] by inserting heurs that humans seem to use to solve those problems. Then, perhaps TM would be in a position to discover more complex heurs

This is ~~[illegible]~~ ≈ approach of Lenat in [redacted] Eurisko, but we know more about (sample size/ nature) of examples and amt of cc. needed to find t. Heurs.

It's risky because it's ~~much~~ harder to tell if you have missed some (perhaps small) but essential concs.

In line w. −1.32−.37 : In English: "Eval of a string, is invariant if we (substitute / replace) part of the string by its "eval"". We need t. conc "Substitution".

Maybe make it primitive (.15−.16)!

30 One way to do −1.29 (nested expression evaln.) Make a set of "primitive" functs, that are adequate to do it. Try to make t. "primitives" factorable into more "basic" functs that are likely to be used elsewhere.

34 In general, this can guide a TSQ design pgm! For each "state pt." in t. TSQ, we write a soln., then try to factor t. solns. into a

38 smaller set of "primitives".

This idea plus t. Learning of various "Definitions" by TM, might be an adequate, not-to-diff't way to write TSQs.

I may want to learn "Lisp" : or devise a lang of equiv. power.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : 744.40 : Try to do -1.34-.36 on ① 743.10-.20 then ② learning to solve linear eqns &
.01 found N.L. eqns. ② quadratic, cubic eqns.

.03 → For Good TSQ: Look at old AI pgms like Stygles symbolic integration pgms.
.03 Express it as a seq. of "state points".

.06 Essentially -1.34-.36 is the way to write TSQ's (at least the Mechanics — the general path — the "goals" of the TSQ are another story!). I have to decrb. each "state pt." w. complete exactness — enuf to be able to express it as a well-defined function. If I can't do this, then I don't understand that part of the induction. In general, I may be able to write a soln. function for a problem & give TQ, but that soln may be of the excessive CJS! It has to be decrd: If I can't factorize, then I don't understand that particular induction.

The language I use to decrb. my solns, "factorizations": This I can invent. — decrbe suitable "primatives" as needed (like "substitution" -1.15, 743.29).

I'd like to spend more time on this now, but I have to get back to the paper.

TSQ So: Main ideas: .06-.16 : .00-.01 for initial TSQ's & .02-.03 for more
.17 advanced stuff.

When I get these tsq's & assoc. seq. of functions, I can tailor Learning Algms. to fit into this particular mode of TSQ construction.

The present Trace on TSQ's starts 742.00: Read this carefully now 745.89
.23 Re-read & till it seems clearer than write TSQ. In particular, Note 742.31-.40 ("all in Polish GPD").

→ A good way to write TSQ's (for both TM's & Mg), would be to make a very large array of probs & state pts. for the TSQ. I would then gradually fill it in. As soon as I had what looked like an acceptable path from one pt. to another, I could compute r. CJS. (Not known ideas @).

.30 To start, the state pts would be in fuzzy English — then → exact equations & the TSQ Array in forward/moved toward completeness.

Keep things in "English" as long as poss. ... it makes "adjustments" easier

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20! : [SN] Since any havr can be expressed in t. GPD, we want to "improve" ("optimize") t. GPD. Can we do this via a trick corresponding to $GPD_1 \to GPD_2$?

We have t. infinite set of all GPD_1's (I guess GPD). They correspond to t. set of all PST's. Then we have some kind of Gorc over GPD's. — we want a PF of t. Gorc over all GPD's. Then we get a PD over which GPD will be "Best" wrt that Gorc. (This will be implemented in t. usual Mt. Carlo method of $GPD_1 \to GPD_2$).

What is t. Gorc of a GPD? We'd like it to be for future problems. [This Q may be related to t. "TM$_2$'s Gorc" problem].

0 [SN] t. discn of .00 is not so "Blue sky" — while it is probly not practical to try to optimize GPD in any very Global sense, still, understanding what t. best method is, we'll enable me to evaluate various approximate ways to improve GPD.

We could optimize wrt t. past corpus of problems. This is mindful of t. "Action Algm Evolution" problem! Th. Gorc for t. QA problem automatically looks into t. future somewhat, because of t. Convergence than working for arbty future Q's! Hvr, this is t. common situation in which "Action evaluation" is easy if t. "Action" is pc evaln.!

Would it be useful to compute t. Gorc of a GPD wrt a particular problem? → See +1.19

20 — I have some Earlier Discussns of t. GPD (including both [PST'] & plis of) & perhaps how to optz it. (in the last month perhaps!) — or 2 weeks!

Drop this for awhile!

24 Look at 739.22 — 739.40; 744.00 → 741.40 }

25 also 738.31 ff

I looked at ≈ .24–.25 refs: Q: Just how do I do "ANL" in that framework?

.28 — Its an IND problem! OZ: I need to Make a certain expression. So I try all poss w. that express as inputs & t. desired optzn. as output.

.29 — w. that express as inputs & t. desired optzn. as output. — also as cc f was

30 — [So t. input is [$Q_i A_i$]," we want, as output a O_j that is v.g. — also as cc I was want successive O_j's of hyper Gorc.

Because of t. unique structure of INDuction, we can get a reasonable attempted soln. by trying O_j's in Least order. — This is easier than looking for good functions to map from [QA] to O_j. (It does not, hvr, have t. possy of ever being ≈ optimal. → So I may want to "look into" This!

Also, using tricks like form induction and "outcast" (738.00) we can get "not bad" induction. (739.22 seems more relevant!)

.36 So we can go pretty far w. these QA problems (& any other induction problems) wo. using t. $GPD_1 \to GPD_2$ device.

9/28/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

Base 386 | 389, 90, 91 ok. 392 marginal.

20.746.40 : From -1.36-.40 we get early, not bad, soln's to QA [redacted] & other induction problems. To improve it, The Lsrch method of -1.28-.29 is what I want.

Actually, -1.36-.40 is one PST — it's gore ↑ w. cc. — So we can try others, but that one will (at first) get almost all available cc., since it have much better track record than any other PST. (i.e. "other PST's" are rote listings of codes for poss O_j's in Lcost order)

.08 From t. operation of -1.36-.40, we will get a lot of ([QA], O_j, cc_j, G_j) quads.

From that corpus, we mite get a d.f. on quads like that for any new [QA] corpus

(typically, t. [QA]'s of diffrnt quads will be "nested" so each is a "prefix" of one of t. others.

Still, this is not a D.f. on Mappings of [QA] → Q_j.)

One idea was to put in "by hand" a lot of good PST's (usually optimizing techniques) & I factor t. opt. techniques & help TM make a stoch grammar for them — than TM can use t. resultant PD for real Lsrch over PSTs (≡ OT's) = (optzn techniques)

O.k., so I have this ≡ GPD stoch Grammar on OT's: I want to improve t. GPD. How do I do this? —

.19 What is its Gore? Well: I don't do it that way! I do it by GPD, & GPD_2! GPD_1

.20 hast. gore of any PD i.e. max(pc. approx.), & GPD_2 is simply a xform of GPD_1.

.21 Go over this again, Carefully! So, t. Gore is: That t. Stoch. Grammar

.22 expressing t. pd on [QA], ... (.08) has as hi a pc as poss! That Grammar

.23 is GPD_1: than I get GPD_2 in t. usual, Mt Carlo, way.

WOOPS! .26

Q: Would .21-.23 be helpful in t. Action Alam. Evaln. Problem?

.26 → In .21-.22 t. SGrammar does not give a pc to .08!

T. (ass elements) of t. Grammar are t. PSTs — Well no; t. ass's are (PST, problem, G of soln, cc)$_i$'s

T. defn. of Gore is then unclear!

.30 There is some Q I have about whether t. GPD_2 is on (PST's alone) or (PST, problem) pairs.

Try it both ways! Well, It's GPD_1 that we "test" — that we "Evaluate".

Defs. M_1 is t. GPD is for arby prob I.e. "i" in M_1 means 1 argument.

M_2 " " (PST, specific problem) pair.

For M_1, Given cc, GPD_1 gives

.37 SN M_2 sounds "redundant," but it may not be! That M_1 is theoretically possl., does not neccly make it t. "best way to go". M_2 can "w. (without) away" but that doesn't mean its best to start w. M_1!

I think I will use M_2 to start.

9/28/02

S.P.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: : So Do M_2 first (M_1 may not be possl.).

01 So: Need to stack Grammar have as ass: (problem$_j$, PST_j, cc_k, G_k)

(PST_i . problem$_j$, G_k) I guess cc is included in prob dern — but it would be well to have cc as separate arguments since we have, empirically, > 1 solve for each problem.

From . Then from .01 to stch Grammar is GPD, .

While M_2 may indeed, "whither away": M_1 become to essentially single PST, [it could well be that the "system as a whole" will remain factored into the $GPD_{1,2}$ part & [PST_i] part. (Likely)?] Likely
[9/5/02: See 761.00;10: At (near) end, TM will only have $GPD_{1,2}$ that includes all (Parenthetically) PST's]

10 One final stochastic Stack/Grammar on the [PST] is a GPD_2. (for both M_1 & M_2 — (I think))

11 But ∃ also the assoc GPD_1 grammar. In the M_2 case, there is a different GPD_1 grammar for each (prob dern, cc i)

There is an ∞ of PST's.

17 One apparent diffy w. much of the forego: We are (theoretically) interested in all of them, wrt the present problem dern. We can cut out almost all of the PST's because their small pc's give them to small pc of being the one selected (GPD$_2$). This could be done in a simple MCarlo way! We have an initial apriori pd on the PST's (≡ "coding lists"). We use this to (Mcarlo) chose one, then we evaluate the pc of the

25 [Consider the ∞ of possl. PST's: It is likely that there is one (of very complex dern) that can solve all of the probs thus far, rather quickly (A.H., hvr. — for this corpus — which is why its dern is so complex!) This sounds ≡ the SM "strategy evaln" problem — exactly!] Looks very Sanders!

"Cross Validation" even if it would usually be "not bad" would not work here — 30 One could always find an A.H. strat. that would survive the "Cross Validation".

Def Suppose I had a soln. to the AAE (Action Algm Evaln) problem: (That for a corpus, a certain algm. has a certain yield. I can estimate how good it will be in future (unseen) corpus. [kind of vague!]) > would this help in the problem of .25 – .30? of the GVS.cc

Another way of looking at the problem: that the df's on the set of strategies, (AA's & PST's) is correct, but in selecting the "Best" ones, one runs into the SOP problem! So AAE & SOP

Maybe very tightly coupled problems!

9/29/02 !! Action Algm Problem. Soln !! (.39 — 750, 14 and following) | SOY: Dept of Problems: .26–.38 749
I.D.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 ! 748.40 : This is a disturbing development! I've been working on t. AAE & SOY problem for years.
Little progress in AAE; some in SOY.
How Critical is this problem? Can I make some kind of "Rutland Derby" soln?
One possl. way: ~~[illegible]~~ Somehow, multiply the prob'y of being max, by t. apriori of the PST. If t. apriori is rather broad & flat, the situation is bad. — But under that condition, one can't do induction — so it is a bad assumption to make!
With a fixed apriori, as t. ssz of t. data grows, it would seem that t. dfs of G's would get narrower & t. results would tend to be correct.

.10 In t. SM problem of Estimating future yield of a strategy, I considered t. maximum amount of "fudge" in a strat: this is its info content. A strat w. n bits of descr. could at max get a spurious yield of ~~[illegible]~~ × 2^k. ~~[illegible]~~ I felt this was overly conservative for SM. — But now if we consider all k bit strategies, it may be that this is not so conservative. If we selected t. best k bit strat, it should be able to get a yield of × 2^k, using Kelly = 1. (I'm not so sure of this, hvr.)
Anyway, I think SM is easier than t. general AAE problem, & I haven't been
.18 able to solve t. ~~[illegible]~~ SM strat evaln. problem! I did have that coding method, that
.19 seemed o.k. intuitively, yet w. careful analysis, seemed not to work! [See SM 1.19:20 (7/27/02) 1.25–.29]

.20 Def: Re: t. Soy (spurious Optimzn. Yield) problem: Say we have a bunch of (n) AA's! We have a joint PD for t. yield n-Vector of t. bunch. We have a way of finding t. PD of t. t. peak G. (Obviously MtCarlo, but also expressable as a multidim. S.). We also have a way to get t. pc of each AA_i's ~~being~~ having largest G (of t. bunch, Also available via MtCarlo or by S); we can then get t. G d.f. for that particular AA_j. Easy if they are "indep": Otherwise "Marginalize" by integrating over all other
.25 AA_i's $(i \neq j)$.
.26 Is .20–.25 related to SOY problem? (in any important way?) In t. "true" SOY problem, we have t. True joint G d.f. of t n AA_i's. ~~[illegible]~~ $P(\vec{x})$; we have t. apparent d.f. $P'(\vec{x})$ (due to sampling error).
We select a ~~[illegible]~~ "most likely to be best" AA_j on t basis of $P'(\vec{x})$. It has a Marginal d.f. of ~~[illegible]~~ G which is in error due to ~~[illegible]~~ ssz < ∞. How bad is this apparent d.f.? —
Well, if n is very large, there will be some poor AA's that look much better than they really are (This is Soy effect)
30 Also, some truly v.g. AA's that look better than they really are; but there are ~~[illegible]~~ fewer truly v.g. AA's, so this is not as important an effect. As n → ∞ (as in T.TM case) this (soy) disparity can be very large! There is probably a tendency to pick one of these "spuriously Good" AA's, & t. effect seems to get worse as n → ∞ (particularly as t. AA's are a "universal set" — so they can have any degree of "A. fitness".)
.38 So this does sound ~~[illegible]~~ = t. TM problem.
.39 ⟶ Can we legitly look upon this problem as a Bayesian estimation,
.40 in which I had (incorrectly) assumed a uniform apriori for ~~[illegible]~~ each of t. AA's being best?
Yes! — see +1.13 for reason

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 749.40: This looks like a correct view! What's not clear is just what the aprip is "of": Is it as -1.40, the aprpd of each AA being "the best", or should it use some other aprpd?

29" Bif v.s 44" old

At any rate, the Big impt. idea is that the aprip is very unlikely to be "Uniform" over the AA space! More likely uniform over some dcrns of the AA space.

However, Another Q is: what do I want the aprpd to be "of"? — e.g. I can have an aprpd of the yield Dif. of each AA.

→ I want an "aprpd" that can be modified by a "summarizing Machine" —

So it is "improvable" w. ~~training~~ "Training".

0 Apart from the details, this looks like the Breakthru of the last four Decades!

\$ Seems to be the main problem ≤ M. (I'm not sure as to whether ≤ 0Y ~~≤~~ (-1.26 -.38) solvd).

13 → The main idea here, is that whenever you compute a aprpd, one must do it in a

.14 Bayesian way & one must have an aprpd.

There is still a lot of Q about ~~how~~ what aprip to use: any aprip of GPD1 or of GPD2?

Also! what form ~~should~~ it should take — what it is dependent on — what kinds a "summary machine" of.

Lets Go back to an (apparently) easier problem: Strat. evaln. in SM!

20 For each strat, we ~~have~~ want a P.D on G. — This is a mean ~~yield~~ / yield, also a σ^2:

Want

What we want (in ln doman) a min σ^2 for a reasonable ΔT — and so we can get min σ^2 for longer ΔT's by assuming ≈ "independence". Lets Gloss over the details here, say there is some / find ΔT that we want the D.f. of G (= In yield) for.

One guess: for all Strategies? aprpd: $\mu = 0$ in ln doman: σ^2 is a ~~the~~ function of coding cost & strat.

Just exactly how σ^2 is a function of code pc is important; but not obvious. Maybe $\sigma = \ln 2^{\text{codelength}}$.

= -ln (aprip of strat). This 2 codelength is ~~or~~ the amount of mult yield from a pure A the strategy:

Its "Available Fudge". This may large or for ~~smaller~~ unlikely codes: This is opposite of

what we want! Perhaps $\sigma = (-\ln(\text{aprip of strat}))^{-1}$

10. σ^2 ≈ ssz of data (normally), so $\sigma \propto \frac{1}{\text{codelength of that}}$ means "ssz" ∝ (codelength)^2

A reasonable approach: say X is codelength of strategy; let $\sigma^2 = f(X)$; $f(\cdot)$ to be determined.

Investigate what happens for various forms of $f(\cdot)$

to 5 f = constant is what I used before ≤ aprip diverges. Probably weeps! $f(x)$ is not the thing that has

mm — to converge — it's only σ^2 of the aprpd.

— But maybe I'm barking up wrong tree! For bayes, ≤ aprip of hypoths = 1 — true coverage.

.36 I maybe thinking about the wrong "Apprch". A simple case with best aprpd of GPD2 — the aprip that

.37 a given PST will be the "best" PST for a given problem.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20 : 750.40 ! Considering "Uniform" apriphd for GPD_2 ! There are an ∞ of PST's to consider & one must (i.e. in Mt Carlo mode) try all of them w. equal frequency to get the distribution of "best G". It may be that there are an ∞ of PST's that get the same by G values for Uniform apriphd, that Mt. Carlo method would not be practicable.

If we use a non-uniform apriphd, what is the relation of the distributions of "maxP" ?

05 A possible Mt. Carlo datum (X) was Mt. Carlo chooses PST (a subset of apriphd on PST's). We then
06 Mc carlo chooses a value of G from that PST. We get a distribution of points: each has a (G, PST_i) label.
What's probly that "PST_i" labeled pts are "best" ?

09 An interpretation: Do K Mtc choices like .05-.06. & count how many times PST_i has been Best, $= C_i$.
10 Do this many times. Then the distribution of GPD_2 of PST_i is $\propto C_i / \sum_i C_i$. Does this vary with K ?
11 Does it $\to$ a limit as $K \to \infty$? What is value of for $K=1$? (Uniform!) Is this simply the apriphd of PST_i. What is value for large (but finite) K ?

13 Well, say G has a sharp upper limit ; G has discrete values only. For large K, the population of
14 G-maximizing sets of trials having G_{max} will be a different dist. for big K.
There is a "large enough K_0" s.t. it's very likely that G_{max} has population of at least 1. for any $K > K_0$
we will get a distribution on G_{max} & the no. of cases each TPT_r has of G_{max}. — So
it will be indep. of K past that of the d.f.'s !

19 So .13–.14 is probly riter ! — the idea of .09–.11 wouldn't work.

20 An eq. that I can't exactly justify, but which may have reasonable behavior !

21 Wt. of PST_i :
$$\int_{-\infty}^{+\infty} dw \left(P_i^{a_i}(w) \cdot \left(\int_{-\infty}^{w} dw \prod_{j \neq i} \left(P_j(w) \right)^{a_j} \right) \right)$$

$$\int_{-\infty}^{+\infty} dw\, P_k(w) = 1 \quad \text{for all } k.$$

a_i is the apriphd of PST_i : $\sum_K a_K = 1$

26 The former case of: finite no. of PST's was dealt w. by making all $a_i = 1$.
.21 does not give same result, if all $a_i = \frac{1}{N}$ (for N, PST's)
If in .21, I replaced a_i by 1 & "a_j" by "$\frac{a_j}{a_i}$" I'd get consistency w. .26

.21 is wrong way : should be for .26
$$\int_{-\infty}^{+\infty} dw \left(P_i(w) \left(\prod_{j \neq i} \left\{ \int_{-\infty}^{w} dw P_j(w) \right\} \right) \right)$$

30

31 To modify .30 :
$$\int_{-\infty}^{+\infty} dw\, f_{P_i}(w) \left(\prod_{j \neq i} \left\{ \int_{-\infty}^{w} dw P_j(w) \right\}^{\frac{a_j}{a_i}} \right)$$
→ also 758.15 see a752.31 !

Note, for small $\epsilon = a_j/a_i$; $X^{\epsilon} = e^{\ln X} \approx 1 + \epsilon \ln X$. $X \leq 1$ so $\ln X \leq 0$.

Say $X_j = \int_{-\infty}^{w} P_j(w)$! $\prod_{j \neq i} X_j^{\frac{a_j}{a_i}} \simeq \prod \left(1 + \frac{a_j}{a_i} \ln X_j\right) \approx 1 + \frac{1}{a_i} \sum a_j \ln X_j$ if $\frac{a_j}{a_i} \ln X_j$ are all small

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

749.45-49

Tom English
Jackson State U...
Jackson Mississippi

0: 751.40 : In SM if one has a strategy, there is always one "theoretical" way to evaluate it.

Take the past corpus, using AHP, make a D.F. on all future cases containing one of the corpus.

Test the strategy of interest, on this future D.F. of the continuation of the corpus.

.03 The complexity of the strat. is irrelevant under these conditions.

Could we do any thing like this in the (More general) AA problems? AA seems quite diff'nt!

Consider PST again. We want to guess how good PST_2 will be for problem, X, w. $c_B = c_{C0}$. If, as in .00-03 we had enough time (actually $c_B = 0$!), we could actually determine G as a funct of PST_2, $\hat{X} \to c_{C0}$ - exactly. We could do this for any PST, independently of the form.

These problems: AAE, SM strategies, PST pooling, SOY, Are related but not exactly the same: I've worked on them a lot, but (except, possibly for SOY) I have no usable (or any) solns.

I should: State each problem clearly (for SOY see 749.26-38:)

Keep a record of state of soln. : Look at "How to Solve Problems" list.

List some special cases of known solns.

List some non-solns -- : why they don't work.

SN A wild Idea! for each PST : if we have a P.D. on G w.r.s. c_{C0}.

Hwr, if the dem is long, the P.D. is less likely to be accurate.

So in comparing PST_0 w. PST_1; if PST_1 has long dem, then

When PST_1 seems better than PST_0 for many W values,

the probty that PST_0 is worse than PST_1 is very high if the ↑ curves were better.

but we have very little confidence in curve 1

This must justify 751.31 $\frac{\partial j}{\partial i}$ are our relative confidence in PST_j v.s. PST_i.

a small $\frac{\partial j}{\partial i}$ means $\left\{ \int_{-\infty}^{w} dw\, f_j(w) \right\}$ is close to 1.

One Complaint about 751.31: If one computes the probty of each PST being Max, using this formula: the pc's will not sum to 1. We could renormalize, but this misses the point.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 752.40: An easy way to test conjectured models: Try them on simple cases w. ≤ 3 or 4 PSTs. Try on SM; then compare w. SM in which one is doing pure prediction (so one knows what the right Soln. is).

A possl. analogy betw problems & non-problems:

Say we are trying to predict stock A. It "could" correlate w. any of the different stocks. It's likely that one would (← Ä.H. ishly). To use the correlation, we have to specify the stock, which amounts to ~ $\log_2 9000$ bits.

In the case of a very large (actually ∞) PST's, we have analogous problem, but the thing being predicted is different. Similarly w. "strategies" in SM: there are ∞ of them.

10

.11 Another promising approach is 749.18–.19: using yield of a strat. as a method of coding the corpus. It seems clear, that if a strat. has yield "significantly ≠ 0" that it restricts the possl. corpi & in ∴ a larger coding method. If p_i is the prob on the i^{th} possl. corpus elements; & the strategy multiplys this distribn. to p_i', & q_i is the trans

.17 pd. ant. corpus; then the coding gains $\sum q_i \ln \frac{p_i}{p_i'}$ bits.

So whether $q_i = p_i$ or p_i' or — neither.

.17 is the difference in KL distance to q_i; of p_i' vs. p_i.

20 Consider a random binary seq. to be the daily Δprice (= ±1). A strat. with yield Y. Is a constraint on the corpus. The larger Y is, the more of a constraint. (i.e. the fewer possl. codes w. that yield).

One method of strat. Just tells whether to buy or sell — so one has a

.26 partitioning of the postfixes of a semi infinite binary string corpus.

.27 W. certain postfix one bets +1; other postfix "−1"; other postfix, nothing. 3 partitions

So any partitioning of postfixes into these 3 categories is a strat.

30 At all times, one will be in the market: either long or short $1.

— we could make this exponential yield if we knew probs of win or lose — but just consider this simple (non-exponential) betting method.

.32 [On the other hand, if one averages a yield of y_0 per bet, one should bet a certain fraction (function of y_0) of one's fortune on each bet for "Kelly betting".] — The value of y_0 can be determined by Kwindoor Rwindo

.34 So perhaps exponential betting is normal.

For the binary seq. of .20, it doesn't make any difference whether we use Exponential betting or not. For a given Y, on a corpus of length n,

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

ED

:00: 753.40: There will be a certain subset of strings that have yield = Y. If K out of 2^n strings have yield y, then the strat has info content $\frac{K}{2^n}$ or $n - \log_2 K$ bits.

If K=1, then evidently the strat specifies 1 string & it has a bit in it.

Also note $n - \log_2 K = n$, as it should. So the ratio of the no. of possible seqns before & after the strat specifies 1 seqn per strat is 2^n & the no. of bits in it.

However this is not a good way to code (normally). For a v.b. strat, we expect a doubling of fortune/yr. This is 1 bit/yr. If the strat. takes L bits to describe it, it takes L yrs of data to verify that it has positive yield.

This means Very simple strats. Or superhy yields!

To reason in the other direction: if a strat has a code w/ no. of bits, ($2^{-r} = \frac{K}{2^n}$), then instead of Y, our real/multiplicative yield is 2^r — i.e. if we did suitable Kelly betting, perhaps using x wins or r winds (as -1.32 - .34) to get pc's.

Actually, this is reasonable!: [If, every d days, we make a bet w. a certain average expected-yield, this is equiv. to reducing the future courses of the corpus by a certain factor.]

To estimate K, (.00) the redundancy of the strat: Run random seqns thru it & see how frequently the yield = Y. The SSE for yield of just Y may be small, so if K is a fn of Y, we can collect data on no. of "hits" of yields near Y, & interpolate (= smoothing the data).

This will work also, if we have a Brownian Motion-ish market. We simulate it like the BS by a random walk of the appropriate step size and ΔT of the market. We apply the strategy to it & see how often its yield is within a factor of 4, say, of Y — & we interpolate in the ln domain. In this case, we want the density of cases per dY or $\frac{d \text{ no cases}}{dY}$. This has to do w. the precision we need to specify Y. (I think I've been using this below, but (.01 - .05) & (.19 - .23) are for a simpler SM model (i.e. binary seq) & the system does seem to work!

→ Note 755.00

Around this time, I should look at the recent simpler arguments vs. the coding method of the "strategy as a code"

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

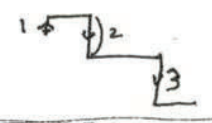
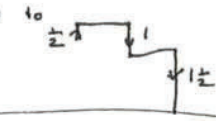
.00 (754.34 spac) In Plg Brownian corpus: t. varc. "per unit time" is constant. To deal w. varying time-varying variance a bit: Devise strategies so that threshold for next bet is always ≤ threshold for previous bet. — Woops! This gives constant σ^2 bets assumed. —

A way that might work: bets depend only on the price history. This helps some, but still, t. σ^2 per hr. could vary a lot over a week, and so our Brownian Mkt on $754.24 would not be V.G. (.18)

.08
.09
.10 A way to study the variation of σ^2 in t. SM signal: plot t. total variation (using some grid size: see what happens if grid size is changed: doubled, ×4, ×8...) Then take a smoothed value of this as "t. local variance".

Q: would this respond (more rapidly/"better") than smoothed RMS? To get RMS, we need some pre-smoothing to get t. mean (μ) - so we can get RMS deviations from it. The total variation via Radon .09 seems to have less "smoothing" in it. A min Grid size could be t. grid size used by the Mkt quoters themselves.

.18 (08) A possl way to deal w. "drifting" σ^2: T. strategies categorize t. past fix (history) in 3 categories (positions of 753.26→.27). These categories depend on the of changes in t (ln prices). Unfortunately $\ln \Phi = \infty$ and $\ln(-\epsilon)$ = imaginary.

.20 I want?:  equiv to  So we are circumventing variations of excessive changes.

A Q is: How well do we want to try to simulate SM signal — i.e. how good should t. "default" D.f. be? The discussn. of 753.20ff on binary strings, is t. Basic Argument — that has to be applied, somehow, to t. more general case.

.30 On one hand Say we have a fixed strategy: If we use a default d.f. w. very small σ^2, the pc of obtaining yield γ w. the strat, will be very (large small). So t. σ^2 of t. default d.f. is very important! But in t. real SM, t. σ^2 is is always varying.

.31 No!

.33 Hmm. Some strats may be invariant under changes of σ^2 — ie. yield indep. — in fact one of CHZ's that I looked into was σ^2 indep. We look at O p in first ½ hr. of Mkt. This scales our betting local behavior for rest of day or more.

purely not true (Same .33)

On another note: Even using t. simple/modified 753.20ff, The Ring that I'm really interested in (corresponding to t. AA problem), is t. d.f. of t. yield, γ of t. strat, over future corpi. What 753.20 says is: "I've found a short code

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

OC: 755.40 for t. corpus — its r bits shorter than a usual decrn. — so I can just yield 2^r if I do Rally =1. " I use the strategy to figure out how to best do Rally back —

This is normally not the sense in using the strategy to back.

Another interpretn: Since this short code for t. corpus exists, it means that if ALP had been applied to that corpus, it would have had at least a yield of 2^r.

Also: It means that t "expected value", the unbiased estimate" of yield for a corpus of length n is $\geq 2^r$

.06 We can look at the problem of predicting t. corpus: we have found a predictor of wt. 2^{-r} = wt of predictor x (pc of corpus wrt t predictor).

0 We have at least 1 predictor w wt. 1: It assigns pc = .5 to 0 & to 1

Consider t. statement. " PST_i had a [redacted] G of G_{ij} for Prob$_j$." Hvr, we are interested in predicting G for Prob$_k$ by PST_i. (G_{ik}) It would seem that this predn. — (t. defn of G_{ik}) would be indep of t. decrn length that PST_k. But perhaps t. D.f. is reflow flat, because we didn't spend much cc on it (if PST_k's decrn is very large).

.18 — Consider t. s-Grammar: $P(PST_i, Prb_j, cc_j, G_{ij})$: G_{ij} as a probabilistic funct of PST_i, Prb_j

.19 — Prb_j & G_{ij} are fixed. so $P_j(PST_i, G_{ij})$.

20 Is it true that if PST_i has a long decrn, then the D.f. will [redacted] tend to be flat?

Well, for rewrite type rule type Grammars (like CFG's) long decrns, mean lots of substitutions, so low P_j — This is "off the mark"

In .14 we have a recursive defn on G, so t. discussion of t. structure of PST_i seems irrelevant.

We have in .18 $\Rightarrow P(\cdot,\cdot,\cdot,\cdot)$ that we've optimized via #14.18.

If PST_i is complex (i.e. has long decrn) then, will .18 take a long time to evaluate?

30 Back to [redacted] .18 Consider a "cut off" so only PST's are included if their apriors are $> \epsilon$. Clearly, for small enough ϵ, t. system will not work (it's too close to $\epsilon = \phi(!)$). So if we had some lower bound for ϵ.

Good! It would seem that in suitable circumstances, one would have to consider ϵ of any/degree of smallness.

So, w. finite SSZ — just what is t. limitation on how small ϵ can be?

[Really sharp ϵ limit is like "Rwindo" & I'd like a "soft" limit on ϵ, more in t. spirit of Xwindo]

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : Hm! In comparing 2 PSTs for likelyness of "best": T. p_i° axiom can be used in

folg. way: $P_i(G_\#)$ is "better than" $P_w(G)$ if $\exists$ PST_i exists and

.03 $P_\# ST_j$ does not exist or $P_\#(G) \geq P_j(G)$. T. probly that (PST_j exists is p_j°)

→ One test of .00 – .03 is, t. crit of [75?.09 – .19]

.05 Another possl. way to do "Max expected G"! Pick pairs of PSTs w. probys p_i°, p_j° (≡ prbys of PST_i, PST_j resp.): Choose a G value for each in accord w. it's P.D. for G. Compare 2, & give a score of 0 or 1

.10 for loser, winner.

Approach (.05–.10) may be ≈ or ≡ to (.00–.03). In both, we do pair-wise comparisons.

A possl. approach to "Pc that PST_i is better than any other PST":

.14 $$p_i^\circ \cdot \int_{-\infty}^{+\infty} \left(P_i(G) \cdot \prod_{j \neq i} \underbrace{\int_{-\infty}^{G} P_j(G)\,dG}_{\alpha} \right) dG$$

Replace α by $\left(1 - \int_G^\infty p_j^\circ \cdot P_j(G)\,dG\right) = \left(1 - \left(1 - \int_{-\infty}^{G} P_j(G)\,dG\right) \cdot p_j^\circ\right)$

For small p_i° this $\approx \exp\left(-\sum p_j^\circ \int_G^\infty P_j(G)\,dG\right) = 1 - p_j^\circ + \int_{-\infty}^{G} p_j^\circ P_j(G)$

.20

Then .14 becomes $p_i^\circ \int_{-\infty}^{+\infty} P_i(G) \exp\left(-\sum p_j^\circ \int_G^\infty P_j(G)\,dG\right) dG$.

This expression is middle of 751.20 – .40 and 752.20 – .40.

.26 A rationale of .14. p_j° is t. probly that PST_j will occur (≡ "exist").

$\int_{G_0}^\infty P_j(G)\,dG$ is t. probly that if PST_j occurs, it will have $G > G_0$.

.29 $p_j^\circ \int_{G_0}^\infty P_j(G)\,dG$ " " " PST_j will occur and " " " PST_j will not have $G > G_0$.

.30 $\prod_{j \neq i}$ " " " is probly that non of t. PST_j's ($j \neq i$) will have $G > G_0$.

.33 $$\int_{-\infty}^{+\infty} p_i^\circ P_i(G_0) \left(\prod_{i \neq j} \left(1 - p_j^\circ \int_{G_0}^{\infty} P_j(G)\,dG\right) \right) dG_0$$ is (better) probly that

PST_i will exist & be better than all other PST_j's ($j \neq i$)

see 758.20 – 759.12 for whittering Criticizing! (Looks pretty bad!)

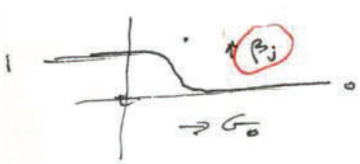
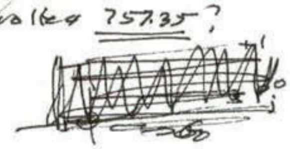
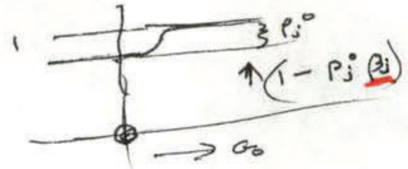
8/4/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20! 757.40 : -1.35 & its "rational" (-1.26 ff) & is a reasonable equ. to investigate.

Try a test $\equiv$ or $\sim$ to 751.09–.19.

Is there a Mt Carlo way to realize 757.35?

~~[illegible] the -1.35 may be backwards in a critical place!~~

$1-P_j^\circ\beta_j$ is the probty that PST_j will not exist and have $G > G_0$.

$1-P_j^\circ\beta_j$ is the probty that PST_j will both exist & have $G > G_0$.

10 → $\prod_{i\neq j}(1-P_j^\circ\beta_j)$ is the probty that none of the PST's will [illegible]

for small P_i°'s it is $\approx \exp \Sigma - P_j^\circ\beta_j$ since $\beta_j \le 1$ & $\Sigma P_j^\circ \le 1$ the product $\prod(1-P_j^\circ\beta_j)$ must converge to >0.

$1-P_j^\circ\beta_j \approx (1-\beta_j)^{P_j^\circ}$ only if β_j is small. $(1-P_j^\circ)^{\beta_j}$ is better, since P_j° is usually small.

.15 751.30 is $\int_{-\infty}^{+\infty} P_i(\omega) \prod_{j\neq i}(1-\beta_j)^{P_j^\circ \div P_i^\circ}$ ← [The idea of using P_j°/P_i° was that if we considered a finite small, finite no of PST's, we would sample from it & would give the result

If we computed ?

20 — first we find $1-P_j^\circ\int_{G_0}^\infty P_j(G)dG = \gamma_j(G)$ for each i

.21 — $F(G) \equiv \prod_{j \leftarrow \text{including } j=i} \left(1-P_j^\circ\int_{G_0}^\infty P_j(G)dG\right)$ Then to get -1.35

$F(G) = \prod_j \gamma_j(G)$ Then $-1.35 = \int_{-\infty}^{+\infty} P_i^\circ \frac{P_i(G_0)}{\gamma_i(G_0)} F(G_0)\, dG_0$

.34

In fact, to obtain -1.35, we would ~~not include many~~ include only a relatively small no. of PST's in the calcns, since most of them would have such small P_j°'s that they would modify the 35 very little.

$F(G)$ is really not very small. $f = \prod\gamma_j = \prod\left(1-P_j^\circ\int_{G_0}^\infty P_j(G)dG\right)$, with $P_j^\circ\int_{G_0}^\infty P_j(G)dG = \epsilon_j$

30

$\Sigma\epsilon_j < \Sigma P_j^\circ$ so $\Sigma\epsilon_j < 1$ so I think F must be $> e^{-1}$ and F can't be >1.

Not exactly! $1-\epsilon$ is $< e^{-\epsilon}$

— see .20 R for defn of γ_i.

So -1.35 from .34 $\approx \int_{-\infty}^{+\infty} P_i^\circ \frac{P_i(G)}{\gamma_i(G)} dG$ (it could be high by a factor of e at most)

.36 $-1.35 \approx \int_{-\infty}^{+\infty}\left(P_i^\circ \frac{P_i(G)}{1-P_i^\circ\int_G^\infty P_i(G)dG}\right)dG. = \int_{-\infty}^{+\infty} \frac{P_i(G)}{\frac{1}{P_i^\circ} - \int_G^\infty P_i(G)dG}$

If P_i° is very small this is $\approx \int_{-\infty}^{+\infty} \frac{P_i(G)}{\frac{1}{P_i^\circ}} = P_i^\circ$.

If P_i° is $\equiv .1$, say this approxn. is not bad!

ABC

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 758.40 : So it begins to look like 757.35 is usually $\approx p_2^\circ$! A terrible result !

758.36 R is between p_i° and $\frac{p_i^\circ}{1-p_i^\circ}$. (probably).

The wrenchest part of this exct. is that $F(G)$ (-1.21) is between $1 \pm \frac{1}{e}$.

$$-1.35 = \int_{-\infty}^{+\infty} \left(p_i^\circ \; \frac{p_i(G) \cdot F(G)}{1 - p_i^\circ \int_G^{\infty} p_i(G)\,dG} \right) dG$$

Since $F(G)$ can vary in size by a factor of e, & thus ↖ expression can vary (at most) by a factor of e from p_i° or $\frac{p_i^\circ}{1-p_i^\circ}$.

It may be possl. to find out what's wrong w. 757.35, by considering the case of a small no. n, of PST$_j$'s of = wt. In this case I (Think) I've now the right Soln. Take a look at 751.31 again.

See 769.33 for possl. applicn. of .13 !

751.31 →
$$\int_{-\infty}^{+\infty} \frac{p_i(G)}{\left(\int_G^{\infty} p_i(G)\,dG\right)^{\square}} \left[\prod_{\substack{j=1 \\ (j \neq i)}}^{n} \left(\left(\int_{-\infty}^{G} p_j(G)\,dG \right)^{p_j^\circ} \right)^{\frac{1}{n-1}} \right]^{\frac{1}{p_i^\circ}} dG.$$

T. Geom mean

The stoyest case for this guess is that it satisfys the constraint of .10–.12 : (.20–.30 Below) is a slightly stronger argument point.

Another perhaps more intuitive expressn. (May be same as earlier one!)

1) Pick a random G value from PST$_i$
2) " a " G " " a random PST$_j$ (using apriv to chuse PST$_j$).

What is pc that 1 is > 2 ?

To criticize .20 consider the standard of n PST$_j$'s of $p_j^\circ = \frac{1}{n}$ for all.
& .20 doesn't seem to give result: It gets to pc that PST$_i$ is better than a "Typical" PST$_j$ — not to pc that it's better than all PST$_j$'s ;
.18 (=75131) does seem to be closer to ; It does to the product of all of the other pc's being worse than PST$_i$.

Consider the "standard" of .26 ! We compare PST$_i$ w. the $n-1^{\text{st}}$ power of the geom. mean of the pc's of the other (PST$_j$'s) being worse than PST$_i$.

The .13 is "not unreasonable"; it really doesn't have any definite reasoning behind it!

It would probably be possl. to express .13 as a Monte Carlo approx., using some other functions than $\sum_{-\infty}^{G} p_j(G)$ as the comparison functions.
Doing so might also give some understanding of behavior of .13 !

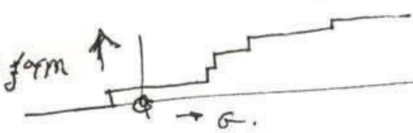
8/4/02

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : Another v.g. approach to checking this: Say I had a bunch of SM strats: $\{S_i\}$
Each has its own μ_i & σ_i^2 for yield. $\binom{-1.13}{759.13}$ gives one soln. STEIN may give another : (which is different from Stein which starts w. raw data).
STEIN is different than -1.13 in that Stein starts w. raw data. ~~-1.13~~ does not.

Well, -1.13 could be empirical data: $P_i(G)$ could be a seq. of δ functions corresponding to 'G's for particular problems.

0 $\int_{-\infty}^{G}$ has the form



Better write review of [AAE, PST evaln., SOY, PST Grammar Gore, Strat evaln (SM), Stein Applicn.

.13 This topic starts at 746.00: We want to improve the (GPD, [PST]) stoch grammar — Goes to 760.10 (15 pp.)
— But we need a Gore for wh. t. Grammar.

.16 → [It would seem that any grammar would have t. same Gore: Namely, Max
.17 pc of corpus × a priori pc of Grammar] elements of Grammar are $(PST_i, Prob_j, cc_k, G_\ell)$ — Q , A
.18 So we just Maxz. 414.18 over t. known corpus.
20 : Is (.16–.18) t. final soln? Would PST's w. long derns give wrong results?

Remember t. Grammar should be able to assign pc's to any PST, Prob, cc, G —
no matter how long t. dern of t. PST (or Problem) is.

Can I use t. idea of .16–.18 to evaluate SM strats? — or any AA's?
It would seem that this implies that complex strats need not have poor expected yield!
→ This seems to contradict the trick method of using yield to code t. data.
Hvr., its not clear how .16–.18 can be applied to strat evaln.

The strat evaln problem & t. PST evaln problem seem similar because in both cases,
30 One can use a very complex, A.H. $\binom{PST}{Strat}$ that gives v.g. results in t. past corpus, but will prob'ly do poorly on future corpus.

In .17R Given PST_i, $Prob_j$ & cc_k one can always compute G exactly
Similarly Given a Strat & a corp, " " " Y " .

T. problems are ① Given PST, $Prob_m$, cc_n to get p.d. for G.
② Given some strat, but unknown corpus, to get yield estimator.

In .13 There are 6 types of problems that seem closely related: I should see if any of t. apparent "present solns" to some of them will help w. t. others!

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : 760.40 : SN Re t. GPD-PST grammar: In early TM, we will have a small no. of PST's inserted by usr. Later, we will insert more, a eventually dev. GPG-PST Grammar. I think these steps can be done "smoothly", w.o. basic Modification of TM. — May be only Lsrch

→ "More" means a finite no. — they will be evaluated as individuals, indip of t. details of their derns. Hvr, each PST can have a special way to evaluate its assess, Problems, CC, G def. — each PST can have its own stack (Probj, CC_K, G_l) grammar. — The t. Grammars can have "shared" concs.

Next, we will begin factoring t. PST derns., so as to ↑ t. pc of overall corpus dern'. This factoring may be difft, so trainer may have help TM by hints, or actually do part or all of factoring.

16 . . . . . Also, (while / when) this is occurring, trainer may add many more PST's.

17 Review of 746.00 → 761.16 (PST, Prob, cc, G)

746.00 – 747.40 : Discn. of whether to use system where GPD is on [illegible]

20 or just (PST, G) 767.37–.40 is reasonable resoln.

748.17 starts difty of "≥∞ PSTi's" and t. Ad Hocness of very complex PST's

Idea that the is example of AA evaln. problem, Strat. evaln. problem, SOY problem — which I haven't adequately solvd. [later, it appears that t. problem seems difficult, & may have "simple" soln! 760.16–.18

④ • 749.26 → .38 V.G. Dern of SOY problem

• 749.39 is idea that in evaluating PST's, one must consider their Apriy: Just how this Apry was to be usd, was at first unclear!

30 Some Attemps! 751.05–.06 A Mt Carlo way; interpretn! Ibid .13–.19 shows it wouldn't work

① 751.20 was eventually decided on as a "good try" — 759.13

② • 753.11 – 756.10 : A Good treatment of SM problem of using a strategy & its approach. Yield, to Code a SM corpus. It does seem to work ≥ 756.06 explains what it actually means

• 756.18 — on continuity in GPD[PST] grammar. — Dern of problem: (Later 760.16–.18 760.16–.18 seems to be correct soln.)

756.30 – 757.35 ← trial soln. to (756.18) 758.00 – 759.12 : Demo that 757.35 will not work

③ • 759.13 : repeat of 751.20 : which looks "not bad". 762.00

0/5/02
ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00: 761.40 (space) 759.20-.40 tries a new model, but gets more reason to believe (which ends up N.G.) 759.13 is 'not bad'.

759.35 — way to analyse 759.13 ; 760.00-.10 other ways to test 759.13

.02 (5) 760.13 : Suggestion that I review AAE, SM strategy evaln, PST evaln (PST evaluation), PST Grammar Gore defn, SOY (Applicn of STEIN idea) 6 Impt., apparently Related, ideas.

(4) 760.16-.18 Idea that Gore for $(PST_i, Prob_i, cc_n, G_e)$ grammar, is just Gore for standard QA problem : Maxes 414.18 followed by discn of how to relate this "soln" to other solvd & unsolvd problems of (760.13)

.10 Discussn: the main, most critical ideas reviewed.

.11 1) Defn of SOY problem 744.26-.38

2) "Soln" of evaln. of SM strategy via coding 753.11-756.10

.15 3) A possl. soln. to the PST amoptzn problem! 751.20 } = 759.13 This is a more exact form of the GPD₁ → GPD₂.

4) Final soln. to GPD; Σ[PST] grammar construction problem — that Gore was same as Gore of 414.18. (760.16-.18)

.20 5) Suggested review of relations of AAE, SOY, SM Strat. evaln, PST evaln, PST Grammar Gore defn. STEIN effect

→ 0/11/02 (Later) (1) I'm not Completely Convinced that 760.16-.18 is a soln. (or "the final soln"). One very Serious "objection": It does not seem to answer the Q of 746.00 i.e. given #2 S Grammars for PST's with ≈ 414.18 Gore.

One could be much "Better" in that it has PST's (or just one PST) that gives (usually) much 'Better' (less cc for same G) soln to problem.

.27 Good → A possl. Approach: That the S-Grammar simply presents the facts. Finding a "good" Grammar is a separate problem: Yes. finding the "Best" PST

.30 under those circumstances can be quite difft, and may need a "creative" soln. — see 769.20 for more discussion.

— But the "Best" GPD, is still the one w.r.t. Best 414.18 Gore.

While it is in theory, possl. to evaluate all of the PST_i, $Prob_i$, cc_i, & obtain an exact G, we are limited in time available & must use approximate PC values.

(It's like the pd. for the 10^{100} bit of π: Given a finite CB, the pd. will depend critically on the CB.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00; 762.40 ; As t. systems seems to work now: It can function as a QA system only.
As I'm thinking about it, Rec update algm. for OM, would be relatively simple.
It would use previous functions & subfunctions of previous solns: Also, it would
use context of 'larger' (proper kinds as it grew in knowledge..
However, the Update system would be static. It would not change.
I don't (yet) see any need to change. All kinds of regys would be accessible
to t. system (I think !). ? (very Quasi-unobservable hrr.!)
[redacted] If I'm Lucky, (tho it may ~~not~~ not be "optimum"), it might be
10 "fairly good" — good enuf to do updating of IND & other OE problems.
Good enuf ~~for trainer~~ for trainer to introduce new PST's & eventually,
have a {PD, PST} grammar. And eventually, good enuf to ~~improve~~
improve (propose new PST) PST for Induction problems.

Look back at 69&.13 → 40 (outline of paper). There is no discussion of t.
special IND update! It's not exactly Lsrch, but it's almost Lsrch, because
we're looking for short pgms. However, t. Pd (in "certain sense") maximizes is
20 invariant! Also, I probly wouldn't be using much Backtracking, so I
should retain several (1) solns. Must are "significantly different." I haven't yet been
.21 able to define this idea.

T. reason for preceding remark: (.20-.21) Next t. update almost always only adds on
to the existing "soln" as new problems are solved. It is possl. for t. update
to "correct" t. soln a lot by observing a new conc. that has [redacted] larger
case count, but unlikely, since new concs ("should be" "are usually") discovered by OSL
(One Shot [redacted] Learning), or well before t. case count is > 2 or 3.
30 [One way that a "breakthrough" (contraction/revision) can take place: A Large CB
search is done, which is able to detect regularities now previously unobserved
because of such small CB being used — i.e., no specific OE or ENV facilities.
As a "stand alone" system, the QA system can be improved
by (adding/modifying) update techniques! E.g. GA (GP) or RANN.
(or regular ANN.) — specific tricks like dividing up corpus (at first, using
in Q's
indices) put in by trainer]

(spec 76500)

10/6/02

ANN and RANN: How to Get ALP values

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

: Say the wts for a ANN are betw. $\pm \frac{1}{2}$. So for a N wt. system, the hvolume of dcmn is 1. Then, if the hvol. of "acceptable" wts is V, its pc is just V. We can balance this against the "precision" of the output in prediction.

If the ms error is σ^2, for k observations pc per obs. is $\sim \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{\overline{x^2}}{2\sigma^2}}$

$\sim \frac{1}{\sqrt{2\pi e}\,\sigma}$. I guess it's necy to have some kind of apriori for σ.

More exactly, the pc of all k obs is $\prod_{i=1}^{k} \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{x_i^2}{2\sigma^2}} = \left(\frac{1}{2\pi\sigma}\right)^k \cdot e^{-\frac{\sum x_i^2}{2\sigma^2}}$

$\sum x^2 \equiv k\sigma^2$ so $= \left(\frac{1}{2\pi\sigma}\right)^k \cdot e^{-\frac{k}{2}} = \left(\frac{1}{\sqrt{2\pi e}\cdot\sigma}\right)^k$.

If, apriori, we knew only that x is betw. 0 & A, then the first ob. costs — No!

Because of the procdn. in the ANN, all we needed to descrb. the data is σ.

Seq. of x_i's which are distributed about 0 w. variance $\equiv \sigma^2$.

Even more exactly, say each x_i is known w. precision $\pm \frac{1}{2}\epsilon$.

Then the pc of each obs is $\frac{\epsilon}{\sqrt{2\pi}\,\sigma} \cdot e^{-\frac{x_i^2}{2\sigma^2}}$.

so the total pc of data set is $\left(\frac{\epsilon}{\sqrt{2\pi e}\,\sigma}\right)^k$.

The ϵ^k factor enters in, however we were coded the data;

so it is not relevant for predn.

In comparing pts in wt. space $(.00-.03)^N$. Each pt will have an associated σ & ∴ a σ^{-N} factor.

Say we have 2 pts in wt. space that are local maxima;

How to compare them? One way: (?): Take hvolum, V_0 (or V_1)

→ in which σ^2 is within $\frac{1}{2}$ of the minimum- (see ¶6)

Compare $V_0 \cdot (\sigma_0)^{-k}$ with $V_1 (\sigma_1)^{-k}$. or $V_0^{\frac{1}{k}}/\sigma_0$ with $V_1^{\frac{1}{k}}/\sigma_1$.

or / (compare) $\ln V_0 - k \ln \sigma_0$ with $\ln V_1 - k \ln \sigma_1$

One could do all of the above (¶31) by integrating over the peaks in wt. space, rather than "the hvol" within the $\frac{1}{2}$ power pts".

space 765.00

9/6/02 HMC (How many Coifs?) 765

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

Spec
00:764.40: T. problem at 764.00 seems similar to how many coifs to use in regression. (Lin or N.Lin.)

.01 The trouble is in comparing V_0 & V_1 (-1.31) for t. 2 dffrnt nos. of coifs.

For t. "new" coif(s) we need an apriori — or a set of values between which the apriori is uniform. In this case we can compare V_0 & V_1 in following way: Say V_0 is N dims & V_1 is N+1 dims: V_1 is a N dim space V_1^N (comparable directly w. V_0) mult by a 1 dim space V_1'. It is V_1' that we need t. apriori for. If a certain segment in 1 dimensional V_1' space, of length l, marks the "½ power pts"; then we want t. ratio of l to t. apri "legal" region of V_1',

Actually, t. analysis of .01 if (V_1 & V_2) isn't v.g. — The regions of interest are not like those dscrbd; they are more apt to be hyperspheres.

[N.B. for hy dimension, most wt. of a Hsphere is close to t "skin".]

I did write a bit within last month or 2 (→ 710.00) on t. shape of $\left(\frac{1}{1+x\cdot(\ln x)^2}\right)$. SEE 740.00-.40

~~It ... x = ...~~

In linear regressn, t. coifs are ~1. In nonlinear regressn (say coiff of products of x & logs of x, t. coifs scale as powers of t. rms value of t. variable to be predicted.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

spec
:0: 763.40 : After I write t. report: I will want to review recent (last yr or so) "ID" notes for ideas relevant to various problems.
Make list of Diftys in TM & for each — tie difty to pts. in report where they are relevant — Make list of places in ID notes where those diftys had useful work on them.

Review of 696 to present: Listing of parts written, etc.

09/019/02 Expo: [for introduction!] Hvr! Not strictly upper Rt border of "Introdu"; it On t. Genzd. optzn. problem. discusses Great variety of optzn. problems

Inversion problems and Time limited optimization problems can be regarded as special cases of a more general optimization problem.
Suppose that we have a function ~~from strings to~~ strings to reals.
real function $G(x,t)$. x is an unknown string
We must find a value for X, which we present at time, t, so that $G(x,t)$ is as large as possible. Often G will take the form $G^*(X) \cdot H(t)$, where $H(t)$ is a decreasing function of t.
An example of such a problem! Suppose we had to design an automobile satisfying certain requirements. We want it to be of low cost as possible, yet if we take too long to design it, our competitors will beat us to the market. We have some sort of function $G(x,t)$ that relates the cost of the automobile and the time for the design to be completed, to its ~~~~ competitive value to us.

Inversion problems can be regarded as having a $G(x,t)$ function in which $G = -\infty$ unless x satisfys $F(X) = S$; otherwise, $G(x,t) = F - T$
we want to maximize G
In time limited optimization problems, $G(X,t) = -\infty$ for $t > t_0$ for $t \le t_0$ $G(X,t) = G_0(X)$, the usual time-limited optimization function, (—).
In section on updating, we will show how to solve this general optimization problem.

019/02 EP

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

: 766:40 : Ont. Introdn: T. main part is an augmented version of t. Abstract.

It is in "RAY DISRAY file: ~ 8/15/02 : T. pp are call "Intro 8.15.02 pp 1&2."

On 718.22 I decided to use mainly stuff in CPM No 8.01 for t. QA, induction section. Following 719.10 there is an expon t. accuracy of "414.18":

It refers to "Appendix B" which is expod on 720.10

T. longer version of t. paper is summarized on 721.25-.28 ← This has additional refs.

726.09 - .24 (.40) discusses just what is Amazing about ALP & t. conv. Dom.

I'm not sure this is o.k. for expo! — ibid (.25-.40) I think this was v.g. → (722.20)

724.00 - .34 is an "outline of sci. Amon search & other srch methods.

724.35 - 725.10 is a general "closing remark" on t. lack of limits to t. system.

725.11 ff is an expon Lsrch: I think I decided to copy parts of ~~on 726.05~~ or 055 85 586 & 2/0 589 for t. beginning of this.

(SN) 731.08 ff t. idea that in induction t. a priori is constant, & that considering "context" doesn't really change this fact. So coding really is "sequential" — but perhaps "Backtracking" is still possl. (= useful). — Perhaps "reparsing" in view of new (cones / datas)?

.21 735.07 - .14 covers stuff written on QA · § I
735.15 - .20, 736.00 - .40 · PST's § II
737.00 - .04 Mostly outlines of § III (GCPD) · Also 704.28-34 : 723.35
GCPD § III Not much material!

.26 (738.31 - 739.17 : Just how does TM work? General flow. alym : ??
738.36 - 739.03 is v.g. model. — Maybe needs more detail? At that pt. I didn't realize that much of IND search does not require update of guiding P.D.) Its t. a priori of ais invariant. Later, when t. induction probs is solved w. better PST's, we will need to "update" → Its still not so clear! What is devised is mainly t. kind of updating, and assoc. search type. For t. "finite" set of PST's that are not being invented — that analysis is ok. — But it doesn't tell ① How new PST's are discovered ② How this discovery process & t. use of t. new PST's fits into that scheme ③ No mention of special Lsrch for early IND probs.

.36 One way to put t. discovery of new PST's "into t. act. we consider t. "act of updating" to be updating of t. PST Grammar, — with pc's assoc. w. various probs, & ee's & G's. — thus Grammar "discovers" new PST's by assigning significantly higher pc's to them than befo

9/10/02 SEQUENCIAL CODING for QA & Time Scores (.20) Maybe BIG DEAL! 768

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

0:767.40 : So, in 738.36–739.03: consider that t. "update" always means update of t. complete (GPD, [PST]) Grammar — but as with previous tests updating, the way it's done, the amount of time spent on finding "new" pst's — this will slowly change as TM "matures".

.06 In t. Forego, IND problems, for a long initial "era" are always done using t. same PST: using t. dfns of concs (as in AZ141) & t. defns of contexts (739.00–.40). Eventually, when many optzn probs have been solved, & we have good PST's for optzn, we will begin to try using diffrnt PST's for induction — & eventually, t. PST's for induction will be part of t. (GPD, [PST]) grammar.

BUT it may be possib to have a stand-alone pure QA system & staying t. "initial era" of .06 and go fairly far. [It might be good enuf for SM! — Alone uses very old historical data, TM could discover things befor tu rest of t. SM & ways does. — It could be a useful kind to TSQ!]

.06 [So, w. B. considerations of 767.36 – 768.11 & 738.36–739.03 it becomes possib to define t. whole system

Perhaps have section devoted to .16 stuff called "putting it all together" (& summary)

[SN] for QA (& possibly ordinary Sequencial predn) we can simply code sequencially — looking for extensions of t. first "Best" code that t corpus pc.

We will retain several "Branch" codes at each pt, if we can, by oversearching — At any pt. in t. srch, we may have several codes for t. recent "Branch". Assoc. w. each is a pc & a cc. We like solns of hy pc & low cc, so we will tend to "remember" (or "Backtrack to"), trials of hyest $\frac{pc}{cc}$. (= lowest Loss = $\frac{cc}{pc}$).

This will work somewhat, if we use t. rite language.

When definitions are made, they are put at t. head of t. code, so all subsequent code can use them. (Start) After we have t. definition set, we can try to reparse corpus, to Maxz pc. This reparsing may be easy: we can have varying amts of "L.A." & try all parsings that are possl. for t. next "k" symbols. "k" tells how much "Look Ahead" we want.

We have a partially ordered set of partial codings continuations of various lengths, We could use t (ln pc/symbol coded) as a score to compare alternative trial parsings. This sounds reasonable Put it "On t. stack" → 611.35

Later very relevant ideas: SUM(CS): 739.22

.30 [SN] 511.26 – 512.21 V.G. Summary of TM ideas! i.e. ideas since 589.
512.22 – .90 Dcsn of "Lsrch"

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 768.40 : Continuing the review,

739.0009 CONTEXT is in searching for IND codes v.g. Very impt.

740: on the a priori $\frac{1}{1+X(\ln X)^2}$: Its properties; How to use it.

How to tackle it for different applications.

742.00 – 745.90 on TSQS 7:

746.00 ff This may be still be a serious unsolved problem! A hour finds some new PST's, say.

How do we "reward" this discovery? Why is the GPD that implements that Hour better

than the GPD that didn't implement it? Did I solve this recently?

761.17 is a review of 746.00 – 761.16 (This is the problem of 746.00)

At 762. . . ± Not 760.16-.18 solved in Case of 414.18 for the $(GPD_2[PST])$ drawer

Now, I'm not so sure! Where is the Motivation to find the PST's of my expected G.?

760.16-.18 may get a v.g. Evaln. for all possl. PST's: so if we ask the v.g. one,

we should get honest advice!? However, I don't see what looks like

"Reward for discovery of v.g. PST's"! (This was the original problem of 746.00!

If we satisfy the Cond of 414.18 wrt $(GPD_1[PST])$, will we get the kind of

"Good PST selection" that we want from GPD_2?

Well a possible soln. to this problem: The Case of 414.18 does get a

good GPD_1 d.f. for the PST's, but the form of the soln, is usually not directly

usable to find GPD_2's (PST's) of my "expected" G. Finding a V.g. (expected)

PST from the GPD_1 d.f. is can be a major discovery process.

It is a difficult optim. problem. Given GPD_1 i $Prob_j, cc_j$, find

a # PST_k → it is, on the average, better than any other PST_l.

We have a certain Time (CC) to do this in. We can't test out all

other PST's, so the Conc. is really not entirely deserved (yet).

See 762.27 –.40 for more discn. of the "soln" of (.20).

In general, it sounds reasonable, but I'm not completely sure of it,

by any means!

759.13 (The Multiple $\prod_{\gamma}$ integral) may be a reasonable

approxn: It may be realizable (perhaps) by Monte Carlo approxn,

We compute $\sum_{-\infty}^{G} P_j \cdot cc_j \cdot dG$ for PST_j's that are reasonable candidates,

Then we may be able to do 759.13 via M. Carlo.

Or, compute the $\prod_{j=1}^{n}$ part of 759.13 $(= \gamma(G))$, then find $a \to \int_{-\infty}^{+\infty} \frac{P(G)}{\int_{-\infty}^{G} \gamma(G)\,dG} \cdot X(G)\,dG$ is Max

10/11/02
IP

770

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00: 769.40 : Well! t. "Soln." of (760.16-.18) (769.20) may solve 746.00 (t. "Discovery" problem), but

Actually, t. "Soln" of 760.16-.18 is t. same as t. proposed "Soln" of 747.21-.22 — which I rejected at that time! I think t. Main criticism is 748.25-.306 —

i.e. if all strats are considered, some very A.H. strats would have very hy scores on past data, but would be unlikely to be good in future

Hvr, GPD has to consider t. Q! Given t. dcrn of PST_i & its past record, & t. records of all other PST's that have been tested, what is D.F. of PST_i for

.0 $Prob_j$, cc_j? (which has not been tested).

.11 Well GPD, can (perhaps SHOULD) look at t. complexity of PST_i as one param to help get t. desired D.F. GPD_1 is a single function that has been evaluated (maximized for 414.18 in theory) over a very large data set of ⟨$Prob_j$, cc_j, G_k⟩ triplets.
Remember this GPD_1 function has to operate (along w. GPD_2 creation) in a very finite time, so it probably should take length of t. PST into account in trying to predict its future behavior

.18

Does .11-.18 xlate into SM strat. evaln.? In t. corresponding problem

.20 — We have a set of strats that have been reevaluated on various sections of a time series of SM quotes. (This includes t. same strat on diffrnt time intervals.) ← This is a bit Mindful of "Cross Correln". I guess what we want as corpus is:

.22 [strat dcrn i, section j of data, G_{ij}, (no. actual data) (length of new seg. of data), G] we want P.d. on G.

This is a corpus of haxtups (like .22) of pentups. We want a function that gives good G values for th. actual data (t. first 5 args are "Q", th. last G is "A").

To what extent should our function consider t. length of strat dcrn?

.30 A funct that would seem to work for SM, would use t. μ & σ^2 of a strat on past data to predict future d.f. of yield. This would give hy 414.18 score on most strats that have been considered. (particularly if we confine our corpus to somewhat successful strats of short dcrns ☺)

.33 (STEIN) What about STEIN? For strats of length ℓ, T. overall average yield will have $\mu = 0$ & σ^2 will ↓ w ℓ. say $\sigma^2 \propto 2^{-2\ell}$. This Gives t. approx. So if t. (mean yield) of t. Strat. is $\mu_0 \pm \sigma_0^2$ This averages w. $\mu = 0$ w. $\sigma^2 \approx \frac{1}{2^\ell}$ so $\mu_0 \to \left(\frac{\mu_0}{\sigma_0^2} + 2^\ell \cdot 0\right) / \left(\frac{1}{\sigma_0^2} + 2^\ell\right)$

$\approx \mu_0 \cdot \frac{1}{\sigma_0^2 2^\ell}$ for $2^\ell \gg \frac{1}{\sigma_0^2}$ | actually — $\sigma^2 \propto \frac{\tilde{\sigma}^2}{2^\ell}$ (for proper dimensionality)

so $\mu_0 \to \approx \mu_0 \cdot \frac{\tilde{\sigma}^2}{\sigma_0^2} 2^{-\ell}$ which looks pretty bad! — tho maybe σ^2 does ↓ as rapidly as $2^{-\ell}$?

774.12 on STEIN Approach

9/11/02 TP

→IMPORTANT←

BIG UNSOLVD PROBLEM .07-.08

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : 770.40: It seems that I've not solved t. problem!

.01 Review of 769.11 – 770.40:

761.17 reviewd 746.00 – 761.16:

NIPS2 [12.20 – .40] (+13.00–.03 a bit) May Solve these Problems! — looks very likely!

At 769.11 I was concernd that t. apparent soln. of 760.16–.18 (t. general GPD on PST's using t. 414.16 Grvs) did not address t. "discovery" of new PST's —

.06 which was hauntingly important. 769.20 ff deals adequately w. this (769.20 –.30, + 762.27 –.40) } Also Note NIPS2 12.13 04 this

.07 Hvr, t. main problem of 746.00 – 761.16 was not solved — t. idea that very complex (A.H.)

.08 strats would work t. problems rapidly w. hy G, yet not extrapolate into new problems. See .30

.09 Some of t. ideas in 746.00 – 761.16 addressd by Q: i.e. t. 3 methods

.10 of 762.11 –.15 ((1) SOY, (2) SM strats evaluated by coding, (3) A funzn of t. GPD→GPD₂ (integrals equ. product.))

762.20 suggest reviews of AAI, soy, SM strats etc. PST eval, STEIN.

770.00 – .10 discusses inadequacy of GPD grammar in 414.18 Gore.

T. rest of t. page discusses possl. approaches that don't seem to work.

Except 770.33 – .40 on t. STEIN approach — which might be made to work. (along w. t. S's a' π of 762.15 → (753.13)

.19 [SN] (Stabindark)! Re t. Sπ eq. in 753.13! Could we put something like it by

.20 using, for GPD, often not raw 414.18, but for each prob'ty assoc assoc w. (PST_i, $Prob_j$, G_k) we take t. a_i power a_i being t. aprip of PST_i. This means that for small a_i t. pc will be ≈1 & not very dependent on t. pc assignd by π. function being opted by "414.18". This means t. function works harder toward optzng. data involvd. w. (simple short) PST_i's

As a result, t. function would be less accurate for small a_i PST's!

Hvr, they are given less wt. in 753.13 (apu) → (GPD → GPD₂).

[It's not clear that .19 help 753.13 any!

.30 I want to keep t. problem in mind: i.e. .07–.08 — That optimizing 414.18 would still give diffty of .07–.08 That Bad, A.H. PST's would get v.g. G distribns wrt. t. past data, yet extrapolate poorly into future.

.30 Is t. PST evaln. problem equivt to foll'g?: Given a corpus [Q_i, G_i]

.35 to find d.f. for G_j given new (or old) Q_j? Here, [Q_i], would be t. [PST_j, $Prob_j$, cg_j] [PST_j, $Prob_j$, cg] in t corpus. If PST_j were constant in t. corpus then we have only [$Prob_j$, cg_j] as Q_j.

Another view: Go back to t. corpus [PST_j, $Prob_j$, G_k] (includes a_k). Say we found a fn shunt that fits t. corpus well. (1) Could we usually find a A.H. PST that would give

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 771.40 : Spurious results for that function? ② Say the funct. was a universal pd. — like prior

Input was [redacted] n = +1

The fact that it's "universal" makes no difference for one input's "i".

Say the 3 inputs were S, Q, R. S decides the stock function. output is A.

If we sum over all S_i, there are probably several S_i that get my score, and yet } ??

extrapolate badly for certain (long, A.H.) Q's (??)

.08 Go back to 771.34-.35 : Say the corpus is $[Q_i^1, Q_i^2, G_i]$ (always G_i is real)

We fit a good funct. viz. 414.18! Given PST, Prob, G Q_i^2, can one always find

a Q_j^1 (A.H.) s.t. the stock funct found, gives "spuriously by" [redacted].

G diff ?

perhaps that's suspenseful: PST's are functions. We have a universal choice over

[redacted] there. So any Turing function is legal.

In .08, Q_i^1 is the function, Q_i^2 the problem, & G_i is the G given. (G/function can be open or closed)

Say we optimized via 414.18, & O^j found/over corpus (PST, Prob, G).

There will always be many (A.H.) PSTs that will get my score. Now O^j could

look at the lengths of the PST's; But I don't see that it has any Motivation

to do this! In particular, if the PST's in the corpus have always been

short, [redacted] O^j has no reason to believe that long PST's are Bad!

Could this problem be due to "No Free Lunch"? I.e. if a [redacted]

induction worked well on one sub corpus, it must work bad on another

sub-corpus. The "other" subcorpus is the one w. "long inputs".

[Is it possl. that a very smart [redacted] O^j could look at a PST & realize that

while it worked well on the past corpus, it is unlikely to work on the new problem.?

E.G. I could do this: I would probably recognize an A.H. PST! But perhaps some

PST's — while quite A.H., are not easy to recognize. The only clue is there length.

It may be that if a TM has enough experience in various parts of logical

reasoning, Math, & Statistics, — that it could be suspicious of excessively

"long" PST's.]

However, TM may not be able to become very smart before it learns to deal w.

this problem — i.e. Being able to solve this problem may be necy. for much smartness!

In this case, we must start out w. some perhaps "ruffle dirty" ways

to deal w. the problem. e.g. In the $GPD_1 \to GPD_2$ known,

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 772.40: I could find a way to cut off PST's of "excess long PL". — perhaps use some trick like that SM method of using strategems to code the corpus.

Another way might be the eq. of 753.13 (S' of $\prod_3$ S ...).

So we use these ruff up dated methods until methods Until TM is smart enuf to figure TM is smart enuf to figure out something better.

This assumes that TM will at this point, be smarter than me....

So it must be a fairly advanced system at this pt.!

One Approxn: Take 10 or 20 PSTs of highest mean G. Restrict to not have long codes for PSTs: Use some perhaps use simple Mt Carlo method to check Dif. for GPD2 (equal wts for all 10 or 20 PSTs)

A reasonable approach to the MAIN Problem: In QA induction, after a uniform set of Q's; say we suddenly start an entirely different kind of Q sequence! The ms error will be a lot. Tho the convergence theorem is still true, the function needed to generate the "true answers" is much more complex than before.

Similarly, if our O³ for "(PST, prob) → G" induction included only short PST's of a certain kind, then it should not be expected to work well w. long, A. H. PST's.

In QA induction as well as the PST evaln. problem, whether a new Q is "near enuf" to the old Q's so that we can expect as good Answers as in past corpus ..... I don't know how to deal w. this.

E.g. Say we have an Encyclopedia-type QA TM, and the answers have been O.K. for Q's about Physics. We don't know if it would do well in Biology — whether the Encyc has much Biology or whether TM knows how to Access Biology info from the Encyc.

ABCD D Pilot
ABCD D Expt
550

In General, we Dont know what areas the QATM will be "good" in from knowledge of its "corpus of Excellence" — (This is true of Humans as well as TMs). In the case of Humans, because of "common sense" based on Humans, we can often Guess well, by "Empathy" & our own Experience How, in "finding the best" of a set of PSTs (or strategems SM), we seem to automatically range over an area where TM's induction is inaccurate, & we pick as "best" a PST (or strat) that looks v.g., but which TM has very low predictive accuracy. The result is that simply taking what (QATM says is) looks like the "best" start PST, is a catastrophically poor choice.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : 775.$01 One "rule of Thumb"! Don't consider PST's that are longer than the longest Corpus PST. Have lots of corpus PST's of near your max length.

"Cross Validation" of choices of "Best" will "reduce" error (at cost of effort), but there will always be sufficiently A.H. dcuns, that will give catastrophically bad results. So "Olde Faithful" will not save us!

It would be nice to have some way of estimating how much accuracy we can expect for QATM A's that are "distant" from the corpus for Q's that are

— How to measure this distance of a Q from the set of Q's in the corpus. Looks like a "Clustering" problem

A STEIN ish Approach: (say SM strat Evaln): Say ℓ is dcrn. length of strats: For SM Data batch size of S. (no. of data pts). Get D.F of yield of strats of length ℓ on corpus of size S. This can be done by Mt. Carlo. Mean will be ϕ. — we want σ^2.

< It is possl. that this is essentially meaningless — since a stat. can be of arby large σ^2, if it bats enuf. We might get around this by having but be kelly-type, & pc of yield be mean since start of sample. —

Some "starting diffys", perhaps.

They

<We can perhaps restrict our strategies so that will not have enormous σ^2s

.19 — .21 is one direction to try. >

For PST evaluation, this is a QA machine, for the corpus, a O^j will give a pc for each A_i We can look at the μ & σ^2 of inpc. of a O^j —

Hvr, most evaluating functns. will assign pc = 0 to at least 1 of the A_i: so $\mu = -\infty$! No! (.35)

We could simply discard those O^j. — Still, perhaps most O^j would assign very small pc to one or more A_i's. Say we put data in pc domain & reject O^j's w/ any zeros.

Actually, what we want is the D.F. of pc's of QA's — μ & σ^2 may not be best way to talk about the d.f.

It would seem that a "pure" STEIN approach might work!

(.26) Actually the d.f. of G can always go from $-\infty$ to $+\infty$ — say: Gaussian D.F. or may allow D.F.'s; so pc = ϕ can't occur. Also, we may want to consider ln pc because we ultimately use the products of the pc's (the "eval'n" (= "pc of corpus"),

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00 : 774.40 : Anyway, back to STEIN : a PST train. : output for each PST, prob, is a / continuous usually P.D. on G. from $-\infty$ to $+\infty$.

So for all PST's of length l, for the k / empirical problems, we get a vector of pc à lnpc's. We then, by MT Carlo estimate get a d.f. of lnpc, for random problem in list, over all PST's of length l. This gives us the needed a priori of PST's "pc of G".

~~For~~ to individual empirical M_i à σ_i^2 of each PST_i, we can be statistically combined w. the ensemble M à σ^2 to get the "true" M for that PST. — we then pick the "best" / "soy" is relevant only when the ordering of expected means is affected. <I forget just under what conds $P_{4,2}$ is true.>

Anyway lnpc domain: Say the M_0 = M of ensemble, σ_0^2 its var.

If M_i à $\frac{\sigma_i^2}{k}$ are data for a corpus of k problems, for PST_i.

Then "true M" $\approx$ $\left(\frac{M_i \cdot k}{\sigma_i^2} + \frac{M_0}{\sigma_0^2}\right)\left(\frac{k}{\sigma_i^2} + \frac{1}{\sigma_0^2}\right)$.

If $\frac{1}{\sigma_0^2} \gg \frac{k}{\sigma_i^2}$ then $M_i \to \approx$

$$\frac{M_i \frac{k}{\sigma_i^2}}{\frac{k}{\sigma_i^2} + \frac{1}{\sigma_0^2}} + \frac{M_0 \cdot \frac{1}{\sigma_0^2}}{\frac{k}{\sigma_i^2} + \frac{1}{\sigma_0^2}} = \frac{M_i \frac{k}{\sigma_i^2} \cdot \sigma_0^2}{1 + \frac{k}{\sigma_i^2}\sigma_0^2} + \frac{M_0 \cdot}{1 + \frac{k}{\sigma_i^2}\sigma_0^2}$$

say $\alpha \equiv \frac{k}{\sigma_i^2} \Big/ \frac{1}{\sigma_0^2} = \left(\frac{\sigma_0^2}{\sigma_i^2} \cdot k\right)$

$\alpha \ll 1$

$$= \frac{M_i \alpha}{1+\alpha} + \frac{M_0}{1+\alpha}$$

So $\approx$ $M_i \alpha + M_0$

$$\approx M_i(\alpha + \alpha^2) + M_0 - \alpha M_0 .$$

So if we are comparing many PST's compare $M_i \alpha = M_i \cdot \frac{k}{\sigma_i^2} \Big/ \frac{1}{\sigma_0^2}$

since k à σ_0^2 are same for all PST_i, we compare $\frac{M_i}{\sigma_i^2}$ only (M_i, σ_i^2 are pure numbers, so there is no "dimensional consideration". in lnpc

In lnpc domain. This, of course, is ok if we have Gaussian D.f.'s.

Since they are not, we may want to use some kind of M. Carlo or MT Carlo analytical methods. — Perhaps Mix w. my method of pc of HR winners!

Note that we have to take 'lnpc's because we are interested in products of pc's.

This result is disturbing, because for large l (PST length), we can

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 725.40 : always find A.H. PSTs w. v.g. $\frac{M_i}{\sigma_i^2}$ that are actually no good!

OH! This is comparing PST's of same ℓ. — But for diffrnt ℓ's we have diffrnt M_0 & σ_0^2. For each ℓ we pick PST$_i$ w. best $\frac{M_i}{\sigma_i^2}$:

Then we compare them wrt. $\frac{M_i \cdot K \sigma_0^2}{\sigma_i^2}$ [redacted] n's have all same, so compare $\frac{M_i \sigma_0^2}{\sigma_i^2} + \frac{M_0}{K}$

for large ℓ (A.H.), M_0 will & but I don't know about σ_0^2. Actually, σ_0^2 may not be so small. [redacted] M_0 may be small.

compare $\frac{M_i \cdot K}{\sigma_{i,\ell}^2} + \frac{M_\ell}{\sigma_\ell^2}$ as assumption of ℓ

Perhaps $\alpha \equiv \frac{\sigma_0^2}{\sigma_i^2}$ is not small so that approximation -1.20 ff. are wrong? — But it's not really important. — The comparisons can be done w.o. approxn.

$\frac{M_{i,\ell} \cdot K}{\sigma_{i,\ell}^2} + \frac{M_\ell}{\sigma_\ell^2}$

Also, I may be able to get empirical Mt. Carlo D.F.'s for the [redacted] ensemble d.f.'s for various ℓ values. — Then do some kind of M. Carlo picking of best PST$_i$'s.

It would be good to go over the general meaning, interpretn, of the STEIN model, & see just how relevant it is here.

In case of Baseball Batting averages: when σ_i^2 per player is ϕ, then estimate is ensemble mean (if available). If $\sigma_i^2 = 1$, clearly ensemble mean is quite important. As $\sigma_i^2 \to \infty$/player $\to \infty$, we use all wt. on individual player.

For Model, we assume we know ensemble M_0, σ_0^2. Then to get d.f. for each player: It is product of (a) a prior from ensemble, (b) a/priori d.f. from times at bat. For Ry problem, the β distribn.'s v.g. because batting averages are on $(0,1)$.

I had some trouble deciding on an Aprior for the Batting Averages Problem — Don't remember just what it was; But small changes in it seemed to give large changes in Batt. ave. estimates! So, we should then be able to use the empirical data to get a good aprior! —

Which aprior gives the best apsif of the data?

5/13/02 ED

No. Params ≥ no. of Data Pts : ∴ .00 – .18

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

→ 778.00

.00 : 776.40 : Large l corresponds to using larger no of params than data pts, & gives nutty results! (→ Not nec'ly so Nutty! — See .03 ff & 778.00 ff) So we should at least limit size of l to be < ∝ no. of problems (or whatever) or whatever

.03 (Tho I may be able to deal w. no. of coefs > no of data pts in a rig. way — T. wt of a model is ∝ [struck] t. the h. vol in h. space of pts that bring us with a h. ball of radius ε about t. data. ∴ — Actually [struck] in this case, I think one should get better predn. than using fewer params than data pts! This is because a greater no. of models can be considered.

[struck] One way to do this h. volume is to assume each param is on (0, 1); or scale it so that is true. This enables us to compare h. volumes in different nos. of dimensions. →

The General ALP method samples from all pts in param space but wts ∝ Gone. A huge trick! By sampling over pooh only, we save time, but get less accuracy. Then go a Gaussian random distance from this pt for sampling; Mult by Gone over corpus. Pick random pts that satisfy constraints exactly. For the predictn. An ... the small mean i.e. ... of predn. are good curve.

Hvr, in order to do predn, sample from space of pts. that satisfy constraints.

— (G) is it possl. to use for pp. to evaluate ANN — [struck] when no of [struck] wts say too > no. of data pts. → 778.00

.14

.19 For report! Since it is "just a report", it doesn't have to be "complete" in any sense. I do want to write section on QA predn,

.20 [struck] Early IND update using defined funct & then defined contexts (sequential coding w. Backtracking).

Then, updating: IND, OZ, IND

For section on Lsrch: copy ∓ 575, 86, 89 or whatever, then add in para on ll Lsrch & Mt carlo Lsrch,

Do discuss PST's, EPD,

.29 Do have Section on "Putting it all together".

.30 But before (or when) I write report, be fairly certain that I have a good picture of t. entire system in my mind, (& in these notes!). ↗

I had a picture of t. system as a whole — evolving: t. dif'nt stages of sophistication of Lsrch, updating etc. — [struck] look in reviews to find it.
[738.31 – 739.17] (767.26 has refs ∓, & discussn.

9/13/02

"CURVE FITTING" .00

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

.00 : 777.40 (777.00-.40 B) 777.18 spec — spec: On "Curve fitting": Say we have an arby no of possl. params & an indep. aprgml. for each. a_i is i^{th} param; it has aprgml. $P_i(a_i)$.

If we have 5 data pts & want to use the first 7 params: We obtain t.p.d. for

.03 the next data pt., by .032 α: chose a random pt. $\vec{x}$ in 7 space using aprgml. It has Gorc $G(\vec{x})$

.04 of fit for the known corpus. Use $\vec{x}$ to predict the next pt & give it wt $G(\vec{x})$ loop to α (.032)

Do loop .03–.04 until the μ & σ^2 of predn. are acceptable.

Unfortly .03–.04 is too slow; very few random $\vec{x}$ will have useful wt.

.04 .091 So instead: β Chose 2 coifs at random using their margmls: Compute the other 5

coifs for best fit. From the resultant pt, distribute many trials for $G(\vec{x})$

using a Multivariate d.f. about that pt. Width of M.V.D should be enough

to allow G to ↓ by factor of ε, say. [redacted] make predn w. wt. $G(\vec{x})$

loop to β (.091).

For linear coifs (of possibly nl. functs), this involves solving 5 linear eqns. in 5 unks:

— No big deal. I think it is a good approx. & much faster than .03–.04.

Hvr, I have to "think this out" in more detail: e.g. the case in which

.20 the large corpus was generated by the first 5 coifs, but we are using 7. When we solve 7 eqns in

7 unks; we get diffy finding soln. We can usually find one, but the Gorc ↓

slowly as we deviate from the soln in a certain 2 space. Orth to that 2 space,

the Gorc ↓ rapidly w. distance.

Getting Back to "Main Problem": T. STEIN soln is 776.13!

.26

.28 Pick $PST_{i,l}$ ∋ $\frac{Kelli,l}{\sigma_{i,l}^2} + \frac{\mu_l}{\sigma_l^2}$ is Max. $\mu_{i,l}$ & $\sigma_{i,l}^2$ are params of G.d.f. of $PST_{i,l}$ — which is the i^{th} PST of item class l.

ensemble of PST's of class l.

.30 μ_l, σ_l^2 are params of

There are k problems that the PST's have evaluated in the past.

we assume $\frac{k}{\sigma_{i,l}^2} \gg \frac{1}{\sigma_l^2}$, but the analysis isn't much diffnt if this is not true,

We are pricing for max expected true μ of PST. Because of 50% effect.

the μ we can expect, will be < this. — But I think it will usually

be the "Best bet" for "best PST".

0/13/02

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 778.40 : T. (?) "solution" to t. PST optzn problem is by nomeans "complete":
What is lacking! (1) A method of (?) all PST's of length ℓ, that's not terribly wasteful. I.e. we want a method ∋ t. fraction of illegal or "obviously bad" PST's is small. (2) A method is needed to find PST's for which -1.25 is large. This could be a big discovery process. — Needing real creativity & ingenuity. (4) Not only "incomplete" but rather vacuous! Evaluating even a few $M_{i,\ell}$, $\sigma^2_{i,\ell}$ would take too long! — Say we are interested in PST's w. CB = cc₀; then each such evaln. takes cc = k·cc₀ and then Actually, we might use a smaller k to save time — which gives a certain undesirable bias for t. resultant srch!

STEIN

But, t. idea of t. STEIN analysis may be correct! That testing on k, problems is always a statistical test of how good t. PST is on t. average (long run). As such, if we consider many PST's, some of them will have apparent M's that are very far from their true M's, & STEIN seems to be a way to deal w. this.

.21 What I'm looking for is a grammar that will "pass thru" fit t. known data of PST applns to known problems & then by giving P.D.on Guesses of untested PST's. T. pd it gives is, for t. most part not A.H., since few of t. proposed PST's have been tested (directly) on t. known corpus of problems. (Tho it could be "A.H." in t. sense that one can see it will work well on certain of t. corpus problems).

[SN] T. [Stein] Analysis of (774.12–776.40; 778.26–.40) may be o.k. for [SM strategy evaln.]. We can often evaluate large batches of strats analytically — Do "hill climbing" on params of them, ect. The idea of fixing ℓ value, getting a "best", then try another ℓ value and getting a "best" seems good. Gradually ↑ ℓ until t. strats start getting

.36 worse.

.37 In General, the PST optzn problem doesn't seem to be in v.g. shape: (.21) ff is about as good as I've Gotten. T. problems of 746.00 & 771.00 It isn't really been solved: tho t. work revised on 761.17 ff & t. work of 771.00 – 779.36 is probly useful in giving insight on t. problem, & solving some special cases.

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

oo: 779.40 : 777.19 -.40 Has a discussn of t. overall "Report". One t. imph. ideas that needs Development/explanation is t. sequencing/coding for induction, (using some backtracking) & "context.

If t. soln to an induction problem can be expressed as t apln of a conc. net as was done in 589, then it would seem that we can (recruit by "sequencing/coding"

Hm: One objection I had to t. TSQ's I wrote in Saarb, was that the concepts used in a subconcept of trials, were always complete solns to problems. I think that this may have to be true if I write TSQ's based on conc. nets — in which each needed conc. is actually a soln to a problem (not a subfunction in a soln.)

Some possl. cases that I might try to get TSQs to learn using them be complete solns. to problems: x^2; translational invariance of recgln. of funcls.; It may be that t. CJS for concs I find in this way are usually larger than those that are always solns to problems.

Also note: I noted this verbally in CJS as I progressed in the Saarb. TSQs: This might have been mitigated if I used "I shall learn" properly, to ↑ pc of concs "used again" for t. first time. — Tho I'm not sure of this — in t. Saarb TSQs, I always "derived" each problem soln. automatically. Perhaps if I didn't do this, I'd get a lower pc of a problem soln. that was "reused". One way to deal w. it:

Any soln. of a problem is a "constructively 'derived conc'" — which means we will try to use it for OSL. In general, any subnet can be used for trial OSL, but funcls that are solns to probs are "more likely" to be useful. We do not have, want to make this, "more likely", part of t. apprp: It will be a "heuristic". ← T. apprp. never changes

SN In 589 when I had that conc. net — as a model of how t. learning took place: One of t. things I was missing, was to "Recognition Algm": t. part that realizes that a certain function

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00: 780.40 should be applied in this problem. T. conc. net as I drew it, was applicable to solving individual problems once you knew that it was needed a new function to solve it. [once you recognized it.] So that stuff I wrote about "Recognition functions" (obs) was really crazy! → But see (29)

As a heuristic trick, I divided the problem into 2 parts; (1) Recognition or how to know problem different from old (2) How to turn Q into A (op). So look at that stuff: One of the old "CPM" trial docs writes about it. Also, look at original ID pp. — There is much confusion, revision, bugs found etc, (perhaps!)

The CPM papers (Sept 8, 01; "CPM".Tex) (Nov 8, 01): are in C:\Ray\ Conprob.Tex

The 'stuff' in ID around 431.00 ff is probably better is viz.: Much of that stuff can be used for "Expo" as written.

Re: the Recogn ("R") funcs! The correspond to "Analysis" in Science! Breaking down domain into subdomains that are "coherent" enough — however enough is.

Merging of Domains corresponds to "Synthesis" — realization that 2 Domains can be interpreted by having common soln.s to their problems.

Initially I will try to have just one R. (Eventually no R!)

Then teach other R's w. indices — then remove indices, etc.

By the time TM has to update R's using the 5 methods of (CPM Nov 8, 01), it will be rather good at induction! (I hope!) — Do say this in the report

29 Re: Conc. Nets: T. learning of the appropriate "R" funct could be
30 part of the conc. net; Both R & the P have to be trained. I don't have clear ideas as to how this is to be done, however!

Anyway, I want to write now: 777.19-.40, 780.00 ff: the overall way TM develops — sequentially. — What the SR's look like, etc.
My present idea is that one could go pretty far w. QA TM only: No very advanced robotic systems: try to stay w. Conceptual ideas! Try to do (29) so that Recogn funcs are also "learned" in the conc. net"

Also, the idea that all heuristics can be expressed as modifications of the guiding PD. — (The basic idea is not used in early induction

The Primary Plan: That if one can make a conc net for a problem, then the problem must be solvable by simple conc. data. only. That use of Context should ↑ pc! Also use of R (which is a kind of Context) should ↑ pc.

20: 781.40: PLAN!

First: QATM only: Use only 1 "R": use Conc Definition: Tentatively define each soln. found (780.29)

See if I can get it to discover concs that are not solns. to problems.

Try context for cont. data. Devise rules to express Contexts. See if it helps Much.

Then R

[A heur for Context data: If find conc. that occurs often! Try to find Context that would ↑ its pc. — Try R's for many different concs.]

Or: A proliferation of prefixes: (Those that are "context" v.s. those that are not) or soft concepts; And on prefixes

.05 DEF Def "A context is any identifyable precondition that may influence the pc. of a Conc."

— A problem type (R) can be a Context. Recent concs inserted, can be contexts

See how fast — ln pc increases w. n, the no. of probs Thus far. See if (R context) helps.

Perhaps start w. (d) induction, then try to teach (5) induction.

first, use Bern. model only. (Perhaps later introduce idea that

Def. of A_i can be any funct of Q_i — How to do this is unclear at present, —

Tho maybe it's clearer than "How to get TM to learn Bern induction.

— To some extent, Bern induction is "built into" the apprd, — but this is just the apprd of the O's functions. This is different than relating the pc of A_i to Q.

{I really have (Thot/written) about Bern induction or Sinduction hardly at all!}

At a certain pt., (after" context is introduced perhaps) I introduce a few R's with indices in the Q's. Then try to get TM to discover relation of Q's to R's so I don't need to put in indices.

Then see if I can get the 5 Heurs for updating, to work.

Perhaps write up the section on QA lang. so it is complete & understandable as is. It will not (yet) contain the updating methods of GPD$_1$ → GPD$_2$ using PST's. These will be introduced after it has learned to solve TNV & O2 probs. (& Generalized Optem probs).

I think this is a much better presentation method than previously — in which the reader would be reading lots of stuff w.o. understanding how it fits together. → (785.30) spec.

00 : 782.40 ! A section or so to follow N8,02 ! (conprol.tex: italics. {/em Analysis})

The discovery of new R functions corresponds to the (Analysis) phase of a scientific inquiry — breaking a domain into smaller parts that can be solved using methods peculiar to those parts

The merging of the R functions in ~~factorize~~ 5b corresponds to "Synthesis" — the realization that several apparently disparate domains, really have much in common and can be treated in a unified manner.

The history of science can be viewed as ~~a continued~~ a continuing sequence of Analysis and Synthesis operations ~~of the day~~

.00 - .12 was added to conprob.tex (in [redacted] RAY directory.) (Oct 17, 02)

This conpro essay deals w. an early part of the induction part. It is not made clear that we start w. $w_n = 1$. For a long time. Also, that Contact must be introduced — a just how $\rightarrow (-1.0g)$

New section: Practical Induction:

The previous section ~~is~~ was meant to give a theoretical background for designing a machine that will [redacted] be able to learn from a suitably framing sequence. We start with a minimal machine and add robinarmaits as its education progresses.

The initial machine will have a set of functions and rules for combining them — as in Appendix 1. At first there will be no provision for redefining previously defined previously defined ~~functions~~ recursive functions — only compositions of the primitives and ~~of the primitives used~~ functions.

Explain how it "learns" is done.

The early updates of ~~all~~ of (1) will be viz. [redacted]² simple Lsearch, variation of using the probability distribution on functions given in appendix 1.

Only now: There ~~will be no employment of~~ the recognition functions of section 1. will not be employed at that point — which markedly ~~simplifying~~ simplifying the updating process.

00:853:00 **BIG Q:** T. Main Heur idea is that all Heurs can be expressed as modifns [redacted] of t. "Guiding Pd": Hvr. QATM's Guiding Pd. is Constant, yet it seems able to incorporate heurs in to its [redacted] Guiding Pd.

So: .00 is "Paradox! Paradox!

.05 Well, superficially, a (useful) Context, is a modifn. of t. Pd. — a heur that [redacted] helps me find good OJ's for 414.18. Yet actually, it need not be, if we already have facilities for defining contexts, in t. defn. of t. [redacted] Language that t. aprip derives from

10 — Could we do something like .05—.10 for (all) heurs? (not just "Context")

[redacted] Perhaps, YES, If we were doing induction only. For OZ & INU probs that are not directly induction,

16 Note that t. aprpd can only accomodate heurs that operate thru modifn of t. Guiding Pd. (which is no real restriction!) But somehow, t. aprpd is not

18 supposed to have info about cc! (It may "leak in" thru t. use of CB < ∞!) (.30)

SN Jurgen's idea of (ĩ) an [redacted] aprpd depending on codes that are [redacted]

20 inclusive of cc or length, may be a good idea: Things that occur in t. world will tend to be of small cc, because it takes longer (more "money") to have ones w. larger cc. Perhaps have · [redacted] pc ∝ [redacted] $\frac{pc_i}{cc_i}$ ≈ Lcost.

I want something like t. probty of something being created.

Hvr, in many domains, t. available cc for t. Generation of t. data, is enormous! E.g. Astronomy, Geology, (Sociology, SM ? "economics") [redacted] Meteorology, Biology?, Physics, ? Chemistry? I Guess, in each domain, one must estimate what t. cc. of t "true Generator" of t. Phenomenon is.

30 (.18) [redacted].18 is probly impt. TM only has real experience w. finite CB's. → 785.00

00: (Spac. 784.30) Re: 3.754.00: I may be confusing t. original apriprd w. t. "summary apriprd".
T. entire corpus is coded, optimumly wrt t. original apriprd. Only t. latest new stuff need be coded wrt t. "summary apriprd."

03 Another perhaps relevant pt.! In t "sequential" coding of t. QA corpus

.06 I have all these definitions: when a new QA comes in, I can try incremental coding of the previous corpus & some of its alternative codes 2/0 (& this has to be done w. reasonable frequency) reparse all or part of t. entire corpus wrt t. existing definitions. This usually shouldn't take very long: T. time required

10 depends on t. "Look Ahead" limit. used. (.22)

.11 P67, column 3. She is disturbed by Wolfram's claim that "the principle of computational equivalence" has vastly richer implications than any single collection of laws in science" — Wolfram's claim

I find Wolfram's claim to be quite reasonable — to me, "computational equivalence" implies that the same techniques for inductive inference can be applied to all phenomena in the arts and sciences. This idea has yet to filter through the

20 collective consciousness of the culture.

.21 (This is part of a review of Wolfram Book.)

22 (.10) A hour can be often expressed as a defined function over a "context" defn.
— So its just a Big regularity in t. data. But often, a hour is specifically a way to do search faster.

So, superficially, it would seem that there are 2 pd's: ① t. original invariant apriprd.
② T. pd used to guide search for good codes. Initially (w.o. hours) they are t. same.

30: 782.30 1) Note 782.30! well, I'm not sure so long, but will see! Note 787.16-24 on t. naturalness of wn's: i.e. many R's. Starts (?)

31 2) Using only one R ($w_N=1$) for a long time, using defined defns & then context defns. Then when a new QA comes in, I do .04 → .10.
This may greatly simplify early TM — yet have it able to solve some interesting problems.

I really should go back to my ideas on Z141 that I wrote Wolff about! perhaps first try it on English (spaceless) text, to see if it works; then see if I can get it to work w. QATM

786.00

0/19/02 ID

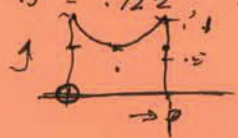
BACKTRACKING .20

00:785.40 : On this "Sequential lrng for QA": I'm uncertain as to what I was thinking about!
If I have "all of t. defns" then this includes solns to all QA_i's, $i=1/n$. If I had to "reparse" — this would be meaningless — T. best problem soln. (= conc.) would work for all QA's ($i=1 \cdots n$).
On way to "Reparse" O_n only. It has a tree structure & its derivations & defns.
A "reparsing" could take t. tree of primatives that represent O_n, & try to reparse it: conceivably using somewhat difrnt defns, or t. same defns in difrnt order or difrnt frequencies. (i.e. more random defns... non-uniform d.f. over t. defns gives hyer PC.) $.1^{.1} \cdot .9^{.9}$ vs $\frac{1}{2}^{\frac{1}{2}} \cdot \frac{1}{2}^{\frac{1}{2}}$ at $n \to \infty$, which often

$.1^{.1} \cdot .9^{.9} = .722$; $.5^{.5} \cdot .5^{.5} = .5$ only $\quad \frac{1}{3}^{\frac{1}{3}} \cdot \frac{1}{3}^{\frac{1}{3}} \cdot \frac{1}{3}^{\frac{1}{3}} = \frac{1}{3}$

$\leq \frac{1}{2}$ $\quad \frac{1}{2}$ "undecidable"

$p^p \cdot (1-p)^{(1-p)}$



> SN Does $P \stackrel{?}{=} NP$?
> Perhaps (probly) there are cases that are undecidable. —
> It "has to do". Bolsanov

Conceivably, I mite rewrite one subtree (or more) as an equiv. subtree of hyer pc. —
T. final functions same but its parts have hyer pc's so difrnt future pc's will be induced.

16 → [So far, I don't see any great value in this "reparsing"!] → 795.25 Good! — Also Note RMS of (795.04–.35)

20 BACKTRACKING! Probly best way to do this: say pc of O_n is P_0 : we need to code (Q_{n+1}, A_{n+1}) ① we first try Continuing O_n. ② If they dont work, we move back on O_n, a pc "distance" Δ (i.e. we consider all continuation of t. prefix of O_n that is of pc $\frac{P_0}{\Delta}$)
say $\Delta = \frac{1}{2}$; so $P_0 \cdot 2$; If nothing found, do $P_0 2^2$; $P_0 2^3 \ldots$
Another possy mite be to start at w. prefix $pc = P_0 \cdot 16$, say, and try random continuations: This would have to be do "w.o. replacement" in t. usual random search manner — otherwise, it's very wasteful.
The idea of "$P_0 \cdot 16$"; T. code for O_n is a linear sequence of symbols: to get to $P_0 \cdot 16$, we simply remove final symbols 1 by 1 untill t. product

30 of pc removed is 16^{-1}.
Unless generation of random continuations has some special value, normally its easier to just try all codes of pc betw P_0 & $P_0 \cdot 16$. These are generated by "branching", so we always retain t. machine state at t. "root", befor we branch. This assures us that we never waste time by repeating portion of a computation.

"a QA soln" could be several QA's that have same O_m soln, hwr.

Another way to backtrack: Go back 1 (or 2 or ....) QA solutions on O_n.
Normally, in solving a problem, we don't take t. soln. of best L cost, but continue
maybe t. "Backtrack index" should be no. of "O_n revisions" back

0/20/02 787
ID

probability → py?

T←2T

i.e. not just best pc.

00:786.40 searching for maybe doublestime, for better solns. we retain all good soln's in many, hvr. (we retain t. machine states at critical branches, so it's easy to code various continuations.)

Doubling Backtrack pc i redoing entire search, will waste time by at most, a factor of 2 — with T←2T (like Lsrch)

Essentially, it is 3c Lsrch, because we will use Tᵢ pcᵢ as CB for t. iᵗʰ trial.

Sometimes back tracking wouldn't help! TM just isn't ready for t. QA nt! we have to modify t. TSQ or abandon that branch of learning for a while.
<we would want to find out just why TM couldn't solve it, hvr. — what concs were "missing" or had to small pc.>

.89

0 In t early part of t TSQ, I will know a "good soln" to each QA. In this case, t. system "has to work" ☺: Later when I don't know t. soln., (or I may know t- final soln but not a heuristic path to it) I'll have to make "local Models" of TM & design TSQ's based on those "local Models". — (Like debugging a large pgm that one doesn't completely understand).

16 T. forgg. discn. is based on a known "Conc net" for t. TSQ. In t. case of a QA, TSQ, t. conc net must include "recognition" as part of t. problem soln. I really haven't thot enuf about that in detail. T. stuff in t. N.8.02 report, is a step. — but I'm not sure it's in t. rite direction. So perhaps that is one of t. main "Lacunae" (missing parts) in this early TM outline.

20

So, for t. report! Do N.8.02 (or latest version) w. associated additions (like t. conv. thm & its proof).

24

25 Add to t. report a much briefer dscn of how t. GPD, PST system works & how t. QA system is used to update it. Then explain how this system can be improved & how at some level of sophistication in soln. of optzn problems, we will get it to work on optzn of 4.19.18 for QA problems, & ∴ for itself.

.29

30 Maybe rewrite introdn. to explain .25 – .29!

. SN To what extent is t. set of defns & contexts a "summary Machine"?: (t. "set of defns" also includes all problems solns (These are to be used for OSL, as needed). It is a kind weak "Summary Machine". With Backtracking it's a much better Summary Machine. — approximate

33 788.12

T. problem of QA induction for d funcs is first thing done: Next, s induction: Much harder, will use continuous params as part of defns. I could do Lsrch w. 1 or 2 bit precision, to find "reasonable" discrete params, then "zero in" w. contin. params. Note that if peak is narrow, t.t is not so good. So one might use random search in param space.
— Yet, we be need S induction: "On t. other hand," if we start w. Baum

6/20/02
ED

Very IMPT → **SUMACS** (Summary Machines) .12

00: 787.40 : Induction only, It may not be diff to search for good structs.
But I have certainly not much worked out the details of search for Bayes models. This kind of thing is automatically done in the implementation of Theapriod via AZ141 — but I need to apply it to creation of structs: which is a new matter. One way is to define objects and contexts. This covers much (not all) of pattern induction.

So: An outline of the report is 782.30 (785.30) Pgm 787.10 — 788.04

788.20
788.20 — 787.09
on backtracking.

This will be it what I will treat in the introduction.

So in introduc., mention that the QA induction system could work as a stand-alone system for a certain class of induction problems. Subsequent work seeking deal in SNU's 02 probs: how the ability to solve these probs can be used to improve the QA system, & improve the overall updating system.

12: 787.33 : "Summary Machine for QATM". The Backtracking makes the system close to an idea "summary machine". Perhaps there are a few improvements to the system that make it a better Summary Machine. For one thing, it should save top 10 or 20 codes to get PD on A. (not just MDL). Backtracking is a way to effectively store many of the best codes. By improving the Backtracking using more stored info, it may be possible to speed it up & make it more effective (less confusion) if more codes available.

However, Go over QATM very carefully: to see if it is a Summary Machine! This is a very nice way to look at it: It may be possible to put other induction systems in the form of "Summary Machines" as well!

Sumacs ≡ Summary Machines

Another interesting/important property of Approximate Sumacs:
For a true Sumac, random input will give exactly right output distribution for future. If it's only an approx. Sumac, (say it represents only a finite no. of codes), then random input will give not quite right future distribution — but it's really not so bad! We can continue the sequence any way (This is done in practical RN induction), Just how this is done, is unclear. I'd like to be able to put in random input & get not really right outputs but for long enough output stretches, it would begin to figure out ways in the sequence that were not in the original Sumac.
One way to do this: the "pure" exact Sumac has say 10 start codes for cases ("unknowns") — each with its own wt. We could add another channel using the original (from T=0) again. Eventually it would begin to give correct predictions.
However, .27 If it very uncertain as to how (or if!), it works! → 784.00

Write a lot about how early QATM will work (Mainly for Me: But primitive!).

For "putting it all together" write up a Plan a sequence of capabilities of the system and sequence of enhancements to the system: (I have a section briefly on this) — but expand it: Also all actual remarks on difficulties & how to (perhaps) overcome them.

0/20/02
ID

0.0: (788.33 spec): A basic Q is: for various ("Potential Sumacs" induction systems): How & how effectively do they Summarize regys in t. corpus? How well do they enable detection of regys in future?
What they (typically) do not summarize is frequencys of ones that are not yet large enough for "definition" status. Being Able to do "OSL" is a special case of non-yet definable cases.

→ One possible Sumac: Just store t. [redacted] machine states of M. best codes. M can be very large: say 10^6 or 10^8. We can do this all in RAM or keep it in HDD w. ≥ RAM M 2 = Parts for swapping while processing. To properly update t. [redacted] this set of machine states, they all have to be tried on each new problem. In a sense, it is "Thorough", but even so, (discounting t. slow speed) is it "good enough"? — i.e. could there not be some much faster Sumac methods that would do much better for certain kinds of regularities?

Using Pr. AZ141 "UMC"; keeping all of t. "reasonable alternatives" at first, may not involve much storage; but t. no. of really trivial modifications will ↑ rapidly w. n. Perhaps by putting codes in t. equive. classes & just store one from each class, along w. size (≡ wt.) of each class. "Equivt." means they extrapolate about t. same.

In t. sugg. I've been considering discrete codes only. For continuous values, sting discrete data, location of peak (and precision to isolate it) & width (≡ wt.) of peak.

SN Most pred. & predn. will be in form of Bernoulli seqs. This rule is best stored as raw case counts nos., since updating & prediction can be easily done fast & exactly, from they data.

A Good Model of Human Predn. is that it is Sumac w. Backtrack, & perhaps a lot of storage of alternate codes. Ideally, t. amt. of Backtrack used (i.e. "Depth") will depend on CC available.

SN I had (in t. distant past) considered that writing t. TSQ was t. most difft part of t. TM project. I then found various diffys in my basic Understanding of TM. At t. present time, the Sumac Model idea & AZ141 (as a form of UMC) may be an adequate way to construct a prelimy TM. that could perhaps learn useful things!

So now, again t. TSQ design! I hadn't before (I don't remember doing so) any consideration of Recognition functions in t. Conc.-net.
In General, it may be that t. "Euclidification" of Human knowledge is Quite difft — even for very elementary Algebra!

I had some recent Ideas I wrote on TSQ that seemed to Simplify t. problem — try to find these notes

-33 → Perhaps it is possible to show that t. Sumac Model using AZ141 & backtracking & all codes saved, is about t. Best one can do in induction? — (Tho it would seem that other optzn. techniques would often be better.) → (790.00 spec

(spec 789.40): Well, perhaps not! T. model of 789.38 will, w. a certain CB, get a certain set of "Go (short) Codes". It does this by exhaustive search in ~~[illegible]~~ order. It is certainly conceivable that there are better ways to look for "short codes". E.G. An Alg. could look at the corpus to be coded & decide that it has a certain class of regularities (or is likely to have a certain class of raggs) — then it would look for those raggs.

An argt. v.s. .02: In order to do that, the system would have to know about a certain class of raggs being "frequent". Discovery of such a class could only be by earlier discovery of certain "short codes" — which could be done by the 789.38 model.

A NOTE on why I feel that the [illegible] INV, 02, (IND?) problems can be solved in [illegible] "not much slower than I know in, the no. of probs".

I expect, that w. adequate context, discovering needed abstr/solns of new problems should be about the same (unless the [illegible] new probs are, massively, "much harder"). Evidence for this: That it is true of Humans; I assume that Humans use some general context-based search for solns as QATM uses. Note that "Context" can be very general.

Discovery of certain Contexts can be [illegible] impt. notes on Concept nets.

T. impt. idea is that identifying proper context usually will adequately reduce the search space. If this is not possible, then a human would also have trouble solving that problem, for a certain problem.

I may be using the TSQ in different ways: Below, QA; & IND/02:

In QA, I have this concept to (partially) order rules needed for prob solving.

— Maybe not so different! In QA using Gumac; we modify the guiding pd so that search for soln. of the next problem is much simplified. This is true in IND/02 as well!

N.B. At the end of the section on QA induction, the reader should know how to use Lsearch (& other refinements) to work a TSQ: As is, the EXPU I've written is not adequate!

The Abstract gives the general idea of what's being done & roughly how things work: Then, the actual paper will start out w "a special kind of 02 problem", i.e. QA.

Perhaps at end of intro (as it's written):

omit: Before we tell how Inversion and generation problems are solved, we will describe a system that

first (search?) Because of the critical role of induction in the general updating process, we will describe a system for [illegible] a fairly General kind of induction based on Lsearch. Though this system can operate as a "stand alone" system with [illegible] facility to work inversion for induction only,

9/22/02 IP

00 790.40: We expect that its ability to do induction will be much improved after it has ~~worked many problems in inversion and optimization~~ been applied to the updating of ~~more general problems — i.e. inversion and~~ inversion and optimization problems.

After discussion of updating in "Analysis & Synthesis". — footnote for P3 of Oct 17, 02

05 (In the foregoing we have allowed $R(Q_e)$ to be 0 or 1 only; and with no overlap of the R functions. It is ~~possible~~ also possible to have a system in which $R(Q_e)$ is a real between 0 and 1 — and overlap of R's are acceptable. We will not discuss such systems in this paper.)

07 In the early training of the system, we allow only one R. — So there is only one "kind" of problem. It is possible to do a fair amount of useful learning within this constraint. Another simplifying constraint is that $O^j(\cdot|\cdot)$ take on the values 0 and 1 only.

~~To start off, we are given Q_1, A_1, our first update problem. We want a function $O^j(\cdot|\cdot)$ such that $a_0^j O^j(A_1|Q_1)$ is as large as possible.~~

This corresponds to a kind of discrete induction. It can be used for training sequences in parts of mathematics — in which there is only one correct answer for a problem. In this case $A_i = F^j(Q_i)$, and we want to find an F^j such that for all i,

separately + separately/line

$$A_i = F^j(Q_i) \qquad (5)$$

(equation number)

20 and a_0^j, the a priori probability of $F^j(\cdot)$, is as large as possible → P 792.03

21 To ~~start off~~ we have only one Q_1, A_1 pair. ~~We can use~~ Lsearch to find F^1 ~~in the following way: Using the technique of Appendix 1, to assign a_0^j values~~ first we add Q_1 to the "arguement list"

25: 292.06 Then we do an exhaustive search of all possible codes for F^j such that the time needed to generate and test F^j is less than $a_0^j \cdot T_0$. (T_0 being the time to execute one computer operation.) [The value of T_0 is not critical] If no suitable F^j is found, we double T_0 and repeat the search. This doubling and testing is repeated until we find a F^j such that $F^j(Q_1) = A_1$. If we can afford the time, we continue doubling for a while longer in the hope that we will find a F^j with larger a_0^j. If we do, we preserve in memory, all of the F^j functions that worked — though most of them will have a lower a_0^j value than the "best" F^j found.

[If we have only random F^1 satisfying F^1]

When the new Q_2 arrives, our guess for A_2 is $F^j(Q_2)$.

[Right margin: If there is more than one F^j in memory (F_i^j; $i = 1 \ldots n$), then we may have several predictions: The relative probability of the prediction $F_i^j(Q_2)$ is $a_0^{j,i}$, the a priori probability of F_i^j.]

When the true A_2 is revealed, we don't have to do any updating if $F^j(Q_2) = A_2$ for one or more of the F's.

33 If this is not true, then we have to update. If we have any F^j values in memory

Omit. { we try them first. If any work, we use the best (highest a_0^j) → 792.07 for prediction, but keep the others that work, in memory. Any F^j that don't work for both Q_1 and Q_2 are removed from memory.

If we find no F^j in memory that work for both Q_1 and Q_2, we start a new Lsearch for a F^j that works for both Q_1 and Q_2.

00:791.40 : I should mention that (when looking for function for Q_1, Q_2, F^j becomes a "defined function" in the AZ141 further down. <For myself at least, I should note that this practice isn't exactly rite — it's an approx. way to do one kind of OSC.

03 :791.21 To start off, we have one Q_1, A_1 pair: We want a function F_1, such that $F_1(Q_1) = A_1$ and a_0^j, the a priori probability of F_j, is as large as possible. Using the AZ language of Appendix 1, we first

06 add Q_1 to the argument list, so the output will always be a function of Q_1. → 791.25

07: 791.33 To update, we remove Q_1 from the argument list. We add to that list, the best F^1 that we've found and replace it with Q_2. We then do an L search to find a function with A_2 as output.

10 If we do, we replace Q_2 by Q_1 in the argument list and see if the output is A_1. — If it is, we have a successful F^2; If not, we continue the search, with Q_2 in the list rather than Q_1. We continue searching until we find a F^2 such that $F^2(Q_1) = A_1$ and $F^2(Q_2) = A_2$.

[This enables us to build upon the functions we've found useful in the past.]

This routine continues as we update Q_3, A_3; Q_4, A_4 ....

Each time we find an F that works with all of the examples up to present, we put that F in the argument list.

In updating, we "backtrack" as little as possible. Suppose F^n works for Q_i, A_i $i = 1 \cdots n$, but $F^n(Q_{n+1}) \neq A_{n+1}$. With $F_1, F_2 \cdots F_n$ in the

20 argument list, we do a search for a F^{n+1} that will work with $Q_1, A_1 \cdots Q_{n+1}, A_{n+1}$.

If we can't find one, we try a search with $F_1, \ldots F_{n-1}$ in the argument list, instead.

EN) T. Long. Sounds a bit unreasonable: why should ↓ content of arg list enable to discovery of a better F^{n+1}? YET I think it Was to! Otherwise no real Back tracking: No close Simulation of SUMAC! ↓ arg list ↑ pc's of things a lot, because individual elements of list have their pc's ↑. This may be discussed in section on "Scaling".

(Don't type this!)

If unsuccessful, we remove F_{n-1} from the argument list and try again, ect.

Has an effect similar to that of backtracking.

Keeping many alternative, non-optimum solutions to problems, in memory ↓ They both give useful alternatives when the present trials fail.

30 : Of the two techniques, backtracking is much slower, but more exhaustive of possibilities.

Having many alternative non-optimum codes gives us a way to estimate the probability of a prediction.

33 (Section) Scaling. As we increase the number and/or variety of our problems, the search times for the simple system just described, increase very rapidly.

Suppose we have just k functions and constants in our list when we begin our training sequence. If we add one function F^j to the list each time we solve a new problem, the probabilitys of a single code element will be roughly $(k+n)^{-1}$ for n problems. In L search, that total search time is proportional to the

time needed to generate and test an acceptable solution, divided by the a priori probability assigned to that solution. For k = 10, n = 20 and as little as 5 symbols in the solution, we will have a search time of about $(10+20)^{+5} \approx 2.4 \times 10^7$ times the processing time of the solution. This would be the factor if we just were successful in searching using all 20 previous problem solutions and didn't have to backtrack. Backtracking once produces squares the search time to a factor of about 5.8×10^{14}, which begins to get impractical. Backtracking further would be completely impractical, with the factor at 13.8×10^{21}.

summit

There are several devices we can introduce to deal with this problem. First, the use of ℓ recognition functions, R_j, effectively reduces the values of n (the number of problems) by a factor of about ℓ.

Another great reduction in search time can be obtained by context discovery. As we have described the AZ language, the probability of a particular symbol at any point in a description is independent of the recent previous symbols, and is independent of the kind of problem or sub problem being solved. By introducing "context recognizers" we can take advantage of the fact that certain contexts, certain symbols are much more likely than others. Suitable designed contexts can greatly increase the probabilities of symbols and greatly increase the probabilities of correct problem solutions — thus decreasing their associated search times.

perhaps discuss English prefix & context dependence.

How do we define contexts? The simplest kind of context is a preceding symbol. In prediction of English text, the "space" symbol is very important because the probability distribution of symbols following it are quite different from the all over average probability distribution.

More generally, any subset of such strings of the 5 symbols may be regarded as a variety of context for the following symbol. Examples of contexts: the last symbol is "sum"; the last 2 symbols are "plus 3"; the entire sequence has an odd number of symbols; the number of symbols is a prime number, ...

insert 794.19-.20

Context can be extended beyond the immediate code string. We can include the problem being solved (Qnm, Anm) and/or any number of previous QA pairs. We might want to include previous solutions to problems, if this information is/available in the earlier part of the function being coded. In human problem solving, a more global context is quite important. We want to know how and where the problem arose — from Chemistry, Physics Math.... These kinds of contexts are important in Machine Learning & we really must find ways to give the system access to such information.

.40: **793.40:** One way to give contextual information of this sort is to give the Q's "indices": Each Q can have associated with it one or more symbols that have contextual information.

One symbol type might tell if the problem is a math or physics or biology problem. Another might tell if the problem involves differential equations — and if so, what kind, etc.

In the infant machine, these indices would be correlated with problem solving techniques.

In a more mature machine, we will be able to commit some of the indices — the machine will be able to induce from the data it received in its earlier training.

.10 In the solving of mathematical problems by humans, a common heuristic is to (very) abstract the problem — to remove apparently irrelevant features and retain as few features as possible to define the problem adequately.

For machines, the problems are often (very) already abstracted. It is necessary for the trainer to restore any useful contextual information. This contextual information tells the machine what areas to explore to find problem solving techniques that have been successful in problems similar to the current problem.

.18

.19 (It will be noted that the R (recognition) functions described are one way of implementing one kind of context.) insert above 793.29

.20

The section from 792.33 – 793.08: Initial portion "Scaling" looks very wrong!

793.33 – .40 may be "not so bad"

.24 insert at 793.035: This would be the factor needed if we only needed to find 5 symbols to add on to the common solution to the first 20 problems. If we have to back track, we retain the common solution to the first 19 problems, but we will usually need more than 5 symbols to find a solution including two new problems. — perhaps 10 symbols would be needed — giving a factor of about $(10+19)^{10} \approx 4.2 \times 10^{14}$, which begins to get unmanageable. With more backtracking or a larger number of problems, it becomes (very) unmanageable.

.30 In the foregoing analysis, we have used "ball park" estimates. In an empirical study of Algebraic Notation learning, using the system described, without backtracking, we did, indeed, find that the search time increased very rapidly with n. ← (Maybe give figures)

.34

Also mention defining subs that were not themselves Soln's.

Probs judged to be of about = diffy for human, should take about (equal) times by machine.

.35 Another apparent deficiency revealed by this study was that the only regularities that the system used in past data were entire solutions to problems. There were no sub-functions in previous solutions that were usable in solving new problems. We feel that this deficiency was a characteristic of the training sequence used, and not a deficiency of the system itself. Nevertheless, if we do (very) have

30: 794.40 training sequence that is adequate in this respect, we still have to augment our System with techniques to discover useful sub-functions. This ~~[illegible]~~ involves the discovery of repetitions of sub-functions in correct solutions, and ~~the evaluation~~ ~~[illegible]~~ calculation of how much the total ~~[illegible]~~ a priori probability of the code is

05 changed by [illegible] defining such functions.

.04R

SN If we "oversearched" each in Lsrch, we would find OSL solutions, but they would not be given the hy pc's they deserve. Whenever we find a soln, we should examine it to see if it has any functs or subfuncts that have appeared earlier in the soln. function, but were not given "definitions".

10 **SN** On BACKTRACKING: Its cheap to save "state of machine": consists of lists of defns / contexts & case counts for defns. Since defns are ~~[illegible]~~ from large common list, we just give address of defn, + case counts. Not much RAM needed! Also, because of the tree structure of Search, each node has a list of definitions up to that pt. Branches add defns. (Maybe some times delete defns?),

15 **S**uperficially, it would seem that there wouldn't be much point to ~~[illegible]~~ "oversearch" & saving results: For backtracking, it would be just as cheap to go back to the node that was used, & pursue the branches (i.e. "oversearch"), but only when "future" events dictate it. One advantage of "immediate over-search" is lost — i.e. good pc values for ~~[illegible]~~ prediction. → Also, one could find shorter codes & those would be better cands. for "i prefer" for continuation of the Corpus codes

20 **BIG Q**: Say one is searching for a code & nothing fits (or only very poor fits) thus far. At what point do we abandon this branch & do "backtracking"?

Pretty much has been written on TREE SEARCH ... but 5 have important extra ~~[illegible]~~ parameter(s): (pc & cc) & "goodness of fit" that combines well w. (pc & cc) → .36

23 From Pearl, it looks like there isn't very much one can do to improve the search!

25 (796.16) One possy is to change order of problems. Another is to (somehow) "reparse" but the presence of the "complete solns" in the definition list, makes this pointless.

Maybe "reparse" ~~[illegible]~~ earlier part of corpus, using defns & pc's of defns, from later part of corpus, but omitting certain defns that would solve the earlier problems immediately &

30 This sounds a bit A.H., but I may be able to justify it, clean it up, define it in

31 a better way.

32 I like the idea of "reparsing" because ① It does seem to be a commonly used scientific discovery method ② I know how to do it, program-wise.

In view of ①, I should be able to fix it up (.30–.31)

The main reason "naive" reparsing wouldn't work, is the presence of those defns of "funcs" that "work on the whole" corpus (or on large parts of it).

.35 .35R

36 (.23) When we figure that we've spent much more time on a problem w.o. soln. than we would expect, ~~[illegible]~~ we then do Backtracking. The decision to backtrack, & how much to backtrack (How many problems solve back) can be "found". There are 2 alternatives to Backtracking: One could try "reparsing" 796.06 ff : See NIPS 184.25ff on Backtracking

→ But at this point, I'm not interested in defining Recursive funcs. → 796.07

↳ In normal Sci: Inquiry: we don't have this constraint: so we should be able to ... well even with those defns of big functions; it should be possible to reparse earlier functions using codes that occur later. Actually, parsing using future problems, is somewhat illegal because it defines...

0/25/02 ID

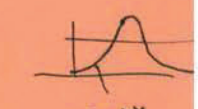
This would seem to be very close to what AZ [redacted] does in evaluating Apri . — It involves function definition & context definition.

00: 795.40 : -1.10-.40 (on backtracking, reparsing, etc.) looks v.g. It looks like t. Lsrch phase of t. QA problem could be done not-so-badly!

Tho -1.10-.40 certainly needs some work : it will probly "Move ontoose" work w.o. backtrack for some "v.g." TSQ's.

dIND → SIND Partis (ing "Bernoulli Induction"; Ram more general type. I need to work on this More)

Hvr, Efficient Backtracking (Note -1.36-.40) is impt & probly



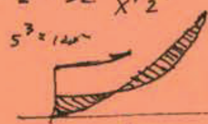
$x^3 2^{-x}$

2

$2^x \geq x^3$

$2^5 = 32$ $x^3 2^{-x}$

$5^3 = 125$



.00 Reparsing is very impt. (-1.25-.35) (-1.32-.35 points way to solution).

(-1.25-.35)

07: (-1.04-.35) R — (on reparsing) : Actually, defining "Reparsing" isn't so hard! (to hard margin)

Tennessee Louisiana Ohio.

Say we have a [redacted] soln to [redacted] an early problem : we express t. soln in primitive constants & operators. We Ran try to reparse this soln, in terms of all of t. defns used in t whole

10 (future) corpus. This would only be useful if I had defined sub-functions that were not complete solns. of problem. So this "Reparsing" can't occur until I have defined enuf "sub-functs" to make it possl.

.14 So, at this point, I have t. problem divided into 3 (phases) "Eras" : The [redacted] first Phase is t. Early QATM: Lsearch in an essentially Sequential manner. The refinements are discussed in 2.00 ff: Tree search, to depth of k. $\frac{cc_i}{pc_i}$. ("k ≡ depth index).

Some Hears ("refinements") [Backtracking, Reparsing, Possibly oversearching to find extra codes (Bursar -1.15 for "objection to this"). Re ordering problems. T [In this Era/phase I] T. Lsrch is blind : It doesn't look at its "Target"; t. sub corpus to be coded.

Era II (Phase) is use of induction to update t. Guiding GPD for INV & OZ probs (by OZ I mean general Optzn).

20 There is a sequence of refinements on how this is done.

Era III (Phase) is when QA induction begins to use [redacted] General PST's to do very General optimization of 4.14.18, rather than t. simple [redacted] Tree Lsrch of .14ff.

Era (Phase) II & III may be mixd, hvr. (??) — well, there is a gradual transition from II to III as the update algm for QA is progressively Modified.

Grace has 791 – 794 + 797

26 797. ... needs more to "finishing" stuff on QATM:

What needs be done! (a) Appendix 2 : t. conv. theorem proof.

(b) Bottom of 794 & 0-795.05) On detecting sub-functs that occur > 1 time at o t.

(c) some text tying (a) into t. paper. (I think I wrote this already!)

30 766. 10-.40 Expo on General Optzn problem (time-varying G(X, t))

735. 07-.21 Biblio outline of stuff written [redacted] (720.10-722.15) ← is Appendix B (or Appendix 2)

I probly want to rewrite t. beginning of Appendix 2, explaining just what $\mathcal{U}(A|Q)$ is.

722.25-.28 gives a biblio review of "T. report thus far". I probly want to change t. intro, hvr.

Instructions for Typing

: Sackheim Plot needs to be typed to fill out dizen of QATM:

~~783.30~~ "Practical Induction" 783.20 — .40 ~~Problem~~ Maybe don't type Res yet: I don't yet know where to put it!

~~791.00~~ following ~~PP~~ on "Analysis and Synthesis":

791.05 -.21
792.03 - .06
791.25 - .33
792.07 - .40
793.00 - .40 (Insert (7.94.19 -.20) ~~at~~ at 793.30) (Insert 794.24 - .34 at 793.035)
794.00 - .18

794.35 - .40, 795.00 - .05 ~~important~~ important, but I don't know where to put it!

The file to which Res stuff is to be added is Conbrobl.Tex 10/17/02

Start w. d IND, defns only of previous soln. ($w_n = 1$)
Refinements
1) Defns of other subfunct.
2) R's introduced ($w_n > 1$)
3) Context defns
4) S Ind a) Bern.
5) b) General Ind.

00:797: Introduction: We will begin with the description of a simple kind of inductive inference system. We are given a sequence of Q (questions), Q, A pairs (questions and correct answers), then, given a new Q, the system must give appropriate Answer.

At first, the answers will be mathematical questions in which there is only one correct answer. The system tries to find an appropriate function F so that for all examples, Q_i, A_i; $F(Q_i) = A_i$ — we look for F functions that have highest a priori probabilities — i.e. they have "short descriptions".

In generating such functions, we use compositions of primitive functions and definitions of sub-functions that we have defined. Subfunction definition is one of the main methods we use to get functions with short descriptions.

At first we only define functions that are solutions to problems. Later we define subfunctions that have occurred several times in solution functions, and functions that define contexts for functions.

Another important improvement in prediction is obtained by a set of functions that define classes of problems, so we can use a different kind of prediction function for each class.

I want also want to mention "context" at this pt.

From Deterministic prediction we can graduate to probabilistic prediction. Among other advantages, it allows us to make errors in information given to the system. At first we will use simple Bernoulli prediction, Then the most general kind of prediction based on Algorithmic Probability

While each of these improvements/extensions of the system brings new challenges, perhaps the most difficult design task is the construction of suitable training sequences for the system. We will begin by teaching elementary

~~The foregoing discussion has been of the question answering system only.~~

~~further improvement of the system~~

algebra — first notation, then solution of linear quadratic and cubic equations...

After the system has acquired a fair understanding of elementary algebra, we begin to ask questions in a simple kind of English, about topics in algebra that it is familiar with. It is easier to learn a new language when you already know one that has many concepts in common with the one to be learned. In the present case, the common concepts will be the definitions of algebraic entities

→ 813.00

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

(703.90) : If the ~~index~~ summation index, j, is a continuous parameter, the sum is replaced by an integral (often multidimensional). If j is continuous and

$$P(O^n_j) \prod_{i=1}^{n}$$

→ T. Introdn. of t. CPM Nov 2001 paper ~~does~~ gives a v.g. explanation of t. ~~prob~~ QA problem. → (115)

Make list of impt. ideas that I want to include in either t. report or my version of t. report. Perhaps do this on computer, so I can insert lines, & move lines.

(10) Also, it explains (to some extent), why 414.18 is a good approx of it. If t. perms. in CPM Nov'01 are, indeed, indep. than their wtd Σ sum being used as approx., means that t. coll. wts are " $\alpha_i \beta_{i2}$ ". — which means, t. larger $\alpha_i \beta_{i3}$, t. more confidence we have in it rel. to other Perms. (Maybe refer to Sol64a "T. all Perms Method".

..... Since each "Perm" must have a different code, t. codes for diffnt perms cannot overlap (I'm assuming "2 part codes" .), so we do, indeed use $\alpha_i \beta_{i2}$ to tell wts. of 2 diffnt Perms.

→ In t. report, I could discuss this a bit. General discussn. of "Optimization": "Just how Good is t. System?"

So, present state of report:

Outline of Entire report: 6% .13 - .90 : I should have put in Section on GCPD.

T section on GCPD could explain about diffnt types of GCPD's

How index nos. are used in Queries.

Th. commonest modes of use of t. GCPD : (1) problem → d to on PST's

(2) expresses Soln. of extensive (induction problem. (3) $GCPD_1 \rightarrow GCPD_2$ used in Lsrch & Modifns of Lsrch. 2 levels of GCPD

(3) Guiding Lsrch & Modifns of Lsrch. (4) Guiding other PST's, e.g. WON-timon optimality of Lsrch. → see (723.35)

Section on ~~How the System works~~ Evolution of the System

→ We will ~~first~~ describe the initial configuration of the system, the contents of its GCPD and AST data structures, ~~& then~~ show how the system works, using these ~~primitives~~ primitives ~~to solve problems~~. Cognitive devices, and discuss the limitations of these devices. → 705.20

perhaps write PS of various Generalizations, Extensions, Expanded discussions of System. Write t. PS's — Maybe put in "Report" But do put in My "Report".

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

Discussion of "reports so far" 722.25 ; & Y. general report outline of 696.13 – 40

The outline has detailed discussion of parts of the system. The Abstract & Introduction Give a general idea of how the all works, but I'm not sure a reader could put this all together! — There is no section telling how the "system as a whole" works.

I did suggest §V an "Evolution of the System" → Maybe "Operation of system" V !

Here we return to the operation of the system as a whole: Starting as a baby & moving Retro Lsrch, Adaptive Lsrch, ~~some general PSTs~~, The seeding problems, TSQ's (?)

perhaps make TSQ design a subsection of Section V

I might also discuss OZ problems & INV problems. — The phrases might be discussed in the section on GCPD — how GCPD differs for various kinds of OZ/INV probs & for probs as well as TM

How TSQ guides system design (The avenues/lessons from the TSQ,) ~~[illegible]~~

Various PST's are designed by user to deal w certain ~~TSQ probs~~ probs in the TSQ.

That initial training of TM is interacting betw TM, TSQ, trainer. Trainer Modifys TSQ in response to TM's response to TSQ. Also hours are discovered by Trainee — But need to be realized (turned into TM) by the TSQ.

→ Perhaps write more on optimality of 414.18 (for my copy) at least, plus Biblio references.

Biblio: ~~Ends on~~ 701.19): The whole discussion seems to be 697.03 – ~~699.40~~, 701.00 – .19

701.19 – .19 may be best here, but the Discuss in CPM NE, 01 Page 2 may be better; Maybe, the idea is that 414.18 is U.S. (maybe as good as we can get), & we want to get as close to it as possible. The larger our total wt, is the "closer" we get to it.

SN 703.00 – .21 discusses QA in which j of O_j is a continuous parameter. (A's are continuous ~~defs~~ of the (Hessian Approximation)) This is crucial

I may want to include this somewhere (Maybe "footnotes" ☺).

Actually, the A's need not be continuous, only j needs to be continuous! In Barnsley, j is the fractal params, but the A are (usually) discrete.

Re: Expo The Abstract & Introduction give a general idea of how the system works; But before the detailed sections on QA, PSTs, GCPD & updating, I think the reader should have a better idea of what the systems does before reading that stuff. It may be possible to have "QA" interspersed in the early system.

I may have said that Lsrch uses a P.D. to guide its search. Then explain that after a problem is solved, this P.D. is (improved modified) in view of the newly solved problem, & mention that this P.D. is meant to reflect the experience of TM to Date, in solving problems of that type. How GCPD is designed so the reader can follow it. The sections on QA, (Lsrch's PST) descriptions understandable by predecessors. The section GCPD (704.29 – 34) is a kind of summary of previous mentions Reading of it. Actually, 704.29 – 34 doesn't say very much. I might go more into it as an analog of Google. — ~~The main idea~~ An implicit idea to get across in what is that There are many P.D.'s that TM uses, but they can be all regarded as one big P.D. (GCPD), which enables ~~efficient learning~~ Transfer learning between various kinds of P.D.'s.

3/2/02 Just used up pilot Explorer #1. — now on #2 724

ID

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University

53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

20 [SN] Some time ago (A month or 2?) I worked on folg problem; I think I solved it!
In update, what one wants to do is optimize t. "Grammar" that assigns "pc's of being best" to all possl. PST's. How is this done? Well first we get a grammar that gets pc for each possl. PST's ā pd. on G ā cc for z t. problem in question. This is a ganzu of $(GCPD, [PST])_1$ to get to $(GCPD, [PST])_2$ We do t. standard integration to get a D.f. over PST's that will be "t. best" — T. nature of t. integration depends on t. problem: depends on how G( ) is a funct of cc as well as X. ← This entire approach ommits diffy of "Correlns" Betw. Cands

.08 — Whether its INV or its OZ

O.K: so Next: Section on [PST's]: see [696.21] for Genl. outline of entire paper

0 1) Define, Derb. simplest Lsrch: $(T \leftarrow 2^k T)$, using pD. or codes"
Derb. II Lsrch (time sharing)
2 " " random. ① $\varepsilon_{j3} = \frac{cc}{pc}$ for Lsrch ② for Genl b house cands $\frac{cc}{pc}$ order is best ③ Lsrch is close to $\frac{cc}{pc}$ order.: Some times best.
Give 3 G-H Thrms: $T \leftarrow kT$, II Lsrch, II random Lsrch.
2) Use of Lsrch for OZ probs. Same
3) One trouble w. Lsrch: pc is too small: How to fix! "Adaptive Lsrch". t pc of soln.
① Update PD after each problem (CJS still works
② Use $T \leftarrow 2T$ After each T, spend T on updating t. Guiding P.d. then Lsrch.
Kinds of Adaptive Lsrch — CJS "sort of works"
We are in a hurry to update because cands may be correlated.
.17 ③ Time share revision of P.D. w. Lsrch. "50/50" share: Do II Lsrch; I think I worked out t. mechanics of this. work on all cands w. min $\frac{cc}{pc}$; when it is no longer min, work on 2 cands w. min $\frac{cc}{pc}$ keep working each recruit new cand into set as soon as its $\frac{cc_i}{pc_i}$ is that of current set. — CJS doesn't work.
20 Say we double $\frac{cc}{pc}$ of t. work done so far: Then we hunt for cands w. $\frac{cc}{pc}$ < new
22. level. In our random(!) search, we only keep track (in BINS") of $\frac{cc}{pc}$ of each cand we examine, so we can know when to recruit it.
We could do random srch, or search in (new) pc_i order.
P.D. ($\in pc_i$'s) are continually changing so t. "BIN" idea of .22 may not work. → See 733.06 for Notes

(This is theoretically "Best" & should be my version of "T. repeat")

(I may have more re: [733.24], in particular [732.26])

26

28 ④ Another method of searching: TM initially selects t. "Best" (via GCPD) PST for t. problem. As it is working on that PST it is (Time share) working on revision of GCPD, in view of t. success or failure of t. current PST. When (and if), t. expected additional time for soln. of t. present PST, becomes longer than t. expected soln. time of a different PST, it will switch to that PST — It will then work on that PST until (or if) a switching criterion is met than sends it to a new PST.

.34

35 [SN] In last Section on General Conclusions, or: "Concluding Remarks"
We have described a very General prob solving machine that we expect can be developed into a system of very great power. The problem solving techniques that we will start with, Lsrch and adaptive Lsrch, will certainly solve many problems, but

9/4/02 ED

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

eventually they will be limited by the scaling consideration of section . The most important feature of the system is ~~not~~ its flexibility — The particular methods used are able to accomodate any possible PST — be it added by the trainer, or ~~[illegible]~~ discovered by the universal search of the system itself. The development of this system will never reach a "dead end", ~~unless the process of inductive inference, itself, is necessarily limited. If such a limitation exists, it is far beyond what we, as humans have done in the past.~~ — ~~certainly a tolerable limitation!~~ The limitation being only those of the Hardware — which we expect will continue to double in speed every about 18 months ~~[illegible]~~ for quite a few years.

12V / 200k = .06ma

10.

11. Levin first describe his Universal sequencial search in 1972 (Lev 72). Though that paper didn't tell how to ~~do it~~ actually do it, the description was enough to suggest several ~~[illegible]~~ implementations.

The problem was function inversion: Given a function f(x) from strings and or numbers, to strings and or numbers: Given ~~a~~ a possible range object, s, to find argument x such that f(x) = s. We assume that evaluating f(x) ~~does not take very much time,~~ is a slowly increasing function of the number of bits in s.

Actually, he just Gave Thms about srch, not alg.

→ Theorem on a lower bnd on how good a srch Algm can be.

For the function inversion problem described in the introduction, Levin ~~Though he proposed a universal sequencial search algorithm (Lev 72). Though he didn't~~ ~~[illegible] give the algorithm in that paper, his description was enough to suggest several possible implementations.~~ ~~described the algorithm in~~ didn't ~~describe~~ define the algorithm in any details (Li.Vit 93 97) his description was enough to suggest several implementations.

1. Consider the set of all functions G, that look at f( ) and the object s, and give a ~~[illegible]~~ trial x as output. Let us assume a probability distribution P(G( )) on all such functions. Then, for a small time T₀ we spend maximum time T₀ P(Gⱼ) generating ~~[illegible]~~ Gⱼ and testing its candidate xⱼ on f( ). If we are successful, we quit. If not, ~~after testing~~ it will take us $\sum_j T_0 P(G_j)$ to test all of the Gⱼ in this way. Since $\sum_j P(G_j) \leq 1$ our total time will be < T₀.

If we are not successful, we double T₀ ~~[illegible]~~ T₀ = 2T₀ and try again on the whole set of Gⱼ's. We keep doubling ~~our~~ Time limit and searching until we find a solution.

8/4/02
IY

JOINT CENTER FOR URBAN STUDIES of MIT and Harvard University
53 Church Street Cambridge, Massachusetts 02138 (617) 495-7908

00:695.40: We make approximate optimum path for -1.37 by ≈ "Lsrch". We move along successive cands, keeping track of total pc thus far, & we regulate Σ cc to be below a certain multiple of that pc: t. multiple ↑ linearly w. time, if we are doing ‖ Lsrch.

[Back to -1.37: Say we were able to list cands in best pc order, in t. sense that After kth trial, total pc of will was max over all poss k trial orderings.

Hvr. t. optimum k sequence is prob'ly not a prefix of t. optimum (k+1) sequence ©!

- Anyway, probably various approx. soln's to t. cand ordering problem will be made clearer by t. consideration of -1.37 — that ∃ an optimum cand. sequence.

0 Fact is, hvr, t. conditions that define t. soln -1.37 do not obtain in "≈ Lsrch":
One does not know all t. pc's & cc's in advance — still, t. concept of -1.37 can be useful.
— In designing Approxns. to be used in modeling and implementing GCPD for "Lsrch".

.13:694.20 O.K. – Back to T. Report.

Main section of report: Introdn: 693.22 – 694.12

15 (I) QA problems: How they are OZ probs: Give 414.18: Give convergence Thm. & Significance. Explain that to define a Q.A. problem we need only give

{ T corpus [$Q_i A_i$]$_{i=1 \ldots n}$, Q_{n+1} } This defines t. problem:
{ Also t method of giving aprips to [scribble] O^n. } To solve problem one needs much more: ∴

∴ defn. of t. problem tells what soln. looks like — not how to find it. → (699.26)

20 N.B. $E_M O^s(A|Q)$, J_{srch} can be controversial: see exp 703.00-.01 (Give ref to Schmidhuber paper on Kapfetness)

21 (II) PST's (a) Desc. Lsch & its variations: impt properties: ∃[CJS]
Also t. 3 GHT's (Gaukin House Thms). (1) $\frac{pc}{cc}$ order is best (2) cjs (3) Lsrch is close to Best: Is ≈ Best if all info is in GCPD & only blind srch is allowed.

Desc. Adaptive Lsrch: (a) Change PD for each problem ..
Discuss Non-optimality of LSrch — (Was discussed in My Version of Report)
(b) Change Pd at T ← 2T (c) Change Pd every several cands. (d) change PD every cand.

Note that this is no longer Lsch; CJS ≡ [scribble] $\frac{pc}{cc}$ [scribble] is not apriori known.
→ see 724.10 – .26 for ≈ 696.15 – .28

.28 — even when soln is known.
(b) other PST's: They are contin srches; they can be any thing: → 761.00 – .16 on mixing [PST] and GPD. mixing
Maybe discuss GA's & RANN's here. or call t. Main PST!
They can call of PST's or call t. (GCPD) see (704.28 – .34) (Also 723.35)

30 (III)

(IV) Updating: (a) Simple updating: 695.15 – .28 (note diffy of 625.28 – .29) I think some kind of naive version of this may be somewhat workable. It is certain that people will try it or variations of it. unclear where to discuss this.
(b) updating using $GCPD_{1,2}$. with $GCPD_2$ being having indiv pc's of cands.
(c) " " " allowing "correlns" (dependencys of pc's of cands)
(d) The General soln. of 695.00 – .07 → (see 723.04 – .10 on this)

(V) → Evolution of System (704.35) ← possibly discuss GA & RANN here?

(VI) Discussn. of TSQ design: Refer to 385, 89, 94.
But give general form of "concept" (call it an "abstraction" not
t use by current lang thry.

[Right margin, vertical:] Functional of pairs of G will care search invariant. — Some? This much be true of any optimum search system. It is not true of Most GA fitness functions (fruiting trait sake). the GCPD ∪ Method

Expo

Abcdefghijklmnop

is expected ... be able to ... perhaps

00: 798.40 The system described thus far, is a pure induction system and can ... some work problems of some difficulty. We will show how it can be extended to work Inversion problems and time limited optimization problems. Both kinds of problems can be solved by Levin's Universal Search algorithm (Lsearch). However an elementary system can only solve (lem small) problems, since the solution time is exponential in the length of the description of the solution.

Let's see

→ (812.00) Lsearch uses a universal probability distribution to Guide the search. ABCDE ABCDE ABCde ABCD ABC AB

6:30 Note that if Guiding PD ... OZ ... as of a different kind & man ... for ... (induction).

SN I need Section on Lsrch: OZ Lsrch & Random Lsrch have not been done "In the Literature". I don't know if ll Lsrch has been; — Also (X-depth) a they factor for ll Lsrch. — depends how one does it, hvr.

SN for NIPS Talk: Give reasons why "it" should work;
(1) The convergence thm: what it means (counter argt.).
(2) QATM; The SUMAC argt: How we should be able to write good SUMACs w. good update algms
(3) For advanced system: t. 3 Grand House terms: Details of how they work, & why relevant. — t. "factor of 2" argt.

But do continue reading Part 379.26 !

0.30.02 Re Sumac: (I did → 378.00 (9/14/01 ... 1 yr ago) → Some ideas on How to deal w. diff kys ... consider this in some detail some time ago: There were 2 competing systems. This does not heavily enhanced! Shouldn't be hard to find; Maincopus summarized at t. top of first 10 or so lines of t. page.

378.00 − 379.26

If one saves "all" of t. O^i, w. their wts, t. wts are easily updated for each new Q_{n+1}, A_{n+1}. $wt_i^{n+1} \to wt_i^n \cdot O^i(A_{n+1}, Q_{n+1})$. .295 If one stores only a finite no. of O^i, wt_i's they tend to not contain t. O^i's that are critical in t. future corpus. T. finite set are used gets "used up". Th. needed O^i's, that are excluded, were excluded because of low wt_i^0 (small a priori)

I do want to go back to the Earlier reference. It is in ID. — I think since I was at IDSIA, but I'm not sure. → found! 378.00)

A way to do a Sumac w. a no. of O^i w. finite storage, is to make a S Grammar (≡ PD) over t. O^i. One such Grammar is the set of function ... & their wts, "up to now". — Also t. set of contexts "contexts" & their wts.

The reasoning at .25–.29 (why a finite C^i set eventually fails) may

38 be wrong. T. ... finite set could all get very low $C^i(A_{n+1}|Q_{n+1})$ yet, if they were not zeros, t. finite set would survive. T. Q is, would t. new set of wts have a reasonable distribution (set of ratios to each other).?

00: 799.40 We prefer deal w. 799.38 – .40, by putting a small exponent on t. O^2 mods!
i.e. update factor = $O^2(A_{n+1}|Q_{n+1})^\gamma$: γ would be small if, on the average
very few of t O^2's did well.

Hindsight's probs! 581.06–.13

.03 [SN] On Backtracking: If one had rite "wts" for all component concs, & one did an Lsrch over t. [crossed out] cands, using Bernoulli pc's — Perhaps it would automatically Backtrack properly!

— Say S_n is [crossed out] a acceptable soln to $[Q_i - A_i]_{i=1...n}$. We have this set of S_i ($i=1/n$) in t. "Bernoulli alphabet" along w. all of the other functs that have been defined. For QA_{n+1}, we would try S_n as a component, but S_{n-1} & S_{n-2} would also be tried.... S_{n-1} & S_{n-2} amount to "Backtracking".

It may be impt. to note that many of t. S_i components have been "used at least twice;" by being used as a component of a later S_j — But S_n has not: its presence in t. arg. list ("precorpus") is by "courtesy" only — & ultimately justified only by **OSL**. — Hvr Q_n does have the likelyhood of legal usein OSL, so we put it in t. "precorpus" to facilitate its use in a new funct. defn.

.18

.19 A possl. way to ~~Get around~~ deal with t. low aprp of t. needed concs: In hunting for Born codes,
20 if we see a certain conc. is repeated a large number of times, we know that even w. low aprp, its a usable conc. One Efficient way to do such a search: Use a (sample / small) section of t. corpus to see how often t. conc. repeats. If we multiply this by t. ratio of entire corpus size, to that which was sampled, & it looks [crossed out] like it's enuf to deal w. t. low aprp; we do t. search for the rest of t. corpus.

— Actually, we don't have to do entire "rest of corpus" we can do as much as is needed to decide about "going further".

∴ So this Mite be one way to express t. "SUMAC problem": How to recruit into the "followd" C^i's, new C^i's that have [crossed out] small aprps? Again, I think I wrote about stuff like this. [crossed out] (378.00ff) — These approaches are different from [crossed out] those being considerd now!

In 378.00 I was concernd w. "contraction/narrowing" of t. defn C^i's, ↑ is ff is one way to help. We do have to continually bring in new low aprp C^j's, if we want this system to work. (I think it should be able to be made "workable" since humans seem to do it! & — i.e. Humans use SUMAC's)

Humans do Backtrack.

0/31/02 ID

00: 800.40 : In 378 00ff, I was concerned w. gradual (or/sudden) [now] degradation of the "set of good codes". Let us look at what happens from t. pt. of view of a TSQ.

In order to "collapse" t. "set of good codes" must not have adequate subfunctions to represent t. soln. of t. current problem.

SAC (set of acceptable codes (a "soft set" i.e. pd)) Its an inadequacy of t TSQ. ... wrt t. "set of ~~good~~ Acceptable codes." T. "set of Good codes" "acceptable" SOAC (SAC) is mindful of t. current hy G population in GA.

For t. latest problem to be solvable, ~~the~~ it must have a soln. that is a reasonable CJS for at least one (& preferably Many)

In General, its very (instructive/revealing) to look at t. TSQ & its effect on t. SAC:

11 (Phase II) Perhaps t. "SAC problem" (of its degradation) only occurs at that pt. in t. TSQ where t Trainer has lost detail'd understanding of the SAC (what concs TM has "acquired") & what concs are adequate for soln. of t Current problem. | Trainer may know what concs are adequate solns, but he will not know How to factor those cones so that TM can acquire them.

14 [SN] A possible impt. Aspect of this problem that I've not Considered:

"A code cluster" has 2 aspects: its pc & its mean cc per problem or (mean $\frac{cc}{pc}$ per problem.).. So, we mite retain in SAC all solns. w. pc within a certain constant factor, of t. best thus far. But we will be worried about Codes that have bad $\frac{cc}{pc}$ (& bad CJS), tho they mite have Good pc's, (Good for predn), they would tend to take lots of cc for future problem solns.

T. way It looks now! That in t. early phase of t. TSQ (Before .11–.14 has occurred) There isn't much of a "SAC contraction problem": There is always a solution that is an extension of a possibly single acceptable code. By oversearching in Lsrch, we can have >1 acceptable code — some of them w. hyer pc's but larger cc's. If we use a CB large enuf to find what t. Trainer knows to be "an adequate soln", then TM will at least acquire that soln.

If TM often finds better solns (faster, hyer pc) than those expected by t. trainer, then TM has entered Phase II (.11–.14) & t. continuation of t. TSQ will probably be made on t by t. trainer based on "Local Models." (These are approxt models of TM & my be sometimes rather "gross".)

In Phase I there need be no "Backtracking": Hvr, by "oversearching" we can have many 11 codes. Some of t 11 codes can be expressed compactly in terms of common ~~[illegible]~~ code additions — perhaps (partly) by cartesian products of certain sets of solutions.

0/31/02 ID

802

ABCdefg

00: 801.40 : If solns are found by Lsrch, I think we want to retain all solns. found. They will tend to be of hy pc a/o of low cc ∴ "useful".

.02 F. Mechanics of when & how to Backtrack : Certainly UNCLEAR!

I want to Model it More on Human Backtracking — & Theory Revision.

In Theory Revision: we back track to ~~previous~~ an earlier problem soln. — that worked (or could be "made to work w. minor changes") — but at that time, was of low pc a/o hy cc. (low $\frac{pc}{cc}$). Certain of t. problems since then are dependent on that soln., so we have Lsrch on those problems, to develop/test that Theory revision model. Certain problems will not depend on that ~~~~ older soln, & we can use t. same solns. as before &

During (t. lengthy) Theory revision process. We continue to make predns using t. Old theory, perhaps w. some "local" revisions (whatever that means)!

It would be good to examine carefully, t. process of Theory Revision (in t. Sci community) Then Correlate it w. Processes in QATM. It is conceivable that it may involve t. kind of inductive processes that are needed in a QATM that updates using t. ~~~~ full optzn model. & rather than t. ~~~~ update model using Lsrch precursor to "Infant Early Induction."

20 I really need to write out in as much detail as possl. (for my own use) The Operation of the "early" QATM ~~~~ Call this Phase I & II QATM!

Phase III is where it uses full OZ technique to Maxz 414.18.

See 801.11-.14 for Defn, discussn of Phase II (this is where TM's response to TSQ is not completely understood by trainer.)

25 — One unclear thing! We can use a fixd (CB time) for each ~~update~~ search = "update" → 803.00 = time spent on previous search (?). or time spent on forthcoming srch? : any way: another way ~~to decide~~ would be "how ~~hy~~ hy a value of 414.18 do we need before we quit?

In t. case of d induction it is perhaps easier — often we are satisfied w. t. first soln. we find, because it is of hyest ~~~~ a prop : (But if it hy small cc, we may be able to find a hyer a prop soln. w larger cc ~~~~ — which justifys "over search".

It may be necy for t. Trainer to decide ~~~~ what level of 414.18 to accept — as part of t. "update problem" Definition".

:00 : (spec 802.25): The Main Idea is that If we have an adequate one net for the problem, we should be able to write an a daily rate tool. "Adequate" means that the Recognition as well as the "soln" algms are in the one net.

Most heur. can be put into net, since most heurs are modifns of the guiding pd (on a simple level) & more generally can amount to "discovery" of new PST's by assigning "reasonable" pc's to those PST's. — In this case a PST is a soft (probilistic) reordering of trials.

How is Impr. during a Run, done? Is it really necessary?

Also, of course, Quick Abort — not being served by simple modifn of pd.

Remember the PD for simple QA is for O_n^*'s not PST's. → But (.12-.14)! functions (to some extent) implement PST's

How., we are looking for "functions" & a function can be defined as a such.

On the other hand. Such functions have large cc, & would be used only if they had very large pc!

There may be a min set of heurs that we have to "wire in ab initio", before the TM can learn enough heurs to get to Phase III.

SN Image Reconstruction: A blurred image should be expressable more compactly by a hy res. image + natural noise — but it's not clear just how!

The original (hy res.) image may have much less redundancy (?).

Reason .12-.14 seems very crude! Can any PST be regarded as a "function"? How, "functions" can't really implement PST's in general, because PST's know what the problem is: The functions I'm using in QA, don't. E.g. in QA problems a PST would know A_{n+1} as well as Q_{n+1}. It would also "know" that a hyp O_n^* restricting Q_{n+1} to A_{n+1} is wanted & but also, we want hy $O_n^*(A_{n+1}/Q_{n+1})$.

The "functions" I'm using only know Q_{n+1} & we list/train/ O_n in pc order.

→ A heur may be able to change the diff. guiding the trials, but that's it!

In phases I & II the sys is "blind" in the sense that it doesn't really know what it's looking for! — other than that hit has biggest apriority possible & (perhaps) smallest cc.

So Phase I, II is only able to try funcls of Q_{n+1} in pc order (or in $\frac{pc}{cc}$ order).

If we modify the pc def., we are no longer solving the correct problem: Except that

If the pc is the P.D. for A_{n+1} (not the pd for the entire corpus) — it's the PD for A_{n+1} in view of Q_{n+1} & all previous QA's. — It's a pc of a SUMAC. The Sumac knows Q_{n+1}, but not A_{n+1}.

N2.02
ID

804

aproprd ≡ "prior"

80: 803.40 So .t looks like t. best Views of Phases I, II is as a Sumac: Discovery of functions à Context for functions is simply a definition of t. apropd / prior.
Could any thoughts be regarded as discovery of certain "Contexts"?

I would Seem that a Sumac approach, using a conc. net, should work --- Phase I would be certainly Phase I, à possibly part of phase II. A by Q is how to get reasonable pc's for t. concs d'cribg. t. latest soln. Context is one step in this direction.

I don't remember having worked on this problem from that pt. of view.

There is t. "recency" context. W. an "X window" on the Recency {or Rumdo ... which means TM only remembers stuff used ≤ T seconds ago} T. "softness" (G Gray) of X window sounds Much Better than Rum Rumdo > .20

So, at present, in Phases I, II, I have Sumac Model: defining functs, ≃ OSC, à Context for Funct defn. à t. Bernoulli model of ACM's — are t. only ways I can find/justify for these Phases.

Whether à to what Extent, Backtracking is useful/feasible, is unclear.

[I could do a brief anal. of "Overscore" to get extra codes, making Backtracking a bit Easier] T. discn. of Backtracking on 802.0 - .20 is good. 800.03 - .18;

11 { We have a really enormous no. of concs. We really have to assign large amt / large number of pc's to them
So Prob t. pc of unique soln is reasonable.
perhaps .20 is one of t. Major problems ... other than designing t. TSQ, itself.

23 Actually, the Scaling problem is, along w. TSQ writing, t. 2 major diff problems in TM.
In doing d induction (past stages I, II), we are looking for / hypc functions such
that Q into A. (Partly: we also need a suitable "R" funct) — but any way, it is t. early
AI problem of finding a seq. of steps that brings one thing into another, or this case
Q into A. The standard methods (Sub-goaling / deriving), backward à forward
chaining à GPS (difference reduction / Vector Gen), à other common techs.
Probly these techniques can be extended to S-induction.

AH! Context Discovery can be implemented by putting t. defns of t. Contexts
in (a / The) "Concept net". One version of a "Context" is a structure
out t. previous corpus (≡ "stabilized / post fix").

SO: When Peter Barnett (at Stanford) was getting pc's ≈ 10^{-15} for his problem solns, I think
that he was not breaking problems down into sub-functions properly: but while that may have been true,
he was probably also getting very low pcs because of t. simple "scaling" & not very "Context"
(whether Context alone (even very "extended" context) will be enough to get reasonable esses —
is unclear. Ultimately, we want to be incremental cc cost of each problem-soln. to be
a good constant.

806.00

PS file: Nov 2, 2002

Generalization of ALP to N dim. lattices

Chris' Question:
Say we have data from t=0 to t=100. What is the probability distribution at t=-100 ?

One simple answer would be to relabel the data, so t -> 100 - t . We then have data from t = 0 to t = 100 and we have to predict at t = 100 + 10 . - which is a standard ALP problem.

However it suggests a more interesting problem: Suppose the data (t = 0 to t = 100) was really taken from a string of data from t = - infinity to t = + infinity. i.e. it really didn't start at t = 0.

In this case, to get the probability distribution for t = -10, consider a universal Turing machine with no stop symbol. Its input is a unidirectional random binary string. It has an infinite bidirectional work tape. Its output tape is infinite, bidirectional: Initially blank; starts on square zero. This tape is bidirectional, but it cannot erase, only write.

There is a certain probability that the machine will print all 100 known bits and 0 for bit 110. There is also a probability that it will print all 100 known bits and 1 as bit 110. These two probabilities give a semimeasure on the 110th bit.

allowd on "output tape".

erasure ✓

No eraser allowd on output tape"

This model can be easily extended to n dimensional lattices. The "known data" can be any set of lattice points: They need not be contiguous - many data points can be "missing". The predicted bits can be any points on the lattice. The "output tape" will be the n dimensional lattice and we use a special turing machine that can move on the output lattice in 2n possible directions and print. The input is as before an infinite unidirectional random string. The work tape is bidirectional and infinite.

"Needs work" ☺

Turing machines may or may not have a "stop" state.

As this is formulated, it maybe that the probability of any particular config being eventually produced is 1.

However, as T→∞, the no. of times a given config gets produced, may have a limit value to T.

We can also use a Turing machine with infinite bidirectional random input tape, infinite bidirectional work tape with both write and erase allowed on the output lattice. Start at any point on the input tape. What is the probability that the machine will, at some point in time, have on the lattice a certain known configuration of data bits, and will have a 0 at a certain specified lattice point ? --What is probability that it will have a 1 at that point ?

This last is of interest because we are using an Unrestricted Turing Machine -- this model is Very General.

Infinite bidirectional random input tape: analog of "prefix set" : No string is a substring of another in the set. However, no "kraft inequality", we could have an ∞ of contiguous, non-over lapping strings of length 2. (or 1). $\sum 2^{0} = \infty$. But, every string must contain the stop bit. So maybe "kraft" still holds. It does hold. And $\sum 2^{-L_i} = 1$ for any "complete" set.

Let L_i, R_i be left & rit halves of an input code.

For any particular L_i all legal $R_{i,j}$ of our two R's. halves for L_i, form a prefix code.

Subtotal wt. of all such codes is $\leq 2^{-|L_i|}$. and the sum of wts of all codes is $\sum_{\varepsilon} 2^{-|L_i|}$

Which is ≤ 1.

N402 · IP.

N DIMENSIONAL ALP

— cf. "Ota AI expert Systems" 804.23–29

10: 804.001 At present, there is plan for early QATM (Phases I & II) [redacted] is rather open — I have to conc. not for functions & contexts of functions. [redacted] (including "R" functions) How much this will slow down the ↓ of pc's of solns, is unclear. I may (probably) have to dig cover new means to ↑ pc's of solns. Can I get a good enuf soln to be able to Go on to Phase 3 "(use of Induction for updating — & soln. of QA as a general optzn. problem, using only those 2 tricks: which are simply ways to define the appropriate .... It is pretty much the state of the Bypassing QA problem — right now.

I will want to write about how Both funcs & contexts must be on the conc. matic ∴ int. T&Q.

10 **SN** N dim, bidirectional prefix sets: A kraft inequality:

.11 Consider all [redacted] subsets of pts on the N dim region. ("strings") $\vec{x} = (x_1, x_2 \cdots x_n)$ $0 \le x_i \le 1$

.12 x_i = 0 or 1. (or it can have [redacted] a k element alphabet). Consider k=2, here to start.

13 Associate w. each subset of .11, the subset of strings of which it is a substring — Give it Total wt. (Give each [redacted] string/wt. of length 2N 2^{-2N}.)

If [redacted] we have a set of strings (M.11) ∋ each [redacted] pair of them has at least 1 pt in common at which they disagree, then they have no common associate strings .13, So the total wt. of that set is ≤ 1. The wt. of each string is 2^{-S}, where S is the total no. of pts. specified on that string.

The above Proof (which needs to be "cleaned up") applies as well if k > 2. (.12)

This idea applies to any [redacted] bunch [redacted] of sub-sets of points! To be a "prefix set", each pair of sets must differ at at least 1 pt. So every pair of the [redacted] bunch must have at least 1 pt. in common, so they can disagree about at least 1 pt. If this is true, then we can assign a semi measure to the "Bunch". ∋ $\Sigma p^i \le 1$. $p^i = 2^{-\text{no. of pts in subset}_i}$.

Otherwise $\Sigma p^i = 1$; How can we tell if a prefix set is "complete"?

Q if we have a prefix set on pts, then for any subset of pts: [redacted]. is it true that; [redacted] that subset can be expressed as a sum of prefix sets in only 1 way? I suspect that [redacted] 8. Then May not be so simple!

(SN) Where we are now: 1) 798.00 — 799.10 has the beginning of a new outline. Needs to be finished.

2) 720.10–722.15 (APPX B²) — rewrite beginning explain just what (t|Q) is .13.

3) see 796.26 – .40 for review of what needs be done.

followed by 1 p. up
808
809
810

00: 806.40 SN It would seem that "in theory", If we have the complete instr. set, that the AZ141 "Sumac" should be able to discover any "pc increasing" tricks that are derivable.

In evaluating the relevance of Barron's paper on Bounded Neural nets to induction: The Q is: what is the relation to Occasion? This gives the amount of "magnification" from one space to another

Could [redacted] the Sumac use Lsrch to discover "Lsrch"? As is, prob. not: The heart that I would use to disc. Lsrch, would first be to notice that $\frac{cc}{pc}$ is some measure of how long it should take to solve a problem! — in this case cc would be some kind of mean cc of the trials — that this cc should be the cc of the "soln. trial" is a surprise!

Remember: Lsrch is the best, if "All info is in P.D.": But the AZ141 search is not Lsrch. — it just puts soln trials in Least cost order, but there need be no heuristic info on the "guiding PC" (which is fixed — and is the apriori). As a Sumac, hvr. some of coding of the past may take the form of a heur. (like "Context").

Rewriting of "Appendix (B)": 720.10 — 722.15

The only "proof" I'll do is 720.08 → 07, 721.09 → 40; (Give 578, +3 corr, and Hutter's Gonzn. to readers.)
I also need to show that $\frac{P}{\mu} > 2^{\alpha_0}$, the apriori of μ w.r.t. the same ref that P uses.

So its only ~1 page plus maybe 722.00 — .15 a/o other discussn of Significance → 808.00

19

SN Thinking about [redacted] this again: We actually do predict the A's sequentially.
20 : 720.07 I want to show $\frac{P}{\mu} > 2^{\alpha_0}$, the apriori of μ. (Actually, I think I showed this in the 599: "2 kinds of ...")
Its in §3.3 of "2 kinds of Prob. — induction" (599). I could just refer to that & actually say that it shows $\frac{P}{\mu} > 2^{\alpha_0}$ ∴ I want to show that the analog of 599 eq (3) is true.

Maybe I should define ALP for QA on the other basis!

In 599 eq (2) does indeed imply eq (5) (i.b.i.d) — but how to go from there to predictions?

It would be easier to prove the convergence thm for 599 type Seqs: the old. eqs (1) thru (5).

So in 599: Just do pc's of the [redacted] D_n

29
30 $(P_U \equiv P_{M_U})$ we want to compare

$$\prod_{n=1}^{h} \frac{P_U(D_1 \cdots D_n)}{P_U(D_1 \cdots D_{n-1})} = \frac{P_U(D_1 \cdots D_h)}{P_U(D_0)} = P_U(D_1 \cdots D_h).$$

$P_U(D_0) = 1. \quad D_0 = \text{Null} = \Lambda$

So it's TRIVIAL!

with [redacted] $$\prod_{n=1}^{h} \frac{P(D_1 \cdots D_n)}{P(D_1 \cdots D_{n-1})} = P(D_1 \cdots D_h). = \prod_{n=1}^{h} P_i(D_n)$$

Appendix 2: 808.00–.03 | 809.18–.40 | 810.00–.29, 810.34–.36

(807.19) spec.

00: 807.40 Start Appendix 2 again!

or equation (1)

close up. no space. — 809.30 looks ok!

Appendix 2:

We will show that [redacted] for an adequate sequence of (Q_i, A_i) pairs, the probabilities can be expected to be extremely good. predictions obtained by the probability distribution of eq(1) will usually be → 809.18

03 Plan: Express the A_i seq. as Z (221.18). Then show eq(1) (and M) can give

[redacted] conditional probs for each bit & space — conditional on all previous info in Z.

So P assigns cond'l pc's to each symbol of Z; as does M.

Then show that the product of these condit. probs for P $> 2^{-2^{k}}$ product of true cond probs for M.

Invoke "Solomonoff–Kinel–Gacs–Hutter" Theorem. SFH Theorem; SFGH

0 Perhaps state SFGH theorem & give its history as a footnote.

(No!) Let $P(A_i^j)$ be the conditional probability of the j^{th} symbol of A_i. A_i has k_i symbols: Meaningless!

$A_i^{k_i}$ is the space symbol s.

(No!) Then [redacted] $\sum_{i=1}^{n} \sum_{j=1}^{k_i} E\left(P(A_i^j) - M(A_i^j)\right)^2 <$ No!

$P_{i,j}(1)$ is the probability that the j^{th} symbol of A_i is 1, conditional on all previous symbols in

A_i [redacted] and the A's before it.

A_i has h_i symbols; the h_i+1 th symbol of A_i is always the symbol s (space).

the zeroth symbol of A_i is Λ (null).

0 Then $E \sum_{i=1}^{n} \sum_{j=0}^{h_i} \left((P_{i,j}(0) - M_{i,j}(0))^2 + (P_{i,j}(1) - M_{i,j}(1))^2 + (P_{i,j}(s) - M_{i,j}(s))^2 \right) \le k \ln 2$.

— The Expected value being with respect to M

Instead of this "E" I'd like to make clear just what E is. Hvr. look at 78 T3 com: T. way E is done makes .23 look very wrong!

23 Is .20 + sum of terms like $M_{i,j}(Q) \left(P_{i,j}(Q) - M_{i,j}(Q)\right)^2$?

Probably False. I could write it as $\sum_{l=0,1,s}$

More generally! If the A_i's have a larger alphabet than 0 & 1 the first Σ will range over those symbols plus the space symbol.

I guess the main trouble with expressing .20 w.o. "E" is that the "Z" sequence ranges

0 over all possible trinary strings: $\sum_{Z} P(Z) \sum \sum \sum (\ )^2$) is correct.

perhaps write .20, then express it in this form. $m \equiv \sum_{i=1}^{n} (h_i + 1)$

Z_l is seq. of m symbols. There are 3^m of them $l = 1 \ldots 3^m$.

35 So in .20 insert $M(Z_l)$ at α & sum over $l = 1 \ldots 3^m$.

spec.

.00 (808.03): To do this, we express the sequence of A_i's in eq(1), and the spaces between them, as a ternary sequence, Z_m. We hypothesize a very good probabilistic question answering algorithm, M, and, using Hutter's generalization of the

.02 theorem of S78, show that the symbol probabilities assigned by the algorithm of eq(1) must converge rapidly toward one another.

SPU [redacted] that all 3^m of the Z_i's represent n A_i's.

Perhaps sum over only that Z_i that have just n "S" symbols. If a says doesn't contain n S's it has $P_i = 0$ from both P & M. It is easy to compute how many

.10 Logical (n S's) $\leq$ Z_i's there are! $2^{(\text{total no. of bits in } Z_i)} \times$ the no. of ways one can place n S's in U places

---

SN If M is a very poor predictor of the A_i's: eq(1) will also "converge to M"!

— I think we also assume that M is the "creator" of the A_i sequence that was actually observed

So .02 doesn't really deal with

.18: 808.03 To do this, we hypothesize that the actual finite sequence of A_i answers $M(A/Q)$ that have been observed, were created by a probabilistic algorithm, M, and

.20 — that M can be described with k bits, using the reference machine of Appendix A that assigns apriori probabilities to all partial recursive functions. e.g.

Any probability distribution that assigns probabilities to every possible A_i, must also assign probabilities to each bit of A_i. Suppose that s_r i.e. is a string of the first r bits of A_i. Then the probability given by M that the $(r+1)^{th}$ bit of A_i is ϕ is

$$\sum_j M(s_r \phi x^j) \Big/ \sum_j M(s_r x^j)$$

$|Q_i$ $|Q_i$ log log of

x^j ranges over all finite strings.

.30 Similarly, the algorithm of eq(1) can be used to assign a probability to every bit of every A_i. — italics, i.e. — which we will call P,

We will represent the sequence of A_i's by a string Z, of these A_i's separated by the symbols, S — denoting "space". Z, then, is a sequence of symbols from the ternary alphabet 0, 1, S. Both M and P are prediction algorithms for the symbols of Z. Using an argument similar to the foregoing, it is clear that both M and P are able to assign probabilities to the space symbol S, as well as to ϕ = 0 and 1.

N. 8.02 C8 — See how this comes out (BAD) — 810

Hutter, M., Optimality of Universal Bayesian Sequence Prediction for General Loss and Alphabet http://www.idsia.ch/~marcus/official.htm

.30: 809.40: We have, then, two probability distributions on ternary strings Z. The first distribution, μ is the creator of the observed sequence and the second distribution, P represents eq (1) and tries to predict the symbols of Z.

For two such probability distributions on binary strings, the corollary of Theorem 3 of Sol78 applies: The expected values, with respect to μ ("Regularizer") of the sum of the squares of the differences in probabilities assigned by μ and P, to the bits of the string, are less than $\ln c$, c being the largest positive number such that $P > c\mu$ for all arguments.

$\frac{1}{2}\ln 2 \cdot k$; ~~k being the number of bits in the description of μ~~

Hutter (Hut 01) has generalized this corollary ~~beyond the binary alphabet~~ so it applies to systems with any finite alphabet: We are here concerned only with the ternary alphabet, but it implies a corresponding theorem if the symbols of the A_i are from any finite alphabet.

The result is

$$\sum_l \mu(Z_l) \sum_{i=1}^{n} \sum_{j=0}^{h_i^l+1} \sum_{t=0,1,s} \left(P_{i,j}^l(t) - \mu_{i,j}^l(t)\right)^2 < k \ln 2 \qquad (13)$$

Superscript. — 0,1,s. — a, c, s

A_i^l is A of Z_l

$P_{i,j}^l(t)$ is the probability, as given by P, that the j^{th} symbol of A_i^l will be t, conditional on previous symbols of A_i^l as well as previous A_j^l's in the sequence Z_l and the corresponding Q's.

t takes the values 0, 1 and s, but it is independent of previous A_i^l's in the sequence.

$\mu_{i,j}^l(t)$ is defined similarly to $P_{i,j}^l(t)$. The (h_i^l+1)th symbol of A_i is always s.

no ?

h_i^l is the number of bits in A_i^l.

The total number of symbols in Z_l is $\sum_{i=1}^{n}(h_i^l+1)$

Σ over l sums over all strings Z_l that consist of n finite binary strings separated by s symbols (spaces). — no (s)

$\mu(Z_l)$ is the probability that μ assigns to Z_l, in view of the sequence of Q's.

k is the length of the (shortest) description of μ.

This implies that the expected value with respect to μ of the total squared "error" between P and μ, summed over the individual symbols of all of the A_i, will be less than $k \cdot \ln 2$. → (.34)

.29

SN eq 13 is a bit funny because it sums over Z_l's of different lengths — but This makes no difference in the result: T. Q is 1 is the "proof" o.k. The Theorem I've been using is for μ's that generate strings of length n,

For an essential proof use method of Gacs in Li+Vit. ~~for that he used for~~ the same measure convergence [illegible].

34 – (.29) Since the total number of symbols in all of the answers can be very large for even a small number of questions, it is clear that the error per symbol decreases rapidly as n, the number of Q A pairs increases.

.34

Gen'l Theory of Problem Solving

00: 810.40 : SN A General Theory of Problem Solving:

Steady state operation: Old Ideas ≡ OI', New Ideas ≡ NI',

- New ideas are formed by combining old ideas in ways that solve problems.

Methods of combining ideas are in themselves, ideas; so we can do all kinds of reasoning by simple "composition" of ideas

To solve a problem, we have to have not only the necessary old ideas to be combined to solve the problem, we have to have good ideas on how to narrow down the search (so it takes place w. as little cost as poss!) for an adequate combination of old ideas. So a good way to solve problems would be to have a idea that looks at the problem and looks at available old ideas and is able to map out a search strategy to find a new idea (or old idea) that solves the problem. Ideas of this sort, I called PST's.

Some ways the things might work: Search is often a v.g. search method — in some cases v. badly. One good auxilliary PST, might be for the PST to look at the problem & give a PD on old ideas that could be combined to solve the problem.

I really have to work this out in more detail: it is essentially the way TM is supposed to work!

A basic idea: Rest is "All info is in the P.D. Rest (such is v. best)". However, the P.D. must include PST's, search functions, etc.

So if we use Unattended Mind Normal Search: We want an alg that looks at the problem, and the set of available ideas, & implements a search strategy. — I think this amounts to a PST.

Perhaps every time a problem (or sub problem) is solved, all relevant PSTs are "updated" — T. may or may not "update" way they behav. PST's.

SN IMPT & perhaps related to forgoing.: We write a short, elementary TSO, that we expect should be learnable by a human in not a long time (very we may be able to test this empirically!). We then have to discover a set of heuristics that are adequate for that performance. If we can't do this: we don't really "Adequately understand" the performance of .31 (Note 816.00ff on "Scaly"

799.00-40, 812.00+
799.00-10

00: 799, 10 : By modifying the guiding probability distribution, in view of ~~previous~~ the system's previous experience, the solutions to future problems are given higher probabilities ~~[illegible]~~. Since the time to find a solution by Lsearch is proportional to the time needed to generate and verify a correct solution, divided by the probability assigned to that solution, it is clear that ~~[illegible]~~ increasing the probability ~~[illegible]~~. Reformulated modification of the probability distribution will decrease ~~the~~ the search time needed for a solution.

?? ~~[illegible]~~ Does this technique modify the probability of solution enough to permit the solution of difficult problems in acceptable times? We will be using several methods to modify our probability distribution. The adequacy of these techniques is discussed in ~~[illegible]~~ section 3 ~~[illegible]~~ on "Scaling." Not really!

In our initial training of the Q,A induction system, we used a particularly simple form of Lsearch that is appropriate to induction problems. We search for productive functions in order of their a priori probability divided by the time needed to generate and test the trial. This search is "Blind" in the sense that it contains no information on the nature of the new induction problem to be solved. ~~[illegible]~~ The search algorithm may know about the task, but it does not know what it is looking for — except that the trial must satisfy a certain test. Superficially, this would seem to be the same as ~~to discern~~ inversion problems — in which we are searching for a string that satisfies certain conditions. The difference between the two problems is that in Inversion, we want only a string that satisfies the conditions and we want to find it as rapidly as possible. In the case of induction we want to find a function (represented by a string) that satisfies certain conditions, but we want the description of the.

This search algorithm is "Blind" in the sense that its trial functions ~~[illegible]~~ depend solely on past data and not at all on the target ~~[illegible]~~ that it is trying to generate. ~~As the system has~~ When the system has had enough experience with certain kinds of induction, we will be able to employ and update more effective search algorithms for ~~[illegible]~~ problems (which are ~~[illegible]~~ kinds of time limited optimization problems and induction problems, time-limited optimization problems and induction problems). In general, these are usually "not" "Blind" algorithms.

N902 ID

~~815~~
812½

799.40 seems more relevant

789.40 ← ?? Seems irrelevant very unlikely!

00: 812:? The system described is this far a pure inductive system, but and we except it to work soup problems of some difficulty, it has a serious deficiency — it uses a "blind" search algorithm looking When it is ~~searching~~ for a prediction function to fit the known data, it's trial functions depend only on summaries of past data, and not on the target function that it is trying to generate. [It is characteristic of inductive inference problems ~~are characteristically characterized by the fact~~ that they are often solvable by this kind of search. ~~We will next describe two kinds of problems in which blind search is usually inappropriate.~~

We can, however, do much better. After our induction system has been applied to the updating of search algorithms for more general problems (i.e. inversion and time limited optimization problems), it will be possible to use it to update more general (non-blind) algorithms to search for good induction functions. One motivation for working inversion and optimization problems is the expected improvement in the general induction system, obtained by sharing concepts used in updating. The induction system itself can be viewed as a kind of "Bootstrap Device" in which any improvements in induction translate into better updating and more improvements in induction.

N.10.02 ID

813

INTRO continued

798.00–.40 has been "Eap'd"
798.00–.40 " " "Tex'd"

: continuation of ≈ 798.00:

The ~~[illegible]~~ system described uses a limited form of Lsearch that is particularly effective

.02 for induction problems. → (.05)

.03 ~~To solve Inversion problems and Time limited Optimization problems,~~ we will use

.04 Lsearch in a ~~[illegible]~~ different, potentially, more powerful way.

.05:.02 → For the first ~~problem~~ Q_1, A_1 problem, we ~~try~~ ~~to find a~~ use Lsearch to find a ~~[illegible]~~ function ~~[illegible]~~ $f_1(\cdot)$ such that $F(A_1) = Q_1$, and ~~the~~ ~~[illegible] a priori probability, i.e. code length of~~ ~~$F_1(\cdot)$ is as short as possible.~~ the apriori probability of $f_1(\cdot)$ is as large as possible. In searching for a solution to ~~the second problem~~ we want a $f_2(\cdot)$ such that $f_2(A_1) = Q_1$ and $f_2(A_2) = Q_2$. and the apriori probability of $F_2(\cdot)$ is as large as possible. However, after we have solved the first problem, our a priori probability distribution changes. It includes the definition for $f_1(\cdot)$. As we solve more problems, the relevant probability distribution used for searching changes in accord with the past solutions.

3 or 4 kinds of learn
1) simpler induction
2) INV
3) OZ
4) Generalized OZ
5) TAF
b) parallel
c) random.

F( ) By mistake

~~Insert .02–.04~~

TLO

To solve Inversion and Time limited Optimization problems we started using a technique similar to that for induction. In the case of INV problems, we are given a strings, and a function $g(\cdot)$ that maps strings into strings. The problem is to find ~~a string~~ ~~X~~ such as rapidly as possible, a string (x) such that $g(x) = s$; ~~[illegible]~~

Conventional Lsearch ~~[illegible]~~ ~~[illegible]~~ searches over functions that look at $g(\cdot)$ and s and generate trial (x) values as output.

As in ~~[illegible]~~ induction problems, ~~the functions F(X,) are searched~~ ~~in order of their apriori probabilities, with a rejection condition that~~ Lsearch uses an apriori probability to guide the search for a suitable $f(\cdot,\cdot)$ functions. After ~~[illegible]~~ a problem has been solved, the solution to that problem modifies the a priori probability for search for a solution to the next problem — just as in induction problems.

O.K.

N·10·02 SD

INTRO
Continued

617 625 8960

[redacted] : 813.40ff After several Inversion [redacted] problems have been solved, it becomes possible to update the guiding probability distribution in a more effective manner.

We have several problems [redacted] that have been solved, we have the functions [redacted] that have solved them and we know how long it has taken to solve each of them. From this data we can use our earlier Q, A induction system to get {\em new} probability distributions for the time it will take for various $F_i(\cdot|\cdot)$ functions to solve a new, IMV problem. From these distributions we derive a probability that each $F_i(\cdot|\cdot)$ function will be the fastest in solving a new Inversion problem. We can then use this probability distribution over $F_i(\cdot|\cdot)$ to guide the Lsearch for a solution to the Inversion problem.

Inversion

For Time limited Optimization problems, we use a similar solution technique. Since induction problems are a kind of time limited optimization problem, we can use this method to greatly improve our solutions to induction problems as well.

This can be put in intro — but refer to "Eq(1) can be solved in 2 ways" description.

Section 1 describes the initial induction system. → (.29)

eq(1) is unclear here at that pt. : It may be possible to explain.

(S.N) Possible opening of intro or in Section 1 (and QATM)

This optimization can be done

Eq(1) can be solved in (at least) two different ways:

First by direct Lsrch for a O^j that best fits.

Second, by a very general optimization technique that uses induction as a subroutine. This is best implemented after the system has had much experience with the simpler kind of induction.

eq. 1 can be solved in largest 2 ways
1) direct Lsrch
2) a very general OZ technique that uses induction as a subroutine
[illegible]
is best implemented after system has much experience with simpler induction

Perhaps have Appendix C on Lsrch.

§1 QATM
§2 INV
§3 OZ
§4 [redacted] Conclusions: Main problems [illegible] design & "Scaling".

.24 : (20)

Section 2 Tells how to work Inversion Problems — greatly

Section 3 Tells how to work Optimization problems. Time limited optimization is generalized to include time varying optimization. Inversion problems become a special case of time varying optimization. We show how this formulation can be used to improve the induction system & [redacted]

Section 4 Discusses the critical problems of writing training sequences and dealing with "Scaling" — the possible need for new techniques to deal with larger, more difficult problems.

{ Perhaps Discuss 811.31 & "If all info Super Hyper wrapper" idea: that system can

is in P.D. Lsrch is ≈ optimum also find any PSF (w/ suitable TSQ).

00:816.40 : Section 2 : Inversion Problems.

In Inversion problems, we are given a string, S, and a function G(·) from strings to strings and we are asked to invert G with respect to S: to find an x such that G(x)=S. We want to do this as fast as possible.

One way to solve such problems is to use Lsearch. (Appendix C describes different kinds of Lsearch). In one simple kind of Lsearch, we look for a function, F_i that maps the description of G(·), and the target string, S, into a trial string, x_i. ($P_0(F_i)$),

$x_i = F_i(\tilde{G}^*, S)$

$\tilde{G}^*$ is a string that describes G(·). It is usually a program. We can use the language AZ of appendix A, to obtain a probability distribution, on all possible $F_i(\cdot,\cdot)$. the descriptions of the

(ommit) We have an apriori probability distribution on F_i,

We have an apriori probability distribution $P_0(\cdot)$ on descriptions, $\tilde{F}_i$ of Possible $F_i(\cdot)$ functions.

From $P_0(\cdot)$ and G(·) and S, we can use Lsearch to obtain a solution, x_i (ok) $= F_i(\tilde{G}, S)$ such that $G(x_i)=S$. If it takes a total time t_0 (cut off time) to Generate F_i from $\tilde{F}_i$, generate x_i from $F_i(\tilde{G}, S)$ and test $G(x_i)=S$, (o.k.) will take about time, $t_0/P_0(\tilde{F}_i)$.

Then Lsearch will The total time for Lsearch will be about

(ommit) At first, the probability distribution P_0 will be obtained from a universal algorithm (say a Turing machine or other universal computer) able to simulate the function F_i, using its description $\tilde{F}_i$. One easy way to get P_0 is to make $P_0(\tilde{F}_i) = 2^{-\ell(\tilde{F}_i)}$. Here $\ell(\tilde{F}_i)$ is the length in bits, of the description, $\tilde{F}_i$.

So time for Lsearch is about $t_0 \cdot 2^{\ell(\tilde{F}_i)}$

After we have solved the first problem, we will put the definition of F_i in the "parameter list" of the language used to generate trials.

When we have solved several problems in this way, the language will have several useful functions in its parameter list. It will usually be possible to find common sub functions that can be defined, to increase the probabilities of past solutions to problems, just as was done in the solutions to the induction problems of section 1. In general, all of the techniques for increasing the probabilities of solutions of induction problem in section 1, can also be used for helping to solve inversion problems.

4P Jan 10 4PM.

: [SN] On "Scaling": In S89 (or perhaps S86), I thot that if a problem had excessive CJS, T. main recourses ① faster, bigger machines ② factor the concs used into smaller pieces ("subchunks") and write a suitable TSQ for these sub chunks.

A possy I hadn't explicitly considered, is the acquisition of the needed concs in an essentially difrnt way — by reasoning, say, or by a novel search method. (Also the possy that the initial "factoring" was inadequate, and a different "factoring" was nec.)

All such prob solving techniques are "PST's" and can usefully be regarded as "concs" to be lrnd — and applied: One must also have concs that tell when to apply these PST's: otherwise rcost is too hy!

We still can usefully use the idea of a "Conc. Net"; but it will have some rather complex PST's in it, along with the "chaperone"? cons that tell when other cons should be applied. "Marriage Broker"? "Head hunter"

All of the mechanisms that I devised for Elaborate Search, optzn: (E.g. the hyer order optzn method; the realization of "correlations" betw. possl. PST's; the approximation of correlns; updating during Lsrch, dropping Lsrch in favor of won (or chosing "most promising" PST) etc.) can (or perhaps should) be regarded as concs in a conc. net.

The remark on 811.31 is also very relevant — We must be able to (find / discover?) (trainer) all needed hours and be able to express them in the conc. Net.

Suppose we have a new problem, $(\tilde{G}_n, s_n)$ and we want the probability that an arbitrary PST, $F_\ell(\cdot,\cdot)$ will solve this problem faster than any other PST considered. We could then use such a distribution for a very goal oriented search.

How can we obtain such a distribution, $P_\ell(F_\ell)$?

00: 815.40 In section (1) we wanted solutions of maximum probability. For Inversion problems, since search time is inversely proportional to probability of solution, high probability solutions will tend to give us the minimal search times and lucky ones.

Each of the $F_\ell(\cdot,\cdot)$ that has solved at least one problem, can be regarded as a General Problem Solving Technique (PST).

In the foregoing the one mode of solving problems just described, It simply tries to imitate PST's or sub-routines of PST's that have been successful in the past. The system does not "know" that it is supposed to find solutions fast. And does. Though this method works

10 to some extent, it can be much improved.

Consider the set of quadruples: $\tilde{G}_j, s_j, F_j(\cdot,\cdot), t^j$

$\tilde{G}_j$ is a string that describes the j^{th} function to be inverted

s_j is the argument for the inverse of $\tilde{G}_j$

$F_j(\cdot,\cdot)$ is the function of $\tilde{G}_j$ and s_j that solved the j^{th} problem

t^j is the time it took

For $F_j(\tilde{G}_j, s_j)$ to generate the correct solution to the j^{th} problem.

We have a quadruple like this for each problem solved

For the new problem, $(\tilde{G}_n, s_n)$ given any $F_k(\cdot,\cdot)$

20 we can use the technique of the previous section, on induction, to get a probability density distribution of the time it will take for F_k to solve $(\tilde{G}_n, s_n)$. That probability density distribution we regard as Q_j and t^j as A_j.

Here, we regard $(\tilde{G}_j, s_j, F_j(\cdot,\cdot))$ as

Q_n is a probability density on the solution time t^{sol}. As output A_n, we get the desired probability density distribution on the solution time t^{sol}. F_k may or may not be in the set of F_k that have been tried.

From the foregoing, for arbitrary $F_\ell(\cdot,\cdot)$ and problems $(\tilde{G}_n, s_n)$ we can obtain a probability distribution $h_\ell(t)$. This is the probability that $F_\ell(\cdot,\cdot)$ will solve the problem in time less than t.

30 Usually there will not be enough data to get a very detailed $h_\ell(t)$.

A three parameter distribution might be of the form:

134 $$h_\ell(t) = \int_0^t a\,N(x,\mu,\sigma)\,dx$$

N is a Gaussian (Normal) distribution on x of mean, μ and variance, σ^2.

ℓ of $h_\ell(t)$ is the probability that F_ℓ will ever (in $t < \infty$) solve $(\tilde{G}_n, s_n)$.

Often we will only use Part of data will often demand that we use less than 3 parameters.

→ 818.00

so: 817.40 Given a set $[\tilde{F}_\ell]$ of PST's and their associated $h_\ell(t)$ distributions for solving G_n, S_n we can obtain a probability that each $\tilde{F}_\ell$ will get the fastest solution (smallest t) for this problem. This probability, P_ℓ is

$$P_\ell = \int_0^\infty h'_\ell(t) \prod_{j \neq \ell} h_j(t)\, dt = \int_0^\infty \frac{h'_\ell(t)}{h_\ell(t)} \prod_j h_j(t)\, dt .$$

While this integral looks difficult (time consuming) to evaluate, not much accuracy is needed and it is relatively simple to calculate, using a Monte Carlo technique. We randomly sample from each h_j distribution and pick the ℓ value which is largest. The frequency with which h_j is largest will be an estimate of P_ℓ. Since little accuracy is needed, we can use very simple approximations to model each h_ℓ.

If the values of the P_ℓ are initially expressed in Monte Carlo form, we can use that form to guide a Monte Carlo search (as described in Appendix C) for a solution to the nth problem. [illegible] describes a Monte Carlo version of Lsearch.

The foregoing may not always be a practical way to solve Inversion problems! The time needed to compute the P_ℓ's of the relevant F_ℓ's can be much longer than the time needed to test each F_ℓ empirically! In this case, we should use the foregoing analysis as a guide to approximating what we want. The main superiority of this method is that the system already knows that it must try to find a F_ℓ that obtains a fast solution.

Need revision

In this case the present model should be regarded as a kind of ideal that we try to approach as closely as we can, in whatever time we have available.

In this case, we must find different methods of solution — but the model described gives the (only) goal of such methods — makes it possible for us to criticize and improve them. A clear advantage of the system described, is that it is not a "blind" search — the system (really knows) that it must find a F_ℓ that obtains as fast a solution as possible.

See 819.13–.15 for a not bad continuation.

[illegible] subsection (maybe Section 2.2)

The last PP was somewhat modified — sounds much better now.

Also, numerous criticisms 823.25–.31.

t. discussion of approach. methods of 820.15–821.30 was not included; 821.09–.24 was deleted from "Report".

"new" ABCDefg / "use" ABCDefg

Subsection to section 2 on Inversion Problems.

00:818.40: **Problem Solving Techniques** The first attempt to map problem descriptions into candidate solutions may be regarded as "Problem Solving Techniques" (PST's). Initially, a ~~they will be put into the system by the trainer.~~ number of such PST's will be ~~inserted into the system by the trainer.~~ obtained by the system using as the "solutions" to suitably simple problems — obtained by Lsearch over PST space, using a universal distribution on functions — ~~(as described in Appendix A.)~~ [illegible] This will give us a set of PST's "known to be (at least occasionally) useful. We can then apply the technique of eq. ( ) or its approximations, to obtain a PC distribution for guiding Lsearch to any new problem — but our set of PST's would be limited to those that → 821.00

.04

10 [illegible]: : Just as we do not expect an exact solution to equation (1), we don't expect an exact evaluation of eq (818.05). We simply want to get as good an approximation

.12 as possible, in the available time.

.13 ~~Eq. (8~~ In such cases, eq (818.03) may be regarded as a description of "where we want to go" rather than "How to get there" — and the reasoning leading up to it. Approximation techniques should be guided by

.15 the goals and subgoals implied in this discussion.

SN t. idea of "A well defined Induction problem", corpus & a priori program; it is clear what is to be optimized.

20 Some items I may want to include in either "T. report" or [illegible] t. Augmented Report (My Version)

1) How to do time series' 2) T. MXT corpus problem & t. Mxd corpus Pgm.

3) Much detail on a) T. 3 Gambo House Pgms & Just why & to what extent "all info is in PD" enables Lsearch to be ≈ optimum.

4) exactly what "factor of 2" means in General solns to problems I.e. devoting ½ time to Lsrch, ½ time to updating GPD(s).

5) The mechanics of optzn of both continuous & discrete params of a system.

30 6) Consideration of t. set of all PST's: Obtaining an PC ~~group~~ that ~~they~~ each one will be (fastest / best) for a particular problem. I've ~~recent~~ written much of this recently, I may have a "Review".

7) A "Preface" Explaining that this is a "Report", not a "paper": Also. ~~That~~ t. Abstract gives t. General picture of an advanced system: a strong subgoal. Before we can do this, we will implement a limited kind of system that works on QA. induction only.). ~~Those t. kind of system~~ The t. Algorithm it uses may ~~be quite able to solve difficult problems~~, be of considerable power, it has the ~~a certain deficiency~~ its use of "BLIND" search algorithm ~~it is a kind of "Blind Search"~~ necessary weakness of "Blind Search".

Taid: Abstract (several warnings — use / start — may Maybe for Gammer'.

Intro B is Total info.

Section 11 Induction Inference is sec 1. TDX

~~Com probl 10/17/02~~
~~" " 2 11/10/02~~
~~" [illegible] this step~~
~~[illegible]~~
~~(this is good!)~~
~~Delete "Abstract"!~~
— This is good.

Take laser pointer!

00: 819.40 : The more advanced system, uses a somewhat different search algorithm that overcomes this deficiency.

1) Simple Induction Lsrch: Looks for function > 414.18 to make search best fncts in Least order.
2) Next: fnct looks at problem descr & proposes soln., we Lsrch on each function.
3) For functs of 2) we "learn" which functs are likely to solve a given problem rapidly.
4) We apply the method of 3 to the "library" of problem-solving methods. (Bootstrap): Self-improvement.

8) Another item! Past $\frac{b \pm \sqrt{b^2-4ac}}{2a}$ takes ... bits to derb: | $\sqrt{x^2+2bx+c}$ or $\frac{b+\sqrt{b^2-4c}}{2} = \frac{b}{2} + \sqrt{\left(\frac{b}{2}\right)^2 - c}$; $\tilde{b} + \sqrt{\tilde{b}^2 - c}$

10 If it took 1 loons per trial, we have to wait ~ 70 yrs to get ≈ X2.18 ... a soln. in 1 second. Got pc of or

This [illegible] results in a way of looking at search in general / show how additional complexity ("a coil for x^2 ... m yrs to wait)

9) Discuss how we get "Logical reasoning" into System. Maybe in section on TSQ's & "Scaling".

---

15 For end of first subsection of section 2:

One way to get approximate P_e distributions much more rapidly:

After we have obtained P_e distributions via eq (818.03) for several problems, over one or more PST sets, we can use the induction system of section 1 to probabilistically extrapolate these results.

A final result of this induction: W (problem descr.) output is PST$_i$, P_i pairs in ≈ P_i order (highest P_i first?) with P_i being probability that PST$_i$ is the "Best" of the set of PST's for that problem.

[margin note:] I mean need $O^i(A|Q)$ assignd form of $\delta \cdot P_d$. another! $W^j(Q) \to$ list of A, P pairs in ≈ P roby order.

Whether or not 15 gets into report, it is probably an Essential Part of the System!

I don't have time now to develop this adequately, but maybe one line in the previous subsection of §2!

GOOD!

29 One nice way to obtain a fast approximate function that assigns P_e values to the $F_e(\cdot|\cdot)$ for a given problem description is to calculate the P_e's for a set of problems, using eq (818.03), then use the inductive technique of section 1, to extrapolate this relation to new problems and possibly new $F_e(\cdot|\cdot)$ sets. The accuracy of the assignment will, as in all induction problems, depend much on the size of the data set, and the amount of time spent searching for a solution.

spec eg

00 : 820. : We have as problem solutions. We can expand that set of PST's in several ways:
One way is to have a set of PST's inserted by the trainer — who thinks they should be good. Another way to extrapolate

Change 8 → G. ()

for end of first part of section 2. u.c.

eg
10. One way to get approximate P_ℓ distributions more rapidly:
After we have obtained the P_ℓ distributions via eq (818.03) for several problems
over several F_ℓ functions, we can use the induction system of section 1 to
probabilistically extrapolate these results to new problems and possibly
a new set of F_ℓ functions. the output of the inductive system

No omit Can express its probabilistic output distribution in several ways: in the present case

We regard the problem as a "Q",
each
and possible F_ℓ functions as a possible "A". The induction

takes the problem description as input, and lists the possible F_ℓ
functions as outputs, each with its associated P_ℓ — in approximate P_ℓ order

22. The system can express its output distribution in several different ways: In the
present case if there are many F_ℓ, a good way would be to make a listing of each of the F_ℓ, paired with their
probabilities with the pairs in rough probability order.

Actually: .22 is usually wrong? Unless there are an enormous no. of F_ℓ's,
We try all of them each T ← 2T round, so the order in which they F_ℓ are listed is unimportant.
If the no. of F_ℓ is large (→ ∞) then we don't do all of the F_ℓ each round:
In the nth round we only do F_ℓ's that have $pc > 2^{-n}$.

See 823.25 -.31 for Serious Criticism of .09-.24

20 : 821.40 : (Section 3) Optimization problems.

There are many kinds of Optimization problems. ~~The~~ The commonest form: we are given a function ~~E~~ $G(\cdot)$ from strings to ~~integers~~ real numbers (strings include numbers as a special case). We have to find an argument x so that $G(x)$ is maximum and prove that it is maximum. ~~Then~~ We have to do this in as short a time as possible. This is regarded as a kind of Inversion problem and is solvable using the techniques of Section 2.

If we are given a ~~time~~ time limit and are asked to find the best x that we can in that time, then we have a "Time limited Optimization problem".

Time limited optimization problems can have many forms. ~~not all of which can be~~

~~For~~ (1) The function $G(\cdot)$ can be given as an algebraic expression that is "visible" to the problem solver ("Transparent G")

(2) The function $G(\cdot)$ is a "Black Box": The system is allowed to present an input to G and it will receive $G(x)$ ~~to~~ after a time, Δ. If Δ is small relative to ~~the~~ one's "instruction time" we will have one kind of solution. If Δ is very long with respect to an instruction time, we will have a different kind of solution.

(3) G may have "noise" in it: ~~to~~ If one presents the same x to G several times, one ~~may~~ can get different output each time. We may have to rephrase the solution criterion and ask for best "expected" output of $G(\cdot)$.

(4) G may ~~be certain~~ vary with ~~[?]~~ time in a known way, so we know ~~G(x, t). ~~ The function ~~G(...)~~ $G(\cdot,\cdot)$ and we need to maximize $G(x,t)$, t being the time at which the ~~sum~~ solution is presented. ~~the function of the solution~~

Designing ~~a system~~ a car in 6 months satisfying certain constraints, having lowest cost, is an ordinary time ~~lim~~ limited optimization problem. But suppose we are given no sharp time limit; ~~but~~ instead, we know that the later we produce the car, the more likely it is that our competitor will produce a competing car. If we have some idea as to ~~how the value of that~~ how the value of our car design depends on its cost and how late we produce ~~the car~~ it, we have ~~an example of~~ ~~time limited optimization~~ a Time varying optimization problem.

~~Inversion problems~~ Ordinary time limited optimization problems can be regarded as ~~be~~ having G functions that are independent of t, for $t \le t_0$ but G ~~[?]~~ $= -\infty$ for ~~[?]~~ $t > t_0$: · · · ($t > t_0$) ($t \le t_0$)

Inversion problems can be regarded as having G functions of ~~[?]~~ the form $H(x)/t$. Here ~~[?]~~ $H(x) = 1$ ~~[?]~~ if x ~~[?]~~ is the inverse of S, ~~[?]~~ and $H = -\infty$ otherwise.

Possibly mention that Humans ~~seem~~ usually seem to solve Inv. probs by converting them to OZ problems.

Also Good: A discussion of "Anytime problems": that to finishing often seems to Give what one wants in ~~[?]~~ practical solns to ~~often~~ problems. Same kind of G & t is needed to define the problem.

(Section 3) Time Limited Optimization.

use PST concept: it was introduced in § 2.

~~Ordinary Time Limited Optimization problems are solved in a way very similar to Post for~~

In the previous section, we described two ways to solve inversion problems. use L search to In the first, the guiding probability distribution is directly modified by previous solutions to inversion problems. In the second, we use the induction technique of section 1 to obtain the probability that each candidate function will give the correct solution to the problem, and we use this probability distribution to guide L search.

While the first seems to be applicable to the solution of only time limited optimization problems, the second method can be used for more general time varying optimization problems.

We will first describe the solution of time limited optimization problems. Then show how time varying optimization problems are solved.

In time limited optimization, we have a function from strings to reals, $G(\cdot)$, and we have time limit t. We must find an x in time t such that $G(x)$ is as large as possible. The string x can represent more generally, the string x_i can represent a number.

These problems are solved in each like the inversion problems of section 2: (§21.23)

We have a set of PST's, F_i, that look at the problem $(G(\cdot), t)$ and given after a requested time, t_i, present a solution candidate, $x_i = F_i(\tilde{G}(\cdot), t_i)$, which is evaluated as $G(x_i)$

Previously $F_i(G(\cdot), t_i)$ ... to test each F_i empirically. In such cases (eq (B18.30)) and the reasoning leading to it, may be regarded as a description of "where we want to go", rather than "how to get there". We must find different different methods of solution.

Solution methods

Method 1: In the resp. inductive procedure. (§21.09 - 21.24) because it would be easier to use empirical optimal (to beat competitors in induction. Approximation suggested by Approach). Doing so would seem to be about as t. Old method "of doing L search! Write up carefully! Another point: Unless "t. System's inductive view, it will be better to use "Method 1" for search rather than Methods (eq. B18.3) to guide L search so Big-O can be only used if its faster than direct-empirical testing and t. induction System is already very Good. "Method 1" may be regarded as one way to try to approximate Methods.

SM T. Soy effect is similar to the "FDA effect" in biasing results by selecting out the empirical "best". — may call Soy → FDA ©. Also $\sigma^2 \to \sigma^2 \frac{N}{N-K}$ Effect.

Also SMA also maybe "The 1024 bit strings see scam": 1 of 2^{10} 10 bit strings is nitty every time for coin flips. Also the "Prayer Healed" experiment(s).

Anyway I MUST Think out .25 - .31 carefully. I'm not sure it's exactly right. — I really need to know just what kinds of approx. Guiding Pd" functs are Logical/useful.

20: (5 Dec 823.19): Initially, we have a guiding probability distribution, $P_0(F_i)$, that is the same for all optimization problems. ← Discussion in Appendix A

- $P(F_i) \approx 2^{-\ell(F_i)}$: $\ell(F_i)$ being the length of the shortest program we have found for the function F_i.

~~For each Problem $G(\cdot)_j$ t_j we will~~ Many optimization programs such as F_i make many trials, and get better and better results for longer run times. In the present case, we will run the PST, F_i for (at most) time $t \cdot P(F_i)$, and use $G(\cdot)$ to test the result. Checking all of the F_i will take time $\leq \sum t \cdot P(F_i) = t \leq P(R_i)$. Using $2^{-\ell(F_i)}$ will give approximate P values so that $\sum 2^{-\ell(F_i)} < 1$

so we have not yet used up all of our time:

We then double our testing times to $2t \cdot P_0(F_i)$ and retest all of the F_i. This doubling is repeated until we run out of time. We then select the x_i that has given the highest $G(x_i)$ value.

---

23 823.19 → We have, as for Inversion problems, an a priori probability distribution $P_0(F_i)$ that initially, is the same for all [Time Limited] optimization problems. We run the generation and testing of F_i and x_i for time, $t \cdot P_0(\tilde{F}_i)$. Since $\sum_i P_0(\tilde{F}_i) \leq 1$, the process takes $\leq t$ time. The x_i of maximum $G(x_i)$ is then selected as output. If there is remaining time, we can rerun the tests using times $2 \cdot t \cdot P_0(\tilde{F}_i)$ ← doubling and redoubling until all of the time is used up.

As with Inversion problems After we solve the first problem, we put the solution function, F_i into the parameter list of the language AZ used to generate the trials just as we did with Inversion problems. All of the methods used to improve the guiding probability distribution for Inversion problems can be used with minimal modification in the present case.

In the discussion of the improved P_i distribution, we are trying to maximize G rather than minimize t, so eq (817.34) becomes

$$h_i(G) = \int_{-\infty}^{G} a\,N(G, M, \sigma)\,dG$$

∈ eq (818.03) becomes

$$P_i = \int_{-\infty}^{+\infty} h_i'(G) \prod_{j \neq i} h_j(G)\,dG = \int_{-\infty}^{+\infty} \frac{h_i'(G)}{h_i(G)} \prod_j h_j(G)\,dG$$

The overlaid changes are the only difference from the eqns in section 2 namely $t \to G$

N20.02 ID

TSQ & Lsrch: | More on GA | i.e. of $\frac{b+\sqrt{b^2-4ac}}{2a}$

→ Transfer learning.

Learning Self-replacement

Deleted from Report that have to be investigated — in Aux Report
1) Method of getting P_S (820.15–821.30)
2) Time Varying Optzn.

Del : → Discussn of [Time Varying Optzn.] :

Say $h_i(G, t)$ is available from a $F_i(G(\cdot,\cdot), t \ldots \ldots)$?

$(G_i(\cdot), t_i)$ corpus available : first consider Gure i $G(x) \cdot L(t)$

So far now $G_i(\cdot)$ we have viz QATM: $h_i(G)$ as a function of t.

So $h_i(G,t) \cdot \to h_i(G \cdot H(t))$

For each G, t value, what's proby that some F_i will have $G \cdot H < G \cdot H(t)$. ?

$G_i(x) \cdot H_i(t)$

for given t, we have $h_i(G)$, + proby F_i will have its $G < G$

Perhaps give some idea how recursive fcns are defined >

My impression: That I originally had a "Soln." to the time varying G problem — not carefully worked out here. That later, I noted my uncertainty about it & worked it out in more detail (rather [illegible], tho). — But I'm not sure of this & remain unsure! → for a more recent Soln, See NIPS 176.02–.27 Approx. Note in Aux report

Best thing to do is leave time varying G out of the report — it would be a nice "Breakthru" to report, but I would better spend time on other aspects of an already much overdue, report. I may have mentioned "Time Var G" in introduction. It was only mentioned in the brief dern of "Section 2" So I cut it out.

NIPS 176.02–.11 is main idea

So the danger: Abstract, Intro, Section 1, Sec 2, Sec 3 the Lsrch

Appx A AZ, Appx B Conv

under Sec 3 02 the danger, Sec 4 TSQ's, Sec 5 gen remarks

Appx C Lsrch.

Preface, Preamble.

To define a common type of Recursion:

Consider defining the factorial function:

$F(x+1) = (x+1) \cdot f(x); \; F(0) = 1$. $f(x) = x f(x-1); \; f(0) = 1.$

$F(x) = x f(x-1)$; → $F(x) = h(x, f(g(x)))$ | $h(\cdot,\cdot); g(\cdot); a, b$.

one mite define this funct by allowing "a" to be a prim symbol.

For now end of APX A:

29 The factorial function is recursively definable by the expression $\mathcal{H}$:

30

$\mathcal{H}$ $F(x) = x F(x-1); \; F(0) = 1$

It is a member of recursive functions a class of functions defined by → $F(x) = h(x, F(g(x))); \; F(a) = b$. $\mathcal{H}$ (new line)

A function of this sort is definable by the list: $h(\cdot,\cdot), g(\cdot), a, b$.

More general classes of recursive functions may be defined in similar ways.

For ≈ and ≠ of A≠xA

Appendix A1

08:825.40 !

01 The factorial function is recursively defined by the expression:

$$F(x) = x\,F(x-1);\ F(0)=1$$

(l.c. 1, 2)

It is a member of a class of functions defined by

$$F(x) = h\,(x, F(g(x)));\ F(a)=b$$

(l.c. 1, 2)

A function of this sort can be defined by the list:

$h(\cdot,\cdot)$, $g(\cdot)$, a, b

09 Formalisms of this kind can be devised to define more general recursive functions.

SN Alternatively, if we allow definitions of funcs to refer to the funct being defined, we can simply include "a, b" after such a definition.

We must define a set of intermediate interdependent recursive funcs by first listing the funcs being defined, then give the defining eqns, then the "boundary conds".

It may be necessary to use "=" as a primitive.

However, a good method of expressing rec. (or other) funct defns, should be closely related to one or more heuristics for discovering funcs of that sort.

On "transfer learning"

Suppose we have two training sequences $[Q_i^1, A_i^1]$ and $[Q_i^2, A_i^2]$, each of length, n.

For extrapolation If we regard the sequences as completely unrelated, we will extrapolate the two sequences using

equation

$$\sum_{j=1}^{j_{max}^{1}} a_0^{j,1} O^{j,1} \prod_{i=1}^{n+1} O^{j,1}(A_i^1 | Q_i^1) \qquad 1)$$

and

equation

$$\sum_{j=1}^{j_{max}^{2}} a_0^{j,2} O^{j,2} \prod_{i=1}^{n+1} O^{j,2}(A_i^1 | Q_i)$$ 2)

We will try to find formulas such that $O^{j,1}$ / and $O^{j,2}$ so that

$$a^{j,1} \prod_{i=1}^{n} O^{j,1}(A_i^1 | Q_i^1)$$

and

$$a_0^{j,2} \prod_{i=1}^{n} O^{j,1}(A_i^2 | Q_2^2)$$

are individually as large as possible. In this case, each sequence extrapolates in its own way, and there is no possible transfer of learning between the two sequences

826.40 : To extrapolate our training sequences using "transfer learning", we simply put all of the data into a single training sequence!

For the 2n+1 th question, we get distribution!

$$\sum_{j=1} a^j_{2n} O^j(A_{2n+1}|Q_{2n+1})$$

Here,

$$a^j_{2n} = a^j_0 \prod_{i=0}^{n} O^j(A^1_i|Q^1_i) \prod_{i=0}^{n} O^j(A^2_i|Q^2_i)$$

All of the information in the two training sequences is brought to bear on the prediction of A_{2n+1}. Any useful functions or subfunctions common to the two sequences, can used to get shorter codes and more weight for the resultant prediction function.

Note: T. [illegible] discussion doesn't make clear that any thing useful has been done in Merging a 2 [illegible]: Actually, there are 12 relevant theorems about [illegible] $pc(A,B) \geq pc(A) \cdot pc(B)$.

[SN] R. Objection to Update Method 2: Not so valid! The function evaluations don't have to be done! They have already been done and we have them in Memory! Not relevant! — But in general, it's not clear that Method 2 (done literally) will really take longer than evaluating the F_i's.

I think method 2 may be one that looks at the problem and gets the probability the df's then integrates to get the final guiding P.d. This final guiding pd. looks at the problem's outputs a P.d. are F_i's, for that problem.

Perhaps I should make it clearer as to what the GCPD is & its 2 "components."

As is, there is no mention of GCPD in the text; just in the Abstract!

Could I have a section on "GCPD" after section on OZ?

We can use indices on inputs so TM can more easily mix corp's — In fact these indices are usually necessary for most arguments, so TM will have some idea as to what to "pool" the data with. This indexing is a very impt idea & is very relevant to Transfer learning.

(Section 4) General Conditional Probability Distribution (GCPD), Transfer Learning and Indexing.

Or maybe part of (Section 1) (on QATM). Actually (Section 1) ends w. a discussion of indexing.

Whenever a problem is given to the system, that problem will usually have associated with it one or more "indices" that tell where the problem came from, or other characteristics the human feels may be useful for the machine's education.

00: 827.40 : Continn of end of §1: (Call this section 1b if this has been inserted in §1 → easier)

One value of the indices is in "Transfer Learning": All of the data used in inductive problems can be regarded as one large "Corpus" (body of data). At first it may be best for the system to consider "sub-corpi" — data that have certain indices on their Q's. Each can be regarded as a separate training sequence, and may can be solved independently. It is, however, better to regard all of the data as part of the same large corpus, so that the system can use common functions in the inductions on "sub-corpi" to reduce the total number of bits to describe the data, thus materially increasing the value of eq(4.19.18)

This "sharing of functions" can be used to link any data in the system. What we have is an Extended General conditional Probability Distribution (GCPD) for all of the inductive data in the system. In sections 2 and 3 we will discuss Inversion problems and Optimization problems. Though they are not induction problems, they have induction probability distributions that guide the searches for solutions to their problems. These probability distributions will also be included in the GCPD.

10 to insertion §2 (Inv probs):

After P2 ... all possl. $F_i(\cdot,\cdot)$

23 Suppose δ is the time to do 10 machine instructions: fetch three P_i's We try all possible F_i's, using a maximum time of δP_i to generate P_y, and x_y and test x_y. This take a total time $\sum_i \delta P_i$ — which is $< \delta$ since $\sum P_i < 1$.

If we don't find a solution we run all the trials again using 2δ as our time limit. We keep doubling the time limit and doing all trials until we find a solution.

The total time for all of the trials will be $< \sum_i T_i P_i$. P_i is the probability assigned to the F_i that solved the problem & T_i is the time needed to generate and test the solution.

52

N21/02 IP

Section 3A (New) (.00–.03) — Done → §3

.00: Since the induction problems of Section 1 are (time limited) optimization problems, we can apply the techniques of the present section to their solutions. In particular, we can use the method that improves the P_2 distribution

.03 to obtain significantly better performance.

Some things I need to report:

1) How trainer can add PST's to system. How TM can extrapolate set as a stoch. Grammar. How trainer can help by factoring (partially factoring) the PST's. — in TSQ section? The idea of a "Conditional Grammar". — Mention in TSQ's.

10

- 2) The idea of Extreme wrapper system using self-replacement. in TSQ Σ

3) In §1 there is/usage discussion of Lsrch: refer to Appx C. (done!)

4) §3 pl add a mention of Lsrch (done)

5) For §2 Inv probs: PST's ... time to generate F_i. (done) with 828.23–.32 replace 3rd ¶

Perhaps in TSQ, discuss GHT's. All info is in PD.

20

• 6) Discussion of GA — maybe in Section where it is first Mentioned — §1 → QATM. We could have section on GA.

• 7) The update/search time ratio fixed at 2. (Significant), — Discussion of exactly what the "within a factor of 2" means. I could have a ¶ on this. — But where? "General Administration" — Gen remarks, Conclusions. Section 5?

8) The idea of SUMACS (Summary Machines). — not done, but put in AR

Get §1 in shape. Get it out. Names authorized. Get Greco to put this on Entropy paper.

30

9) Bibliography
3 Lev 72
4 Li V 93 (PP)
5 Li V 97 (PP (Lisp-like lang) NIS 2001)
2 Koz 95 Sec 1.
Sol 78
Sol 86
Sol 89
1 Hut 01 — ApxB

10) Augment intro w. names of new sections

00: 829: 40: ~~Stata~~ Appendix C: Levin's Universal Search Algorithm (Lsearch).

Levin's original paper (Lev 72) gave some properties of Lsearch, but didn't tell how to do it. The Li, Vitanyi book(s) (LiV93, LiV97) gave more details ~~a few ways~~ — including some ~~[illegible]~~ ways to implement it. We will discuss three methods of implementation: Limit doubling, Parallel search and Random search.

(3 of [illegible]) refers to Lsrch mostly.
Be Barbara Coglio
rad. to Appendix C

Subsection 1 Limit Doubling & Lsearch

For Inversion problems: Given $G(\cdot), s$, to find x in minimal time, such that $G(x) = s$.
We look for algorithms (PSTs) $F_i(\cdot,\cdot)$ such that $F_i(\tilde{G}(\cdot), s) = x_i$
Then we test x_i to see if $G(x_i) = s$. ~~[illegible]~~ The time needed to generate F_i, generate x_i and test x_i

~~The limit doubling method of Lsearch was described in section 2 (Inversion problems)~~

~~and~~ ~~The~~ Section 2, on Inversion problems, described the application of ~~[illegible]~~ the limit doubling method to inversion problems. ~~[illegible]~~

Section 3 described its application to time limited optimization.

Subsection 2

~~Subsection 2: Parallel Lsearch~~

For Inversion problems: We work on all PST trials ~~[illegible]~~ in parallel, time sharing our CPU cycles. The fraction of ~~time~~ time spent on the generation of F_i, generation of x_i and testing x_i

is P_i, the ~~[illegible]~~ a priori probability assigned to F_i.

This method is at least twice as fast as the limit doubling method.

~~While this method is ... if the P_i of the F_i that solve~~

omit: with careful programming, This method is about $-\log_2 P_i$ times as fast as the limit doubling method — P_i being the probability of the ~~[illegible]~~ P_i that found the successful x_i. However, it

requires ~~[illegible]~~ much more memory ~~[illegible]~~ and/or a clever disc swapping scheme ~~[illegible]~~

For Time limited optimization, as with inversion problems, we work on all PST ~~[illegible]~~ trials in parallel, using a ~~[illegible]~~ time share ~~of the~~ fraction of P_i to ~~[illegible]~~ PST, F_i.

We quit when time has run out, and pick the PST that got ~~[illegible]~~ best output ~~[illegible]~~ at quitting time.

Random Lsearch is a variety of parallel search. ~~Being only twice as fast as the limit doubling method~~ ~~We randomly ... with~~

With probability P_i we randomly chose the PST, F_i, and we work on it for a fixed time T, ~~[illegible]~~ If F_i is ever ~~[illegible]~~ again ~~[illegible]~~ chosen, we ~~[illegible]~~ continue where we left off, spending another T on it.

T should be ~~[illegible]~~ small, but "large" with respect to time needed to switch from one PST to another.

: 830.40 : The properties of random Lsearch are about the same as those of unparallel Lsearch. ~~Parallel Lsearch is probably harder to program.~~ It is particularly useful when the value of P_i is available in ~~[redacted]~~ Monte Carlo form — e.g. F_i can be generated by a Monte Carlo procedure, say, by randomly selecting operators or opcodes ~~[redacted]~~ with probabilities ~~[redacted]~~ that correspond to their a priori probabilities

Maybe comment.

~~[redacted]~~ Section 4 on Training Sequences discusses some of the properties of Lsearch and conditions under which they ~~[redacted]~~ are about the best way to solve problems.

10

SN While Lots of info can be put into the Pd that guides Lsrch, the GPD (which is supposed to contain all of this info) is not the same as this Pd.

I really have to clarify this in my mind. Perhaps leave it out of this version of the Report. (I like to think of GPD as "containing all of this info", but it seems that the GPD does not "guide Lsrch"; Part of GPD "guides Lsrch" — what does it do? The rest can be an Array ~~[redacted]~~ stochastic knowledge base.

Also "I. factor of 2" : ~~If~~ If one does ~~//~~ Lsrch the "factor of 2" is in the update/search ratio. I really didn't mention that in the report.

20

30

00: 831-40: Section 4: Training Sequences.

~~We will first discuss some properties of Lsearch — the principal algorithm that will process our training sequence.~~

In an earlier paper (Sol 89) we discussed design of the training sequences at some length. The goal was to find a function that would solve the paramount problem. In the early training of the system, the trainer would know a solution function for each of the problems presented. This function would be expressable as compositions of subfunctions. The subfunctions were composed of subfunctions etc, until we got to the primitive functions of the system. The training sequence would first give problems that were solvable by simple combinations of the primitives. The solutions were then usable for the next round of problems that had solutions moving toward the final goal problem. This succession of problem sequences continued until the final function was attained.

fixed 2/11 — could be much clearer! — Not clear!

While training sequences of this sort can solve problems, they can't solve problems of much complexity in any reasonable time, for reasons discussed in section 1.3 on "scaling". Learning subfunctions that build up to the final solution to a problem is an essential part of a training sequence, but it is not in itself adequate.

In section 1, we mentioned use of "context" to speedup problem solving. These contexts must be learned and we must design training sequences to teach them.

"Context" may be regarded as a kind of "catalyst" that makes certain combinations of functions more likely. There are probably many other kinds of "catalysts" used in human learning. We must find them and find ways to teach them to our system.

An important part of training sequence design certainly involves "factoring" of problems into parts that are useful in themselves. These parts must be further "factored" and so on. ~~The motivation for Object Oriented programming "reusable parts"~~ ~~is in this direction — the programming language "FORTH" — "reusable parts".~~

The "reusable parts" orientation of Object Oriented Programming was to follow.

NB → Finding "catalysts" is closely related to heuristic programming. We must find observations ("contexts") that make it more likely that we will choose the desired components of a function that solves the problem.

83.40

00 : 83.40 : [redacted] There is a somewhat different direction that training can take in

In section 2 on Inversion problems, the system had a number of PST's that had been successful in past problems and it had to decide [redacted] what probability to assign to them in working on a new problem.

The trainer can greatly augment this set of PST's by including techniques that the trainer has found useful. The system would then learn what kinds of problems to which these new PST's could be applied. It would [redacted] correspond to having a student memorize a set of PST's, and learn how to use them.

[redacted] We would like the student to understand the PST's [redacted] — how they were constructed and why they work, so the student could invent new PST's that were more appropriate to new problems.

Correspondingly we would ask the system [redacted] to regard the set of PST's as the "acceptable sentences" of a language for which it must find a [redacted] stochastic grammar — which it could use to extrapolate the set of PST's. This process would be much facilitated if the trainer first "factored" the PST's into common useful subfunctions.

A very experienced system might be able to do this without external help, but in the early development of these systems, external aid of this sort would be needed.

833.10: (section 5) ~~Probabilistic Methods~~ fixed Efficiency of the System

Subsection 1 ~~On~~ Updating

~~After the system has a~~ After the system solves a problem, taking time t, it must update the associated probability distribution. This involves looking for repeated ~~sub-~~ functions or subfunction in the solution and/or evaluating ~~recalculating~~ equations 818.34 and 818.03. ~~The latter are inductive~~ ~~Recursion is an abbreviation for~~

Both of these tasks are open-ended in the sense one can spend any amount of time on them. A good approximate solution is to spend as much time on updating as one spends on solving problems. This is within "A factor of two of Optimum" in the ~~sense~~ following sense: ~~Suppose that by using a different ratio of problems.~~

Suppose that we have a system that uses a different ratio of update to problem solving time and is better than our system. Then if our system has a clock that is twice as fast as that system and uses equal times for problems and updates, it will ~~be~~ at least as good as the other system, since at any time, it will have ~~spent~~ spent at least as many machine cycles as the other system, for both problem solving and update. ← results ← for

Subsection 2. ~~Ultimate limits of this system~~ Ultimate limits of the system

Normally, in the course of its operation, the system looks for better PST's. Since it uses a universal language to describe candidate PST's, there is really no limit on what PST it might consider. It might find a PST that is much better than L search, in which case, it would ~~still use the system~~ nominally use L search, but since that PST might close to one, so that all most all of its time would be spent applying this new PST to problems. It will have ~~replaced~~ effectively replaced L search by the new PST.

Subsection 3. Genetic Programming.

In section 1, on Inductive Inference, we mentioned the possible use of Genetic programming (GP) for updating. ~~The Genetic Programming~~ It is a particularly good kind of ~~problem~~ problem for GP, since we are always working ~~the "same" on the same problem~~. We can use the same population but ~~must change~~ ~~the "fitness" each~~ each new problem given to the system changes its "fitness function" somewhat.

Current GP systems are much improved when the fitness function as well as the mutation and crossover ~~algorithms~~ algorithms are allowed to adapt to the problem being solved and to the ~~state~~ nature of the extant population. The techniques of section 1 ~~can be~~ can be used to implement this "adaptation".

~~Mutation and Crossover are a very restricted set of techniques to obtain~~

DO: 834.40 In GP, New populations are obtained from old by Mutation and/or Crossover — which were originally attempts to simulate organic evolution. These operations can be usefully generalized. Mutation can be considered to be "induction from a sample of one", and Crossover, induction from a sample of two". Classical statistics is not enamored of small sample size. Bayesian statistics can deal with small samples, but the prediction depends much on the aprioni distribution. Appendix I tells how to obtain a reasonable aprioni distribution and how to update it with newer data. Sections 1 and 2 tell how to update it.

It is clear that we don't have to use samples of 1 or 2 any sample size up to the size of the current population can be used.

The induction methods of sections 1 and 2 can be regarded as a kind of GA using the entire population of PST's to create new PST's for global fitness function changes as each new problem is encountered, the entire population can be parameters

for May Report

Critical items

1) Fix intro so it doesn't have Lockheed. Look ✓

2) Put in eqn. nos.

3) Bibibiography.

4) Help: IDS/A, bus, Prof.

eqns: §1 eqns 1 & 2

3, 4, 5

§2 6, 7 (refers to 6, 7)

APXA (8) refers to (8)

B (9) 2 more pgs) 6 M.

§3 refers to both 6 & 7 only 2 refers to §3 but from both

§17. → (6)
§15.03 → (7)

Sec nos. when:
4 new
5 TLOZ

School doc
2
3

2 updating / should be Scaling / see subscripts at section 1

00: 835 [SN] In dealing w. correlns betw. cands in Lsrch: One way was to make x equiv. classes of cands. One member of ay equiv. class would be chosen & given all Lsrch cc of all other cands in t. equiv. class. This is a way of dealing w. t. fact that Pcost is often >> 2^{-kcost}. But this equiv. class idea puts even more wt. on t. single cand.
This can be way to really ↑ effective RC's a lot!

[SN] Sumacs can be a vg. approach to Lrng. w. t. AZ441 logo, I have step in this direction by including Context defns, — another strong step.
In general, Historically successive Sumacs can all be universal, but t. "constant factor" by which each one deviates from t. needed soln. to a problem can ↑ exponentially w. historical time. This "Constant factor" hypo in "Kolmogorov studies" really makes most of t. work irrelevant to Lrng. & A.I.

Notes on proofreading IDSIA Report:

1) The pnos 440-443 are for LIV 1393 not 1997!
p2 abstract: transfer lrng: where is this discussed? last ¶ of §1 (I omitted this! in original printout)
pp10, 10 — on Context: should be lined to discussn in Appendix 3 (Scaling)

---

p6 eq(3) looks bad! (eq3-3): $P \quad R^u \to R^i$

P8 update method 5b: "Merging P_k^i's. We also have to "merge" t. assoc. R's.

Now R = "OR" of old. However I did already say that we "merge" P_k^u funcs and assoc R funcs. So lot it go.

P7 on updating: "Modify R^i so New $R^{i}(Q_i) = R^i(Q)$ But: ..." How this is to be done is not stated. Lsrch was mentioned earlier but not explained. Then p. 8 we modify R funcs & P funcs, but don't say how we do it. On p9, Lsrch is introduced, but reader doesn't know it's Lsrch that's being explained. Is p9 t. first mention of Lsrch?

Tentatively: (p9) After ~~the~~ first ½ line!
To start off we have only 1 ~~eq~~ Ag pen. We will ~~use a primitive form of~~ show how to Lsearch to find a function ~~parts~~

How to deal w. this? One way; (not unnatural): Do t. simple case of one R & discrete predn first

---

~~We~~ We will first describe the ~~operation of the system for a very constrained type~~ early training of the system, ~~then deal with more complex cases;~~ using simplified problems, then we will ~~describe~~ tell how to ~~start~~ update for more general kinds of problems ~~types~~. In this early training, we only allow one R...

P8 near end. copy to end of P9 Then Introduce more general case: Start at top of P7

Somewhat rewording.

836.401 Intro after simple case: ~~When we are given a new QuesAns pair in a~~
~~For the more complex problems~~ for more complex problems with several R^i and many P_i^k, approach

we ask, how is the system modified?
Start top of P7, elim. first sentence.

¶ More Generally Suppose we have a more complex system with several R^i and several P_i^k.
When we are given a new.

The foregoing techniques must be considerably modified for systems with having several R^i and associated P_i^k. when we are given a new.

P17: Appx A: Introduce "Argument list" sooner.

we start up .... function definition. We will call this the "Argument list". In the course of our calculations we will occasionally add to and delete items from the Argument list. Some examples of primitive functions could be .x, sin,

T. Argt. about "factor of 2" can be sharpened if we use 11 or random (such as do regular i.e. update "simultaneously")

Bayesian

Bays Bayes

CLRC, Royal Holloway, U of London.

Acknowl: I would like to Thank IDSIA for the very productive month and their facilities that inspired this report, particularly Marcus Hutter and Juergen Schmidhuber for comments and discussion.

N25.02
ID

S38

Wksp: Nicolo Cosa-Bianchi: (Univ Milano, Italy)
Universal Reason, Game Theory, Path. recogn.

Ilya Neumann (U. Calif Santa Barb).
Universal Lrng.: A View of a Bayesian.

07 Remarks on OOPS (Juergen's paper)

1) Mention BAD title.
2) P5 (Osborn) "new invention": This is identical to Lenat!

10 He uses Ostrich w. a P.D. that changes adaptively: Is this [redacted] a distraction?

2) P8 on "frozen memory": This is an attempt to get around the fact that in an older system of his, any buggy program could be destroyed by a random "write" instruction. It seems like a very A.I. soln. On the other hand, it would seem to be a good idea to store (permanently) all good solns to problems

3) P15: 3.3.4. So perhaps he is saying that OOPS works w. proper tsq (only)

4) It is not clear that "Optimal" suppose to apply to ordinary problem solver.
One interpretation is that it takes a bunch of problems and decides what order in which to solve them (This was my original interpretation)

5) P4, because universal search can take much time → no reason to avoid universality

6) I'm not sure, but I suspect that what J means by "Backtracking" is: sequential search of a tree, depth first. We use the usual trick to do this (see "ocs").
It works w. any radix — even changing radix.

ABCDEFGHIJKLMNOPQ
AB

7) Could I use his "prefix string" idea to implement Success?

8) P32 re. Probly changes ("Bias shifting") instructions: It seems that this is not a very easy to do this — but I must look into it. It is into "TI Credit assignment problem" (probly a "wrong problem")

9) P20 He uses 1.5 GHz pc: ~1000 clocks per "time step" seems very inefficient!

10) P20 [redacted] ff discussion of exploratory results may be useful.

11) I want to find out how he chooses bias (Also note 21.3 ff re "Bootstrapping" pc values; Also note PC) P's 2,3,4

12) First find out how it works w. constant bias, then → P32 perhaps

---

How it may work: [redacted] ("w.o. changes of Bias") : M machine can request a token (alphabet input) or work on a problem set. It chooses a stop state: Always goes to stop state when it thinks solved all problems — but then announces this fact. It may stop sooner — which is a "failure". I think initially, all tokens have pc. Also: Saves all solns/pgms to problems in "Frozen" Memory

To change pc's: new tokens can be defined as substrings of [redacted] existing token set. Also other ways to change pc's of tokens.

It may be just normal execution of trial functions, & revision of pc's is done as a unified task.

— Possibly Good: but elizn. can also be very useful. — see 2839.0(±.02)(6).

So "between tokens" Machine can ① access frozen memory (access of)
② Invent new tokens (≡ strings of primordial tokens)
③ Reassign pc's to tokens
④ Work problem "Compute" This could be on problem also on how to change pc's.
⑤ Output some of problem.

Since one input is a token & a token can be any string, a lot of computing can be done on "one token"

→ Is the only requirement to system Gets is its correct Solutions of problems & insertion of these problems in Frozen Memory? — Also if P^n solves all probs up to n, P^n is used as a trial prefix for problem #n+1. I do use pc's of solns as additional f.b. info.

---

10 $\Sigma\ 1 \cdot p_1 + 1 \cdot p_1 p_2 \prod_i (p_i+1)$ = converges! its min is 2, its max is e.

for min, $p_1 = 1$; for max $p_i = \frac{1}{n}$; lim, $(1+\frac{1}{n})^n \to e$

11 $1+p_i < e^{p_i}$ so $\prod 1+p_i < \prod_i e^{p_i} = e$. If the largest p_i is δ then the product is between $(1+\delta)^{\frac{1}{\delta}}$ and e

---

It would seem possible to use .09—.11 for $T \lessapprox 3T$ but actually, the time to do a "round" is always ≤ e and if the factor is over

14 < 1, Then the recursion is over. The < occurs whenever we have the termination of a trial before time out — e.g. it does something illegal or

19 it is clear earlier that the result can't be a solution to the problem.

20

What was Minsky about when he calculated $B \equiv \frac{1}{\text{time of exhaustive srch}}$? P14.25 $\approx \frac{pc\ soln}{cc\ soln}$.

Did he want to modify all pc's so that he got B for pc of soln? —

i.e. modify pc's by something like numerator → num + k, denom of all → denom + k*(no. of tokens).

This revision of the pc's would reflect the fraction of trials that aborted before "Time out" as in .14—19 i.e. [B v.s. $\frac{pc\ soln}{cc\ soln}$] ratio of these 2 would give "index of non-time-out aborts".

Moda discussns ① Title ② typos on ~ page 0. ③ confusion about < srch data, what cjs is etc. ④ $\prod(1+p_i)$

20 5) Series discussion: ⓐ Just what was his pc modifn. scheme? (I may know)

31 ⓑ would R's scheme work?

ⓒ was his conclusion that "working the 1, n, 2n problem helped solve tower of Hanoi" logic?

ⓓ I say how Minsky's search times are really grossly large.

.00 Re: 8.39.30-34 : The idea of using info in old solns to help solve new probs & help calculate pc's of next tokens.

.01 In my report, the system does have (about) all old solns "in the right list" — so it does do partly what .00-.01 does — but does it use 'old info' to completely recalculate pc's of next tokens? Is this a good idea? Normally, I think J uses "Leaps rule" for pc's of tokens: The "boost" instruction, is rather Grossberg, as a method of how to recompute pc's!

I think his system is more general than "discovery of contexts" for new tokens to control pc's of new "tokens" — { The I did have this idea of "Generalized context" categorization of patterns. — so maybe his system isn't any more general than mine! }.

10 New Commands: I think his system is designed to solve INV problems only.

The w. my "contexts", I had a more limited idea as to how context could influence pc's of next tokens. The result would be that the system would learn Phrase Rules, but would not be very general. J's system is very general, but takes much longer to learn.

Also, it may be hard to train because the trainer has no good idea (& s. model) of how to train it.

As is, J's system can try any & invent all rules it likes for assigning pc's to tokens.

⇒ we know a priori that only such rules will not work. We have a way (context) to generate rules that are very likely to work. (Unclear as to whether there are rules outside this class that work).

20 Another Possl difference: J. uses strings to represent pgms, syntax; I use functions/trees.

Re: Optimization problems: Prob'y J's system could be easily adapted to do it as my section 1 does it: For the more complex "marked" modifying the quickly pd ... I don't know if it would be possl.

SN In my original work on the QA archive, I "decided" that it could n't really learn to work on optimization problems: That if I gave it examples, & its "best guesses" as to how to generate those examples would be to do or "approximate" the pgm that created those examples.

However, Not so! If TM had "proper vocabulary", deriving the corpus would be relatively easy, & deriving the pgms that created the corpus, rather diff/t. (low pc, by cc).

30 So it would depend on TM's state of development, as to whether it were to discover "the ideas of optimization!" Whether it would discover it "completely" is unclear.

An analogous idea: Discovering "Laws of Algebra" would be easier if TM took long time to do multiplications, division — maybe add & subtract also: So the laws of Alg. could save TM a lot of time. However, Arithmetic, the "laws of alg" are harder to discover.

90.8
90.8
998.8

III

: His General "pc of token" modifyer seems like a good gener. of "Generalized Context"
to pc. of next token is a function of present "prefix" and probdescn.
The corpus for this induction is all (problem descn, prefix ~~prefix~~ to ken) triplets.
Q A may be pc known as part of corpus?)

Could this induction create new tokens ?
ABCDefghij Klmnopq

(1) The his result about T. "tower of Hanoi" being more solvable after "1-2" was solved!
I'm a bit incredulous of this & I want to think about it more.

(2) I feel his execution times ~ ($10^9 - 10^{14}$ ops) is excessive! for a prolong tsq,
I would expect ~ 100 ; maybe 10000 ~~must~~ at most.
I feel that his **TSQ** is not V.g., in the sense that he isn't really controlling things

(3) "That it should take 1500 clocks to execute an instruction is rather wasteful — but at present time,
you are not particularly concerned with speed — you just want to see if and how it works.

Probably OOPS could simulate my method of assigning pc's to tokens by just pgmg it
& using its pgm as a "Token" — w. perhaps much initial "Boosting".

Out. "Success" of his pgms: the "boost" inst seems critical to deciding which tokens will
get reasonable pc's. Look into this!

QASM

The ~~description~~ ways the system deals with induction has several phases of
sophistication. In the first phase "The early training" (p. 8 of my version)
Induction is pretty much an inversion problem! ~~we want to find a function F' such that~~
we want to satisfy eqn (5) with as large a_0^j as possible. Since we do trials
of F^j in approximate order of a_0^j, ~~therefore~~ the first F^j we find that satisfies
eq 5 is usually acceptable. The trainer may want to "oversearch" for a
F^j with larger a_0^j. In this phase, we use the limit doubling technique of Lsearch —
we start out with small ~ tau using a_0^j ~ Tau as time limit, then
double the time limit in next round, etc.

Did I send this to Marcus?

In the next phase, we do induction problems involving probabilities
induced on the A_i. We want to get a maximum value for eq 1 (p4).
In this case, using a tau = Tmax as in section #4 — which
is what you (Marcus) suggested.

(SW) And randomness besides. my & J's approach — like has it yet tried to do induction problems, — only inv problems.

Note that .20 - .30 is not really "Lsearch" — which was designed to solve inversion problems only. Section 1 induction! find O^j's ~~s~~ eq(1) is as large as possible.
look for O^j's in a_0^j order.

Hvr for "induction," we might regard {Q_i as a problem descn, & A_i as an acceptable answer} — as an INV problem. We could use same formalism for induction as well —

D.1.02
ID

~~[illegible] doesn't take to work that way.~~

Not so standard! t. criterion of soln prob'ly has to be Maxzn of 4.4.18: which is more like a standard OZ problem !

00: : xforming it into a (standard) Inv. problem. In this form, the Guiding pt. could be modified by t. "2 step procedure". The difrnce betw. this & t. standard use of Lsrch of optz, is in t. idea of "t. solution"! Here t. "soln" is t. function that maximizes the value of 4.4.18. In standard OZ, t. soln is t. output of t. optimizing function. } NO! t. output is t. string argt. X of G(x)

In t. present case, t. search is very "Blind"! t. system doesn't know what A are, until it makes a trial. It is "unblind" in that it "sort of" knows $A_1 \cdots A_n$ when it's looking

.08 for "A_{n+1} (since it is? Soln is not "Blind" wrt $A_1 \cdots A_{n+1}$)

841.36 → 842.08 Looks at t. inductive method of §1 of report & asks how

10 it differs from Lsrch. — how it might be regarded as a kind of Lsrch.

I may have to explain just how t. method of §1 differs from that of §3.

Maybe make transparency?

On difrnces betw. OOPS & ~QATM;

1) The double search scheme for general solns, v.s. search for extensions of t. prefix that worked w. probs & Thm.

I would just do search ~~[illegible]~~ (solns to problems, Thm in) in "Argument list" (→ Frozen Memory)

Since I probably wouldn't have a "Boost" inst., I probably would have to have a relevant

20 problems with some common solns. to "Build up" (gradually) what J. would do w. "Boosting".

It may be that this slow build up is necy, & that J.'s TSQ is exceedingly A.H.

— (That "Boosting" worked O.K.) ← Amazing!

2) I have been [illegible] considering a TSQ w. much smaller cjs's than J has used, usually ~ ~~[illegible]~~ 10^2 to 10^5. ~~After TM has learned a lot of Algebra, say,~~

I expect to have a v.g. understanding of what to expect from TM in t. way of solns.

That eventually I will get to pt. where I don't understand how TM is solving problems & I will have to make "approximate models" to guess at good TSQ's —

30 to guess at why TM can't solve a particular problem ... etc. — t. same problems as a human teacher. This particular approach ("small" cjs) makes experimental TSQ's easy to implement.

The TSQ J uses w. runs of .3 days to 3 days — one would not "casually" try a new TSQ if it costs a few days. (Even J's pgm could be certainly speeded up by a factor of at least 30 ~~[illegible]~~ or perhaps 100 with careful assembly pgms.

As is he has about 1500 machine cycles per "instruction time" ~~[illegible]~~ This is certainly reducible usually it requires really careful pgms. 7.2 hrs → .72 hrs = 43 min; .3 days = ~~[illegible]~~ 1 hr.

~~3 days~~ [illegible] 3 days → 7.2 hrs. Also ÷ 10 in 5 yrs.

But ~~[illegible]~~ this misses t. point! That you want to be able to "track" t. machine's mind for as long as possl.! When you can't — this should be

next page is
NIPS 1.00

.00 because a machine is solving problems that I can't solve. (Obviously, a machine can solve a million problems that I can't solve, but I'm not considering that kind of problem. There are lots of very difficult algebra problems that I can't solve in a day.)

Major Q for it: where does he go next? what problems? — what would be good to & how could he continue? Experiments are expensive.

possibly ↓ cj's by ↓ no. of users.

(SN) An efficient way to do OOPS in ||! Have N machines: Each one does every N'th branching point for tree. ?? Not so clear.?! Hmm, first machine can put "markers" at branchs where another machine has been assigned. This

.12 Tree search is easily done using "stack".

.20

30