P:15.02

ID NIPS

Previous page is ID 843,846

Go ID 843,40 Inverse of Conv. Prim:

01 1) Any distribution always that gives $>\phi$ pc for any string, is universal. → 2.12

2) It seems easy for Distr. on over finite strings, but way harder for Infinite strings ("continuous" univ. D.R).

3) If conv. Theorem is not true, there must be at least one μ_0 for which * P'S78T3 is false! (ie. $b=+\infty$, $P(x_n|x_0) \to 0$) ↓

4) in S78T3 we can have $b \doteq b(n)^{\frac{n\pi}{n}}$ bit in seq. so if conv. Prim is false, ~~it~~ only need be false for $n \geq n_0$: ~~P~~ for $n < n_0$, ↑ Prim can be true but error²/bit will be only bounded by $\frac{b(n-1)}{1}$ 6

5) if 1) is true for infinite strings, then for any positive γ, we can do $P \to \cdot P^{\gamma}$. (normalize constant). for large γ this means pc ratio of ~~0 to 1 or~~ $P(0)/P(1)$ will $\to +\infty$ or 0. So the system becomes deterministic, rather than stochastic.

20

Hm, ~~int.~~ conv. Thrm. 2Σ err² $< \ln p_0 \to \gamma \ln p_0$ so error is worse: we should do $\gamma < 1$ if $\gamma \cdot$ distribution converges.

22 6) If ~~M = M0 Prim~~ $b = \infty$ ~~P~~ $P_{\text{approx of } p}$ must be zero for some n_0,

which gives KL distance $= \infty$ for strings of length $> n_0$, but that P could be ~~fine~~ for $n_0 > n_0$! No! P will ϕ for all $n > n_0$: — P assigns probes to strings of length n_0. If $P = 0$ for string s then $P = 0$ for sa,

26 where a is any string, finite or infinite. ← This doesn't seem right! i.e. it should be possible for a Prim to give ϕ pc to something that occurs, then give pc's $\neq 0$ to subsequent bits in a usable way. → 2.00 →

32 7) ≈ EN in that proof that eventually there will be an error that is below normed ↓ unnormed pc: My "sort of counter proof" was for $\sum_i p_i(v)^2$ bounded. But this would also for 2Σ KL distance $< \ln p_0$. So ratios (context) have to converge.

— But look at this carefully — seq just what converges! It may not. Not My proof of $(pc)^2$ converging would only for ↑ seq, not ↑ KL distance.

D.15.02

NIPS

(8) until 1.26 − .32 is resolved, it's not clear that ∀. violation of ∀. conv. Thm" need be so bad! The possible idea is that 1.22 s/o 1.30 IOU, One troublesome issue is that PDS is v.s. ε true, which scores of pc = 0 accurate → .031 → .160

(9) Another Q: if a P.d. is violated, is it nearly true that ∀. conv. Thm violated? As Kolter suggested, if ∀. PDS is computable, then can make a corpus or set of corpi, ⇒ ∀. pc ≤ ½ for every bit of ∀. sequence. ≡ random or worse than random. Then if its computable, it can't be universal, ∴ ∀. conv. Theorem must be violated. ∴ which puts us back to 1.26 − .32; 2.00

(10) woops! ∀. uniform a priori assigns pc > 0 to all strings yet it's not universal.. b(n) ~ n: b(n) ~ 2 log n is tolerable.

(11) We could have a universal d.f. but add in just a finite no. of φ's or a "finitely computable set" of φ's. — However, ∀. D.f. is such that it is not "computable",

Busy D.f. (∀. uniform D.f.) yet it has no "zeros" —

(12) We can have a computable d.f. Eg b(n) > 2 log n, but does in << n

(13) Another approach: If we could list off Cpdt (computable fcty measures), then we could beg them in order: but they are not recursively enumerable — So whichever way we try to list them, we will miss out on some cpdt cpms (& of could have found ∀. constant regions).

(14) In paper on this: how to compose ALP approx w. other approx methods) I think I sort of did it in ∀. Bayes paper; it is ok, but not clearly enuf. Trivial Problem seems to be "Model uncertainty". This whole thing might be a paper on "Two kinds of Uncertainty" (Model, Simple interval uncertainty

→ 21.00

D·11·02

NIPS

: What to add to Report:

1) Real stuff I wrote comparing to Latex.

2) Discussion of OOPS

3) Tell how secret's Appl audit A are "stand alone" — secs on Int. & Oz. tell how to update much better using Oprah duction — & ultimately how sec't should update itself.

4) Perhaps pseudo prog on how >a confounder GCPD; Advanced updating work.

5) Begin work on my "Acquired response" — list of simple variations (improvements) & Refs in Notes.

6) Put talk itself on Web, but add details / garbage slides — & make slides (better, when needed).

7) Send copy of report + OOPS to

- Ivan
- Kurt?
- 2 Grad students at MIT
- Henry Linkenman
- Schaffridge.
- Jass Marton
- Louis Marton.
- Prof Lesly Pack Kaelbling & she has probably seen both.
- Will Gersh. !
- Gary Wolf.
- Eric Winfree
- Phil Apley
- Dick Herter
- Guy in Berlin (Cox. grad stud of Turing..
- Guy in London UK. (Cole w. video Tomasz) called him
- GAMEMAKER MAN
- Steve Wolfram
- Minsky
- Carlos
- Murray
- Alex
- Dowe
- Wallace.
- Guy at Santa Fe Inst.

8) Get Latex a/o Pdf version of OOPS to send out w. my "Report" — Look at J's website.

Criticisms of OOPS:

1) S.22 ! He doesn't have trng in to put text computer, now PC's for Tohoms.

2) He has large CJS: probably will soon find it very difficult to write TSCJ's (4. "Autistic student" problem

3) Scaling problem w. "Boosting" ~~useful~~ previously useful items.

4) S.32

Compare to ~~the~~ on ~~after work in this~~ area. Related work. not read

(I have ~~read~~ related Work on Info. Lang & is may be a problem.

D·16·02

ED
NIPS

CLUSTERING

This was in response to a proof (maybe at NIPS 2002) that clustering was impossible to do in an exactly rigorous way.

Gen'l clustering: Give a data set; each pt. of set has several parameters. $\vec{x}$ is N dim vector $\vec{r}_0$

A "cluster" is a way to a) pick a pt in R. space, b) find a scalar function $f(\vec{r}_0, \vec{r}_i)$ ($\vec{r}_i$ is data pt. $\vec{r}_0$ is "center of j^{th} cluster")

$\Rightarrow F(\vec{r}_0, \vec{r}_i) =$ max overall second arguments. Perhaps $\sum_j F(\vec{r}_j, \vec{r}_i) = 1$ & $F(\cdot) \geq 0$

so $F(\vec{r}_j, \vec{r}_i)$ is a probability that $\vec{r}_i$ is in cluster j. (kind)

Anyway: the idea is that this is a way to describ. a set of points.

How a clus goes: ~~If every data pt is assoc. w. any cluster j all data~~ ~~One way: each pt.~~ If there are j clusters & n pts., each pt. can be describ j ways. We want F's such that the total info in the clus of pts. functions F plus the j $\vec{r}_0$'s. plus the info in the data clus, is min.

The pc of each data pt. is some of its pc's wrt. each of the j clusters:

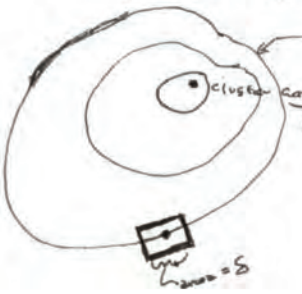
We want product of pc of i^{th} pts times (pc of j^{th} center) × pc of $F(\cdot,\cdot)$. to be max.

We can compare it w any other clustering of those pts.

Is entropy a way to describ. the set. ? — Yes! .22 — .40 shows how.

.21 is unclear! The clustering $\sum p_i = 1$, but how does this enable us to describ. each of the pts. wrt. their center of the cluster?

Say we have a 2-dim space ($n=2$)



What we want is the prob of being in a data pt. so that it is within an area, A, We want $p_c \cdot A$ for small A. [($p_c \cdot A$) will be $\propto A$ for small A.]

So what we really want is that $\int_{\Omega} F(\vec{r}_0, \vec{r})\, dV = 1$. ($V$ is a volume element in n dim space)

We want the product of prob densities of the data pts to be max, when multiplied by pc of the $F(\vec{r}_0, \cdot)$ terms.

The actual (pcop) clus of the data set to any accuracy can be obtained by multiplying above pc by δ^n (for n data pts & we want to know each pt. to within area of δ). — but the δ^n factor is indep of the clustering method! So the choice of clustering method does not depend on δ.

SN Perhaps Got pgm ≈ OOPS ?, but with many of my cores.
— Perhaps add a stack or a Fault dictionary. — So my SZ ~~system~~ lang → Form.
Theory Teaching ~~ANL~~ ANL, algebra, ~~etc~~ — in ≈ order needed for doing human Algebra. text. If CJS have a problem is excessive!
find out why. → (Note (8:25 2.0 8.09)) Try doing it at a hy level.

Hvr, I Denth scaling will be a problem. I will have to ~~text~~ put heuristics in text ↑ pc's of soln. ~~Context~~ Gen'd "context" will help, but I certainly need. Much More!

SN(Re: J's method of modifying probys of tokens: This can be done in a Universal way: (J. talks as if what he has was universal, but his rests for pc Modifs are minimal ... If he expanded Prim to Universality ↑ probs of ↑ system's discoveries may be very useful & enormously!)

Hvr, if ↑ ~~natural~~ reduced lang is already univ. Can considerations of this sort (which includes ↑ Gen'd idea of "Context") add anything useful to ↑ system? Well Context seems to help alot. T "Universality" ● argt never includes cc. — So perhaps J's idea of an aux. (univ.) system to modify pc's of "token's" may be a good way to ↓ cc. of solns.

I think 12.35-36 may solve this in Best possible way!

Actually, Gen'd. Context" may be about as good as one can get — & have I mean this in most General sense, of Given an arbitrary prefix (≡ "context"), what is ↑ pd over possible / continuations?

This is ↑ General problem of sequential Extrapolation. We can use various simplified concepts of ntup data. or, that general method of string extrapolm, using all ~~of the~~ previous solutions.

[It would be a good idea to try ~~doing~~ this to see if it works at all] i.e. this method of extrapolm. i.e. what computer science concept is this?

Also try doing my most recent version of ≥ 14.1. — (w. recent corrections I have) was at least one simple error ---- I may have accidentally fixed it to work.

YES!

AQ about .25 - .3 : Essentially, we are extrapolating in "successful" pgms! I.e. we have a set of strings that have been solns to problems! We obtain a pd. on new strings by ↑ method of Sqg (2 kinds of ~~AI~~ ...)

D-18-02

NIPS ~~ID~~

12.35-.36 seems to be ? Virt. Genl. Soln. to this problem!

This forces

20 the system has no idea of what's being done — much less a concept

"Optimality" — If is, hvr, a ruff & dirty way to work on this problem. It extrapolates

22 the "set of solns", & no idea which solns were v.g. or very marginal — or what problem was related to present problem

J. would (perhaps) mention it in a different way! He would try various methods (.10)

(PSMs) for assigning/modifying pc's of new techniques. Models? He

would then reform the ones that worked best. This is a simple

but very slow ■ method of proc.

0:02 ■ A possl. way to improve this: T. data set being used in previous

driven was ~~one~~ a set of soln strings, $[F_i]_{i=1}^{n}$: we want to extrapolate this

.12 set. A slightly wiser way: Given $[\underset{\text{desc}_i}{\text{Prob}_i}, \underset{\text{of } i}{\text{Soln}_i}]_{i=1}^{n}$,

.13 Given prob_i — to get good pd. over soln_i's. This is (ruff) operator induction.

It is better than my old concept of "Gauge Context" — The gauged

context ~~could~~ could include problem desc. — & so Gauged context could

be a "ruff & dirty" approx. to .12 - .13.

← This may not be a nice way of picturing it!

20 **N.B.** There is in the system not 5.25 ff & 6.10 - .12 : a kind of "BADness"

If is, to some extent, a "self confirming hypoth" — The regularities observed

in the past, are used for future trials & found to be "reenforced": This is not

the best way to do induction, but it is probly [much?] faster than the correct way.

24 T. correct way (ALP) would consider all possl combins, then do something → 7.00

If is interesting that J did not give all solns equal wt. for predicting

next token & ~~Manually~~ gave them wts. of ~~1 for solns 1 thru n-1~~

30 $\frac{1}{2} \cdot \frac{1}{n-1}$ for solns 1 thru n-1 & wt. of $\frac{1}{2}$ for soln n.

I was thinking of using = wts for all solns. — so n-1 times as much wt. for soln n, as for each of the others.

His reasoning was that soln n did do all of the problems. Also since he only

had to ~~change~~ "add on to" that soln, ~~he only had to~~ & not check each of problems

That took much less time for trials. [I'm not quite sure how he did the "not the nth soln contains" trials. — Check this.] Exactly what was done is critical in

deciding whether the system really did anything interesting in solving (low) ergs

after $D^{4,4}$ recursion.

.00 ; 6.24: Like compare d.f. of Gorc's of continuations. At first Glance, this seems diffl't,
.01 but maybe not so diffl't! We have various (probdesc$_i$, complatt soln.desc$_j$, Gorc$_{ij}$) triplets.
The Gorcs can be diff't. instances for diff't problems, & still be part of
the same (data) corpus. Anyway, we use the data of .01 to indicate the Gorc distribution
for all possl. token continuations of the present prefix, wrt the present problem.
From this Gorc D.F. we can get the pc. of any particular token
.10 continn. giving max Gorc.
.00 It is a bit "fuzzy"; I want to write it out in more detail. — It
looks (in present form) easily forgettable"
.04 Corpus is all known triplets of .01. Gorcs include time for soln.s
of Env. problems (~~For we have to make Gorcs neg. time~~ No!), as well as any other
kinds of OZ problem Gorcs.
Looking at it from a very slightly diff't viewpt: We have:
(probdesc$_i$, postfix$_j$ (this for continuation token TK_k) we want D.F. of ~~Gorc~~ wherever
.20 Gorc type is specified by probdesc$_i$ → (as corpus for this prediction, we
use data of .01). → 9.00

SN I'd like to be able to use "negative" info — Usually this
takes form of knowledge of "No soln." for $\leq$ time T_j" for a given problem, soln. attempt.
If it is a INV problem. — but for Optzn problems we always (?)
have a "Gorc thus far" for any optzn technique. This can be $-\infty$, if
no "trials" have been done yet.

SN For the primitive instr. set (like for FORTH) would best to be a brand
.30 in a RISC cpu — but I think there aren't any any more! — Maybe
Caruso? — Pick a fast stored inst set to start — Then the system
cuts down on the set it uses — picking an optimumly fast subset &
giving it most of the pc wts. — we could retain other insts at low
pc's — (but possibly very hy pc's in certain "environments" (contexts/postfixes).
— Actually, this is what J. did, both "ends" (i.e. selection of hy
pc insts ~~under certain~~ for certain (contexts) — but he didn't use cpu opcodes
directly.
.37 Perhaps use inst. set of 8 bit cpu like 6502!

→ see 12.35-.36

→ 9.00

T. induction problem, w. pc for best "calcn. of .00 - .10 will be
the Main (almost entirely) Math problem. Such system will work on. Its the same as
the problem of ordering BCPD$_1$ ⇒ GCPD$_2$. Actually it is "T. same" in all ways! (8.10)
space

D.19.02 "704 TM" 8

ID

NIPS

[redacted] m the early History of project.

10 : 7.37 : I may want/need, a hardware clock. to time progs. It may be easy to have a counter running at cpu clock rate, or $\frac{1}{8}$ clock rate. — just pickup clock signal somewhere & try to find fast counting. ckt. as "off t. shelf" item.

At first, System will be very slow, using many op codes per "instruction", & I can use relatively slow counter.

op The idea of t-ing it a ≈ 704 TM w. idea of 5.00-.07. (.21)

0 : 7.40 ! — Its t. only meta problem, & it includes all Lsearch!

NB The present "Advancing Rush" of optimism, assumes t. pc's of cands are uncorrelated. T. result is a very (limited/stupid) TM, I guess!

Try to find this. → I had that very neat fast H.W. idea for TM : Look it up now! It was a large very parallel machine that moved addresses around. It may have also created various H.W. functions that were physically duplicated if they were used much — Then these "devoted" for diffnt functions when other functs were more sorely needed.

1 (.06) → I think t. essential idea is that t. 2 level GPS update can automatically include any recalculation of pc's of tokens that OOPS (in theory) does.

olde Sched. { lv Vanc 2.16 — arr 8.15 | Chi dep 8.95 } 288 8.85 Expected arrival in Chi

25 → T. Big trouble w. 5.00-.07 is discovery of needed heurs. T. system may be able to work interesting/diffct probs, but it will not approach human skill, (except by substituting (Hardware) Speed for heuristic insight)

My watch is ≈ 3 minutes

Perhaps study heurs: list them, try to find common Concs — try to make "Grammar" for extrapoln. : J. Pearl has book on Heurs (that I may have: — I'm not sure how also much of rest of the community ↔ Relevant, Srvr: Try

PM PM 9.28 ; 3.15 Eastn time 2.128 2.195

30 This could be a main path (say of J.), because it will be able to win on various "Benchmarks" U.S. other Mach Lrng. Systems — Enyc of A.I. Both t/one I have and t. new one

In general, .31 will probably be most impt. path. The "Heurs" are a "premise" but nothing (perhaps) in way of provable superiority. Usually normal pgmmg skill & buying of Big Computers will be preferred, over reliance on development & understanding of Heurs. Perhaps they will try insertng heurs, whenever possible.

(85.05-.11)

D.19.02

NIPS

8:16 Aur.

Flite UA 516 : :
Leave 8:45 : Board 8:15

How to get there?
Gate.??

8½ pounds
and hike
8:15

8

Lv 21:45 9288 9:15

Lv 21:35 9:058

00: : So: perhaps impt. "Political" problems!

Re: Juegen: Main guess is that he doesn't understand Operator induction, much less my formal Soln. (& more toward a practical soln. of it).

IBM MAINTOP W. Dig Hitachi?

05 Re: T. Boost instruction & its Application to enable emphasis of tokens used by previously Successful gens: The T. Sq. was certainly not long enough for discovering the "Heuristic": It would have to have several/many cases of its expected use before its "definition" were done. The case in OOPS is not even One Shot Learning! —

10

11 It is simply inserted by human, not "learned".

As is, there is very little (= only Boost instruction) for usefully Modifying pc's of Tokens. My Guess is that J. might want to add more instructions — but first they would have to have more or less A.I. implement heuristics that he feels are useful. As his system is now — There seems to be no way for it to Usefully learn new Heuristics or any other useful Algms for modifying pc's of "Tokens":

20

22 A Major Criticism of OOPS is that it really has no provisions for its learning how to assign good pc's to Tokens. He says it has all these capabilities of modifying pc's of its Tokens — but provides no good way to make this come about — other than by human-programming in Heuristics, invoking new Tokens.

SN In Learning Text Book Algebra (5.00-.07), it may be easy for

30 If to know/learn Logical "reasoning".

To "identify" Heuristics: Try to see why certain "steps" or whatever were "obvious" to me. — Also, Look at Newell-Simon on "Human Problem Solving".

32 : : Compare Theoretical vs. actual cc of solns to Towers of Hanoi (as shown in OOPS paper): See if it really found anything — in its learning recursion" whether pc of trailer of file "Proof" cost as much research as it. and. Found.

D.18.02
~~ED~~
NIPS

00 : 7:21 : I want to work out details of [7.00 – 7.21 / 7:00 – 7:21] : see if it's a more or less complete Soln. of TM (w.r.t. TSQ, of course!). → .12 – .14 seems correct!

02 Then look at various "early" approxns that move toward that Goal. → .07

Also, look into possl. use of Big Computers to reduce diffty of TSQ design. (Also enables larger CJS — which usually means! more "creativity")

07 .02 → Actually, the original 2 part GPD was a very good initial approxn. to 7.00 : 20

The only real difrnce is that somehow, the PST is actually created

0 by GPD$_{1,2}$!?! Just how this is to be done, is quite unclear!

Perhaps a kind of modifn. of the old GPD$_{1,2}$, [PST] sort idea?

12 It would seem to be necy to consider an enormous no. of PST's, & do the standard older GPD$_{1,2}$ on them. From the DF on PST's, the d.f. on "next token" can be determined. While this does seem theoretically correct, one must

14 clearly also find reasonable approxns — otherwise the CC is sounds

19 way too high. The Goal, here, is clear. → spec. 10.00

20 .12 ff looks reasonable, is not far from what I was doing before (GPD$_{1,2}$), but considers All possl. PST's: The obtaining of PC's for tokens (remember a "token" can be a large Macro) is one possibly useful elem of the problem.

Assoc w. infinite no. of PST's being Optimized is what is probly a Big SOY problem. — [see my second unwillingness on (SOY Big)! — The present (.12 ff) problem, rather than an (aggravation equivalent) of the problem, may point toward a soln !!] (Ideally! → see 10.00

30 SN Can Cross Validation (Colco Faithfull) get good estimate of PC error in ALP?? … Cross Validation may assume Stationary Data Stream

Also, it is mainly for perhaps average error, not error of a particular prediction!

SN Perhaps main item in Talk is Report! That Lsrch is Adaptive Lsrch can be used in sub systems (stand alone systems) of § 1 & Appendix A: Adaptive Lsrch is necessary — Lsrch w.o. it is excessively diffl.

SN In normz v.s. Semi measure version of Conv Theorem: The rate of the 2: "b's" in the normz constant. Does this const is this construct a funct of "n" ?. Comply size? Can it → ∞ w.n? If so Normz can be much better than conjectured.

~ D.22.02 (Sun)

NIPS

SM, STEIN : (.25) : koza's GP : 21

00:9.19 (spec) : ~~SEL~~ 9.12 ~~-.19~~ -.19 may be the long sought soln to the general SOY problem : I haven't yet figured out how to deal with it in Printz nor of PST's. — I don't even know if it's a Big Problem, hm. — Maybe trivial.

I do various approxns. to 9.12-.19, ~~drop~~ — the degree (style) of approxn, depending on how far TM has progressed.

So the idea is 7.00 -.21, then 9.12-.19

Re: Soy : 2 aspects of the problem : (1) Is it really the "BEST" choice ← (2) What is the actually expected yield diff.? Here I'm only concerned w (1) (I think !).

The present problem (7.00 -.21 =, 9.12-19) differs in another way from classical SOY, — I'm now only interested in the probty that a given PST will be "Best". → .19

(SN) In section (of Report) on "TSQ", explain how TSQ is to train trainer as well as TM. for trainer "Debugging".

.15 Perhaps a "Main Problem" is: **How Logit. are the diffs of G of the PST's.?!**

SN In doing GP using "SPACE" for Gen eval'n (~.1" per eval'n!) Koza should have spent (about) as much time on selecting cards as he did on Gen eval'n! (and/or) did approx Gen eval'n, to start out].

As I remember, one of the last big hurdles w SOY was in SM, w. large no. of stocks to bet on. STEIN would seem to be very relevant! — Seems to be a deal soln, since it gives many yields. — BUT I have considered this a Place where serious diffys! I don't remember just what the Ray were! Gaussian D.F.'s would seem not bad. I could use $e^{-x^{\alpha}}$; $(\alpha < 1)$ for fatter tails, if necy.

Anyway : If STEIN does compress the data better than other ways, then USE it!

SN I had this idea about how (organic) Evolution might be sped up, by having something like a computer try to extrapolate from the past. HVR. A big problem is how to get info about past success/failure in the system! It needs F.B. —

A weak kind of F.B. would be the set of (≈Genes) Macros found useful in the past. That rearranging of genes as chromosomes so as to closely synergistic Genes is a way of speeding evoln. Are there other ways to take advantage of available F.B. info, like that?

Could one find a better way to use past info?

ALex suggests: The older a individual is, the more likely that his genes kept him alive. — But evolution is interested in not accurately age, but how many babies has for! Could be proportional, hm.

Bayesian analysis of the data. Did it (allready) run into some trouble, in that the expected parms?

So: Say the PST d.f. is legit, & I have it for all possible PST's. Integration gives the P.d. for "Best" PST — to be used for Lsrch. At first Glance, Hwr, the pd. is not in the best possible form for Lsrch. Maybe it is OK! We need the pc's of the PST's in ruff pc. order — We don't need the "Token" pc's For Optem. problems (~ 02, say) we have the CB = 400, & spend cco P's on the 1st PST. If we have time left, we double cco & 2nd extra time on the 2nd PST (for || Lsrch) or, we just run the whole thing at 2 cco limit.

For Inv. probs, we do do doubling (or || Lsrch), but we really don't need pc's of "tokens"

So this looks like an adaptable TM soln, (except for Corrolary betw. pc's of PST's) If the d.f. on PST's is Legit & the SOT effect doesn't invalidate the Analysis!

T. Jaeurs will try to find PST's to analyse (to compute d.f. of G) that have large "expected" G's. After a few PST's are evaluated wrt. a given problem, it may become clear that we can (assign a ruff single parameter to each PST giving it "expected G" — This is to tell where to look for "Good PST candy".

We might look at the G d.f. of the best "PST Plus for" and order PST Geurges w.r.t. the probly of "Beating" the corro.

ID 779.37 (& preceding discussn) was "pessimistic about" the state of PST optem problem: Just what was "wrong"? Rita now it seems OK! ⇐
It was "T. problem of 746.00 & 771.00

T. complaint of 771.07-08 was that there were very complex, A.N. PST's that would work past corpus problems w/ hy G but not extrapolate to present problem.

Well, if it was clear that the present problem was not like the past ones — then there is no difficulty.

Reading 746.00 ff .... its not easy to see just what the problem was!

Maybe a difficulty at 771.07-08 / 746.00 ff was the idea that a PST would "look at" a problem & decide on how to solve it: So the G (P.D. corro) of a PST was to be problem independent!

747.26 – 40 discusses 2 PST systems: In one ① evaln of PST is problem-independent. In other ② PST evaln does depend on problem.

Recently, I've been thinking about ②; But ① has advantages of the D.f. on PST's, being same for all problems, so presumably less frequent Updates (?).

In both cases, we update Beta for each problem is "Lsrched" (≡ "solved").

I may have felt that ① & ② were somehow equivalent — so if there was trouble w. ②, there would be trouble w. ①. — I'm not so sure they are equiv. Hwr!

b.23.02

NIPS2

GREAT BREAKTHRU! .35-.36 → But Note 18.33!

: Perhaps t. ① v.s. ② Question depends on whether I consider QA's to be One Grand QA seq. or a "sub. QA seq. for each problem:
It's "Grand" QA seq. One can have "indexing" of Q's to help TM recognize from:
Itvr, I write concludes of a "general problem" Inv, or OZ, or IVP : &
leave TM decide how to deal w. it. (≡ type ① : 11.32)

In type ② (11.33) T: problem is dramatically — divided into 2 parts: Getting PD over PST's as a function of (problem demo) ⓑ Having PST work problem that is already somewhat appropriate to it. The PST's become specialists.

In type ① (11.32) t. PST's become generalists.

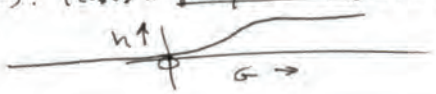
I'm finding it hard to get into the spirit of 77.00 (see 11.24) {I'm beginning to understand 11.24} we have PST's w. very low pE (very A.I.) that solve many problems. — ~~How so we must expect them~~ So, if we pick one of them, & ask how good its likely to be on t. new problem — ~~from~~ t. answer would seem to be indep. of the fact that only PST is of low ~~pE~~ eprip.

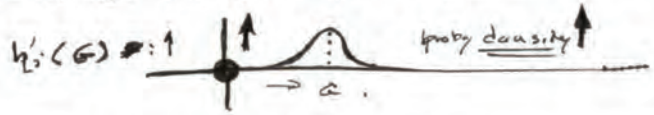
The, actually, low eprip PST's ~~are~~ don't extrapolate well — so the fact that they did well on a certain set of problems doesn't make it so likely that they will do well on the new problem. ~~[redacted]~~ Is STEIN of help here?

Well, Let us first Define the problem(s) Clearly!

1) We have this infinite set of PST's — many (almost all) have very small eprip.

2) For each ~~[redacted]~~ PS_i (≡ Pst_i), we have $h_i(G)$. There is a prob that PS_i will have ~~its~~ Gen < G ~~[redacted]~~ when given t. problem of interest.





for OZ probs, t. CB is fixed. we want t. d.f. for t. G for ~~the~~ PS_i for that problem (problem demo includes CB).

For a "few" PST's we will have empirical data on G's for various probs, T_k ← ∃ CB.

→ finite as opposed to t. infinite no. of PS_i's for which there is an empirical data.

[PS_i , $Prob_j$, T_j , G_m] Using this data set, From { $Prob_0$, T_0 } (Q) we want a P.D. over all pairs { PS_k , G_m } (A)

Q: Not Quite! May be we want $\frac{Prob(PS_k, G_m)}{Prob(PS_k)}$. i.e. $Prob(G_m \mid PS_k)$.

I think we want $P(G_m \mid Prob_0, T_0, PS_k)$ ← one or More O^j

So that's it: first find $h_n \equiv O^j \Rightarrow \sum_j a_0^j \prod_{i,k} O^j(G \mid PS_i, Prob_k, T_k)$ is as large as possible (no given PS_k)

then use $h_n(G) \equiv O^j(G \mid Prob_0, T_0, PS_k)$ | where exactly, $h_n(G) = \sum_j a_n^j O^j(G \mid Prob_0, T_0, PS_k)$

$a_n^j \equiv a_0^j \prod_{i,k} O^j(G \mid PS_i, Prob_k, T_k)$ ← use mean of all O^j's.

→ see 18.33 for Counter Argument

35

36

OOPS conclusion .16

0 : So how does hr of 12.36 stand up to the criticism of 771.07-.08 ± 11.24-.25 Maybe no relevance at all! — But then 12.36 may help solve the other problems that were discussed in 746.00 761.16 & 771.00 ff ← various; see 771.01 for list of Reviews

03 On 771.07-.08 I was perhaps thinking about SM & Strat even.

04 769.11 was concerned w. "discovery" of good PST's. 771.06 seems to think this was solved at 769.20-.20 & 762.27-.40. At present I can see it as a discovery indeed!

After we have found a good Oi in 12.35, finding PST's w. hy (xi) expected G, is a Big discovery process!

0 Actually, finding Good PST's can be integrated w. getting pc Protocol PST is "Best Bet". Once one or more good Oi 's are found. We begin to look for PSTi's w. good looking "hi(G)" curves. — using some ruff "goodness" criterion. When we find a bunch of PST's that look good, we can do an approx assignment of "prob of Best" to them. Looking for "Good Looking" PST's (after good Oi's have been found) is a OZ problem — so it will fits top criterion!

1C SN Criticism of OOPS: T. Irng. mechanism for modifn. of probys of tokens is rather rudamentary (first order Laplace rule (no new definitions). This criticism holds for any other "Irng" in the system. In General, he has no way for dealing Combinations of Concepts, unless may appear as solns to problems. The he says any kind of operators that could be added to the system as primes "tokens".

0 I think the only trick he now has is the "Boost" instruction: So he can search w. an excessively large no. of insts, then (slowly) solve some problems: T. insts used can then be "boosted" to be essentially the subset of insts used for new problems. So "Boosting" will several problem solns can give a new, useful "prime" on the tokens.

Unfortly (I think) he only has 2 kinds of tokens: (1) primitive insts (2) "frozen" solns to problems (But check on this).

J. wants me to be more specific on how I do $GPD_1 \rightarrow GPD_2$: Giving eqns. 12.35-.36 for both INV & OZ may be adequate. — I'd certainly want to do it for my own reference. Rite now 12.35-.36 plus ideas on corrections & how to deal w. them, seems to "finish off" the non-TSQ part of TDL!

133 For TSQ, I could adopt a "Don't look inside" pt. of view & try to have TM try to track a "human" concepts sequence — along w. standard hand concepts. → TM

T: hard pwr!

So: Perhaps Make a study of search Heurs used in "Early AI": Try to find ways to give a TSQ for Ir to learn! — Just acquiring having PA heurs isn't enuf: TM must be able to learn/invent new heurs as it moves into new domains &/or the myopic complexity of old domains ↑.

long time problems!

Be for .33 ref. Go over 746-761.16, 771.00 ff

Note 3.oct on "what to add to Report"

: Sections on OOPS: ID § 36.07 – §43.12 5.12

SN **Anthropic Principal in Statistics:** That ~~You are asking~~ More are people asking if Q already eliminates many "conceivable" priors. That one should have no rule in if a prior is ~~perhaps~~ of probability zero à of questionable meaningfulness.

Also, even such a def. would not be of practical interest. ~~Likewise~~ There is no problem distribution in which I could use such info.

→ 52.00

SN on OOPS: If. Boost just has rather hy pc, The pc's of tokens will be ∝ of f. freq. w. which they occur in all ~~the~~ old "frozen" poems. Perhaps not a bad idea! Short out we'll be over abundance of ~~most~~ primitives à use this trick to get better pc's for ~~f.~~ tokens: Q: There would include freq. of "Boost" — so this is a kind of recursion: It's not clear how it works out! Perhaps if one starts w/ a normal (max entropy) pc for "Boost" it will eventual converge to f. Boost value. On f. other hand, if one starts w. a hy pc for Boost, it may converge to a different value (or, depending on ~~f~~ Q: it may oscillate – or drift on doubly !) depending on ~~f~~ Q.

Or, f. pc of Boost may converge to some value, but very slowly!

Going over OOPS can take a potentially indefinitely long time. Perhaps Boost! write out f. 3 or 4 similarities & differences to QATM then give some positive & negative criticisms.

Later write up criticisms (= like a Reviewer) to J. (eg. .10 – .19

One (apparent) Criticism of OOPS: pc's of tokens are actually all "Global": Her doesn't really have even context dependence! — Her does have some "(very) long term average context dependence" — But this is not (I think) enough to deal w. f. scaling problem. J mentions a kind of context: f. open paren ( ), can only have certain things following it — but I don't see how OOPS ~~does~~ is able to get f. needed context dependence. I'm not sure enough of how OOPS works!

Mention this to J., but not in report — One possible way for J. to get context is via "push pat" & "set pat(X)" instructions. "pat" is an assignment of pc's to all tokens. So: we first present pat on to stack & set pat #7, say. Then set pat(X) ~~can~~ X can be dependent on present "context" in an appropriate manner.

In QATM, 12.35–.36 may deal w. context

35 Re: OOPS division of stack into Prefix solving all n V.S. problems solving all x ($i=1 \ldots n+1$) having $= time$: Closer to the way f. Sci Community does: Backtracking (= Resort resolution) fraction α^i on Prefix solving Pro n; fraction α^{i-n+1} on Prefix solving Pro i; ~~always~~ α may $= \frac{1}{2}$ or some empirically determined constant. Look in to way OOPS does it: Unclear as to just how much resource allocated to various "processes".

:20 : SN I had 2 models for GPD ↦ [PST] In one, ① PD was indep of problem-dern. In ② secret it was direct of problem-dern (including CB ...).
My Gen/i soln. was for case ② only! See if it works for case ① as well.

Look at comments I sent J. comparing 2 systems.

:05 I think I had many PDs:
0) OOPS was being pissed
1) OOPS had no concept of "optimal" (links to ①, ②) : Also It didn't work in induction problems – but could (too not as bootstrap, & miss if it had GPD1→GPD2
2) I think I did PC assignment of Tokens better than QATM (but actually probably not.) This may be true for early versions of QATM
3) My TSQ's had too large CTs — Making TSQ construction hard — (Actually in OOPS it is open succeeds can put in any PC modifying MSB, Headings. — This enforcement Mechanism is that PC modifying MSB that are (useless useless) will tend to be (not occurs occurs) in successful Pgms
:0 discussing unintended consequences problem — Something student
4) Also, Compiler simularly's (Frozen Man" V.S. "Arg list"
5) use of Stack Caching V.S. Function Traces (Lazy) { Very non-critical difference } → Choice of cut sets in initial TSQ } is of some import.
6) J's use of "prefix Pgms" { Is it possible to have a primitive > not a "prefix" } → 3.14
7) Differences in set of initial (Primitive Cones ... Primitive Cones) . If calculate an input distance, it is very apt to change.
8) Re ② J's choice of PC-modifying insts is critical: This set he uses is not (universal). ↳ it may be: his PC modifying insts are probably perfect for Turing Universal Systems
I was not voluntarily use to calculate PC's is rather general & certainly Universal. ↳ It should get 2 links from Bad as well as Good Events,
9) OOPS Doesn't (really) learn positively from its errors — it just tends to "avoid the bad" not "avoid the bad".
:9 10) Scaling problem of Boosting, when there are many frozen PD's in the Harmony
:0 SN How to do GPD, [update] When some of OJ's work w. Some only of corpus is other → see 20.04ff
∴ OJ's work w. most of corpus only. (Unclear as to what I was thinking about!
SN Did ... Friend of Laura: M. Livingstone's book effectively show that the normalizing constant for {semi measure → measure} must be Unbounded?
:26 SN When I put some of problems onto "Arg list", this is a way to implement "OSL" One Shot Learning"
Its not exactly correct, however. — It ↓ PC's of various tokens, even if it is not used in OSL! Also, the OSL use may not be legit — depending on size of corpus, size of Ring data structure, etc. However, in any case, the way I computed the PC's of tokens is Legit
:30 So at least its legal (i.e. ΣPC ≤ 1) — but I may not be getting the hyper PC's for the best coders.
:34 On the other hand, Lsrch automatically normalizes, so this may fix PC's again.
However, on 22.22 OSL is sort of "irrelevant": This doesn't seem to jibe w. .26 !

D29, 02
NIPS2

10 : There are 2 essential additions to t. Report:

1) Related work: Lenat + OOPS.

2) Expansion of domain of GPD$_1$ → GPD$_2$ mechanisms.

expand about sections I, Appx A barely mentioning "stand alone" system — as "front-end" of OOPS system.

Re 1) I'll write this brief discussion of Lenat, & 15.00 → 20 lines & some critical comparisons to OOPS.

for 2) I want to make clear that this (i.e. Phase II & 2) is a very powerful and general way to implement

Adaptive Lsrch.

Somewhere, I should discuss the need for study of Contexts — so it can learn quickly from previous trials. / Use of Clustering to simplify search space.

While I will eventually want to go over the OOPS paper more carefully, I'm in a big hurry to get Phase 2 additions in so I can send out the Report. → 3.26 for recipients

3 Remark: T. pg. 83 not whether a learning technique is "Bias Optimal": For some time, it has been known that Lsrch (called Oracle by J.) was "bias optimal" within a factor of 1 or 2. The big Q is, "How Good is the Bias?" The Update procedure determ. The bias was approved by the ML community

really forces the Search to ~~[illegible]~~ error ~~somehow due to best bias for Lsrch that can be~~ obtained ~~with~~ wrt t. available data, ~~for~~ within the time limit available for update.

J. makes no claims for Optimality of his update ~~scheme~~ search — which amounts to his asm for modifying the **pc's of** Lsrch. At the present stage of development of OOPS, the method of updating depends entirely on just what pc modifying instructions the system has been given. — at the present time, there is no claim as to ~~[illegible]~~ optimality of the OOPS system for modifying pc's of Lsrch — i.e. "updating the bias".

Best bias for Lsrch that is obtainable within the specified time limits for updating.

D ~~We will~~ In comparing Alpha to OOPS In comparing our present system

We will compare In comparing ~~OOPS~~ Alpha to OOPS the two systems, we will

prefer to call our present system as "System Alpha"

OOPS and Alpha are similar in that they both use adaptive Lsearch

35 to solve problems. An initial minor difference ~~is~~ is that OOPS solves only

Induction problems — but it could, with some modifications, solve optimization problems. (Maybe not so easily! — it could solve induction problems — I guess it could

see initially often pursued by backtracking of Lsearch but failed to be general enough

because to try PST's for optimization) Alpha solves Induction, optimization and Induction problems

Induction problems are formulated as a variety of optimization problem, but implicitly & tackled problems are solved by special (forms of Lsearch).

A more serious difference is in the method of updating the probability distribution that guides the Lsrch.

Initially, the two systems are similar in this respect. The probability of a "token" is obtained updated using "Laplace's rule" — roughly proportional to its total frequency of use.

Both systems augment the set of tokens by including any program that successfully solves all problems "up to now".

Alpha has, in addition, facilities for defining new functions. OOPS has facilities for "Boosting" — increasing probabilities of tokens used in one or more of the successful past programs. In OOPS, the token modification instructions are designed by the trainer, (and are ultimately "reenforced" by the Boost instruction.) ←?

After Alpha has had some success in inductive search, (using a special token (T6.35L) meaning before) this induction facility is used to update the probability distribution for search. It attempts to do this in the best possible way — in view of the available data and available search time. This is probably the most important difference between Alpha and OOPS.

In OOPS, any augmentation of the updating system beyond Laplace's rule, depends upon special instructions added by the trainer. In Alpha, token probability modification

Modification of the guiding probability distribution is obtained through the probability distribution. universal ~~induction algorithm~~. It's optimality is limited by ~~the fact that~~ only a fixed amount of time is allowed for ~~updating~~ the induction process. The performance of both Alpha and OOPS are further conditioned on initial choice of instruction set and the sequence of problems used to train the systems.

Another Big Difference is "scaling" (5.19) Important for ~~early~~ Alpha OOPS & Early Alpha. Alpha has "Conceptual contexts" to help deal w. it. OOPS will have to add special insts to deal w. scaling — we may be able to use Concept Groups.

In "Advanced Alpha", the induction-based update algorithm can be regarded as a further generalization of Conceptual Contexts. — We expect that it will have fewer scaling problems than those that are essentially unavoidable. be able to deal with scaling problems about as well as is possible.

Omnit

The more mature Alpha does not directly assign probabilities to tokens: It directly assigns probabilities to PST's that are to be selected via Levin search.

.36 The mature Alpha assigns probabilities to PST's — it does not assign probabilities

.37 to tokens directly — though such probabilities can be inferred from the PST probabilities.

→ In discn. of α: mention (Stage I is Phase I, Section I, & Appx A. Stage II is Phase II sections on Inv, 02.)

Remember we 2 defns: ① α. ② Stages I

1-1-03
NIPS2

GA: .25 : Looks Impt ! 20:18

Discuss Distance betw AZ topics. "forth". → 20.33

SN Th. Self Conforming hypoth problem. If a certain PST is fairly successful, TM will pivot it much wt. on new problems & collect much data on it. — Much less data on other PST's. — So it might tend to not realize that another PST was really better & eventually switch to it. → .16

From short sample of behavior of other PST's: if they look promising (by induction) TM wants to give them longer trials. However, before TM could really "logically understand" (i.e. do logical reasoning) PST operation, it probably wouldn't be able to see (recognize) that a certain PST that did poorly w. low CB, would, indeed, do very well w. larger CB! → .16

SN What would be useful analogy: Rocket Sled: Uses conventional rockets to push up to critical velocity — Phase I → Ramjet: for QATM!

Phase I → Phase II : Phase III (must be "logical processing" or "correlations betw Cands")

16:04 Some ways to Elim. "solid" cands hypoth (1) random choice of TSP cands.

On Pc of PST forming cands, use $\gamma < 1 \cdot ((pc)^{\gamma})$: If we use random Lsrch, is it poss'l. for this to diverge? — what happens if $\sum (pc_i)^{\gamma} \to \infty$.

If $\sum (pc_i)^{\gamma} = \infty$, it will be impossible to create MC. Carlo trials!

Is there a way to "flatten" the PST d.f. so that normally, least likely cands are more occasionally tried? (stochastic)

If we have a S-CFG; and we multiply all xfm pc's by $p \to p^{\gamma}$ w. $\gamma < 1$, This will "flatten" the d.f. — Make all Branches of closer to $\approx$ pc for each sub-branch? — But may be not very close to flattening to entire PD!

SN GA for GPD_1 & PD_2 cuda kernels using actual corpus a 414.18 — (i.e. for α_1)

Also (crossover Mut) as in introduction w. ss of $\binom{n}{2}$ — implement this using GA & 414.18. & "usual corpus" usual corpora.

Is it true that most any algorithm or system that can get "reasonably O.K. Induction", — (enuf to do $GPD_1 \to GPD_2$) can be used to start off & system, so it becomes eventually (soon?) independent of the original induction system? — A no longer used in N. If so GP is a good candidate for such a system. Well, if finds a good O^j to generate $\{C\}$ pd's (GPd_i) Run $\to$ GPD_2 to choose/generate good PST's. → 19.07

SN Re: the "Soln" to the Soy problem of 12.35-.36 : The counter arg. to the best looking cand" is the best choice as is: we have 10 independent random vars, w. zero mean & $\sigma = 1$, we have 1 independent random variable w mean .2 & $\sigma = .01$. If we pick the best of the 11, one of the first 10 will almost always be "the best" — yet Var_{11} is best one overall.

I'm not really sure that .33-.36 is really relevant to 12.35-.36, where we are

00 actually interested in which ~~stand~~. t. probty of a given stand, having ~~max~~ max G (or ~~other~~ whatever)

In 18.33-36, If t distributions were of "G", we would be interested in what GPD$_2$ gives us: t. IIB variable will indeed be very unlikely to be "Max", even tho it has a higher M than t. rest of t. variables.

In SM applicn of SOY, we want t. variable w. best mean, & PB is quite diffrnt

06 from 12.35-.36.

07: 18.32: T. "Good PST's" would (initially) include the "O.K. inductive" method of 18.28 that got t. system "up to speed". Eventually, our PST's would be found that would be better & t. pc of t. original "O.n. induction" would fade into $\approx \phi$.

0 { I may have a bit of trouble explaining just how t. updating process automatically, simultaneously ~~[illegible]~~ revaluates pcs of PSTs & "creates" new ones. The O^{ij} for GPD$_1$ gives a pd on all poss PST's. & T. "creation of new PST's" is an additional pgm. that ~~[illegible]~~ Given t. O^{ij} that gives a pd on G of a PST$_i$, to find a PST$_j$ that is "better Good" — This is t. process of "discovery" of good PST's. It amounts to an OC problem.

Section(s) of report on GPD$_1$ → GPD$_2$:

17 P12 §2: INV probs.

how to top of p 13 (2nd line)!

eliminate lines 2, 3, 4 of p13

After "$(\tilde{G}_i, s_j, F_j(\cdot,\cdot))$ as Q_j and t^j as A_j.

20 — Using Lsearch, probabilistic we took for/functions, O^i, ~~[illegible]~~ such that

21 $a_0^i \prod_{j \le k} [\,] O^i(t^j \mid (\tilde{G}_j, s_j, F_j(\cdot,\cdot)))$ α

being

Is as large as possible, z_0^i the a priori probability of $O^i(\cdot\mid\cdot)$.

24 The j summation is over all of the known $Q_j = (\tilde{G}_j, s_j, P_j)$. $A_j = t^j$ pairs.

[illegible] would back or more [illegible]

"From t. purging ..... t. probability density $h'_\ell(t)$ that $F_\ell(\cdot,\cdot)$ will solve the problem at time t." — now. $\#$ operating Fcns

29 $h'_\ell(t) = O^i(t \mid (\tilde{G}_n, s_n, F_\ell(\cdot,\cdot))$ β

30 $h_\ell(t)$ is the probability that F_ℓ will solve $\tilde{G}_n, s_n$ in time greater than t.

31 $h_\ell(t) = \int_t^\infty h'_\ell(\tau)\, d\tau$.

32 "Usually ....

Q: What is better, a soln. w. probty p_1 at time t_1
or " " " p_2 " " t_2?

There is a partial ordering if $p_1 > p_2$ & $t_1 < t_2$.

Is $\frac{p_i}{T_i}$ reasonable? It's probly for unit time. (Looks like Lrust criterion!),

for a curve, $h(t)$ maybe would like ~~max~~ max $\int_0^\infty \frac{h(\tau)}{\tau}\, d\tau$. or min $\int_0^\infty \frac{\tau}{h(\tau)}\, d\tau$

Nipsz

$= h(0) = p$

: prob'y of soln $= p = \int_0^\infty h'(t)\,dt$; Time of soln if solved $= T = \frac{\int_0^\infty t h'(t)\,dt}{p}$ } $\frac{T}{p} = \frac{\int_0^\infty t h'(t)\,dt}{\left(\int_0^\infty h'(t)\,dt\right)^2} = \frac{\int_0^\infty t\,h'(t)\,dt}{(h(\infty))^2}$ } we want min.

HA! If in 19.21 I only include data on Fc(·,·)'s that succeed, I will get a very inaccurate result! Including unsuccessful data is diff(k)! How to do it: we only know for each case, that $t_{soln.}$ was $> t_{quitting}$. — we know t_{quit} in each case, but t soln can be ∞.

One Possl. way to deal w. this: instead of using large $h'(t_i)$ at t. time t_i of soln, for t. diff problem, for each factor in t. product involving unsolved problems, optall time t_α

we want max $\int_{t_\alpha}^{\infty} h'(t)\,dt = h(t_\alpha)$. for unsolved unsolved problems.] i.e. this is t. prob'y of "Failure to t" prodn.

Does this problem occure w. OZ probs? I think not usually! Tho **for** less than certain CC's, a PST could give no G at all. **No G** could be or "$G = -\infty$" one of the possibilities that t. prodn. function considers — say for small cc or for PST's that are not much good for t. kind of OZ problem being solvd.

So OZ is closely analogous to IND in this respect. — "No G" corresponds to "No soln" for a Copenhagen ce. (or "$G = -\infty$")

necessarily this is a standard OZ problem. Not an inductive problem.

O.K. — so we have this $O'(\cdot|\cdot)$ funct. from 19.19:

As an OZ problem, we want a funct that looks at $O'(\ |\ )$ and t. problem $[\tilde{G}, s.]$ and finds $F_a(\cdot,\cdot)$'s w. $h(\ )$'s $\Rightarrow$ .00 $\frac{T}{p}$ is as small as possl.

Actually, what we want is a bunch of fa's w. large $\frac{T}{p}$'s & we then compare their $h(\ )$'s to get prob'y of each being "Best". As a result .21–.23 is not a "normal" OZ problem. Many kinds of PST's solving OZ problems give a seq. or exact of hy G. PSTs of this type are appropriate for .21–.23. But there are other PST's that give a single output after t. CB has been attend .... Such PST's are much less appropriate here.

Next: try to write this into report in intelligible form. → 23.00

SN One reason I used "AZ", rather than "FourT", say! AZ is a "Personal" lang, so subtleties are unnecessarily hid. What things in form would be "Memorable"? — well things in Dictionary! This is an impt Q. A language is good for TM to t. extent that regys in its poems are easily detected. Actually it's "most common regys" that need to be easily detected. Any regy, no matter how obscure & diff. to derb, in any computer lang, is a logical regy. — but if it's hard to derb, it is accorded low appr.

21.00 is about "1.00 – 2.00 : Uncomputability" of ALP → 22.00

1.4.03

NIPS2

easy — Kan, Tversky surv:
accy, — ① Sci Amer Jan 82 pp160-173
② Science Jan 30, 1981 · pp453-458

w. some Bayes.

00:2.40 : Say we have a statistician & we give him a prediction problem. He looks at problem & , from his experience, he picks a bunch of functions to optimize w. that data , & he says " That will take 10 min of Mechanical time."

Since the time limit is finite, I know there are other effects that would fit the data ~~as~~ well, but have longer cc. (Or even that there are some that fit as well in same or less cc)

At any rate: If data has components appropriate to those "not-used" models, then ~~the~~ statistician's models will have errors — ~~of~~ of unknown size.

.09 On the other hand, using RLP, we are given the problem, we choose our own, using
10. our experience; but we also ask how much time to use. In both cases, we have data that can have ways that we can't see. — why is .09 better than .00? If we have lots of time, it could be better — it could "recover" .00?, & then get extra models. (The statistician of .00 wouldn't know what to do w. extra time if we gave it to him. ?)

1.00-2.40, act address the "incompatibility" of ALP & suggest that other methods of estimating pc are not better! But: Is the RLP actually better than usual methods of pc estimation? One way RLP is better: ① It may give better estimate for given cc ② It tells what to do w. any extra cc available (& #!)

20.

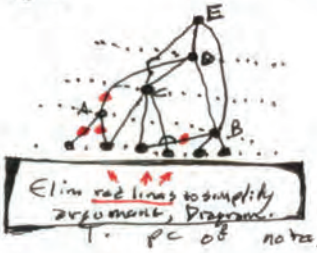
30.

6-22 mL .1733 $= \Sigma \frac{\Delta i}{F_i}$

1.4e $= \Sigma \Delta i$ * 2.95 mL $\frac{\Sigma \Delta i}{\Delta F} = 184$

0 (space 20.40) **Re:** Forth "Dictionary". In both Forths: A-Z, we can write a function & use it directly, or define it & use it. Normally, Definitions should not be made unless t. thing defined has accord-border.

What has to be done: In constructing new trial tracks: we (somehow) have to randomly try subfunctions (cf t. "function thusfar"). This is equiv. to OSL. → But see .22 ! (que sera sera)



Elim red lines to simplify argument, Diagram.

Suppose D was an initial problem soln. from t. primitives: (& A, B, C were used, they were not defined" (put in dictionary or in "Arg list" or in "frozen store")

We would like E, to be a function of D (defined) & C & B:

0 By "retro defining C & B. e.g. pc of node D

T. pc of nodes C & B are obtained by considering them to be "Tokens" like t. primitives & D. One trouble in this is that pc calculations after C & B were first constructed (b.c. not "Defined" will be incorrect if we "retro define C & B. .28

2 Actually, I may be already interested in OSL. This is because in t. update phase, TM knows t. right answer, so SSZ is at least 2. In t. prediction phase of TM operation, if it does not use OSL, it will be less good at prediction, but this could be fixed up ≈ A.H. for problems in which we have lots of time for predn. phase. Normally, predns take little time, since no updating is done at that time.

But see 15.26 for what looks like counter-examples

0 T. point is, here, that W.O. OSL, TM's updating will not be distant from with OSL. Tho, for actual Prediction, & this is often important, OSL is important.

.2 → **For Long term Growth of TM, OSL** ABSOLUTELY is A irrelevant — A I am mainly interested in long-term Growth. After TM has grown up enuf, ~~we~~ can give it t. problem of "What's t. Best way to do OSL". For long term Growth, TM needs do the prediction. The pc's assigned to corpus on update .... This is all t. "checking" I needs on T. time needed for update this progress

28 .12 → **An approximate soln:** When E refers to C, its pc is ≈ twice what it would be otherwise! **T. Reasoning:** In defining E: we mention "D" which has been defined!

2 C and B have not been defined, but it does take many, many bits of defining them which is already be covered by pc of "D" D's pc for C & B is mult by $\frac{1}{\text{total \# symbols this size (} \equiv \Sigma\text{)}} = \Sigma^{-1}$.

So we get Σ^{-2} factor. E's pc ≈ (pc of D) mult by $\left(\frac{2}{\Sigma}\right)^2$ since C & B have occurred twice. So .... (This probly needs to be done but I don't have t. time now!).

0: (Space 20:32) : Starts @ 19.17 : From 19.32 ..... Usually ... to Deutsch data .... 3 parsons. — They:

01 Given ~~[illegible]~~ a good $O^{\dot{}}$ function, there are an infinite number of $F(\cdot,\cdot)$ functions ~~that~~ ~~it could~~ for which ~~it might~~ could give ~~[illegible]~~ $h'(t)$ distributions.

We would like to focus our attention on F's that are likely to be "good".

A rough ~~measure of goodness~~ figure of merit is $\beta = \frac{\mu}{a}$. It is something like the ~~probability of solution~~ expected cost of solution, and is similar to the ~~[illegible]~~ Conceptual Jump Size of ~~problems~~ solutions to a problem. ~~"very small" means "very good"~~ Small β is good.

$$a \simeq h(0) \text{ and } \mu \simeq \int_0^\infty t\, h'(t)dt / a$$

$$\text{so } \frac{\mu}{a} \simeq \int_0^\infty t\, h'(t)dt \,/\, \left(\int_0^\infty h'(t)dt\right)^2 .$$

2 ~~We will call this the figure of merit of [illegible]~~

Given the function $O^{\dot{}}$, the problem of finding a set of F's with small β values is a time limited optimization problem of the type that ~~we will discuss~~ is solvable by the methods of ~~[illegible]~~ the next section.

Most time limited optimization techniques give a sequence of trial solutions of increasing merit that can be used in the present context.

18

This is the text of Theorem (Theorem eq. (7))

→ ~~[illegible]~~ Given a set of good F_α's and their associated $h'_\alpha(t)$'s, we can obtain the probability that any particular F_α will solve the problem faster than any other in the set. This

0

0

To be added to
nipsoz.tex — done 1.603 · 530 pm

Ray/ nip add. tex

00 $\alpha \equiv 19.20-.24$

Using L search, we look for probabilistic functions O^i such that

$$a_0^i \prod_j O^i(t^j \mid (\tilde{G}_j, \overset{\text{l.c.}}{\downarrow} S_j, F_j(\cdot,\cdot)))$$

is as large as possible.

a_0^i is the a priori probability of O^i and the j summation is over the known $Q_j = (\tilde{G}_j, S_j, F_j)$, $A_j = t^j$ pairs.

$\beta \equiv 19.29-.31$

10
$$h'_\ell(t) = O^i(t \mid (\tilde{G}_n, S_n, F_\ell(\cdot,\cdot))$$

$$h_\ell(t) = \int_T^\infty h'_\ell(\tau)\, d\tau$$

$\overset{\text{l.c.}}{\downarrow}$

$h_\ell(t)$ is the probability that $F_\ell(\cdot,\cdot)$ will solve $\tilde{G}_n, S_n$ in time greater than t.

$\gamma \equiv 23.01-.18$

Given a good O^i function, there are an infinite number of $F(\cdot,\cdot)$ functions for which it can obtain associated $h'(t)$ distributions. We want to focus our attention on F's that are likely to be "good". A rough "figure of merit" is $g = \mu/a$.

I is something like the expected time cost of solution and is similar to the "conceptual Jump Size" of solutions to problems. We want μ to be as small as possible.

20

$$a \approx h(\phi) \quad \text{and} \quad \mu \approx \int_0^\infty t\, h'(t)\, dt \,/\, a$$

$$\text{so} \quad g = \frac{\mu}{a} \approx \int_0^\infty t\, h'^2(t)\, dt \,/\, \left(\int_0^\infty h'(t)\, dt\right)^2$$

Give the function O^i, the problem of finding a set of F's with small g values is a Time Limited Optimization problem of a kind that is solvable by the methods of the next section. Most time limited optimization techniques give a sequence of solutions of increasing merit that can be used in the present context.

30

00 : Noted impt. pts. to perhaps include in comparing OOPS, α.

1) 20.33 AZ v.s. "Farm" (vs. on OOPS).

2) 20.04-19 α_2 (second phase) does use info about what trials failed as well as Q was last successful. (This is not explained in report, but it is a characteristic of α_2, I.M.O.

3) 18.28 / 19.32 : Next a great variety of true systems would be O.K. for α_1 : If Rep could learn to do induction .... incommunicably.

4) 16.30 - 17.40 : A long Expo on α v.s. OOPS.

10

5) 16.13 "Bias Optimal" Why this (unimpt.) importance of "Good Bias"

6) Perhaps use "RAMJET" analogy : α_1 is to get up to speed, α_2 is final ramjet.

7) 15:05 - 19 has a list of 10 ideas : 3:00 - 3.20 has some more General ideas about the report & related items.

8) Biggest Difference! α is a (Partially) Recursive system: A little has been analyzed in its behavior using simple problems. Learning algebraic notation, evaluating algebraic expressions, using a stack-based language, much like OOPS. The CSS of various Problems was computed. The CSS of various problems was computed. Scaling problems were observed. The effects of scaling were observed. Techniques was not applied to more difficult problems. While some analysis was made of more difficult problems CSS was not computed.

20

OOPS on the other hand, has been applied to more difficult problems and had success in demonstrating the importance of early training, in acquiring more complex concepts.

Perhaps closest to the present system (which we will refer to as Alpha) is Schmidhuber's OOPS. To compare the two systems, we will for purposes of comparison we will call the present system "Alpha" and consider two phases of its behavior: Phase 1 is the QA induction system of Section 2.1 using the AZ (average of Appendix A. Alpha has acquired enough skill in induction to implement the improved updating technique of Section 2.1.

30

start →

enters "Phase 2" when it

It can be considered as a "stand alone system" to do induction, or more general without

to improved updating technique Section 2.1

see p. 25

The behavior of Alpha is analogous to the operation of a ramjet engine: A slower, less efficient propulsion system is used to get to critical velocity — at which point the ramjet begins to operate and the initial propulsion system becomes unnecessary. Phase I is needed only to get to Phase2 — at which point the updating techniques of Phase I are used less and less.

A variety of induction systems would be adequate for Phase1, i.e. Genetic Algorithms, Certain types of Neural nets, SVM's, a modified form of OOPS, and some possibilities for incremental learning in OOPS, is not designed to do induction, but it could be very easily modified to work in induction problems as a possible implementation of Phase1.

(Monday 1 year)

Though OOPS is designed for induction problems only, it could be very easily modified to work induction problems — as a possible implementation of Phase1.

OOPS and Phase1 of Alpha are very similar. They both use universal distributions to solve problems using Lsearch. (footnote: Schmidhuber's terminology is incorrect: What he terms "Oops search" is really Lsearch; what he terms "Lsearch" is really "SIMPLE", a minimally complex program devised by Li and Vitany; to illustrate an important characteristic of Lsearch — i.e. its ability to solve all solvable inversion problems within a constant factor of the speed of an optimum solution. The "constant factor" in SIMPLE is however, much larger than that for Lsearch.)

The language used by OOPS is a kind of stack language, related to FORTH. The language used by Alpha is AZ, similar to LISP. Both of these languages can use subroutines to compress code — an essential feature in incremental learning. AZ was designed so that AZ is a functional language: Functions are represented by trees, and all subtrees are legal functions. In AZ finding common subtrees in a large functions, enables compression. In the language used by OOPS we guess that it is less likely that common subprograms (that had not already been defined) would occur with usable frequency, an analog of this technique would often be useful.

BoB systems use incremental learning to update the probability distribution that guides search. In most parts, the update systems are similar, while the update systems are similar. Alpha uses context of some kinds to deal with scaling effects. We are uncertain as to how OOPS approaches scaling — though apparently, it is a serious problem. OOPS was able to solve the Towers of Hanoi about 1000 times faster, by using search patterns (a kind of probability distribution over problem trials) from the solution to an earlier problem. Since there weren't many "earlier problems", it was relatively easy to find the appropriate pattern. If the system had solved 1000 problems before the "Towers of Hanoi", finding the correct pattern would likely have taken 1000 times as long — thus canceling out the gain in search time.

The most serious difference between OOPS and Alpha is in OOPS not having any concept of "optimization" — it has nothing corresponding to Phase 2 of Alpha. Phase 2 has many useful features. It is able to use its updating scheme to improve its induction, which further improves the updating scheme, and so on. It is able to use information about failed trials, as well as information about success.

- ~~It used data from previous problems to assign probabilities~~
- It is able to "invent new PST's as well as assign probabilities to them with respect to an attempt to solve a particular problem.

Another difference is that Alpha is largely a theoretical system. The behavior of a variant of Phase 1, using a stack-based language (much like OOPS) has been analyzed for the evaluation of algebraic expressions (Pau 1994). Scaling effects were observed when we computed the CJS's of problems of increasing difficulty. While some analysis was made of more difficult problems no CJS values were computed.

OOPS, on the other hand, has been realized as a computer program — demonstrating the importance of early learning in facilitating solutions of difficult problems.

- It really tries to find the optimum solution to a problem, rather than an improvement over the best previous trial.

∞ : 20.04 − .19 is an important ~~simplifn~~ of the system: Consider inv. probs: (20.04 − 10 =)

Marcus Hutter: Then "h'(t)" into can be divided into 2 parts.

Can I somehow integrate this into the idea that

A correction for p.13 of PS document

Suppose $O^i * (\cdot|\cdot)$ is a probability density function of the type discussed in section 1.

~~Then defining~~

define $h'_{i,j,k}(t) = O^i(t \mid \tilde{G}_j, s_j, F_k(\cdot,\cdot))$

This is the probability density (according to O^i) that F_k will solve G_j, s_j i.c. at time t.

$h_{i,j,\mu}(t) = \int_t^\infty h'_{i,j,h}(*)\,d*$

is the probability (according to O^i) that F_k will take longer than t to solve G_j, s_j.

Then using Lsearch, we look for O^i's such that

$$a_0^i \prod_j h'_{i,j,t^j}(t^j) \prod_\ell h_{i,\ell,t^\ell}(t^\ell)$$

is as large as possible.
a_0^i is the apriori probability of O^i
The j product ranges over those cases in which F_j has solved G_j, s_j i.c. at time t^j
The ℓ product " " " " " " has spent time t^ℓ but has not yet solved G_ℓ, s_ℓ i.c.

It is notable that this update schema uses data on past failures as well as successes.

continued → "Usually there is not enough data. . . . (commit subscribt on h, h' ;

$h_\ell \to h$. ack.

(only if abstracted) it says: "Perfect data will often demand ~~words~~ <3 params"

also note http://cyc.com/

D. Lenat "Cyc: A Large-Scale Investment in Knowledge Infrastructure" Communications of ACM 38, no. 11, 1995

Communications of ACM 33(8): 30-49, August 1990

D. Lenat et al "Cyc: Toward Programs with Common Sense"

refs NI crimes

http://www.sover.net/~michaelp/nlc-publications/icg285/index.html.

J. Schm 2002 July

http://www.idsia.ch/~juergen/oops.html.

Optimal Ordered Problem Solver TR IDSIA-12-02, 31 July 2002
J. Schmidhuber.

W. Paul; R. Sol. 1994

Pavl, Wolfgang, Solomonoff, R. Autonomous Theory Building Systems, Annals of Operations Research 1994.

D. Lenat.

D. Lenat, R. Guha Building Large Knowledge Based Systems, Reading, MA: Addison Wesley (1990).

1.9.02
Nips

η { on p. 13: We have a quadruple line this for each problem solved.
We also have an identical looking quadruple for problems <u>not</u> solved. Here, t^j is the time spent before the trial was ~~[illegible]~~ discontinued

---

Revisions: 26 Nov 02 original report
9 Jan 02 : Section 2.1 on "improved Update system" augmented
Section 6, "Related work" on Constr. & OOPS added,
additions to Bibl. & ext web-ref. for Cramer added.

Mailing list for 1.9.03 report

1) Kurt Alan Steinkraus — kurtas@ai.mit.edu

2) Ivan Chardin — chardin@media.mit.edu

3) Alex Gammerman — alex@cs.rhul.ac.uk

4) Natalio Krasnogor — Natalio.Krasnogor@nottingham.ac.uk (note ↑ u)

5) Gary Wolff — gerry@informatics.bangor.ac.uk

6) Doug Campbell — [redacted] dugiec@yahoo.com (note "i") tmp.

7) Tom English — tenglish@jam.rr.com

8) Henry Lieberman — lieber@media.mit.edu

9) Oliver Selfridge — ogs@media.mit.edu

10) Phil Apley — PGA@ALIENLANDING.ORG

11) David Dowe — dld@mail.csse.monash.edu.au

12) Sepp Hochreiter — hochreit@cs.tu-berlin.de

13) Eric Winfree — winfree@hope.caltech.edu

14) Steve Witham — sw@tiac.net
sw@tiac.

15) Marvin Minsky — minsky@media.mit.edu

16) Carlos — cazhv@nyc.rr.com 1.16.03

• 17) Jess Morton — jmorton@igc.org

• (8) Lewis Morton — lm@lmcd.demon.co.uk

19) Will Gersch — ~~Gersch@Hawaii.edu~~ gersch@ics.hawaii.edu

20) Murray Denofsky — mur252
~~mdenofsky~~@yahoo.com 1.16.03

21) Alex Sol ... — als

22) Cosma Rohilla Shalizi — Shalizi@santafe.edu

• 23) Kirby Smith — ~~kirbysmith@mindspring.com~~ | try ksmith@ieee.org.

• 24) Irv Wieder — WiederSci@aol.com

26) Marcus Hutter — marcus@idsia.ch

27) Juergen Schmidhuber — juergen@idsia.ch

28) Richard Gardner — rdl@rtuh.com

29) Gerald Sussman — gjs@mit.edu 617-253 5874

1.10.03
NIPS

00 : Call t system of t. Nov 9, 03 report [redacted] ALPHA). (ALP+H).

Consider Alph as Phase1, Phase2. At first Glance, we have much chatter how Phase1 works: It does have to be able to do induction, to t. extent of finding O^i ≠ eq s.s (t. first eq n ≤ 2.1) is beyond.

.05 ☞ SN Hm, even Phase2 is not t. "final t M," We have to deal w. "corrolns" betw.
.06 pc's of PST's. I'm not sure that even that is t. "final system". But not (.11 - .12)

But anyways, in Phase 2, t. system is able to discover stock loops for t.
09 [PST] Sort — which (I think) make explicit consideration of "Context", irrelevancy,
10
.11 — Also (I Think) Phase 2 is able to replace itself — (t. Ultimate "Wrapper").
.12 If .11 is true, then t. "context" consideration of .09 becomes no longer necy, — also t. "corrolns" of (.05-.06) would be taken care of.

.14 Re OOPS: Use of prefix codes for Goals: T. way it supposed to work:
A particular prefix code q^i is able to work all problems [redacted] thru n^{th}. After working a problem, it halts. So, over t. problems in put 1 thru n, the q^i's form a prefix set.
For a new problem not in t 1 thru n set, t q^i's may no longer be a prefix set.
This seems like a nice way of being sure that all new trial q^i's will
20 work probs (1 thru n); if we make all new trial q^i's; old q^i's w. post fixes on them.
Hm, I should think that "backtracking" would work as 14.35 :. (spend time [redacted] $\propto \alpha^{j-n+1}$ on q that work probs 1 thru j [redacted] $\propto$ maybe $\frac{1}{n}$)
What $\propto$ should be is unclear. Perhaps find out empirically?
Actually, t. function & Mechanics of this "Backtracking" are unclear! Say one never Backtracked.
Would it be possl. to somehow "go on", w.o. revising t. General pgm. thus far"?

SN Perhaps it would be well to work on how TM gets Strucs, since
30 this is [Part of t. Goal of Phase 1. A Basic seq. can be defined by
.31 (a) t. n type (3 overlap rules) (b) num-denominator (c) numerator.
.31 is not so clear! Perhaps Basic seq. for a radix k alphabet is clearer.
List symbols in alphabet, followed by concatenants.

SN Jass, M. asks can it go into [recurrent, repeated] loop of "improvement of GPD," Then improvement of improvement → improvemental GPD → etc ? I.e. for "improvement" read "modify."

SN Also [redacted] $prob \doteq 2^{-l(O^i)}$ l is length . . . was unclear to Jass a "naive reader"
.40

NIPS.

Impt Argt.

or more exactly
A kind of R LP : A kind of Comp. Bnd
A way of trying models before CB occurs

e.g. A method "other than RLP"

.00 On the Q of 1.00 − 2.40 : Any computable Prob distribn. can be regarded as an approxn. to ALP — which is the "True" p.d. If there is a code for the data that is much shorter than that given by this computable p.d., then the error in the cPD, with high certainty, is large.

Whether such a much shorter code exists is unknowable ∴ whether or how large the error in the cPD is, is unknowable.

.05

.07 Counterargt. "I don't agree that ALP is really the correct probability".

{ V.S. .06 : (.00 − .05) can be rephrased by saying: Whether there exists a "much better model" for the data is unknowable. More exactly! We can never get to the point where we can say "There are no significantly better Models"

Thermodynamic entropy difference between state A & state B $\equiv \int_A^B \frac{dH}{T}$

dH is incremental heat added in path from A to B. In a sealed Box,

$dH = dt \cdot c$ (c is no. of molecules) So $\int_A^B \frac{dH}{T} = \ln H_B - \ln H_A = \ln \frac{H_B}{H_A} = \ln \frac{T_B}{T_A}$.

How Thermodynamic / Entropy relates to "Informational" entropy is unclear.

In Cover's book, p33 he defines Thermodynamic entropy as "ln of no. of possl. Micro states of the system"

However, he expects probabilistic transitions from state to state, so

But any way; If one knows Temp press, vol of Gas, one can compute its entropy change to new Macro state.

Actually in Thermodyn., If T P & V are Constant,

On $ functions: "Many" distinct forms? (1) 3 input func (2) Bernoulli params: normalized denominator.

(3) explicit functions $P(A|Q)$, for normalized A, Q, $P(y|x) = F(x)$ A fix'd normalized pd (or unnormalized) $\int_{-\infty}^{\infty} f(x)\,dx < \infty$. & $f(x) \geq 0$. usually $f(x) > 0$ strictly.

(3.2) a common form $F(x) \to f_a(x), f_b(x)$: i.e. we express 1 or more params of the dist'n, f, as functions of x. More exactly: Q is of form S, X; S is a string that defines the problem; S is the continuous set of params in the problem. "S" may say the params are in coeffs of a Taylor expansion, or any other parameterized expression.

ω à

1.22.03

NIPS

Criticism of OOPS: Its method of modifying its pd. doesn't look universal to me.
(My "Phase I" isn't universal either — which is one reason I have Phase 2.)

OOPS gets its pd [redacted] on functions by (1) modifying pc's of tokens (2) defining new tokens.
However, its actual [redacted] limitations/definition of what kinds of pc modifn is allowed, is unclear.
At one (time) time I was uncertain about whether the system I had, had Phase I limitation — I think
I finally got a system that was universal (into recursively).

ON SUMACS: A sumac has 2 input (characteristics) functions (a) prediction (b) updating.
The ease of each is input; & the accuracy of pred'n. (which interacts w.r.t.
" accuracy (per unit time) of updating.)

So study various sumacs w.r.t. Phase characteristics 2 or 3 come to mind:

(1) True [redacted] unc based CB=∞ sumac
(2) (1) with finite CB: Problem Backtracking & Possl.
(3) AZ-type Lang: Bernoulli pc's of "Tokens"; New tokens definable: Backtracking possl.
(4) OOPS language: Stack based.

Remark: In General, Backtracking is an attempt to deal w. fact that CB for
each sumac is finite — so we don't know all codes for "Corpus Phi-far".
In 3 or 4 if we had CB=∞, would we have a universal d.f.?

In (3) I admit "oversearching", so any code, is eventually findable; & it's lang is indeed
Universal & answer to .19 is "Yes"

In (4), I guess it's lang is universal, but he doesn't allow oversearch, so "NB" for that system.
If OOPS allowed "oversearch", then .19 would be "Yes".

In both AZ & OOPS, we are considering [redacted] D-induction only.
Since [redacted] OOPS uses Lsrch, given enuf time, it will find any short code possl.
Having found that short code, does it extrapolate well? I think that OOPS is pretty much
to same as Phase I Alpha! It finds a (short) function that (solves) its 1st few problems.
~~This is not a theorem~~ So perhaps the Conv. Theorem of Appendix B applys to OOPS—
If he ever survived enuf!

However, his problems are all different types: So, for the "a"b" problems the examples have
to be indexed; similarly for the sources of transl. — In which case they do qualify
as QA problems & the Conv. Theorem holds!

The OOPS doesn't "oversearch", it does "backtrack", so T. Conv. Theorem may apply
(Or maybe a "less exact" way); Q: How does Conv. Theorem work with "D-induction"? — Still space

:00 : Since "Backtracking" is impt, Its impt that we do it right! My impression of a good (possibly) way is: if $Q_1, \ldots Q_n$ are t successful pgms, then it spends fraction α of remaining time on Q_{n+1}, fraction α of remaining time on Q_{n+2} ... etc. Its not clear what α should be, or what constitutes "A problem" (so fraction α should be α) Is t universe "ab" set a "single problem"? Is R is sampling fr. "Trainer" should decide? (i.e. is t our distribution a part of t TSQ definition? — it could be.)

1) Oops doesn't give rules pc's to successful pgms $Q_1, \ldots$

2)

W.O. special "pc modifying instr" (other than definitions & Bernoulli pc assignment) we should get good pc's for functions. Yet his "Boost" instr. was critical in solving t "Towers of Hanoi" problem. So Clearly certain instrs can be very impt in learning certain kinds of TSQ's. These instrs can be "Heuristics"!

:10

Hvr, for a TM to discover theorems: It has to have "Requisite Variety" of probs in its TSQ.

:12 : 33.40 For D-induction, error in $(pc)^2$ is either 0 or 1. — so we count no. of errors. T. expected no. of errors is bounded — indep of corpus size, if there is a "short code" for corpus that doesn't grow w. corpus! (corpora = plural)

This D-induction may be related (or identical) to Chaitin's ideas about "how much math" one can get from "how much axioms". The "errors" are t. axioms (?).

Anyway, if one has a large corpus of theorems w. no errors, & t. derivs (= axioms) have n bits, one could have n errors in t. theorem's applications, (maybe) — its "Expected no. of errors"!

:20

Hvr. "Expected values" when $p=0$ or 1 are quite different from Statistical "Expected values"! I should end up w. a Math Theorem (not a statistical theorem!).

Maybe put Conv. Theorem in RL form — (is stronger) than $p \to 0$ or 1.

Actually, unnecy to go at it from t. conv. thm — I think its possible to do it directly. Consider sequential prodn: (D-prodn). we have a long sequence derivd by k bits. T Deriv is "M". We have another deriv, k bits long that derivs a corpus of length n, created by M. (M can create a corpus of arby length $(=\infty)$). Say k has correctly predicted m bits of t. seq: if $k < n$, then "some" prediction going on.

:30

Tho t applicability of t Conv. Thm to D-prodn — not so (sequential, bag induction & QA induction) is interesting, I don't want to spend time on it now. Maybe I will write paper on "Convergence theorems."

Anyway: what I was working on, was t. Q of just how good OOPS was. W.o. special pc modifn. instructions, it seemed like it would give a regular Universal D.f. The pc modifn. instrs could give a wildly biased Univ. D.f. — T. Q is — would it be a good Bias?

Re: SOMACS. By limiting Backtrack depth, we make it impossible to deal w. certain "errors" (= find certain Raggys). What kinds of Raggys are these, and

Can we modify to Sumac update alg., so it can find such reggys?

Try to find raggy types that Normal Sumac backtracking would not find.

Some raggy types: (1) Linear/non linear pseudo.

One way to do it incrementally: for short sequences, try to get good corrs (Reggy will not be v.g. — to optimum will be Broad.) As more data comes in, refine optimum params.

So Prog could be viewed "incremental rung". T. trouble w. starting w. a larger Corpus, is that optima are narrow & ∴ hard to find. Tes.

W. non-linear optzn; say neural nets, (AKN & RANN) there will be several optima, so one must save many of them — then discard some as new data arrives.

Saving many is equiv. to Backtracking. Instead of finding several optima for each data chunk, just store the first one found (along w. data set address) so one can "Backtrack".

On OOPS: Backtracking: Hypoth: that given problem$_{n+1} \equiv A_{n+1}$,

If we spent time T attempting to solve A_{n+1}, by extending $q_{i'}$, then we should

if "KT" " " " " " $q_{i'-1}$

k should be indep of i'. We can then solve for T as a function of k.

Total time = $T + kT + k^2T \cdots = T(1 + k + k^2 \cdots) = \frac{T}{1-k}$. So $T = \text{Total available time} \times (1-k)$

time spent on extending $q_{i'} = T \cdot k^{n-i'}$. So we have to decide on k a priori.

T is a function of k.

In backtracking to q_{n-1}, we must first find an extension to q_{n-1} that solves A_{n+1}.

The first extension we find must give q_n, but we must overreach past q_n (starting where we started when we discovered q_n) to find at least 1 more solution for A_{n+1}, i.e. q'_n, so we can try to extend it to solve A_{n+2}.

Alternatively (more time consuming) start with q_{n-1}, try to find extension that solves A_{n+2}, then try to find further extension to solve A_n.

How much time we spend on extending q_n v.s. q'_n is not clear; maybe related to k?

Whenever we discover $q_1, q_2 \cdots q_n$, we save the state of the system, so we can do "over search" from that point.

A big difference between α and OOPS! In OOPS, the use of prefix codes assures that if a prog/soln Prgo solved problems set S_0, then any extended version of that prog will continue to solve S_0. (Contr. other hand if we give a Prgo a problem not in S_0, & it has output & stops or stops w.o. output, w.o. asking for extension, then there is no extension to Prgo that will solve that problem.)

In a functional lang., all (usually) inputs will give outputs — some will not stop, then ∴ no output. If we have Prgo (as above) & it solves all probs in S_0, then if we give it

20: a new problem: It may give output (usually it will); But extending P. code of Prgm, will usually modify some responses to older problems in S_0.

Essentially, there are 2 parts to understanding OOPS! ① T. Mechanics of "TRY." ② how the Language works: how it recalculates new pc's for tokens, etc.

One impt. idea is use of prefix codes, to reduce time spent on testing new Cands.

SN It may be that by suitably doing 11 Lsearch's duplicating/saving states at Nodes one should be able to get better than bias-optimal of 1; perhaps better by factor of length (in tokens) of solution (?).

Is the OOPS formulation more amenable to expressing S functions?

On S functs: for continuous PD's: say I have a (gray-block uniform) pd on (0,1) interval. Any monotonic function w. (0,1) as Domain, will give us a pd on its Range. What I (have made) is not Normed pd's → normed pd's.

I can probably translate this to discrete d.f.s.

In Continuous case: for each x, we have a (distrib) function. so f(x,y) gives a distrib. mapping for each x from y. common (?) domain to a common (?) Range. It is y's domain

What we want for discrete case: each discrete Q gives a distrib (usually) d.f. on the discrete A space.

For each discrete Q we have a discrete P.D. over A space. In general, A space covers all possl. Discrete objects. So it is a p.d. on all strings (finite alphabet perhaps) — It could be a grammar. It could be a Universal D.F. — perhaps it would be desirable for it to always be a Univl. D.F.

NB Just because a D.F. is Universal doesn't fix it down very much! All $c_i = 0$ Sums are Universal, but they exclude a recursive set as the TSQ Grows!

A Universal grammar can start w. a finite number of primitive functions & a discrete Pd. on them. In AZ I have a (I hope!) univl d.f. on functions. How can I modify AZ to get a d.f. on Strings? Well, OOPS does that! OOPS also asks for a new token occasionally.

Perhaps use OOPS formalism to create General Purpose UMC → Universal P.d. — i.e. to generate all possl. P.D.'s.

Perhaps have a Machine that's a mix of AZ & OOPS! Q, R → A

Q input is AZ; R input is OOPS.

Note, not 3 input UMC; one need only have 1 input tape: T. machine itself decides on the boundaries betw. the pgm of O; Q & R.

Each There has to be some convention on where Q terminates: maybe a "stop" symbol.

10 : 36.40 : T. machine/has no output until it starts. It can have some output w.o. knowing the entire value of Q — but usually (I should think) it waits for the end of Q before starting to "speak". I could have a special output, (that can be erasable) that tells when t. machine wants another input bit.

35 BUT I think my idea about "the form(s) of S-functions" was that I would express them pretty much the way that Scientists normally express S-functs — then possibly gen. the methods to make them Univl. (if necy). If they're not already univl.

So introduce problems w. S-funct solns. that one would expect a person to be able to discover. Then find ways to express these S-functs in a common, General way

I should wait until I begin getting problems of that sort — then write S-functs that seem "Natural" to a human:

-15 Initially, Use Bern P.P. for Discrete pred'n. à la (pred'n. of center) + d's for continuous functs.

Re: My Understanding of α_1 is correct!

1) I think I understand α_1 (phase 1) w. one R-funct. >1 R as I've written it may not be so good, Using R's w. pc's that overlap might be better.

α_2 is 0.6. to solve extent "In Theory", but I don't have good general (form for S-functs (the see above .05–.15) also t. "correln. between sounds" is a problem — later: "Clustering" may help. — Cross corr between some sounds may help.

2) Oops Understanding Oops has several parts:

a) T. General (org. used. How pc of tokens are changed; when does system requests now to them?, etc. : As a sub-Q: Just what are good sorts of primitives? (This last is also a Q on α_n)

b) How does Oops solve a bunch of problems? or more exactly, having solved a bunch, how does it solve a new one? My tentative understanding:

2 methods, using time-share!

(I) try extensions of $\$q_n$ (q that solved all prob's)

(II) start w. t. bunch of (n+1) problems & try to solve them in ||. T. only thing used is t. pc's of tokens "plus" t. frozen solns, $q_1 \ldots q_n$. Methods of "editing"

33 ?(= "Modifying, combining" t. q's must be generized: J. expects Oops to

34 learn how to do this — but her doesn't suggest TSQ's to "influence it". the way want to insert them "By Hand" — Perhaps may need tough techniques & have system "learn" which are usable

T. feasibility of this route (II) would seem to be the present question. 33–34

Otherwise, the use of prefix codes q's having time share as b^{-i} (for some $b>0$) looks better than (I): extensions to q_i would try to solve probs $\neq$ their (i+1) in parallel (time shared).

1:3003
NIPS

I scanned copy at 128 samples/sec.

00 : In OOPS, I'm not clear on how it does "TRY" in (1) on all tasks. It has a certain prefix, q, & it wants to know if q will solve all of t. tasks. If it does not, & at least 1 task asks for extension, then we have to find an extension that solves all tasks not already solved by q. To test then all on q alone, we can just spend t time on each task. If one fails (i.e. asking for extension) then q fails. If they all work, fine, (end): If tasks for extension, then we try a token. All tasks asking for token 1 must be satisfied by that token — or else we try next token, & next etc, on till we find one that all say is O.K., then we can go on to next token, etc, w. backtracking.

T. exact mechanics of search involves remembering that a certain extension was needed to solve a certain problem b_1. So if that extension is bad for b_2, then we have to find a new soln for b_1 as well as a new acceptable extension for b_2.

10 Of course, during this search, t. pgm always has access to info in frozen memory: pgms that have worked for certain subsets of tasks. It would seem that one would want to augment frozen memory w. new subsets of tasks that have been solved — even if they don't solve all of t. problems in "R". T. larger no. of probs a pgm solves, t. more "interesting", this — more likely to be "frozen".

I could use similar search strategies for α_1. T. main difference betw α_1 & OOPS: In α_1 t. end of a (function) pgm is the same for all inputs. In OOPS, t. end pt. of a pgm can depend on input.

It would be well for me to understand/appreciate t. differences

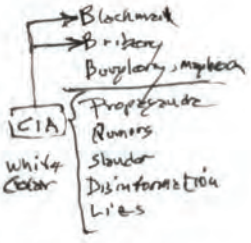
Another thing (Advantage?) of α_1 is that t. Multiple "R" model does correspond to t. way Humans do it — so it would be easier for me to teach TM,

20 Also, I may want to pgm something like OOPS, but much much faster — I would then try simple IS Q's — but w.o. my actually knowing T. soln. in advance!

Perhaps t. way OOPS gets new to locns: If A is t. top address in t. pgm, it just says "Go to A+1". — So we have to put a new inst. there. If there are A instructions in t. pgm thus far, t. pc of such a Goto would be at $\frac{1}{A+1}$ (perhaps).

REAL

Is there an Advantage of Functional (i.e. "non-prefix") codes? Is it more "Versatile"? Is discovery of subroutines such a Big Advantage? i.e. is this Difference betw. "prefix" "Really There"?



30 If it is not really a difference, then one should look for substrings in OOPS q's that are repeated.

Another Q About OOPS: Is this method of Modifying pc's Legit? T. idea of having objects in frozen memory & using their statistical characteristics in various ways to Improve/modify t. pd on tokens. At any pt., this pd. is completely characterized by t. set of tokens & their counts i.e. t. no. of times they have occurred (t. numerators & (Σ) denominators). He does mention changes of pc's after certain symbols, like "(" — but I don't see how he implemented this. [he mentions this on 11.25 ("Example....

→ 39.

MEM: .00

6 v.s. 4-2
1 v.s. 7
Great Schmarty kopf

1) Unsure about UN
2) Don't mix sci, politics
3) Don't want to appease Iraqis.

: MAX ENTROPY METhod

It seems to me that MEM ≡ Max Likelyhood!

Say p_i are the unknown pcts of various symbols in a corpus.
We have N_i is case count of symbols.
We have/known constraints on the p_i.

Then we want MEM to find the set of p_i, →

.07 $\prod p_i^{N_i}$ = max subject to known constraints. This is Max Likelyhood.

This is also MEM!

More exactly, the d.f. of the p_i (w. finite ssz) is simply $\prod p_i^{N_i}$ subject to p_i constraints.

.10 This gives a precision of the $\vec{p}$ vector.

Maybe not! According to R. Christensen!

MEM says chose $\vec{p}$ ⇒ $\sum p_i \ln p_i$ is max (subject to constraints)
— so the N_i of .07 would not be "known" — (or used!).

In this case, $\prod p_i^{p_i \cdot N}$ = max, subject to constraints on $\vec{p}$.

N may be unknown, but "large".

Max cross entropy (See / Shore & R. Chizal 3rd. 1986) is — This is a paper, not a Book.

.20 $\min \sum p_i \ln(p_i/q_i)$ where q_i are apriors of the p_i.

.21 : 38 to More on OOPS! this method of solving r_{n+1} ← problem(n+1): Divide out 2 ways: 3 times each.

1) Try to extend q^n to solve r_{n+1}

2) Try to extend λ (null) to solve $r_1, \ldots, r_{n+1}$.

Actually, this is problem .21 (2), but it is not "obvious" that these are the best 2 ways — or that either is particularly easy! It amounts to a WON problem: = "Add" next, but w. much aux info — i.e. that we know that q^2 solves $r_1 \ldots r_2$.

→ J. Says or implies in the paper that .21 is only off optimum by a factor of 2;

.30 I see no reason to suspect this is true; it should be proved or it's like has a reason!

I have, in α_1, proposed a way of solving this problem: it may be in classes of optimality, but its not clear that .21(1,2) is any better or worse. (The J uses a pre-pixlays of a kind distinct from those used. — (i.e. 181 α_i a pgm exists at some place for all inputs).

Nips

L-V book 1993 p 237 Lemmas 4.6 & 4.7 (perhaps) Does L-V have a proof or discussion?

Also p 220 Lemma 4.1 & discussion

p 223 anticoding theorem ... semi-measures. ... Theorem may be relevant.

looking Levin / Zvonkin perhaps.

If I know his result in §7.8, maybe problem is L/Z paper.

ON Normalized Universal D.F. For a continuous DF (Levin showed that if you Normalize them, then there is no one that is > all others (w. Thm constant factor).

If the normalizing constants were bounded, then this couldn't be true; so Normzn constants must be unbounded (may be not for all corpi, but for "most"?).

That the normzn constants is infinite has strong effects on why the Normed form of univ. dist. is better than unnormed form: i.e. its error is much less. — in fact, it begins to look impossible! The Expected Σ error bound is $-\ln P(u)$ where u is the generator of the data. If $\ln A(n) \to \infty$ { A is normz constant (≥ 1), n is no of bits in data } so Σ (error)^2 = $-\ln P(u) + \ln A(n)$.

$\Sigma \Sigma \text{err}^2 \to -\ln P(u) - A(n)$ · which $\to -\infty$ as $n \to \infty$!

Bound unbounded.

$P(u) \to P(u) \cdot A(n)$ so

say $P = 2^{-10}$; $A = 2^{+2}$

$-\log_2 P = 10$ $\quad -\log_2(P \cdot A) \quad -\log_2 P - \log_2 A = 10 - 2 = 8$

Maybe $A(n) \to \infty$ only if $P(u(n)) \to$ 0

So, for

Perhaps Levin should (.00-.01) if we considered all methods of Normzn, not just the type I used.

Another Possy: But my proof of Sol 78 T3 is incorrect! That the univ. form is not true for the Normalized Universal D.F.

$\prod (1+\epsilon_i)$ $\quad \Sigma \epsilon_i^2 < \infty$.

$\Sigma p < \infty$

$\Sigma \sqrt{p}$ can be

First study ratio of Universal semi measure to arby semimeasure. (including normed measures).

Univ semi measure = to any semi measure + positive value w. finite area

Normz Univ measure = Norm const × to × any (semi) measure w. finite area + positive . . .

The many semimeasures have finite areas, the lengths of the descriptions can be arbly. long.

N.B. all normzn methods are not equal: Some poor ones: (1) always choose 1, (2) always choose 0, (3) choose 0, 1 w. = probs (4) choose 1 if $P(0) \geq \frac{1}{2}$ otherwise choose 0. This last is attributed. (5) if $P(0) \geq \frac{1}{2}$ choose 1; otherwise choose 0.

Not all methods of normzn have diverging constants! Only the one I used by a normzn constant!

Anyway: Look at the Li-Vit refs at top of this page to get ideas of why Normd Univ. dists "have no "Best"".

2.2.03
NIPS

FIRST DATA

FIRST DATA MERCHANT SERVICES
20 MALL ROAD
SUITE 350
BURLINGTON, MA 01803
www.firstdata.com

Things to do in Revision of report.

1) contents; w. some comments,

2) Long discussion of Differences ~~in~~ Betw. OOPs à la TSQS & real differences approach to problem.

3) Discussion of how to use info on Failures.

4) Go thru ~~the~~ annotated paper carefully — fix typos, etc.

5) Do put in section that evaluates what has been done — what is implementable — what isn't (yet). (see ~~to~~ Feb 7, 03 letter to J. giving overview of import — i.e. how to implement s-funcs à la ~~the~~ TSQ to get them to work Phase 2 GPD options.

FIRST DATA

FIRST DATA MERCHANT SERVICES
20 MALL ROAD
SUITE 350
BURLINGTON, MA 01803
www.firstdata.com

.00:00 taken centuries for Per Scientific community to acquire, and I'd rather not have Per's team take that long!

[redacted] F —

G — See my letter of 27 Nov 02 for a discussion of just this point.

H — I've written this up, but have to find good way to insert it into the report.

.10

I — Am now working on 2 talks: One "popular" lecture for/The London, Kolmogorov Celebration; Another more technical seminar to be given at Royal Holloway Computer Learning Research Center.

Will try to finish up the report as soon as I can — perhaps after [redacted], Per's Kolmogorov Lecture

Much thanks for comments!

.19 45.00 45.00

[redacted] Have been looking at OOPS. You propose several ways to get a [redacted] single solution to [redacted] r_1, r_2, ... r_(n+1); given q_1, q_2 ... q_n.

It is not clear that any of them is in any sense optimum.

In general, L search is about back if the only information you have is in the preceding probability distribution. While auxilliary information, like is in your probability distribution, information how to use this data is not. Under q_1 ... q_n; ~~universal search NEED NOT be optimum, but it's not clear as to~~ these conditions, L search is not near optimum.

~~[redacted] what it is optimum in this sense~~

I don't think you should claim that your search technique is [redacted] "optimum within a factor of 2". We really don't know how close [redacted] to optimum ~~the~~ any of these methods are.

.30 At first I thought that you had made a "Great Breakthrough" when you described to use prefix codes for your programs. A bit of thought made it clear that prefix codes were no panacea. The codes I used in "Phase 1" were also prefix codes — but each legal code ended with a "stop" symbol. ~~Your prefix codes differ from mine, in that~~ My prefix codes differ from yours in that the prefix set is independent of the arguement. In your codes, there is a (possibly) different prefix code for each arguement.

Comment — I'm uncertain about how to explain

2.2.03
NIPS

FIRST DATA
FIRST DATA MERCHANT SERVICES
20 MALL ROAD
SUITE 350
BURLINGTON, MA 01803
www.firstdata.com

OE Answer to J's letter of 13 Jan 03:

A Actually, I think that's eq 6 and associated discussion, is about the only part of the paper where I give examples. I will, try, make it clearer that the definition of Be being proposed is independent of just what model one uses.
Also, I really should have more examples!

B "If it's enough still", when it is able to do what needs to be done in Phase 2 — i.e. in section 2.2, first equation (unnumbered): it must be able to find good $O^\wedge i$ functions.
Usually the trainer will judge when it can usefully try to maximize this expression — just as the trainer has to decide when the system is ready for ANY new problem in its training sequence. If the trainer gives Alpha 2 a problem that is too hard for it to solve, it will not solve it — which was best, timewise.

C This is certainly a good idea and I will put in a section that discusses these matters.

D The idea of self improvement: The system normally spends half of its time solving problems using search, guided by the GCPD. Half of its time is spent "improving the GCPD" by trying to get better $O^\wedge i$ values for the first equation in section 2.1. By "self improvement" I mean "improving the GCPD" self improvement.

The reason you can't implement "self improvement" is that I didn't explain how to vectorize stochastic functions like $O^\wedge i$.
In the new section on "How" the waves got there and what needs to be done", I will discuss this.
Also, you need a framing sequence that will get you to phase 2

E True, Also note that Phase 1 of Alpha using only one R can do their examples, by building various nifty functions into the single R. However on both OOPS and phase 1 of Alpha, it takes a lot of training for it does this well. In the "Multiple R" system, I am training the system about a very important heuristic device used by scientists, and saving a lot of computation time, I have no idea as to what kind of training sequence I'd need to get the system to discover this trick. Also several heuristic tricks have

But a solution by r_{n+1} is not always obtainable as a refinement of q_n.

The utility of this kind of prefix code hinges on the frequency with which extensions of q_n do indeed solve r_{n+1}.

The great feature of your prefix codes is that if q_n solves $r_1 \ldots r_n$, then any extension of q_n will continue to solve $r_1 \ldots r_n$. There is no reason to be sorry, however, that extensions of q_n will be any good to solve r_{n+1}. A possible advantage of your prefix codes: If you use r_{n+1} as argument for q_n, the program could stop with useful output or do something illegal or print some thing flashy clearly wrong before it asks for a new token. This will save time, since we then know that no extension of q_n will EVER solve r_{n+1}.

In the prefix codes that I use, the prefixes can never be extended, since they end in a "stop" symbol.

Mention has been made in Rather recent The advantage(s) of declared languages

It may be possible to devise a system that has the advantages of both extendable prefix sets and functional languages.

Does T. 3 input ones do this?

It subsequences in non-functional languages are tokenized tokens, then there is no guarantee of uniquely finding a sequence of tokens

Looking at pp 20-21 of OOPS: After OOPS solves $1^n 2^n$, it has this solution. But when it try to solve T o H, It remembers pc's of tokens for but it doesn't have to devise a code to solve both problems!

In α_1, I would have to put "indices" in to Q's so TM could tell which problem was which!

In both of J's problems: $1^n 2^n$: the extension scheme was not ultimately used.

— But did the pc's may inspired help obtain the final soln?

Towers of Hanoi

In T O H, extensions were never found to be solns.

In $1^n 2^n$, the final soln did not benefit from earlier "extension" solns: as it, took 28 times as long as it would if it had not been imposed by the in-line solns.

So it looks like "extension" did not help solve any of these 2 problems!

My values of pc's as $(70)^{-n}$ are not ribo: J biased the initial token pc's so they weren't not all = . — see p17 of OOPS! Hm, he doesn't say just what his initial bias was!

How may know simply Boosted certain insts. for Boosted 3 insts for $1^n 2^n$ as then unboosted Runs Boosted

So it looks like: for these 2 problems, the extension property of his 5 different insts for T O H languages did not help solve the problems.

His choosing an initial set of insts to "Boost" amounts to selecting which insts will be (mainly) used in the soln of the problem.

Assigning certain insts low numeric "names": This should be done more carefully: It may indicate a fact in the addressing system: The steps of the insts can be controlled entirely by transfer (which J does do by selective "Boosting" — but doing a distinct approach to tokens for each problem — doesn't sound so good!

NIPS

5 → 3
P A
10 hrs.

10

X! Fabanacci.

.00 SN Learning recursion usually this is done by hinting that f(n) = some funct of f(n+1) or f(n-1) & f(n-2) — Not by trying various recursive functions.

Also, addition, mult, expn ... could be tried as recursive functions.

These could be in unary notation $(n \equiv 1^{(n)})$. Then it could seem to x/t. unary to it's internal operators on integers

2 000 V.S. $6^4 = 1296$

Re: J's assignment of (names) low nos. to certain insts so they would be addressed w. more likelyhood!

He should arrange so that instructions that have pc's of being pitch stack that are x. normal/taken pc's — in terms of how numbers are constructed, seems unnecy & Bad!

.10 Pg1 last ¶: (§4.6): He gets "..." on stack then does boost(q) — so he boosts q^2.

This should not be done that way! q^1 then q^4 should have = likelyhood of being Boosted.

Unless, there is a special reason to reboost a particular q^n.

Is "BOOST" somewhat like "Recency" context?

On pg2 J. says that bz2, dec, boosty have been Boosted! I don't see anywhere up there!

Also x. frozen code to x. recursive soln to $1^n 2^n$ problem was supposed to keep

.20 address "2" why "2"? There were 4 bolongs to x. $1^n 2^n$ problem. — the 4th was x. recursive soln.

See Page 17 : But why dec & not inc? (it has both wts). i.e. c3 → c2 viz "dec".

If he used inc instead of dec, it would be diff(t (unlikely) to get "c2" — which is what he needed.

Superficially x. argt seems fair. The instructions should be overadequate

In general, (x. idea of starting w. an "overadequate") set of insts, then have it solve many simpler problems to see which insts were most used) seems good.

.30 Then, for more diff(t problems, recursion, say, is needed; so solve easy recursion problems to get needed insts by end of pc's.

ABCDE

I had this idea (long ago) that TM would have to learn special techniques for solving very diff (long search) problems. — ABCDefghijklmnopqrstuvwxyz αβγδΔΣμξζχφΘπρ

That Diff(t probs could be "essentially diffrnt" from easy problems.

Note: If we have a overadequate set of insts, then we solve a bunch of problems; If we arrange x. pc's diff(t problems so that x. pc of x. enlisted set of solns. is max, we will get "x. straight rule" to complete x. pc's! So Lap's rule is probly fairly good.

2.7.03

• NIPS...

Letter to Jueyan.

:42.19 : I noticed on your website a discussion of the demise of refereed journals. While I pretty much agree with your ideas on this, I think that an additional problem has to be addressed. As is, there are a very large number of papers accessible — some of interest to me — some not so interesting. Finding the potentially interesting ones might be done by an extension of Google, but finding papers that I actually want to read, is a much more difficult problem. Refereed journals cut down the search space a lot (admittedly, in a procrustean manner!).

At the Dartmouth workshop, we discussed the idea of "editors" — [illegible]. Anyone could set themselves up as an "editor", and list papers that they thought were good or bad, and why. A person could then ask a search for papers "recommended by A, B and C and not panned by D" or various other search constraints.

I'm not sure of the mechanics of this, or whether this would work. To some extent, when we write a paper with citations, we become an "editor" of this sort — but often it is difficult to find useful references in a paper, to the papers cited. — i.e. I may know that paper x was cited in paper y, but finding the actual citations in the text of y may be difficult — and often a paper will not tell much about a paper it cites.

Anyway, the general problem of finding papers that a person really wants to read, remains an important problem — even with a lot of refereed journals around. We must find a way to deal with this problem — which becomes even more serious without refereed journals!

1 PM 29° snow 2-4" 32° ky

.00 : Some random Comments on OOPS:

You proposed several possible ways to combine/get a single solution to $r_1 \cdots r_{n+1}$ given $q_1 \cdots q_n$. You then picked 2 of them and spend equal time on each, saying that the resultant search was within a factor of 2 of optimum.

I'm sure the list of possibly good ways to use L search to solve problems is quite long, and I don't know which is "the best" (if any). — I'm still working on this problem. That either of the 2 you picked should be optimum —

01 it is very unclear!

When I first read about your use of prefix codes for programs, I thought it was a great idea! More thought convinced me that it was not at all certain! The apparent advantage is that if q_n solves $r_1 \cdots r_n$ then any extension of q_n will continue to solve $r_1 \cdots r_n$, thus saving lots of time in choosing new trials.

The question is: how frequently are extensions of q_n able to solve r_{n+1}?

02 I have no good ideas on this. In the 2 examples you gave, $1^n 2^n$ and towers of Hanoi, you would have done better without considering extensions at all. It may be that with a properly designed prefix language that it is, indeed, more likely that an extension of q_n will solve r_{n+1}.

Prefix codes/languages seem to be useful in describing universal probabilistic functions. I have developed a functional language with prefix properties (inspired by an idea of Chaitin's) — but I don't yet know if it's really practical, ... Needs work!

03 There is some possibility that you may be able to double the speed of OOPS by changing a few symbols in one instruction! — This would change it from a limited doubling L search to parallel L search. — However I'm not yet sure I understand your program .... I may be completely wrong about this.

(omit — looks unlikely. Anyway, may need >1 change!)

Preparatory to running towers of Hanoi, you boosted

At a certain point you boosted by 2, dec and boostq. Why not "boost inc" also? It would seem that inc and dec should have about equal probability. "dec" was specifically useful in creating the (C3, dec, boostq) which enabled the particular solution of TOH that it found. Without it, this particular solution would not have occured. If you had used "inc" instead of "dec", it may have found some other solution — Though

it seems less likely. Probably it would be best to boost both "dec" and "inc" —
This would reduce the probability of solution somewhat, but not much.

I don't understand just which solutions to problems you "freeze". I initially thought you froze all successful solutions. In this case, there would be 4 of them for the 1^n 2^n problem. So why is the code for the 4th solution, "2"? —
It would seem that "4" (or perhaps 3, where starts with zero) would be appropriate.

~~[later~~ say it pops were 76543210 + bumping of frozen! while TOM(5) is being worked. | Not universal number
123456 123
13th 8th

I am still trying to understand some details of OOPS, so that I may be able to usefully criticize, and possibly improve — and possibly get good ideas on how to do Alpha I. → 48.0 new PP

~~We'll write more on this later~~

I want to make a list of my comments (+ or —) on OOPS! Also parts that I don't really understand (like which items get frozen, when).
My main immediate Goal now, is to test α_1 "off the end".

Def: α_1' is α_1 ~~that~~ that can work 5 induction problems. ABCDefghij ABCDE abcdefg.

Perhaps a main Q about OOPS! Is the "boost" heuristic any good? —
If good, could it be improved? What's the best way to do a boost? (if it's a good idea at all!).
As is, I decide to freeze certain codes. Which codes to freeze are unclear.
Having frozen them, they are in ~~memory~~ frozen memory accessable for & boosting a/o for editing & use in pgms. (perhaps "boost" is a kind of "editing ?!?").
Presumably, all code ~~chunks~~ chunks in ~~the~~ frozen memory have = pc (as in SAARD ANALYSIS)
Boosting the ~~push~~ single instr byz, dec, boostq = — how is this done as part of the system? Sounds a bit A.H.!. He may have just decided that ~~push~~ an initially uniform applied over all tokens was unreasonable, & that this would be one way to do this that. (Hm. "Boost" is an extreme change of the p.d. on tokens ~~that~~ that for an instant system! As the system matures, the 73 (hypothetical) occurrences of the token will be less important — In fact, in general, just how to best do a "boost" is unclear.

: [redacted]

01:47.11 The main immediate concern is the boost operation: Using boost you were apparently able to multiply the speed of solution of the Towers of Hanoi problem by a factor of 1000.

Questions: Was the use of boost legitimate! i.e. will it continue to help solve difficult problems? What is the best way (if any) to use boost? If it is imperfect, how can we fix it ... etc.

I use something like boost in [redacted] Phase 1 of Alphe, but in a very "muted form".

Should I modify it?

Will write more on this later

— Ray.

..... Perhaps ~~start talk~~ introduce Univl d.f. via ~~discussion~~ Sol 86

:00 : Kol talk: Start w. section of [Amended Amended Kol lecture notes] about discovery of Universal D.F.

"He was surprised to learn of my earlier discovery of the universal distribution and its applications to inductive inference and prediction of all kinds. He told his colleagues about this and so for a long time, my work was discovery was much better known in Russia the Soviet Union than it was in the rest of the world.

[redacted]

At first, the Universal Distribution seemed to have two serious flaws.

First it was incomputable and second, it was dependant

~ ½ hr. for part to end of page.

Do an outline of talk, first.

The discovery

1. First explain how U. universal D.F. is obtained from any Universal Again.

Then how Kol discovered it comp.

" how he found out about my discovery.

Discuss:
1) An Empiric Principal for Induction
2) Heuristic Programming" Aspects of [redacted]

[redacted] Univl. D.F. had 2 Appeal flaws:

2) Depends on just what Umc. used.

1) It was "incomputable".

Discuss incomputability: i.e. that probability (Kolmogorov) is incomputable. —

That any computable analog of probability will have errors not of unknowable size, unknown size & unknowable size. at least

~~Those~~ There are 2 kinds of errors in probability estimates:

a) ~~[illegible]~~ → ~~pc uncertainty~~ data uncertainty (essence of — we don't know rest of data)

b) Model error → Model uncertainty — we have not ~~ruled~~ ruled out all "reasonable" models.

Study of Univl. D.f. of ALP ~~has~~ have revealed much data uncertainty that pc is incomputable — that all computable approximations have errors in form of unknown sizes. — which can be arbitrarily big.

[Give (01)" as example of Model. uncertainty.]

Discuss Dependance on Umc.

1) 2 kinds of Apriori (a) Philosophers (zero info to start)
(b) Statisticians (Info before Data is known).

Explain w. examples.

Perhaps discuss Statistical Anapriori Principal. | What to do if you have no info?
What to do if you have no food, water or Air: Do nothing — you Die.

That ALP tells how to use past experience in terms of info in hand usage to reconstruct a reasonable a priori.

So dependance on Umc. is necessary & desirable . . . we need some way to create apriori info.

2.8.03
NIPS

2 aspects ① Genl. Theory of the system
② Heuristic pgmg: How to find hy pc codes.

00:49.40 0) **Second Part: on Alpha.**

1) Discuss General Goals! Very capable "scientists assistant".

2) 3 parts α. phase 1, α phase 1 advanced, phase 2
d produ sprocty
Inv, Oz, v.s. IND

} Not clear that this is good way to devise a up project.

3) Give equ. for ■ α_1 :≈(4.14.18)
Also 'Secondriverum': and explain:

4) Tell about convergence Rem.

5) Discuss Heuristic pgmg. — a "fast" way to solve (Σ) ... find short (≤ hy pc) pgms.

6) Discuss Phase 2

Discuss Lsrch:
exponential cc
"cheat" by the size of pgm.

Computer Chess v.s.
Human Chess ~~kasparov~~ 1-3 ~~M~~ moves/sec
~~2 new Bds/sec~~
~~3. Fritz/sec vsn~~
Deep Blue ~ 3 M ~ a million Bds/sec
~~(Pik Kasparov)~~

Deep Blue may ~~million~~ 20 M Bds/sec.

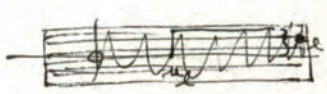
Perhaps: Early:
Discuss Identity of
a) pc determinacy, Estimators
b) Lrng.

p $7.00/sem → $2.00

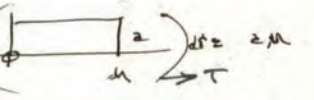
Good!

(d.p.s w. same $\frac{a}{\mu}$ are given same ordering by this method; i.e. same pc of being "best".)

: An approx. soln. to the problem of ordering $h'(t)$ functions: Section 2.1 eq (7) p14.

Assume each $h_\ell(T)$ function of the form  

i.e. $\sigma^2=0$. In this case, eq. (7) is easy to evaluate.

Actually, too easy! Integrating: $P_\ell = \int_0^\infty h'_\ell(t) \left(\int_t^{t+a} dt' \right) dt = a\mu_\ell$ 

we cannot simply μ shorter. 2

$\mu = \min(\mu_1 \cdots \mu_{\ell-1}, \mu_{\ell+1} \cdots)$

$a = \prod_{j \neq \ell} a_j = \frac{1}{a_\ell} \left(\prod_j a_j \right) = \frac{A}{a_\ell}$

Maybe 2 cases:
case 1: μ_ℓ is smallest of all μ_i's
" 2: " not " " " " .

So make a in order of μ_ℓ's size: So μ_1 is smallest in our Pyrex

case 1: $\mu = \mu_1$: $a = \frac{A}{a_1}$

" 2: $\mu = \mu_\ell$: $a = \frac{A}{a_\ell}$

If we integrate out $h'_\ell(t)$ (a & funct. $\times a_\ell$) we get zero for all ℓ but $\ell = 1$!

so t h w. smallest μ get all of t wt. ! — seems unlikely; also $\leq \beta_\ell \neq 1$.

I don't exactly know what's wrong w. this anal. sim, but another way to think about it:

t_{end}, will occur at $T = \mu_\ell$ if it does occur.

Its pc of occurring at all, is a_ℓ.

In order for t_{end} to be best, μ_ℓ has to be the smallest of all μ_i that occurred.

Again say we arrange i s. for all μ_i's, if $i<j$ then $\mu_i \leq \mu_j$. $i = 1, \ldots$

see .23 if $i<j$ implies $\mu_i \leq \mu_j$ — i.e. some μ's are $=$.

$P_\ell = a_\ell \cdot \prod_{i=0}^{\ell-1} (1-a_i)$ = probty t_{end} will occur & no faster (less μ) end will occur. probty

This assumes that all μ_i's are different. If several are identical; we have

This assumes zero var for all d.f.'s: What is effect of $\sigma_i^2 > 0$?

If >1 end have same μ_i, .21 holds but the product is over all ends w. $\mu_i \leq \mu_\ell$.

Special case.
"Special" split by
" " one must be

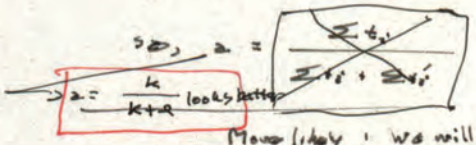
for $\sigma^2 > 0$; consider each μ is split into 2 or more cases of diff μ — Prize } Maybe might to do 2 split at least 2.

doesn't complicate the solution much.

So we find ends of by $\frac{\mu}{a^2}$ & then use .21 to induce a p.d. on them for L search.

A more direct way to approx. μ & a for a end!

Say we have many trials for the end. Successes at $t = t_1, t_2, \ldots t_k$ | failures at $t'_1, \ldots t'_\ell$.



$\mu = \frac{\sum_{i=1}^{k} t_i}{k}$. | $a = \frac{k}{k+\ell}$

More likely! we will get algorithm that will find best a & μ for a end, & correctly $\frac{k}{k+\ell}$

N.B. I have to do the above analysis for failures as well as success: Prize is to be modified eq (5.5) p13 (first eq. on section 2.1) Maybe not: Prize eq. obtains $h(t)$ & $h'(t)$

t_i's Rarely obtained.

2.9.03
NIPS

:00 (space 19.09) : epistemology
Anthropic Principal! A certain initial σπερμ + suitable TSQ → Array of Modern Man.

W.o. proper stand. TSQ, we would not be here to ask QS about σπερμ!

[Situation similar to that in physics: If laws/σπερμ physics were not close to what we observe, we would not be here to observe them. However in physics we have to do experiments to determine what these laws & constants are.

In a sense in studying Learning, we have to determine empirically what man's σπερμ is & (perhaps guess at some initial σπερμ & T. TSQ that led him to his present σπερμ ("state").

for talk: Explain weak Anth. Principal in Physics: Then Anth principal in Learning.

10
11 TSQ's in OOPS: prob1 = 1^2^n; prob2 = TOH:

It looks like the early examples & on both probs were unnecessarily complicated.

13 [In both cases, the early problems were used only as "tailoras" to help find the (minimal)
14 prob1,2(k) had a recursive soln (usable for real fn.)

Btw Byt. way: Just what was the 1^2^n problem? Was it that TM find a pgm w. n as input & 1^2^n as output? The TOH was, I think, from initial setup piles of disks, to legally transfer them to new piles.

It looks like .13 - .14 is true. We would have saved some time by giving
20 prob1(3) than prob2(3) (I think it was solved recursively in both cases, but check)
prob1(4) wouldn't have taken much longer than prob1(3); but prob2(4) could take
2 times as long as prob2(3) so the T. TSQ prob1(4) prob2(4) would be
much slower.

24 In having long of recursive fcns (I wrote about this recently) the mode of
long is, say prob1, prob2, prob3 . . . T. long of the recursive fcn that occurs
26 when TM retrieves value of soln of prob3(k+1) to soln of prob3(k).

In OOPS TSQ, the only thing that OOPS saved from prob1 (6) that enabled
it to solve prob2(3) was that the instrs used in recursion were (defnp, ci, callp, c2, endpp)
30 used (1). — Just which instrs? [Notation: CFG(k) ≡ 1^2^n(k) ≡ prob1(k)
≡ TOH(k) ≡ prob2(k).

Using this fact our way have gotten factor of 1000 improvement (but looking this more carefully using the fact .24 - .26 the factor of improvement should be much larger. (much smaller CJS).

While OOPS got "factor of 1000", CJS was still very large (I would say "too large")

N.B. T. 6th trial for CFG( ) was successful
" 3rd " " TOH( ) " "

space
55.00

00 Tutorial talk: Part 1: (secs 49.00–.40)

Derb. discovery of ALP, Univ. Dff. Tell how it works, draw [resistor] — unpredictable output tape — is [redacted] main constant.

What is PC Phat machine will write "x" before stopping (if read/write tape).

U. IO machine may be easier to derb.

Give

$$P(x) = \sum 2^{-l_i}$$

$$P(x) \approx 2^{-K(x)} \quad \text{.....}$$

Tells how one up cores" use by many of you more! time for.

Discuss K complexity [redacted] in this approx. Tell about K's interesting Random nesses in properties of Complexity .... was surprised to learn of [redacted] earlier work on inductions

Also: w. ALP: Overfitting or underfitting never occurs; it fits just right

No need to divide data into fitting set and testing set. just 1 set is needed

Accuracy: Discuss Conv. Theorem.

Diffys: ① Incomputability: [redacted] have π or √2 not computable exactly but approximable to arby precision. [redacted] ALP also approximable to arby precision.

But one never has used assumed ent error. It is theoretically impossible to know what the error is.

This might be regarded as a Bug in ALP! Actually it's not a Bug but a Feature, Feat makes it better than other prob eval methods:

Let me explain! Ordinary/computable Methods of prob Evaln. ~~all have errors in them or under~~ you compute it and your techniques. If there are better models of the data though you have used, your estimates will be in error. In all cases, it is impossible to know if there are better models available.

Perhaps discuss ssz-error uncommonly v.s. model (uncertainty: error): ⎍⎍⎍ $p = .5$ v.s. $p \neq .5$.

$\frac{1}{2n}\left(p^2+(1-p)^2\right)$ [redacted] $- p^2 = \left(2p^2+1-2p\right)\frac{1}{2} - p^2 = p^2 - \frac{p}{2} + \frac{1}{2} - p^2 =$ [redacted] $\left(\frac{1}{2}-p\right) = \sigma^2$ $\sqrt{\frac{p(1-p)}{n}}$

MeanSq. − Sq.ofmean. means $(p+1-p)/2 = \frac{1}{2}$?

0, 1; $\sum X^2$ pn is so $\frac{pn}{n}$ = synth mean of sq. p = sq. of mean. ; means p so $p - p^2 = p(1-p)$

[redacted] $p(1-p)$ a σ² per bit. $n\,p(1-p) = \sigma^2$ for entire seq. $\sqrt{\frac{p(1-p)}{n}}$ = experimental = $\frac{1}{2\sqrt{n}}$ for $p = \frac{1}{2}$

So ssz error: $\frac{1}{2} \cdot \frac{1}{\sqrt{n}}$, stands down.

Compare w. Model uncertainty error $\approx \frac{1}{2}$.

ALP is honest about uncertainty: other computable methods are not.

ALP suggest how to fix error: suggests new better models.

other computable methods usually don't.

Summarize Conclusions

Next: Difficulty: ALP depends on choice of UMC.

Convergence Thm says $\sum$ error $< \frac{1}{2}\ln 2 \cdot b$ $\overline{(\text{error})^2} < \frac{\frac{1}{2}\ln 2 \cdot b}{n}$ v.s. $\frac{1}{\sqrt{n}}$ for usual statistics

Here "b" is very much UMC-dependent: So ALP is very "subjective".

2.8 " "
Ar
20 ...

: Meaning of Aprip

1) classical philosophy : case of No apri info < No Aprip, Air — what to do? can occur / No apri info — what to do. — very unlikely to occur.

2) Practical Aprp: Info before data is seen. — change of apriori during life as one updates after each problem is solved / data predicted.

What we have w.r.t. large amt, but not well known.

What we have as Adults is to some extent known: we can use language of our Science to construct instruction sets for universal computers. That useful vocab of a science has much info in it . . . . to be used for aprip.

In Physics we have Anthropic Principle. 52.00 That laws & constants of Physics, as we know it, happened to be such that Intelligent life can evolve, and thrive. This seems to indicate only a narrow set of values for these laws & constants are possible — otherwise life would not occur.

In a similar way, initial life in universe has to have certain aprip built into it & a "reasonable" TSQ to bring it to present day, intelligent man. Assumption?

In Physics, the laws & constants have to be discovered & determined.

? { In statistics & learning theory, the initial apips have to be guessed at & we may be able to trace the growth of apri "knowledge" in Man. } ?

[Actually, this evolution to Man may be part of the Physics "Anthropic Principle".]

In the case of laws of physics, small deviations from present laws would destroy man — But I suspect that modifications of initial apriori & the TSQ could be rather broad, & still intelligent humans would evolve.

For the Bayesian View: Non Bayesians do consider apri info, but they don't put it in form of a PD of the sort I'm describing. —

One common way to put in apri info is to limit the model set considered — give all other models apri = 0. There will always be models that have not been considered that might give much better probability values. we are never certain this is — we can't ever be mildly confident that there are no such models.

If the models seem to work satisfactorily — Bayesian statistician should be happy — There will always be (within the background) the possibility of discovering that he has been living in a fool's paradise!

Using the Universal Distribution, we are much less surprised when radically new models are discovered!!

00:52ff : OOPS (cont.) : Additional point: Say OOPS calculates pc of each token it obtains: One interesting point here OOPS instructions can change state of system much (much Many Many). Boost does this.
simply presenting a token changes system a little (it + num/dennum of various token pc's) — but this is a small change compared to [redacted] "Boost."

In what sense does it above give a Universal dif.? If it does, does it have any advantage over a lang (like AZ) that doesn't have such large jumps in pc's?
[Yes, if I properly implemented OSL, AZ would have "sudden" large pc changes for tokens].
Also if AZ does succeed a v.p. conc. that is it hadn't known before, token pc's could change much.
— If AZ rarely would have many big discoveries (since all pc's of a K concs must be recalculated (other than OSL)) — this is less likely in AZ.

SN on "Negative pc's." As TSQ continues, TM can discover new cases, which could "correct/modify" pc's calculated in past. Could "negative pc's" (of computation, ideal trial) be a way of implementing this "correction"?

A better way to interpret "boost": Because of this effect way what N illegal if N>K Boost looks at stack, if there are K legal anymore for Boost, sometime used later N
Boost N mod k. Instead, if there is anything on stack, this treated as an integer N and → N mod k.
it's a name of an instruction. This is more "correct" ppgm but does not change pc
But in general, Boost should give about equal probly to all frozen ppms, (unless one wants it to be able to single out certain ppms for recognition)
safety don't but more fuzz!
2.22 an finance.

T. first time Boost is used, it really narrows down insts that are at all likely!
Since this is usually a small set of insts, we can quickly do all of them. If one of them is a Boost, we can segment the instructions each.
5 3
6 2
1 1
2 0
3 ...

For 1"2" we start with boost of c1, c2, dec, boost, byz. 5 insts
For TOH " " " " " c3, c4, c5, dec, boost, byz. 6 insts

Initially, the coins are mainly limited to the boosted insts, w. 1 or 2 others used. Other insts have pc of maybe 3 "boosted" insts. $\frac{1}{7}$ for boosted $\frac{1}{7 \cdot 73}$ for unboosted ≈ 25 × 7×7

Essentially what OOPS did in solving TOH! It found 1"2" ppm first and insts needed for TOH coin.

For a recursive solution, def[redacted], call[redacted] and end[redacted] are important insts: That they should be useful for both 1"2" & TOH is not surprising.

It took 10 times as much time to solve (recursively) TOH as 1"2" (TOH used 1"2"'s pc's via "Boost".

SO GENL Conclusions: T. OOPS mechanism enables use of tokens of "successful" ppms, but in a rather exclusive way: So if a ppm is picked pretty much only tokens from that ppm will be given reasonable pc's: Other tokens could be (in pc equivalent) ≈3 or 4 boosted tokens!
This is a rather sharp adjustment "of recursion" way to select out tokens:
A "softer" way would not ↑ more be equiv to Q_N (=73) case counts, but something less — perhaps the denominator could be ↑ by a certain %, & numerator ↑ by some (absolute) ↑ as denom.
Also note that in More Mature TM Rep $\Delta = Q_N$ will be less imps. (Rep it amounts to Q_N ratios)

There is some attraction in the Boost idea: That a trial should consist (mainly) of the tokens in a previously successful ppm.

56:00 spac

$\frac{9.3}{4.5 \cdot 10^{-3}} = 2k$

00: 55.90 (spec) Another view: that Boost keeps together tokens that are "synergetic" — that work well together (like defnp, callp, endnp used in TOH in 1^n 2^n, recursive solns.

01 Inotod that if one wants to get token pc's so as to (postHock) maximize pc of all solns — right. pc's will be the straight rule (or the Laplacian rule perhaps) — which is what AZ is opps (w.o boost) gets. → How can we insert the synergy effect of .00 .01? .oy

94 [I suspect that the success of the 3 (.01) insts in a single recursive prim ... isn't enough evidence to justify using them in a "Boost" only one case.] → 57.20

07 One way to perhaps get more wt. for the recursive 1^n 2^n soln: Use this prim on many values of N as in TSQ

oy .04 In .04 If the "Boosted" set were certain probabilistically we'd less with, it would seem like a more reasonable approach: As is, one's much more sharply constrained to the "boosted" soln(s) T: "softer" boost of 55.34 seems more reasonable.

On the initial boost of byz, dec, boostq. + C1, C2 or C3, C4, C5.

The initial 5 or 6 insts can do little by themselves, so they must boost some previous q^i to obtain a useful inst. set. So we have taken q^i boost + byz, dec & C1, C2 or C3, C4, C5.

This means we can parameterize the integers to the that ranges q^{2^i}'s boosted. — Also have those integers (i' byz) available w. the boosted set of insts. We also have the most of 6 insts, but they have a pc of (maybe) 3 boosted insts.

---

19 Q: Can the system afford to boost > 1 q^i set of tokens? It turns out to the pcost amounts to boostq; & some with insts. My impression is that the addressing of q^i to "boost" is as by: that all should know about things pcost. If illegal addresses for i (in boostq ... q^i) are quickly recognized as illegal & ignored, then they cause no trouble. — would like to be sure that all the legal i' have about same pc's over. Probly the thing could be modified to do this: (see 55.12-.15) One way, after insts boostq is put on stacks, any integer up to byte i of the new q^i is legal token. (This is in line w. OOPS computing the pcost of the next token) — We may want these integers to be "tokens" in their own (numerators & denominators for the pc).

10 This would resolve .19 a bit: > 1 boost could be possible. 2 or maybe 3. — Rao probably 2 & wouldn't want "diversity" in instructions — there are some things like "crossovers" in GA. A 3 way crossover may be unnecessary.

On "weakening" boostq: As is boostq amounts to the additional cases/counts for boosted tokens. One kind of weaker boost would be a PN case counts w. a +1. 28 M w2k ≈ 15 M

looking at page 22 of OOPS! Train of pc of soln. to TOH with boost is w.o Boost! ≈ $\frac{73^4}{15k}$

$\frac{\left(\frac{1+73}{7+73}\right)^3 \left(\frac{1+73}{12+73}\right)^7}{\left(\frac{1+73}{7+73}\right)^3 \left(\frac{1}{7+73}\right)^4} \approx \frac{\left(\frac{1}{12}\right)^7}{\left(\frac{1}{7+73}\right)^4} = \frac{(4+73)^4}{12^7} = \frac{4^4}{12^7} \cdot 73^4 = \frac{73^4}{15k} = 28 M$

$\left(\frac{1}{7+73}\right) = 1.95 E-3$

$((4.73) \times = \times =)^{-1} \rightarrow 1.37 E-10$!

$(4.73)^{-1} \times = \times = \rightarrow 1.37 E-10.$

2.92

≈ 2.83

2.12.03

NIPS

OOPS,

00:56:40 So ratio = $\frac{734}{15K}$ for tracker on. i.e. pc to same. $(15K)^{\frac{1}{7}} = 11$ i.e. $73 \to 11$. $\left(\frac{1+73}{73}\right)^7 = \left(1+\frac{1}{73}\right)^7 \approx \left(1+\frac{1}{73}\right)^{73 \cdot \frac{1}{73}}$

if $73 \to < 11$ we have much log: if $73 \to 1$ we lose for $73 \to 1(\ ) = 2^7 = 128$

factor of $\frac{15000}{128} \approx 120$ factor of loss! I don't understand why we lose/lose any Prog!

OH! we lose because of the initial boost of

Woops! for boost < 73 my figures are wrong! I have wrong domain math.

$\left(\frac{1+73}{73+11\cdot 73}\right)^7$. $\left(\frac{1+x}{73+11\cdot x}\right)^7$ v.s. $\left(\frac{1+x}{73+6\cdot x}\right)^3\left(\frac{1}{73+6\cdot x}\right)^4$ for $x=0$ the ratio is 1.

fixed pt eq

73 [illegible] 689K

in F:/SM/OOPS1:

for $x=0$ I got ratio $= 2.57\times10^{-6} = \left(\frac{1+x}{73+11\cdot x}\right)^7 \Big/ \left(\frac{1}{73+6\cdot x}\right)^4 \to$

$2\cdot\left(\frac{1}{73}\right)^3 = 2.57\times10^{-6}$

The reason $x=0$ has ratio 73^{-3} is it involves the pgm c3 dec boost, which $pc = \left(\frac{1}{73}\right)^3$ — total loss since it [illegible] nothing on $x=0$.

We extract cost of (c3, dec Boost) → # symbol soln of TOH

W. 7 " " " " above.

so c3 dec boost has a "drop" value i if $x>0$ it # pc of next 7 symbols.

If $x=0$, we only have to drop cost in no. in pc of 2 symbol soln,

20 :56.07 : On "synergy": A very basic idea in TM is that good new ideas are formed by combining old good ideas & parts of Pgms. If we run after 2 boosts (see 55.22 ors.. " "

We can regard Pgm as tentative recombinations of old "Good ideas". & "Boost", but

is very vague about how to do the recombination, and it doesn't preserve order or spacing of tokens in the "boosted" pgm. For $1^n 2^n$ his soln was (defnp, c1, calltp, c2, endnp

for TOH soln was (defnp | c4 calltp c3 c5 calltp | endnp)

The TOH soln had same order as ("2" soln, but it also had one extra "calltp".

If OOPS Did a lot of rare-event pgms, it might realize that defnp was a good start, endnp was a good end of a recursive pgm: Rest one or more "calls" were usually used.

I really don't understand either pgm, so I can't really tell how one factors advantages or ruggys in the other.

If we assume the end symbols defnp, endnp, we only need (c4 calltp c2 c5 calltp.

If we choose tokens from list tokens from list c1, c2, c3, c4, c5, calltp. (6 symbols)

it's pc is $\left(\frac{1}{6}\right)^6 = (46656)^{-1} = 2.14\times10^{-5}$ rather large!

SN OOPS Doesn't really learn much from previous prob. solns: only small biases in Token pc's (viz Laplace rule) plus gross pc changes in distribns via [redacted] Boost. Humans certainly better transfer wisdom into from previously solved problems. My own approach is to try to put any knowledge I can think of into a form that TM can acquire, use. — Do we TSP heuristics from Pgm Theory ... possibly via "Hints".

<cut?>

Kol. talks.

10: 54.40: This Lecture is going to be about three things:

1) The Universal distribution and Algorithmic Probability.

2) Its relationship to Kolmogorov Complexity

3) The application of Heuristics to a system for Machine Learning — (which has always been my main interest.)

One way to introduce the Universal distribution is by considering the first "No free Lunch" theorem. A very natural way to do prediction of sequences of symbols, is to assume an apriori probability distribution all possible sequences. The obvious first candidate is the Uniform distribution: All sequences equally likely. Turns out that this is very bad. Given the future is independent of the past: No matter what happened in the past, the next symbol has the same probability as any other. Back to the drawing board!

If all strings don't have same probability, what about descriptions of strings all having same probability? This works not quite well.

A better way: Suppose we have a long sequence of symbols: for example, binary symbols — and we want to extrapolate it. One way to do this is to hypothesize that the sequence was created by some algorithm — then use that algorithm to generate the sequence. There is a nice way to do this. We start out with a universal algorithm, that reads in put strings into output strings. "Universal" means that the algorithm can simulate any other algorithm using a program that is a certain fixed length longer than the other algorithm (input to algorithm being considered).

(draw pictures).

Since we don't know just what the algorithm was, we use a Universal algorithm — one that can simulate any other algorithm. Since we don't know just what the input to the algorithm was, we assume all inputs to be of equal (likely) probability. This amounts to saying that the sequence we want to extrapolate arose created as the output of a universal device with random input.

Assumptions: There are many possible forms of universal algorithms! Our best known is the Universal Turing machine — And most any computer is a very good approximation.

This is the Universal distribution — the output of a universal device with random input.

The distribution probability distribution on output strings can be used to extrapolate any finite string of symbols.

later → Symbolically, this looks like the long-sought "philosopher's stone" — the device that will answer all questions.

In more concrete terms, if X is a finite string and we want to assign probability to it, find a set of strings s_i such that $U(s_i) = X\ldots$. Then $P = \sum 2^{-l(s_i)}$

Perhaps the main application to Sci Method: Choice betw. Competing theories that both adequately explain data.

: It can be also modified to extrapolate an ordered set of finite strings — (the legal sentences in some unknown language.

If we use this technique for induction, the errors in probability obtained decrease very rapidly with the amount of data used.

We can use this model to determine just how many coefficients to use in linear or non-linear regression. — Thus solving ~~one of the most serious~~ a very serious problem in modern statistics.

When we use this method for prediction, there is never any problem of "over fitting" or "underfitting": the fit is always just right.

When we do prediction with data we don't have to divide the data into a "training set" and a "test set". We use all of the data for training and the expected errors are always just right.

It would seem from all of this that the universal distribution is a kind of "philosopher's stone" ~~[redacted]~~ that solves all scientific problems.

There do, however, appear to be two Bugs in the ~~idea~~. First that the ~~use~~ probabilities in the universal d.o.f. are ~~[redacted]~~ incomputable

Second, that the values obtained, depend on just which Universal machine was used to generate it. It is my reader's that most of these are Bugs — they are ~~[redacted]~~ very desirable features.

Let me explain: The real difficulty is that probability itself is incomputable. ~~Any theory of~~ Any thing claiming to be a model of true probability must be also incomputable.

A bit more detail: The value of π or $\sqrt{2}$ are "incomputable" in the sense that we can never find exact decimal values for them, ~~The Universal distrib~~ but we can make approximations that we know will eventually have arbitrarily small error. In the case of probability, we can also make approximations that approach zero error. In the case of π, however, we can always know a limit on how large the approximation error is. In the case of probability, we can never know a useful limit on the error size.

In ~~statistical~~ probability Estimation, there are ~~two~~ tw kinds of uncertainty:

1) sample size uncertainty
2) Model uncertainty.

To illustrate sample size uncertainty: Say we are given a binary sequence and we want the probability that the next bit will be 1. We could just take the relative frequency of 1 in the sequence and use that as a probability estimate. ~~We can~~ The expected error in such estimates is proportional to $n^{-\frac{1}{2}}$; n being the sequence length. — ~~As~~, when we have a large number of symbols, $n^{-\frac{1}{2}}$ becomes very small.

Nips

Suppose, however that the sequence were 0101010101010 (1) The previous analysis assumed a simple Bernoulli model for the sequence and would get a probability of about ½ for 1 being the next symbol. If we use a different model, in which the probability of a symbol is based on the previous symbol, the probability of 1 will be close to zero.

In general, Better models will give better predictions and better probabilities.

In applying the Bernoulli model to (1), we had an error in probability of about ½.

In general, there are an infinite number of possible models. A relatively small number of these models would be considered reasonable. Many have more parameters than there are data points.

But these models cannot all be tested in any finite time.

Of the many reasonable models, some are very difficult to test — they require potentially infinite amount of time to test. For this reason, there are usually better models than one's we've tested, yet we cannot know how much better they are! — Thus the essential uncertainty in all probability estimates — notably those associated with the Universal Distribution.

[Instead, maybe use shorter exposition in Sol 86 in talk; Emphasis perhaps to... large. (long) exposition would be better.

So Start talk like Sol 86 starts:

Alice in Kol: Announcement: After Kol. was surprised . . .

There were 2 apparent difficulties in the Univ. D.P. (also, at first, accuracy)

1) incomputability

2) Dependance on reference Univ. Adopt Bayesian View of apriori: Don't try to justify Bayes.

My impression is that neither of these are a legitimate complaint about the Universal distribution — they are reflections of the universal Probability itself. Not only are they unavoidable — but one mostly doesn't want to avoid, but understanding of them can make better probability estimates.

First: Incomputability :::: π is "incomputable act:

. . . Solomon in Sol 86, p474: (Do mention "Completeness" of ALP/Univ. D.P.

Use "Public Safety" etc.

Second: Dependance on Reference Univ.

Univ. D.P. is an Apriori d.P.

Apriori means at least 2 things.

a) Universal Apri — same for everybody for all situations: Of interest to philosophers — not to some.

b) Applied Apri that one has before one sees Data. This is the same kind of Apriori that I'm interested in: After one inspects data and finds some regularity, the Apriori is modified ("updated") to reflect those regularities. The updated is contains a history of a particular person's sequence prediction sequence — the very particular universe that he was concerned; plus those that he was born with.

How a particular How we get the apriori that we're born with and just what it is are

[redacted] things that I will not discuss in this lecture:

We can approximate [redacted] the universal Turing machine or universal Algorithm. This can be done in many form: One form is an ordinary computer language. In this case we can express our a priori information to some extent, by making [redacted] choosing the instructions set so that the processes

Very compact, regularities that we have observed [redacted] – that we feel are important.

The language APL [redacted] is notable to express unfortunately commonly used Mathematical concepts in extremely compact form.

The dependence of the expression the univ. alg. is something [redacted] the evaluation

to express our personal appeal by choosing the characteristics of [redacted] Turing alg.

So [redacted] To summarize, Incomputability and dependence on reference Alg. are not really Bugs in the system. They are necessary parts of the probability concept that we want to learn to [redacted] to take advantage of.

[redacted] I have attained one way to use the design of universal alg. to express needed apriori info. We can take advantage of incomputability by using this formalism to suggest new inductive models. Incomputability implies that there are always models that one has not yet tried, and it gives a somewhat different suggestion on where to find new models, e.g. by considering [redacted] models that are rather simple, but may be very time consuming to evaluate.

SN I think that I should outline just what I want to cover. As I write ("On line" so to speak) I tend to wander off into branches that I can't complete. Just list topics, indicating just how far I want to go into it. Then try to find [redacted] fast ways that do each section.

- Intro
- Sol 86
- Barc
- Nips talk
- Nips Abstract.

SN
- Discussion of all over system: Introduction to [redacted] How do "well defined problems"
- Use of [redacted] classical "Heuristic search" to solve it — using TSP.
  - Discuss Lenat's AM: Artificial Mathematician: Goal was to find interesting concepts, Heuristic
  - Data [redacted] Terms. It had sets of rules for how "interesting": something was, also rules to extrapolate: A few rules to find new interesting Pieces" from old
  - Ok it is a bit out of basis more [redacted] stupidly defined: we want to make a [redacted]
  - Certain problematic expressions [redacted] Lenat had very rudimentary understanding of induction — we have a much better model & much better search procedure.
  - Ok has to solve stupidly defined specific problems that we give to it. At first it solves them thru "standard" search procedure, using a G-CPD to guide its search. After it solves a problem it tries to improve its G-CPD. It does this using Heuristic tricks that have been either programmed into it or that it has learned from its experience in problem solving.

Perhaps shorten it by [redacted] we ref. to Lenat, but do have ref. to Heuristic Progs.

For the more parts, heuristics are implemented as modifications of the G-CPD.

So : outline: 60120 :

Derb. Univl. Df. : There there are 2 apparent diffs.

1) Incomputability 2) Subjectivity.

show 1) is property of probability itself

2) is natural to Bayesian statistics: Non-Bayesians have other ways to include a priori info —

— but apr info is essential . Attempts to make it non-subjective of necessity result in poorer induction —

Much more limited options .....

I will discuss a particular application I'm most interested in — Machine Learning.

Introduce as new form of classic Bayesian approach to Inductive Inference.

3 elements: Univl. D.P.; Lsrch; Heuristic Search. Earlier: explained that Induction = finding shortest

Lsrch = shortest Code; to improve Lsrch.

Q: "Is Heuristic Srch" a good approach to explain?

Applicn of Univl. D.f. to Machine Lrng:

1) Derb. QA problem: Say a great variety of induction problems can be put into this form. "Q" can be strings or nos of interest. Search for "A":

Formalization $\Sigma_1 \Pi \; O^*(A|Q)$.....

$\Sigma_n \Sigma_{A_{n+1}} O^*(A_{n+1}|Q_{n+1})$ $a_n^n = \Pi_{i=1}^{n} O^*(A_i|Q_i)$ — wt of O^*

To find good O^*

We use Lsrch in simpler way, guided by univl d.f. or O^* functions .....

We are able to make math proofs in approximate probability style

One may : Start w. trad. learning network;

Discuss Rod's not learning network used for induction; It used Learning; Rod said parity induction wasn't "MATH problem". Another possy: at first trials there was no evidence to believe that use of net for induction had any accuracy at all — it had same accuracy as to suppose that for long strings of bits, it was equally likely to converge to either value.

In 1968 I found a proof that it would converge very rapidly to either value. Simply stated: if there was any regularity in a data set, the universal distribution would eventually find it using a relatively small amount of that data.

"Eventually" refers to computer time, not amount of data used.

I called this property of the universal distribution "the completeness" property.

From here go into discussn. of the Nips talk: Perhaps discuss extensions of conv. theorem.

I'm going to start out with a discussion of the Universal probability distribution and some of its properties and how it is related to Kolmogorov complexity.

The birth of this idea was immediately connected with two events: First, my publishing a long report on Machine learning. I had discovered the universal distribution in 1960 — as a direct result of some earlier work I'd done on Machine learning and Chomsky's early — 64.00

1) Canonical UTM!
random input!
At time T:
What is probability that output will have prefix, x ?
What is probability that a stopped output will have prefix x .?

Kol Talk: Start: Many years ago — in 1960, I discovered what we now call The Universal probability distribution. At the time, it seemed to solve all of the problems of Bayesian Statistics. It gave what looked like a Universal apriori probability distribution on all possible strings of symbols.

At first, there seemed to be three main difficulties:

1) I wasn't really certain that it gave the correct probability values, or that it would work with a reasonably fixed data set.

2) The values of probability it gave were incomputable

3) The distribution itself depended on the choice of Universal reference machine or reference function.

I did have some ideas that gave upper bounds on error and dependence on reference function, but the bounds were far too large to be of practical utility.

Five years later in 1965, Kolmogorov discovered Kolmogorov "Complexity": The Kolmogorov complexity of a string is the length of the shortest program for a reference computer machine that will produce X as output

Kolmogorov Complexity of X is Length of S: S is shortest program such that $M_R(S) = X$

Kolmogorov was interested in this complexity as a way to define randomness and as an interesting mathematical concept in itself. My use of the idea of this sort for inductive inference came later.

I was surprised to learn of my application of these ideas to inductive inference.

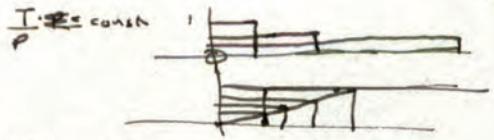
It seemed strange to me that Kolmogorov, who had in 1933 written the basic formalization that created the theoretical basis for modern probability theory — who had contributed so much to practical statistics — would not notice that his complexity could be used to approximate a priori probability, i.e.

$$P(x) \approx 2^{-k(x)}$$

Note that these probabilities are always integral powers of 2, so they are usually in error by a factor of about $\sqrt{2}$ — an extremely large error.

This may have stemmed from considering it as a mathematical rather than practical problem.

He did, however, publicize my ideas and so for many years it was much better known in the Soviet Union than in the rest of the world.



:62.40 worn on formal Grammars:— both were [redacted] about 1956

My main interest at that time and subsequently, has been in understanding inductive inference well enough to get a machine to do it very well.... Essentially Machine Learning at a very high level —— "Strong A.I.".

5 years later, in 1965, Kolmogorov complexity was discovered (what was called Kolmogorov Complexity Thought it was very closely related to the universal distribution, Kolmogorov was not aware of my earlier work. His interests, [redacted] in this area [redacted] was in defining randomness and understanding complexity itself. In complexity as an extremely interesting mathematical concept. He was surprised to learn of my earlier applications of these ideas to inductive inference. He did, however, publicize my earlier work and so for many years it was much better known in the Soviet Union than in the United States.

I'm going to start out by talking about the Universal probability distribution, and how it's related to Kolmogorov Complexity. I will discuss some important

I'm going to start out by discussing the Universal probability distribution, and some of its properties —— and how it is related to Kolmogorov complexity. Then I will tell about some work I've been doing in Machine learning that employs these ideas. Then I will talk about describe the application of these ideas to Machine learning.

Many years ago — in 1960, — I discovered what we now call the universal probability distribution. It is the probability distribution on output strings of a universal computer with random input. [redacted] It gave a probability distribution over all possible strings of symbols. It seemed to solve very serious problems in the foundations of Bayesian Statistics.

Five years later, in 1965, Kolmogorov independently discovered "Kolmogorov Complexity". The Kolmogorov complexity of a string of symbols is the length of the shortest program for a reference computer that produces x as output. Kolmogorov complexity is closely related to the universal distribution. If K is the Kolmogorov complexity of x then 2^{-K} is an approximation to the probability obtained by the universal distribution.

Initially, Kolmogorov was interested in this complexity as a way to define randomness and as an interesting mathematical concept. He was surprised to learn of my earlier work on inductive inference. He did, however, publicize my work and so [redacted] for many years it was much better known in the Soviet Union than in the United States.

For several years after my discovery, I wasn't really certain that the distribution would give good probability values. I had lots of heuristic arguments, but nothing very certain.

This all changed in 1968, when I discovered the conversational program.

→ Nips talk page 1 #2.

The CTS oracle

The time needed to solve a problem will be, T_i / P_i. Here P_i is the probability that GCPD assigns to the successful PST, and T_i is the time it takes that PST to solve the problem. For hard problems, P_i is too small and/or T_i is too large. — to solve hard problems by transforming them to easy problems. This is done by "Adaptive Lsearch." The idea is that we use solutions of easy problems to modify the GCPD. Lsearch cannot solve hard problems directly. — But it can solve hard problems indirectly by first making them small.

For hard problems the CTS will be impractically large. — T_i will be too large and/or P_i will be too small. Lsearch cannot solve hard problems directly.

.." It is too hard"....

.. Solvable by Lsearch

In the early education of the system, updating is done by modifying the GCPD so that previous solutions

In the early phases of education of the system (which I will call "phase 1") the system looks for regularities in past solutions and uses them for biases for future searches in view of these regularities. It tries to "what worked well in the past". It has no concept of "improvement" or "optimization".

In phase 2 — in a more advanced stage of education, the system tries to get the best GCPD possible the available training. — it really "understands" the goal of optimization.

Phase 2 updating is a problem for Oracle induction.

There is a tendency to avoid this issue by pointing to the uncomputability of the Universal distribution. — But this is not the problem. The incomputability of the Universal distribution is not the problem — the problem is that computable probability Lsearch is incomputable — it always has some of uncertainty.

2.08.03

Notes

401 lecture

Jump to p 101

pp 67–100 don't exist.

: The way it works: Suppose M(S) is a universal computer

* and S is a binary string, S acts a program for M. Then M(S) represents the output of the machine, after it stops. If M(S) = X, X being a binary string, then S is a program for X.

If we give as input to M, a string that started out with S — the machine would read S, output X and stop. — So if S was only 10 bits long, only the first 10 bits of that particular input are relevant.

If we give a random input to M, the probability that it will begin with S, is just 2^{-10}. If there are other programs for X, and they have lengths $L_1, L_2 \ldots$ then the probability that M will output X when given a random input is just $\leq 2^{-L_1}$ — the sum of all the ways X could be produced. If L_1 is the shortest program for X then 2^{-L_1} is an approximation to the Universal probability of X, with respect to M.

In 1965, 5 years later, Kolmogorov independently discovered Kolmogorov Complexity — it was the shortest length of the shortest program that could produce that string.

"Finally Kolmogorov . . . "

The way it works, M is a universal computer, and S is a binary string, such that M(S) = X — i.e. S is a program for M that produces X as output. If L_S is the number of bits in S, then the probability that S will occur as input is just 2^{-L_S}.

It seemed to solve all kinds of induction problems and seemed to be the ideal way to solve difficulties in Bayesian statistics.

2-19-03

Addition
Addithorne

Ideas to talk about

The MAN Group
Hedgehunt : 17% guarantee. "(calyon:initial(dep.al)

From 66.40:
pp 67–100 don't exist.

1) Model Uncertainty (01)^M example
2) Anthropic principle in [redacted] Epistemology
3) Discn. of (3rd) diffy w. ALP (≅ Univl. D.F.)
i.e. "Subjective Entity"
4) 2 measures of Simp : Also how Non-Bayesians insert simp into
5) look at things in Sol 86 : Also possibly Sol 78 ! ← for symposium.
6) How to Unmachine model helps one design a prob —
Helps unify disparate probs. (by having common theme)
7) OOPS : How it works. Then talk about 1^n 2^n → TOH start TOH first : Code of n loop.
Expl'n Boost : Eat n root-finder : took n steps.
8) Discuss of "Not a Big Beta Factors" Perhaps explain that when a Stat. Mech. Course covers only the error estimate due to SSZ, [redacted] (perhaps ruthlessly) omitting a truly / possibly very large error source of Quantum Size : The "model Uncertainty".
As a way to explain this, perhaps use Models #5 from Sol 6f2 : (Σ of all poms).
This would make it very Simple. Σ all semi-measures. finitely derivable computable measures—
very many models many measures
Diff to explain about the partial/recursive PEMs. Perhaps "review" (Formal) Prior properties. The Univl d.f. itself is a p.m. "PEM" (computable measure)
An easy way to think of the Univl d.f. as the wtd Σ of the "PEMs" assoc. w. each poss/.
Time.
One way to look at the PEMs : Associate a CB w. each & use Leech to estimate pc values. — So far some (maybe all) ones it never quite settles for finite t) as to what the pc is
Consider the Univl. DF as a PD. ones approximately : At any one time, never will be a set of computable poms that have been tried // Also there will be a set of p.m. being tried but have been tried w. certain CB's.

7:30 p leave
7:45 p
8:15
15
) stay sure,
9:11
6½ hrs
8:15 → 45
22:35
22:36
22:38
22:39

2.19.03	140 A.M.	220 6:19 4 1/12 (54)	100 6 1/2	14:40 (15)	30 2-3	1631 (42)	140 6-1/2	2214 (24) B	2.22.03	180 245
2.20.03		22.14 23:45	125 25 154 4-1/3 (34)		150 2 1/2 (60)	420	250 2-1/6	621 (115)		
2.21.03		245 2:42	266 437 4-1/12 (51)			10308 B				

2.23.03
PEMS

2.20.03

Kol FR

Thurs	Fr	Sat	Sun	Mon	Tu	W	Th	F
~~20~~	~~21~~	22	23	24	25	26	27	28
							Kol 530P	Sem. 10A

- Dec 30, Jan, Feb, Mar 30 — 3,4,5 seem to be Missing!

③ Accuracy
② Incomputability
① Subjectivity.

Idea of Kol talk! Do = NIPS talk, but add!

(Dad) 1) Discussion of real difference between (U.D = ALP) approach & usual Bayes & Non-Bayes. Go into just how UD is superior — how it adds ~~many~~ new P.r. PEMS. How it gives insight about what's going on.

2) Discuss "Subjectivity" of Bayes: ~ kinds of Apriori
a) Philosophical = single universal Apriori — I don't "Have any use for it" —
~~true~~ meaning — I don't like it, as if (I had it) I ~~would~~ have no application for it. (If it existed) applications.
Discuss "If you have no info"
① "If you have no arm to break: — it cannot be broken — not a universality problem.
~~Nothing can be done~~ = ϕ can be done — not a universality problem.
But β (no arm) occasionally occurs
α no info ~~...~~ it may occur if it did create world quickly, ~~create world did not occur quickly~~ creature would quickly dies.
Living creatures always have a prior.

Where it comes from! —
The / ~~Auto~~ weak Autograph purveyed in Physics. (we know some restrictions — on what is possi. — but very uncertain)
" weak Autoed P " " epistemology (less restricted — more possible
viable ways — but I don't know how to characterize them — (use of logic, languages, instructions to insert info; ~~...~~ all computers (computer languages) Lisp, APL, JAVA, (postscript), PEARL, (Reason), Boltzmann sorted-based)

③ 4) Discuss OOPS More. ⑤ Maybe discuss TSQ

Kol. "Seminar."

Discuss Phase I v.s. Phase 2 ! Go into more detail on How OOPS & phase 1

Discuss Eqns for Phase 2! Show how both positive & negative cases are dealt with.

Discuss future problems: "Long duty Search" Joint def. on all PST's?

What could it mean? It isn't a joint Def. on PSTs being "Best" but on pc's of

Soln. at times T_1, T_2 (good!) from this, we obtain changes during working of problems. A big Q is how to do dimensionality of search space: Cross correlation & clustering are 2 simple ways.

32 {
34 { Actually, the fact that pst_{13} has solved a problem $prob_7^{(sop)}$ at $CB=T_1$, can much modify ~~pcs~~ def's of other PST_s's on $prob_7$. ← (It I immediately update when soln does not occur when T >> expected —
Solving would give "long" in a big way). Say we expected to solve prob7 w. PST_{13} at time T_0 & at $T_1 = 3 \times T_0$; no soln: This fact could modify considerably, e.g. of soln. of several other PST's — (If we did a "local update" when 32–32 occurred

2.20.03
Keith

2:42 start processing
2:12 ...

3

Becky called 1708 : will be home ~ 11 AM
Thinking about ferris wheel;
says there are several related play places
near our hotel,

$a = 0^{th}$ moment
$a = 1^{st}$ moment / z

σ^2 maybe relevant

00 : SN Looking at the μ, z of the PST3 wrt Proby ; how long is soln. not further ... before we decide that (μ, z) are proby wrong? If $EB = T_1 = 3 \times \mu$, say, or even just $> \mu$; we suspect that this thing failed : its $pc = (1-z)$.

It is not clear that even if $T_1 >>> \mu$, but if $z = .15$ or less, that we really didn't know any thing new (much).

11 PM Gone

We should get much more info if z was closer to 1, & $T_1 >> \mu$.

— A possl. result of this Analysis: we would try to get PST params w. "z" close to 1 if possl.

10

?? [Yet : it would seem that failure for $\left\{ \begin{matrix} PST_{17}, Prob7 \\ z = \frac{1}{2}, T_1 >> [illegible] \text{ to } \mu \end{matrix} \right\}$ should say a lot about PST's "similar" to "PST_{17} wrt prob7"] ??
("close to")

If this failure occurred — proby we would do normal Lsrch thing & go to next "try pc" trial .... Should we somehow have noted failure & used it to "modify" z_i's of "nearby" (highly correlated) PST's, ?

17

→ 4.10 / 4.00

SN In the z, μ approxn. of $h(T)$ distribn:

20

T_1 "proby of Bayback" was ordered by $\frac{z}{\mu}$. — proby NOT true: see 4.02 ! — Rather use z

relatively simple formula to get the proby & ordering, but I think it could/should be possl. to simplify it further, to make it easier to implement. Consider only 2 $\mu_1 z_1$, $\mu_2 z_2$:

Say ordering of μ's is μ_1 μ_2 (such) (larger) ; proby of μ_1 winning is z_1.
" " μ_2 winning is $z_2(1-z_1)$

proby of either winning is $z_1 + z_2 - z_1 z_2$. (a familiar expression!).

30

We could normalize but for Lsrch, this is usually unnec. — it normalizes itself.

If we have a bunch of $\mu_i z_i$ pairs, the only relevant thing is ordering of μ_i, & z_i's assoc. w. that ordering,

35

— so essentially an ordered list of z's. (μ)

z_1, $z_2(1-z_1)$, $z_3(1-z_1)(1-z_2)$, $z_4(1-z_1)(1-z_2)(1-z_3)$

p_1 $p_2 q_1$ $p_3 q_2 q_1$ $p_4 q_3 q_2 q_1$, $p_5 q_4 q_3 q_2 q_1$

~~z_0~~ z_1 (initial) $z_2 \frac{(1-z_1)}{z_1}$ (factor) | $\frac{z_3}{z_2}(1-z_2)$ (factor)

The W in probs can be > 1, but if W is very >> 1, ...

2/20/03
Keith

note $2\frac{1}{2}$ L/hr 25L/hr ÷ 4 d/hr ⇒ 80 K/hr.

DO: Test if production p's & q's should be done for PST w. max $\frac{\#P}{M}$ seems unreasonable!

But I had a bunch of PSTs w. same $\frac{z}{M}$ & they ended up all w. same prob. of "win".

If $\frac{z_i}{M_i}$ = constant, will $z_i'\left(1-\frac{z_{i-1}}{z_{i-1}}\right)$ be constant? No reason for it to be — so PS results w. .01 more not general — they were true in M12 one case only!

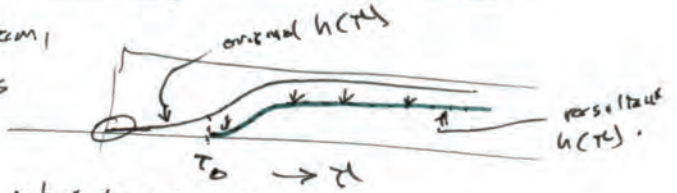
While $\frac{z}{M}$ ordering may not be true, it does have some heuristic wt. & I decided to use it for approx ordering of PST's. — Look at my reasoning for Prob 3.

It was partly the "max pc of win/permit runs" — but I think there was another "proof"/"reason".

∴ 3.17! If PST$_1$ w. M_1, z_1 has failed for $CBST_1 = 3\times M_1$, Then PST$_2$ (say) which happens to be highly correlated w. PST$_1$ (wrt to same problem), will have its M_2 moved out to $\approx 3\times M_1$ at least.

T. reasoning here is not 100% sure, but I think we are sure now that PST$_1$ will fail for times $\le 3M_1$. (probty = 1).

Syst. $h_1(T)$ corresp for $T \le T_1$, was



If we work Prob$_1$ for time T_0, what does the curve then look like?

It should seem that it would be ϕ for $T < T_0$.

If we look at $h_1'(\ )$ may be its clearer.

Looks like $h_1(\ )$ curve is just toward so for $T \ge T_0$; $T < T_0, h(T) = 0$

$h_1(T) \to h_1(T) - K$ $\phi = 0$ for $T \ge T_0$.

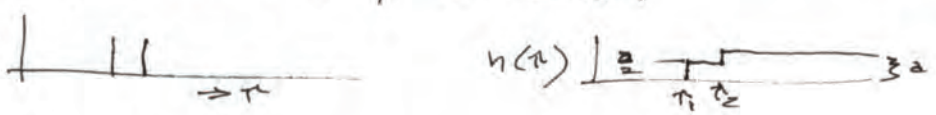
This for convex true also, for any other PST highly correlated w. PST$_1$, wrt Prob$_1$

T. above results (-10 off) also for $\sigma^2 > 0$, if $\sigma^2 = 0$, then if

→ PST$_1$ fails for $T > M_1$, then it has failed completely.

N.B. T. for. discussion. looks related to WON!

If we approximate $h(\ )$ by 2 steps:



If we fail in $T_1 \le T < T_2$ Then $h = \frac{a}{2}$;

" " " $T \ge T_2$ $h = 0$.

We can approximate $h'(T)$ by any no. of δ funcs.

TAXONOMYS Needed, Of much Utility!

(1) Functions: D. Funcs | Ranges, Domains | Funcs vs. Relations | P.R. Funcs & Total rec. funcs, P.P. Funcs | Methods of representing above things (recl. back to "Data Structs" Room).

2) PST's: for various pubtypes: INV or Optza General. (Including Time/Varying).

3) Coding Methods: Maybe closely related to S funcs. Compression Techniques

4) Regularity types.

Main approach: How to Create all poss. n^{th} level functions:

Zero level: Constants & inputs.

Functions: N_k, k any funcs. (k inputs, 1 output) | say # k = 1 to 2 (or 3).

Say we have M(l) new pts at l^{th} level.

" " " $\sum M(l) = S(l)$ total no. of pts at level l. $\quad S(l) = \sum_{i=1}^{l} M(i)$

$M(l+1) = \sum_k N_k \left(S(l)\right)^k \equiv \sum_k N_k \cdot \left(S(l)\right)^k = \boxed{\sum_k N_k \left(S(l)\right)^k}$ ← a very rapidly growing function!

So: If looks like the "Concat not" diagram S89 was wrong a/o not applicable to compositions of functions.

Consider Compositions of functions as done in AZ. This may be an Alternative Way to construct all funcs of a given "depth":

To use the Concat not Graph in Solgg: Each distinct n-tuple of cases combined would have several ways of being combined: Which would be correct, but in general discussion, would be as if was originally, & would be correct.

: More detailed inserts for Kol Talk

slide 1.5

Insert 1.① [redacted] $X = M(S_i)$

$L(S_i)$ = length in bits of S_i (no. of Bits)

$2^{-L(S_i)}$ is probability of input S_i to $M()$

$P_M(X) = \sum_i 2^{-L(S_i)}$ is total probability of all programs for X . . . .

[redacted] $P(1) = \frac{P_M(X1)}{P_M(X1) + P_M(X0)}$

? ~~Shall L mean "it includes partial recursive funcs, as well as recursive funcs."~~ ? No.

slide 2: add: Converges faster than $\frac{1}{n}$.

Advantages of OOPS ① it really is considering P.R. Finch
② "boost" can give 2 mutations & 1 "crossover". (2 boosts is expensive: $pc \approx \frac{1}{n^2}$; n = no of pgms "frozen Merry".
③ It could give 3 way crossovers if he used a "milder" Boost".
[w. large Capacity computer, 2 or maybe even "3 boosts" might be considered.

20: ::: Discussn. of Mechanics of incomputability.... { This part might come when I first Mention Incomputability, If would also explain why U.D.F. is "superior"
(Go through of talk, see if there is time for this).
The showing how U.D.F. can create things not considered by other stat techniques seems important.
This may take too long to do it effectively.

Well, continue on other topics: 1) Subjectivity Previous aspect.
2) OOPS. 3) TSP's.

0: 1) Subjectivity: Comes after incomputability discussn; better start of L's each discussn.
Meaning of A priori:

Data: P.D. can't be created once data seen has been analyzed.

14 The D.F. changes during the problem solving. As soon as problem is solved some data is used to solve a problem, the A priori is "updated" to reflect that solution.
So we have a continuously changing a priori Prob. distribution that reflects the life of the
The problems solved during (this) life.
A philosopher may ask: is there not a Universal a priori probability distribution — in which you have no prior information?
Suppose I have no a priori information?

10: Well: let me give an analogy:
If I had no food, water or Air to breathe, what would I do? — very little to be done, I would quickly die.
If I had no a priori info there would be little that I could do to solve practical problems. Fortunately, we don't get into this situation: we are born with fairly good a priori knowledge of the world we are to live in. This a priori info enables us to learn to walk, to talk and communicate — which would be very unlikely if we didn't have this a priori information.

The nature of the a priori info is not difficult to Get a person has, it is difficult to characterize.
Universal Probability distribution
29 How we normally have to use only part of our a priori info. For a specific problem, → 8.02
10: 8.14 we can usually decrease. Machine languages that are used generally (UTM), relevant to our problem.
These machine languages will have subroutines or subfunctions in them that have been found to be useful in the fields of that problem. E.G. fortran, Lisp, APL, Java.
APL has math — enables very short programs to express very useful Sci and the functions, Extensive libraries; Maple & Mathematica & Matlab have sets of functions particularly useful for certain problem areas. — In general, universal machines can be furnished with features that we've found useful in dealing with problems we perceive to be the domain of our present problem.
What we perceive to be the domain of our present problem.
So we can use our choice of features of our / Universal / machine to express our personal a priori.

:30 : So we can use a choice of Universal Machine to express our own prejudices

:01 apriori.

:02 : 7.29 We may have strong ideas as to what functions would be useful to solve this problem. In this case we would augment the construction set of a universal computer, with those functions. If we are less certain of what functions are needed, we might use a set of instructions that has been developed for empirical/general prediction methods — Such sets of instructions are available in libraries or packs of Macintoshes, Maple or MatLab. The more distant the machines used — If the instructions in set are not relevant to the correct probability function, then the code length for the correct function will be very long and slow probability — $\pm 1/n$ so will the convergence rate will be slow. — the it will converge, even really, to the correct values. → For large n the convergence theorem tells us that the errors in probability will converge more slowly to the correct values — But they will converge.

:14 22.22

SN Augmentation of discussion of "Accuracy" of U.P.

:17 [insert at P3.09 : It would be well at this point to take a big more a back errors in prediction".

:20 In statistics, a prediction method for empirical data will have 2 types of errors!

First error due to finite sample size. Say I have a 100 bit sequence of mostly zeros binary sequence and I want to know the probability that the next bit is a 1. I would just count the # of 0s and 1s — Say there were 50 1's and 100 symbols altogether the probability of 1 is about $\frac{1}{2}$ and the expected error is $\pm \frac{1}{\sqrt{n}}$: n is the number of symbols so our uncertainty in the probability of $\pm$ would be 1.0.

This, however, is only based on the data — a simple Bernoulli model.

:30 If I looked at the data more carefully, I might notice that the sequence is "01" repeated 50 times. A better model for the data would give close to zero for the probability of 1 being the next symbol.

In the first case, we had rather poor prediction. The first error due to finite sample, was only about .05, the error due to incorrect model was close to $\frac{1}{2}$. This is the second kind

In general, the superiority of the universal distribution is in its treating model uncertainty in a more realistic way than do other prediction methods.

2.22.03 [redacted]

Kol. Th.

109 9

Perhaps in A.I. ½ of progress is in SW, ½ in HW!

Maybe for Seminar Talk!

[redacted]

20: Model Uncertainty.

List of items I want to include, that I should "write up":

1) On "Subjectivity (Insert 22 3.09)" "accuracy" : 8.17 – .40 : Meaning of SSZ v.s. Model uncertainty

Give [redacted] GW^to example.

.06 2) "Subjectivity" 7.10 – 8.01 a) Defn. "Info before data is seen" — Info before data is (known / seen)

b) Q : philosopher : zero data --- then what? (NFL Theorems — No Free Lunch) analogy to "no Am" → no Rug.

[redacted] How mature form w. apri info. Talking, walking, analysis of visual/acoustic scenes.

0: Inserting info into U.D.F. ← it's hard to describe, don't go into detail. | C++ Libraries.

Maybe mention C++ libs, Fortran, Lisp, Maple, Mathematica libraries of functs, ALP as compact way to represent math.

In Living creatures: where does a priori come from? (Organic in response to Environmental evolution) — Inter-action analy. between Physical & Biological entities.

Idea of evolution! Also w.g. suitable "TSQ" of events, Challenges, by intelligence would not have developed.

16 "S.S. Anthropic Principle" in Epistemology ← perhaps Rapid "Throw Away" line.

3) OOPS: First discuss Phase 1 v.s. Phase 2. OOPS P 1 & describes TOH

0: propose a bit for Kol lecture; More detail for Seminar. " " 1 & " " $1^n 2^n$.

possibly explain of [redacted] "Boost"

Make list of insts used in $1^n 2^n$; TOH: How Boost was effective.

~~Initially Boosted for both: (~~ for ($1^n 2^n$; ~~...~~) (C1, C2) (C3 C4 C5 for TOH) ~~)~~

~~$1^n 2^n$ soln defnp C1 callfp C2 endnp~~

6^{10} =	6 = 2.3
12^{10} =	4.7
e^{17}	$\frac{17}{2.3}$
36^5	
$36 \cdot 38^4$	
$= 36 \times 10^6$	
3×10^7	

Init boost for both: [byz, dec, boostq]

+ Boost for $1^n 2^n$ only C1, C2 — " Boost for TOH only C3, C4, C5

10: Soln of $1^n 2^n$: [defnp, C1, callfp, C2, endnp] 5 insts.

soln. of TOH : [C3, dec, boostq defnp C4 callfp, C3, C5, callfp, endnp] 10 insts

Only used boosted insts.! So from original pc = $\frac{1}{73}$ per inst → $\frac{1}{7}$ or $\frac{1}{12}$.

2.22.03

Kol·Th

110

OSL .00, but .02-.06 looks much more promising!

One that Lrng

00: : [SN] Could I list various ntups or functions on the "agb list", but on relatively low pc's. These would be functions that have executed only once, say, & are somehow to be used in OSL.

02 Tho, normally, the way OSL is implemented: A part of a function occurs. Then we look in corpus, to see if that part has occurred in the past. — if so, we can try OSL, to see

04 if it's o.k. Qes alternatively.

(NB) .02 looks like L-Z compression routine: so we might be able to devise a

06 Very fast implementation! Nacy to apply the routine to Function traces, subtrees, hier. would Polish Notation" cover much of it?

10

12 OK What Happened in OOPS: (How TOH obtained factor of 1000 in speed, from solving $1^n 2^n$.)

T. initial boostop byz, dec, boostq. ~~made these 3 insts~~ Even these 3 insts almost all wk. Then boostop C3, C4, C5 included Occurs in the "effective search insts" So we have predict "C3, dec, boostq" fairly likely: The new additional boosted insts are now [defnp, C1, calltp, C2, endnp] ← successful succ corresp to $1^n 2^n$.

from this, the sequence [defnp, C4, calltp, C3, C5, calltp, endnp] has an acceptably

20 by pc. succ rec. seq to T.O.H.

defnp, calltp, endnp and the numerics, C1 .... C5, byz, dec, inc could be used to construct fairly complex recursive functs.

Note that "defnp" & "endnp" always come in pairs, at the start & end of a def defn.

— One should be able to build this back into the Lang. used by the OOPS.

Examples for discn. of model uncertainty,

26 → Let me explain: ~~But~~ But first a brief discussion of uncertainty in prediction.

into 2.27 There are two kinds of uncertainty in statistical results: The best known is the uncertainty in probability values due to finite sample size. If you have n bits and half of them are zero, then the probability of a 1 being

30 the next bit is about $\frac{1}{2} \pm \frac{1}{2\sqrt{n}}$. The larger the sample size n, is, the less error in our probability estimate.

The kind of uncertainty I'm going to talk about is not due to sample size but due to "Model uncertainty." For when analysing empirical data, there are normally an infinite number of models that can be used to analyse the data — some will give good predictions, others will give poor predictions. In any finite amount of time, one can only evaluate a finite number of models. From the Universal ~~Study of Pro~~ Universal distribution tells us that whenever we find a model

2→12.00

2.22.03
Kol. Th

111

Things to be typed

for §1.06: Suppose we have a ~~finite~~ string, x and we want to know its universal probability with respect to machine, M. There will be many inputs to Machine that will give x as output. Say s_i is the i^{th} such input. If s_i is $L(s_i)$ bits long the probability that s_i would be produced from a random binary input is just $2^{-L(s_i)}$. To get the probability that x will be produced by any of its programs, we sum all of the probabilities of

$$P_M(x) = \sum_i 2^{-L(s_i)}$$

from 0 to ... → 13.00

(Hoyle and Shaw Slide S1.5)

~~To do prediction with this, we would have to fairly sample.~~

~~If x is a string set~~

for §. 2.27 after "feature": 10.26 − 1.40; 12.00 − .03
(explains "Model uncertainty" abit)

I am not sure how applicable this is to inductive inference.

He did, however, publish it prior to my discovery and solved many years it was better known in E. S. U. Russia

2.22.03
Kol. Tk.

12

→ In a finite time one can only consider a finite number of models.

10: [redacted] /0.40 That seems to fit the data well, we can't be sure that there is not another model that will give much better predictions. There is no way to avoid this. The incomputability of the Universal distribution is related to its evaluation [redacted] all possible models and taking an infinite amount of time to do this.

03

3.26 of talk: Discussion of Subjectivity

[redacted] Not only a need [redacted] Bayesian statistics be subjective, but it is desirable that any other methods of inserting apriori info, be subjective.....

10 List a few Desires: ① vs. accuracy ② no need to [redacted] divide data set into training set and test set ③ Never problem of overfitting or underfitting.

[redacted] Apparent disadvantages:::: ④ Data need not be "stationary"

① Incomputable, ② Subjective

Advantages: Includes pr [redacted] models. [OOPS is only pgm I know that searches over p.r. Models. Some G.P. systems Might.

For Random version of Lisrk, Use 1 shot M.U. to get fixed time Spent on each trial! Usually in computers, setting a fixed time interval, ensures known cut off, is diffic't.

20 for 4.31 of kolTalk. ~~When people see the expression~~ ~~T 2^L, they often become frightened.~~ → When T 2^L is the CTS of the problem, the time needed to solve it by search.

People are often frightened by the expression T·2^L. If it says that the time is exponential in the size of the solution. for many people this is equivalent to unsolvability. However, in A.I., the sizes of [redacted] solutions are usually quite small, and the dicotomy between "polynomial" and "exponential" one is not a useful [redacted].

30 If all the information and bias is in the probability distribution, and the only way you can control the search is by probability and time of trial, then [redacted] about the theoretically best way to order trials is $\frac{T_i}{P_i}$, CTS order, : smallest first. This will give [redacted] least expected time for solution. [There is no better way to solve the problem. [Search is a good approximation to this optimum technique.

40

2.24.03

113-13

2

→ To repeat, $P_M(x)$ is the sum of the probabilities of all programs (for machine M) that could have produced x, when run on M.

00:11.07 : ~~To use this distri~~ It is easy to use this distribution for prediction: If x is a binary string, then the probability that 1 will be the next symbol of x is just:

$$\frac{P_M(x1)}{P_M(x1) + P_M(x0)}$$

: $P_M(x1)$ is the probability associated to x1 by the universal distribution using machine M as reference.

2½ min

~~The accuracy of the universal distribution~~

~~Associated with the accuracy of the universal predictions using the universal distribu~~

09 While the accuracy of the Universal Distribution as a predictor is certainly ~~not~~ an important feature, there are other important features:

~~Some other important features:~~

- We can do prediction on non-stationary time series.
- ~~This technique finds~~ We obtain an optimum solution to the overfitting (underfitting problem

(20" Because of "optimum fitting", there is no need

- ~~If it is not advisable~~ [to] to divide data into "training set" and "test set".

~~This can create expected errors~~ We obtain very good error estimates for the future by analysis of the training set alone. Since we do not assume stationary data, we obtain a probability distribution for the future that is not necessarily correlated with the past.

~~There is no universal program~~

- It is possible to use partial recursive functions to model our data.

To my knowledge, no one has actually tried this — but the system I will describe later, will do it. Whether it gets better results than using only recursive models, remains to be seen.

0 While all of these are very nice, there seemed at first to be a serious problem: — ~~That the universal~~ The universal probability was incomputable.

~~The way I used~~

23: (insert 14.04) One "easy to describe" version of Lsearch ~~describes~~ exhaustively tests all possible candidate ~~solutions~~ X's in parallel — by time sharing. If the universal distribution assigns P_i to ~~the candidate~~ test that string, then it will give a fraction P_i of the time sharing ~~to that candidate~~.

0 While this version of Lsearch is very fast, it uses lots of memory. Another ~~slightly slow~~ somewhat slower version of Lsearch uses much less memory.

~~The problem with this is the~~ The key problem: the time it will take ~~to find~~ before a candidate solution is found.

1	6'
2	2½
3	2:50
4	(5:10 1:40 18' = 2
5	2'40
6	(how? 2'10" 2'45"
7/8	6'

31½ min.

↓ I may not want to include this — I might go over 4 when I introduce PST's

PST: Given a problem, a PST is any algorithm that attempts to solve that problem. The domain of a PST, is the description of the problem.

~~The domain of the PST, is the description of the problem.~~ For ~~certain~~ inversion problems as we discussed before, in which we are looking for a string x, such that the Given function f( ) gives ~~it~~ know. In version problems are described by a function, F(·) and a string or number s. For example, The problem is to find a string or number x, such that F(x) = 5. for example, the problem to solve sin(x) = .5, ~~where x is~~ . If F(x) = sin x and s = .5 While search itself is ~~a problem solver~~ one kind of PST that can solve Inversion problems — but there are many other PST's that can solve ~~inversion problems~~ these as well

~~Another kind of PST~~ Optimization defines another broad class of problems. Given a function G( ) that maps strings to reals; to find a string x such that G(x) is as large as possible. In an important subclass of optimization problems, ~~Solution~~ it is necessary to find the best x possible in a certain time limit, T. The problem is then defined by the function G(·), and the time limit, T.

There are a great variety of PST's that have been developed for optimization problems ... Artificial Neural nets, Simulated Annealing, and Genetic Algorithms are a few fairly general PST's of this sort. 2min

36 [Insert at 8.32] On OOPS! A big difference between OOPS and my "Phase I", is that OOPS has been actually programmed, and an interesting result was obtained, showing the utility of training sequences. In the particular set of problems it solved, there was a factor of about 1000 speed up in a solution, — depending on order in which problems were presented to the system.

2-24-03

Seminar

10. (1) Read 1 part of abstract on How system works. ¶'s 5, 6 & perhaps 7. (Abstract taken from NIPS workshop Hand Out) — 0

0 → 1) Define Inv, Oz probs, (from Sol 86?) | Equs on slides

a → 2) Desb. 1, 2 or 3 kinds of Lsrch to solve [redacted] INV, OZ probs.

b → 3) Discuss Phase 1 update: use of freqs of tokens & of usd pgms & usd subfunctions, subpgms. — α

~~3) Problem is to ↓L of soln; That "Update" is suppsd to do this.~~

c → 3.5 Desb. soln of Operator induction ~~w some detail.~~ — β

Give **Equs:** In Both forms

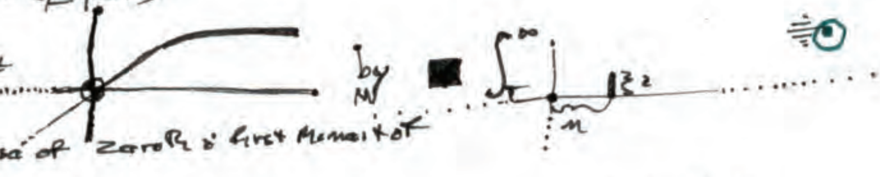
$\sum_{j} a_n^j \prod_{i=1}^{n+1} O^j(A_{n+1}|Q_{n+1})$

0

d → 4) Go into Phase 2 update: Show detail d equs both for h and h' (i.e. negative as well as positive cases). — γ

e → 5) Give "Integration Equations"

6) If we approx



~~space con~~ use of Zeroth & first Moment of

d.f. Shortest Soln.

10 ...... We can quickly obtain prob'ty of each D.F. leading to [redacted]

better approx by use of 2 point or 3 point D.F.

(73 insts) Optimum (Ordered problem Solver.) → Desb. inst sets

Smalltalk(?) Functional(?) Forth v.s. LISP ← similar ways to [redacted] — δ

7) [redacted]OOPS: Differences & Similarity to "Phase 1".

The 2 problems TOH & $1^n 2^n$. | See 10.11 for details on OOPS pgms.

Tried TOH for 2 wch: no [redacted] general soln.

Tried $1^n 2^n$ got soul. soln after $n = 1, 2, 3, 4$. (recursive soln)

Tried TOH & [redacted] soln for $n = 1, 2$ then recursive for $n = 3$.

30 How "Boost" works mechanics

" wa 1st kind of "Mutation", but usa Lsrch rather than random choice w. replacement.

2 Boosts could give crossover — ~~but is unlikely~~ — but its rather wasteful because many cases will end up w. identical defns.

Differences betw. Phase 1 & Oops in TSQ Construction. (perhaps)

Maybe not so wise to give details on Many problems! Best to give details on a few problems.

2.25.03

116 16

Models for kol talk.

P1 Convergence Theorem: If there were any regularities in a data seq:

Ex. Univ. D_F. would eventually predict — using a relatively small amount of the data it would give about the best possible probability values for a given sample size.

Proof of convergence theorem was for sequential data — like Time Series

Analysis. for normalized models:

Later Peter Gacs showed . . .

Various Hutter

- more general sets of strings

Recursive Enumerating system.

Discuss Kol's not having noticed the application — complexity to inductive inference.

Discuss Criterion for Goodness of induction:

1968: Wallis paper.

How it was very exact, reconstruction, development of earlier paper.

rigorous?

win 98 · double click → Term discovery of proof of goodness of induction.

Proof for time series induction using Norzd M univ.d.f.

Peter Gacs should cover. Term true for unnormed Univ DF.

Here I have Univ Df. accepted interp

Mathematicians happy — Nature accepted Normed Univ D.F.; rejected

accepted

reason — Lewin

But in general, while there may be no "best" Normz D.F. — any one

sense of Prom is very much better than the unnormed

sense it is derived from. The Normed measure

from a practical standpt., it's always better to use the normed

measure for people.

Recently Mr. Hutter discuss: Arby alphabet,

Various different loss funct. — one of them was of much utility.

Discuss Kol's not having discovered Application of

Use name Ray Post Pydoye

Host name → Santre Unix

from Santre.

If: server has

users office

Acct-name:

ray

ray@santre

2.25.03

for p18 = 5th

~~17~~ 17

Discussn. of Incompatibility. — leaves as is.

6/19

118 missing?

insert:

Inc. is property of ~~probability~~ summary probability itself!

That it is not a small effect! That we waster how many models you've tried

That [redacted] there may be another model that is significantly better —

But you would find of you spent 10 minutes more searching!

Subjectivity — [redacted] Not all statisticians are Bayesians

~~7.10~~ 7.10 — 8.14 9.06 — 10

06 : **OOPs**: Compute ratio of pc ↑ for $1^n 2^n \to TOH$ v.s. TOH first as a function of no. of insts; say 15 insts actually used in trials: so 73−15 = 58 not used. so let N_Q go from 73 to "15" & see how this ratio ↓ from 1000 down..

b Normzd Unnl. P.D. Go over this argt. that Normzn Constant for Unnl. P.P. Must $\to \infty$. This now Seems supposed! i.e. If we have μ = normz measure = $\lim_{n\to\infty} \frac{P_n}{P_n(U)}$

P_M = unnormed semimeasure then $P_M < k\mu$ iff μ has a limiting dens.

If t. normzn constant is bounded, Then ..... Well it can have diffrnt Bnds for every diffrnt. M. say t. pc of M w.r.t M was P_0, normzn constant could be P_0^{-r}, "r" a real >0. so "length of unit" of normz constant is r x length of $(\log_2 P_0)$

6 A large normzn const. means that + but "very often" as "U"; but we know $\lim_{i\to\infty} |P_i(U)^2 \to 0$. However μ never has M's. The Wierd thing is that if t. normzn const $\to \infty$ would it P_M be >1? No! t. normzd P.D.

I think it is true that t. Normzn constant can be very large (Solovay): in which case, t. normzd P_M can have much more rapid convergence to μ than t. semimeasure.

But I really have to go over this!

5 It seems to say that $-\ln P_0$ (in t. convergence) approaches $+\infty$! and clearly can't be >0!

Discussn of my initial search for single Unc.

At birth, all living creatures are provided with an apriori [redacted] from Heredity, Envt. determinacy ability of creature to live to solve many problems

In more intelligent animals, neonates are more helpless — need much learning by parents, rest of envt. Less intelligent creatures are born with more of needed behavior pattern at birth ("instinct") based less from learning.

— probably
— prefaces modifiability by learning.

---

other "better known Theories than in V.S. (Page 1 near bottom)

insert at 1:36

15 → I was puzzled that Solomonoff had not thought of using these concepts for inductive inference. ~~Leonid Levin, who was one of~~ He had certainly done very important ~~his stud~~ work on probability — both basic theorems that ~~the same validity a reasonable basis for~~ One of his first great works was the axiomatization of probability theory — in addition, he did done much other work on practical applications.

I asked Leonid Levin ~~why Solomonoff~~ how it was that Solomonoff, who was one of his students at that time, had missed his important discovery. Leonid suggested that inductive inference was at that time, not a really "mathematical problem." I wasn't much satisfied with this idea. Thinking about it later, it may have been that at that time there was no good general definition of induction — no general criterion for how good an inductive system was.

~~I have thought about~~ After my discovery I tried to find a good criterion for the accuracy of my prediction method.

Around 1968 — 8 years after the discovery, I finally thought of a good criterion:

Suppose we have a probabilistic algorithm that can be described in a certain finite number of bits, and this algorithm produces a long sequence of symbols according to its probabilistic rules. Then we have a general induction system that gives probabilities for each symbol, in terms of the previous symbols. For a good general induction system and a long enough sequence, the probability given to the symbols by the 2 different methods, should be very close.

While the formulation of this criterion seemed reasonable, I was at first unable to prove it.

In 1968 I was asked to review a paper on Inductive Inference, by David Willis. Though I was familiar with his ideas, I found the paper difficult to read. It took me about 6 months to really read the paper properly.

Willis had taken my proposed system for induction and made it into an exact, rigorously defined system. He had an error criterion it satisfied, which was a kind of error criterion — but it was certainly not good enough to convince anyone that the system was always good for prediction. He showed that the average ratio of the correct to the correct probability to the estimated probability approached zero as the length of data sequence increased — the individual probability ratios could, however, be quite large on quite small.

I was then able to improve this result to show that the expected values of the square of the differences in probabilities between the correct and the estimated values approached zero as fast as $\frac{1}{n}$.

I called this the convergence theorem.

2.26.03

12 | 21

Ray>

really made it clear that the universal distribution & are very good probability estimates.

Perhaps show eqn.

Eventually I sent in my stray manuscript recommending

that Willis paper be published with no revisions — but that other 2 editors who had really quite expertise in this had rejected it — Ray felt that it had little to add to my original paper! As an aside: I later found out who the other 2 reviews were. — Ray were both very expert in the field of the paper. I wrote Willis telling him what a great paper it was and suspecting that he sent it to another journal. He did & it was published 2 years later.

Willis paper may have been a bit ahead of its Time:

About that time, the interests in the mathematical community were in what one might call "True prob

My original convergence theorem was for a Normalized Measure on sequences of symbols.

Many mathematicians were unhappy with my definition Normalized Distribution.

start 26 20

Introduction:

I'm going to start out with a description of the Universal probability distribution — some and some of its properties. — Then a brief description of a system I've been working on. In Machine learning — a program designed to take advantage of the described features of the Universal Distribution.

2.26.03
Wed

122 22

after incomputability.

00 ! Insert on Subjectivity. 4.27

Andrew [apparent difficulty with the Universal distribution] is its Subjectivity.

Karl Popper.

When the Universal distribution is mentioned, there are 2 possible meanings of "Universal": First that the error will converge to zero rapidly if the algorithm generating the data has a small finite description. — This is true for all such generating algorithms.

This is what I mean by Universal Distribution.

0 Another interpretation of Universality is that we can use the same Universal distribution for all problems. This is what is called a "half Truth" — The Universal distribution will work for all problems, but for most, it will work poorly — the errors will converge very slowly.

When I speak of a priori distribution or a priori information — what I mean is information available before the data in the problem is known. The universal distribution, and the information it contains, changes during the life of the statistician. → 7.14 - .29, 8.02 - .14

22 So? The subjectivity ... the fact that it is based on choice of which Universal machine to use — is characteristic of all prediction systems based on a priori probability distributions. The choice of universal machine and its instruction set is a necessary parameter in the system, that enables us to insert a priori information into it.

So ~~this should be~~ The dependence of the universal distribution on choice of machine is not a Bug in the system — it, too is a Feature!

2.26.03

123 ~~25~~

20 : Addition.

For Seminar: write a bit about OOPS. Just structure, don't do. See p 15 for More on Seminar, including OOPS.

for End of Kol talk:

Derb. Motivation for inv / o2 probs: Rest Maxwell, Simon only find small class of inv. probs. — Rest inv + o2 cover almost all problem types

Every workon Inv as a probability!

(1985 All reasons Not Progress relevant to A. I.!
How they move ≈ nice. (But even that was wrong!)

How The "Updating" of which PST to use, also how to work new PST's: U. Avl. DB can't be used to get a good solution

How best to update

How previous Methods of Updating just mixed to 'do it what worked best" — Rest we can actually get Machine to search for optimum ways to update.

All these Goodness are applicable to Univl. D.F.

(probly we never Plan out of).

For Seminar: More on OOPS (See n PE on Reg)

2.27.02 | 50 1:47 | 170 + 43 / 3+13 (58) | 170 749 / 3+0 (55)

Q: in Phase 2 update: any way to use my approximate 2 param h fcn's to use Negative data ?

2.27.03

124

Many years ago — about the time I found out the Universal Distribution, Newell and Simon devised GPS — GENERAL Problem Solver — a program that was meant to solve a great variety of problems.

In fact, it only solved a small subset of what we call "Inversion problems". But perhaps its most important defects were that it that it didn't learn. had no concept of probability and it was absolutely unable to learn

NS .04-.06

My own interest has been in a much more general type of problem solver:

The system I

The AI community was pretty much taken with the Newell and Simon approach and for many years there was relatively little work in AI involving learning or probability.

About 1985 at an annual meeting of the American Association for Artificial Intelligence, a vote was taken and it was decided that probability was in no way relevant to Artificial Intelligence!

The next year a protest — A protest group quickly formed and the next year there was a symposium at the AAAI meeting devoted to "Uncertainty in AI". — Most of which involved probability and learning — This became a many and

This symposium continued to be a yearly event.

As part of the protest, I gave a paper on applying the Algorithmic probability Universal distribution to problems in A.I. — Algorithmic probability is what you get when you use the Universal distribution to problems in A.I.

This was the beginning of the system that I've been developing

Since that Time

NIPS PP The System

2.27.03

125-25

"The system was designed to ... NIPS PP1, PP2, PP3

NS.16-49 do end.

Much of my work of the recent years has been in understanding developing and understanding the updating system that enables the General Cumulative Probability Distribution to learn from both successful and unsuccessful problem solving trials.

The last talk I gave at Royal Holloway was at a symposium on the Importance of Being Learnable. — It was a description of some ideas I had on how to transfer learning — how the learning in one Domain could utilize information from other apparently disparate Domains.

enables both direct and transfer learning from both successful and unsuccessful

Subjectivity — That it is much dependent on just what Universal machine is chosen. While it is known that differences in universal machines only introduce differences in probability that vary by a "constant factor", this "constant factor" can be about 10^{100}. The subjectivity, while the "constant factors" of the order of One" it is not small.

Seminar: Quick fox!

Say: I will describe a very general Prob. solving System that I've been working on for some time. PP's 1, 2, 3 from Abstract | IJCV, 02 feb:

Then talk about its properties.

Another tool in our problem solving techniques is Levins Universal search Algorithm which we will call Lsearch.

00 : 10:47
49.
58.20 audio 7
11 00 Start.pg
11 12 Start 11
11 16
50 ~ 1/4u.

How OOPS works! Stack lang – like Forth. 2 or 3 stacks.
"Tokens" are instructions. New tokens can be defined by pgm. Pgm = seq. of tokens.
73 tokens to start. PC = 1/73 to start! PC's using Laplace's rule new, common common.
Each successful pgm is stored in Frozen Memory (ROM) can be referred to:
Boostqinst:

Pred Slide c1 ... c5, byz, dec, boostq.

How OOPS works:

His Stack language similar to Forth: 2 or 3 stacks.

"Tokens" are instructions! New Tokens can be defined by programs.

10 Programs are sequences of tokens.

73 Tokens to start: probability of each to start = 1/73.

Laplaces rule used to update probabilities:

Numerator = 1 to start common denom = 73 to start.

Increment numerator, common denom in steps after each use of a token.

Probabilities of program = product of probabilities of its tokens.

Probability used to Guide Search.

Each successful program is stored in "Frozen Memory" (= ROM)

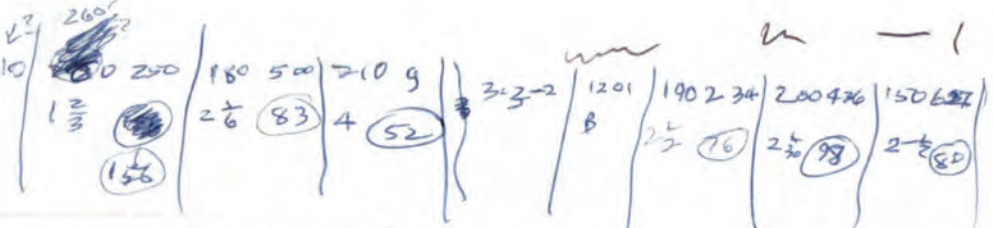
30 How Boostq instruction works:

Boostq instruction looks on stack, gets address of program in ROM:

Boosts all tokens in that program by 73 (Both numerator and Denominator)

Boost as a "Mutation"

22/05



.04 To show no computable pfm can calculate PC in accordance w. Conv. Theorem!

Say Z is such a pfm. Devise say W! $Z(X, W)$ will give PC of W[th] behaviour X. Devise say W, $W_n = f(Z(W, n))$

.07 $f(x) = 0$ if $x > \frac{1}{2}$ else $x = 0$.

We can Make a ~~pfm that~~ Proof like that to show there is no pfm. This can fail if a Wang can't stop for all inputs.

Say $Z(X) = 0$ if x will ~~eventually~~ cause M to halt; never halt

$= 1$ if never stop.

~~Consider~~ Consider pfm $F(Z(X))$ — this a function of X

$F(Z)$ ~~does~~ looks at Z; stops if $Z(X) = 1$

∞ loops if $Z(X) = 0$

This doesn't seem to work! — Yet .04 - .07 seems to work!

3/11/02 Some Expansions/Modifications of KOL talk!

20

21 1) We can use Univ. P. for non-stationary data:

data missing, Multidimensional space of data points extending finitely or infinitely in ~~all directions~~ negative as well as positive directions

2) Re remarks on over/underfitting: True only of models are made before data is seen. (This is true in general of other types of induction, but is easier to implement in ALP)

So I may not want to make these claims.

Because we can get ~~WWW~~ good predictions using truly optimum pfs, ~~this point~~ there is no problem of over/under fitting. ← this may relate to Arians?

can't be divided data...

PRS-6 on Subjectivity

Maybe talk how ALP is actually done & contrast w. other methods

More on "In computability": Perhaps Give Example of MDL:

~~Pick a set of~~ models: ~~Pick~~ Pick best of those models (must ub.)

Model error: set of initial models ~~that~~ very probably did not include

a model that is much better than any in the chosen set.

Use of Univ d.f. gives us a Prog> ① awareness of possibly very

large error ② Suggestions on how to reduce that error.

~~For~~ As an example, ~~suppose~~ we have a data set and

do prediction, ~~from use MDL on a data set~~ we/ pick the best

model in the set using MDL. No matter how well the model seems

to fit, it is possible that there is a better model outside the set of

models considered, and ~~we cannot estimate the probability of~~

~~this possibility~~ There is no way to estimate the probability of this being true. It leaves an error

of unknown size.

SN On leaving during Lsrch, compute h_{cand_i}.

Say $cand_i$ has up to time T in fact solved prob$_k$.

~~If~~ we can use this fact, to recompute the h curves of rest of

the cands wrt prob$_k$. [If there are a bunch of cands that are

"correlated" w $cand_i$ (wrt prob$_k$), then their h curves

will be the ones updated — (in fact we may only ~~the~~ code

2 or 3 models of the h curves of cands that were "correlated"

w. $cand_i$).

(Anyway: After that "big" (partial) h updating, we will tend to

~~have~~ try cands not correlated w. $cand_i$. So there is ~~a~~ a

desired "easy practical thing" — long during Lsrch.

Good!

Note Bene: T. "correlations" of .26 are empirical results about

the h curves that were computed. They were based only on the same old data on

(pst$_k$, prob$_k$, T$_{ok}$) triplets of the past is often of a "04.1"

29.12

→ 29.12

10 : Make Expanded version of "report" with various improvements & discussions! Like for 2 param version of h, h' & how to use it for both positive & negative cases.

04 Also 28.20 – .40 ↑29.12 – .40; 30.30 looks very impt. & should be included.

Perhaps "even now" expand it: Make sure I understand it.

3 forms of $h(T)$ appear:

1) Gaussian.

2)

3) Gamma $\propto x^{\beta} e^{-\gamma x}$

Gamma d.f. ↓ $\left(\prod_i \propto_i x^{\beta_i} e^{-\gamma_i x}\right)^{i_2}$ also of form $\propto x^{\beta} e^{-\gamma x}$.

just so each is determined from first 2 moments of data.

0 On GGPD/[PST] update: Σ1.20

12 : 28.40 : More exactly: After we know T_{ij} for failure of $PST_{i,k}$ (If has child has $T < T_{ij}$), we have strong new data for

so we can get integrals "easily"

$$\sum_j \left(\prod_{\ell \neq i} {}^{a_0^j} O^j(T_{\ell,k} \mid PST_\ell, prob_k) \right)$$

failure prob by time t $= \sum \left(1 - \int_0^t h(t)\right)$

$$= \prod_{\ell}^{a_0^j} \left(1 - \int_0^T dT_{\ell,u}\, O^j(T_{\ell,k} \mid PST_\ell^T, prob_k)\right)$$

is probly of all known failures.

12 We multiply this by the original prediction for success of cands (= PST's). seems better now failures info

20 & we reoptimize wrt O^j to get a new O^j or set of O^j's:

This gives new h's for many of the PST's. The PST's that happen to be "close to" PST_i (i.e. known by TIK failure for problem k) will usually get worse h curves (smaller $\frac{x}{\mu}$; say) than before, i so though it's more likely that PST's not "close to PST_i wrt prob") will have the most likelihood of success for the next NTel.

If we repeatedly chose most likely PST then work on it only till (by simultaneous updates), the most likely cand has been changed, & jump to new cand. Then this looks no more like Lsrch, but more like WON! It certainly isn't Lsrch. → space, 30.30

3.2.03

(30 → 20

1 The Universal Distribution and Machine Learning.

← insert title after title.

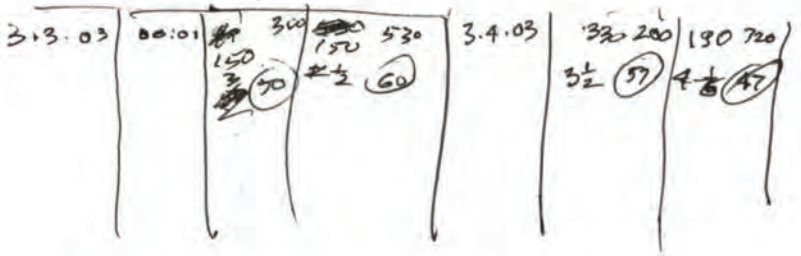
Start of ~~[illegible]~~ lecture.

There will be two main topics of my lecture: ~~I'm going to talk about Two main things:~~ First, The universal distribution and some of its properties: Its accuracy, its incomputability ~~[illegible]~~ its subjectivity.

Later: I'm going to tell how to use my distribution to create very intelligent machines.

It is often possible to obtain predictions using a truly a priori probability distribution (obtained before the data was known). Under these conditions there is no underfitting and overfitting — the data need not be divided into "training set" and "test set" — All data can be used for training and the expected error will be unbiased.

The data need not be stationary; Subsequences of data can be missing; The Data can be multi-dimensional — extending finitely or infinitely in all positive and/or negative directions.



29.40 : its probably much better than Lenat.

[T. (apparently) BAD aspect! It seems too "Elitist" : Very little random, crazy exploration.

Re: Implementation! In 29.29, we will normally have several O^i's for each "sol. ext. exp. ~~[illegible]~~ while we are working on a particular PST, the relative wt's of the ~~[illegible]~~ "Hamiltonian" (wrt wt) O^i's will change — This aspect of "updating" can be rapid/efficient/easy to implement. Actually finding new, better O^i's is, of course, Much Harder.

3.27 replaces by 30.20-.22
3030 replaces by 30.10-20
5.11 : insert 28.08-.16

32.00

3.2.03

Unsolved probs! .15

13) ~~34~~

∞ : Re: [Grad students, ect!] What are major problems in α?

Easiest to work on is Phase 1. To pgm simple TSR!. Look at CJS's.
Try to find correlns (at all levels, of all kinds) to 1 cjs. (Also develop a "R" system (recogn. functions) or/better, using correlns.)
perhaps

Perhaps ideally pgm "SaarbTM" for ANN: Argument re inst. set:
See if we can use correlns. to get at capable CJS's.
(It may be that we simply need larger CJS's)

Try to get at W. Publ's students work. (Peter Bergmann). It's probably in German, hmm. Maybe one of MIT Grad students can read German.

→ Work on Poem(s) for [Sphere]. To do this, first make up bunch of problems. Then 6. "Necessary" forms of S results will become clear. → 37.00

.16 .15 → Some impt, unsolved &/or incompletely solved problems.

1) Test Lsred given cc or pc of a given cond. — but there can be many equivalent conds — would like to give the set of equivt conds, one trial via very large cc (or no of equivt conds). At present, no way to do this except: Use of Laps rule to assign pc's to Tukeys & perhaps → 32.00

19 20 SN: [For any version of report] Write up just how sCPD/[PST] complexity updated "as a whole": how Trainer can help systematically insert/vary factors set of v.g. PST's, to be extrapolated. NB we are mainly interested in optzn. problems a' Plans are many known.

---

28 for to(Lecture Report:

(Any finite decidable. PEM that gets an accuracy of better than $\frac{1}{2}$ p_c error
must for all finite decidable ~~strings~~ data sources which
30 is incomputable. (A real will better-than average better than random for all finitely decidable data strings, must be incomputable.
If a PEM gets ~~le~~ p_c's v.g. for ~~some~~ data seqs.
it is attracted to do well for 6. rest of the seqs. — for all data seqs:
- Given the PEM there must be a computable, i.e. given a string w, finite
dcm of b bits, and a data string decidable by d bits, we can construct
a data string that out w.r.t. known data string it has error $\geq \frac{1}{2}$ for all
subsequent bits predicted by Pacio p_c

~~Factor~~ So! Any ~~Alan~~ CPM (computable proxy Measure)
bounded
For all μ w. ~~bounded~~ — Imp ~~Pacio~~ p_c must have p_c error $\geq \frac{1}{2}$ for all bits.

sacc 35.00

Mike Gherkstein
Gherkstein@ucl.ac.uk
University College London.
020 7679 2000
Mike at work +44 020 7679 2888 (2838)
mi.gherkstein@ucl.ac.uk
Internal Phone. 32888
0207 679 2888

3.3.03

132-38

LOSS FUNCTIONS (.20)

:0 : (Updated early) 31.19: Other ways to assign pc's to Cands (other than "raw coding")
~~Good ways to bet~~ — Also t. method of 28.20 ff (Irredundy Search)
may help, by not trying equivalent Cands that have failed.

2) Trying to find "Conditionals" for t. pc's of tokens, to deal w. this "scaling" problem. "Association", Genzn of "Recognition Functions" in Phase 1.

3) How to get some effects of Boosty ~~re~~ restriction in OOPS, but more tractably. When a set of tokens is "Boosted" There is an impl. "Togetherness" (association) of t. set of tokens boosted. — The "association" of (perhaps) synergistic tokens, is not obtained in the simple A2 Idea that I've described.

3.5.03 SM On "Loss Functions": There is now much extensive Rsch on How to bet w. various "Loss functions": This may be identified to my classic "AE" (?) Algm Evaln. Problem! — A special case is SM strategies. A more General Case is an All-over Algm for Running TM — to optimize some future loss function on it.

As for what Others are doing on this: Do they assume "Stationary TS"? What kinds of "Loss funcns" do they consider? Look at that paper by
30 T. Cover-Bianci assoc w. t. NIPS/workshop I attended — to a review of what they were working on.

My initial/original impression is that of (probably) Quickly Flatter: Just, if one assumes t. data will follow t. Univ D.f. & bets accordingly, one will "do well". However, direct calculation t. Un. D.f. (i.e. doing prediction) Rather than optimizing t. loss funct., is an el. approach & would like to try also to optimize loss funct. Directly — OR as in SM, have way to evaluate (in some unbiased way) poss'l. "strategies" (i.e. "ways to bet"), → (Spec 34.00)

3.4.03

133-33

$3102 , 407 = $120

Look into this.

28.20 ff

28.20 ff plus 40.03 must solve this.

Main problems: (1) K cost v.s. P cost :(Special case of correlated cands. Cands

I don't know if I will help w K cost v.s. P cost.

Step 40.03

It may help!

(2) Defining good contexts, to ↓ Scaling problems. This could be major step in Making SAARB TM (ANL) practical. → 39.00

(3) Finding, expanding SAARB TM. — use OOPS for ideas — some (26) (ANL is other parts)

Somewhere between getting a gold star in Kindergarten and the Nobel Prize.

(4) Analysis of "Boost" in OOPS : can I find all essential features wrt

(5) Put them in optimized form → Seen 38.30 ← NOT BAD!

"Boosty" → Good way(s) to implement repetitious S-Funcs → Seen 37.00–38.50 (looks v.g.)

In "Boosty", t. current ▪ no. of equiv. re positions of each token is N_{eq}.

A better way : t. no. of repetitions is k (present documentation). Initially, k = 1,

but after TM has been running awhile, we try to optimize it over past data.

(An very expressive option!) Try first w. T=1, k=0, k=.5, ect.

Examine other features (s) of "Boosty" — In particular, t. cross-correln.

Pet. token frequencies. One way to do this task: If a pgm has been "Boosted"

we immediately test this correln, so that if any of t. boosted tokens is chosen,

either in future (or past of t. present cand), other tokens of t. boosted pgm have ↑ in pc.

We could have a "adding effect, so if 2 of t. boosted tokens are chosen,

t. other boosted tokens have even more ↑ in pc.

T. idea of Boost (a kind of crossover/mutation) is that "this set of tokens tends to work well together"

& find a better?

SN T. troubles w. SAARB TM : 1) Only fixed successful pgms used as data (no subdefaults)

(Partly because I used Kemeny TSQ ... No Jumps of > 1 "Layer of Concs")

2) Scaling — pc's of gens began to ↓ rapidly, w. small variety of no. of concs. in Many — Contexts of various kinds could help Much.

3) Set of insts, was very small, not near Universal.

4) Perhaps use ideas from OOPS

SN I may want to pass my latest version ("fix") of Z141 — so as it can put spaces into specialized English Text. Even note that I really had one "cut off" — w. any corpus size, it would start to combine words that occured often together. Corpus (size) must be large enough to discover prefixes, suffixes and many words of which they are parts — but not so large that words are often combined to create ("spurious") compound words (à la German ☺).

SN Name for Talk / Handout : "T. Importance of being Unlearnable"

LOSS Functs (cont'd) .00 SM .00 ff
SM .00 ff

space
50 : 32.90 : What about this: each strategy has dcrn length (nits) of P_i S_i .31
00 so w. $1 in Bank, we bet P_i on S_i. (This looks like a nonmyopic Cover's Universal Betting Scheme ... for cases apriori Uniform dst. i.e. P_i = constant.)

Th. trouble w Cover's scheme (& perhaps a fortiori, this one), is that it takes too long to "get up to speed" — But, it may not, if I assume seq. is not stationary (& look for regys in its non-stationarity dcrn), AND, do parallel betting w. other stocks" & allow cross-stock information. .70
10 [This is a v.g. Gauss. of multiple time series Analysis implied by the Mxd Corps Thm]

The idea was that the stocks could have a certain amount of shared info, plus a certain amt. of individualized info for each stock. A better way (perhaps) for many stocks would be to make a tree w. each stock being a "leaf".

a & b share γ; a & b & c share β
d & e share δ, & a & b & c & d & e share α.

20 Are there more General ways to decide how info Units to be shared?

This looks like a v.g. way to deal w. the non-stationarity & i.e. "small sss" available in SM!

26 : .05 That "upto speed" may not have been the problem: the problem might have been that yield/yr. was much less that my approx" scheme (strategy). Perhaps at present time, such low yields would be acceptable, since SM yields (as well as other security yields) are very low now .36
30 51 : .01 Some "stocks" could have large maximum yield: It may be easy to fix stocks so they have bounded max yield — (unlike original Bank)
— So we are sure and yield will be large. — But it would be best to log-bounds & allow Gambler's Ruin" we can't have yield < (original Bank),
36 : .30 Actually, the Covers final yield was ≈ better than best single stock yield.
I think the convergence rate to this was slow — slowness of size of hypoth space, which was a (M-1)-dim object (M stocks). — like space of M-1 dimensional w.t. vectors.
It would converge faster, if we bet on each stock (same amt.), or bet same amt of → (Spec 36.00)

3.5.03

1 35 ~~35~~

(spec 31.40) There should be a nice compact way to say this:

1) For any CPM$_\theta$, there will be finite discrete sequences for which that CPM will have errors of $\geq \pm\frac{1}{2}$ in prediction: i.e. ~~worse than~~ equal or worse than Random choice

2) This is not true of incomparable probability measures: eg. Conv. Turm. gives errors that $\to 0$:

How ALP differs from any particular CPM$_\theta$:

Say we have a particular CPM$_\theta$. It has been obtained w. data set D_θ. ~~Using Data~~ Typically, it will have been obtained by dividing D_θ into a training set, & will give a certain value to expected future error, (assuming ~~that~~ that D_θ was stationary)

Using Same Data, ALP will obtain many CPM's — among them, perhaps, CPM$_\theta$. Its predictions will be close to those of CPM$_\theta$, but it will have used all of D_θ for "modeling". The expected error it gives for the next prediction is unbiased. It makes no assumption ~~about~~ that D_θ has a stationary. Tho its error estimate will usually be smaller than that of CPM$_\theta$, and be "unbiased", it will be on average found that the true error due to Model uncertainty may be much larger (?)

If more data is given, ALP will be able to give new estimates of wts of its models, in accord with this new data, ~~example~~ usually giving a better error than CPM$_\theta$, which remained irrelevant as new data was obtained.

N.B. We could modify training set for CPM$_\theta$ (i.e. size limit) to improve extrapolation.

If time is available, ALP will be able to suggest new directions to search for better models. (Need not use "try, test" routine)

SUMMARY

Main Difference: ALP gets somewhat default models from CPM$_\theta$ — but a smaller, unbiased error, is more important about error claims, than that is CPM$_\theta$ better. Also suggests new areas for exploration.

HMM! in ALP, the search can be "Highly Biased" — while CC=00 ALP may have unbiased error, Bound CC<00 ALP (= RLP) can be very biased, I should think.

→ 45.00

3.5.03
3.5.3
3.5.3 SM

$\frac{1}{m^2}$

10:34.40 each of m^2 pairs. This bets $\frac{1}{m^2}$ on each pair, from usual Cover's (unchanged) system on each of the m^2 pairs. The idea here is that the density of the species bet on, is much higher than in Cover's original scheme.

SN In cover's original scheme: A possible way to compute it: for $m=10$, use only 10 values of each vector component $0, .1, .2 \ldots 1.0$; So 10^{10} components to be updated each day. Certainly possible — perhaps not to time consuming.

$m = 8$ would certainly be feasible this way.

Using present day CPUs, it might be possible to update 8 PFs simultaneously!

→ An even faster convergence is to bet $\frac{1}{m}$ on each of m stocks, "Buy & hold".

— A classic strategy. — Final wealth will not be as large, of course.

So, it may well be that by using sub-simplex $m = 1$ or 2 or 3, (rather than the full no. of stocks being bet on) we will get faster convergence rates — enough to make it a better way than using the full set (& perhaps) much more computation time.

3-6-03	0000	175 137	180 257	240 502	240 353
	$1\frac{2}{3}$ (105)	1.20 (134)	$2\frac{1}{10}$ (115)	$3-\frac{1}{9}$ (103)

SN Re: Advantages of ALPi Newton keeping several hypothetical codes — enables one to quickly pick another code when the formerly best one "fails". (When this is done, one must reoptimize supply of all codes —

Look into just how practical (Physics.) — I did look into this some time ago, but review of its relevance — look again! — It might be a useful (practical) way of doing evolution

3.5.03

NIPS

15 45 p : plane leaves

spac
0:21.06: On Spaces!

One general form: (≥ 3 input vars):

02 T. original 3 input func. had 4 inputs: (1) An input that modified by form of the func — it contained the [illegible] continuously changing "A priori" info.

04 (2) Q (3) random input.

03 Well! .02 – .04 is (1) my old 3 input func (2) T. classic output D.f. of A "in view of Q": "Given Q, how many bits are needed to create A."

Did I ever realize that (1) & (2) of .03 were the same? — They are not exactly the same. The (1) input of .02 is not [illegible] ordinarily a univ. d.f., but a O^j — $[O^j(A_i|Q_i)]$ which need not be universal. { It may be well-defined univ. — e.g. so every input has pc > 0. fixed A

→ T. exact details of O^j will probably depend on how O^j was found! ←

Initially O^j will be for d functions — but assoc. w. each d-pair, there may be > 1 way of implementing it; & some ways to get known A_i's, that extrapolate directly from the highest pc O^j. — These will naturally have much less pc. (Of course in "true" S-Prada, they will have reasonably large pc's.)

23 3.6.03 A passing way to do this: Func has 3 inputs: O^j, Q_i, R: output supposed to be A_i. We find a O^j (3) & a way to test $(O^j|at|R)$ = best for $Q_1 \to A_1$. Then we try R w. same O^j, to try to get A_1. Or we immediately switch to Q_2 & w. same O^j, to try to get A_2 output.

29 What we want is O^j & $|O^j| + |R_1| + |R_2| = \min$ ($R_{1,2}$: inputs for $A_{1,2}$ outputs) ≡ O^j's & R_i's

(I don't think there's any point in trying Q_1 & Q_2 in reverse order) To main goal is to start by keeping track of O^j's that give $Q_1 \to A_1$: we remember codes ≥ (O^j)'s $|R_1|$.

Next search for R_2 of larger i (larger i), we keep track to those earlier O^j's.

v. g. 1, input very ideas!

Generic for Search O^j's & R_i's. $Q_i \to A_i$'s.

I think there is an efficient way to search the space for solns w. larger i (larger Q_i set). (8.25.03: Not so obvious! 138.18 R 138.25 ≥ type of a soln.)

So: we given a $[Q_i^{A_i}]$ set, we end up w. a set of O^j's that will do it w. "minimal" ≥ pc cost. Type of a A_i can be obtained by ⓐ using same O^j & different R_i's. Also different O^j's & various R_i's. Then search for the best codes for a given Q_i, A_i 38.00

36.3

Nip?

6 7 8 9 10
Th F Sat Sun Mon

flite → 6:55 PM (6:55 pm) Gate 18 4403

flite 239 7:30 138 38 Gate 18

16K <

935

00: 37.40: can be done in a different manner, considering the 2_0^j's of the various O^j's.

Essentially PC order for O^j, $[A_i]$ set.

So, this sounds like there is a rather simple search that is really Lisp, that is about as good (theoretically) as can be done.

The mechanics of how I'm to implement this 3 input UMC is unclear. I did have one way to do it, but using a Lisp or AI-like machine, but I didn't like it. Would it be easier using Ivegan's (OOPS) formalism?

SN Check up on OOPS' prefix machines: What is criterion for an output being a legit "trial"? Does it have to stop? — Does it have a stop state? (Ans going were always finite strings. Anyway, looking to OOPS system a' second.

I could use it for a 3 input UMC. → Nolo ● 3TM (Ξ 2003TM) 318.32 2003 TM 3 stuff-loading optional

Suggests that OOPS system can be modified to 3 IU

For a real UIO UMC, how to search in $|O^j| + |A_1| + |A_2| \cdots$ order is unclear.

all 3 inputs are prefix sets, i.e. "usual" way for a computer is to allow the system to ask for another token ad libidum.

18 3.25.03 T: usual way to do ... IBR ... Worry: "Branch" until a certain CB is encountered, then back track to closest unexplored branch.

A possible way, using OOPS system: ① The O^j pgm is a regular OOPS pgm.

② Q_3 is a read only input — it can be copied, but a copy can be modified.

③ $R_{\blacksquare}$ is perhaps an input like O^j where the machine can request new input by giving to address p_{n+1}^R, where p_n^R was the largest address thus far for R).

23 — How to implement the search routine of 37.29 – 38.03 w. such a UMC, is unclear.

Actually, it may be rather easy, since its $|O^j| + |R_1| + |\blacksquare| + |R_2| \cdots$ has to be minimized, i.e. access all "prefix code" type inputs: Since we want to minimize R_1, we look in it first. (Shortest, first)

2005 29 If may be necessary to Devise a special "multiple trace search" w. 1 trace early for O^j, R_1, R_2, R_3 ... This routine will be used when to consider a "next Q".

We can ↑ $|O^j| + |R_1|$ a.r.t. by ↑ $|O^j|$ i/o $|R_1|$ or, by considering short codes for R_2 as well.

I'm not clear on the order in R_3? yet!

3.26.03 Not exactly! More like order $\sum 2^{-|R_i|}$ in R_i.

$P_C(O^j) \cdot P_C(A_1) \cdot P_C(A_2) \cdots$ w. such a UMC, is unclear.

I don't see this!

So Try for Less U in UTM

BOOSTQ Just what features do we want/need that Boostq suggests? Once a pgm has been successful, not only are tokens of that pgm more likely, but other "similar" orderings (w. perhaps gaps) & other similarities, are more likely. Those are "Mutations" (Ξ Modifications of the successful pgms). If we use 2 pgms, we have a genetic crossover (Ξ induction w. $SSZ = 2$).

This is a good way of thinking about it.

Re: Mutations: many mutations are induction using OSL (one case) — also subroutines, etc. This is rather worse reduction, but it's better than "Nothing"!! Hvr. Induction w. $SSZ = 1$ is interesting!

Its a type of A induction! (as a S-function of S_{n+1} in view of mutation) — so we want one or more O^j's to do: O^j (mutation | mutated space)

— a P.D. on mutations in view of mutated.

→ 41.00

3TM 319.10

3T 323.21 for Bibl. Summary

Much more SERIOUS Solution to the problem

10:33.02 ! Context of tokens : Foundering w. Scaling (of the kind).

As I see it, "Context" is an enormously General Concept. A discrete Context (not fuzzy Gray, Stochastic) Problem Is (one kind, anyway) is a branch of subsets of the previous corpus — Each subset is a (discrete) context.

A Σ-context is, for each token, a pd over all previous possl. corpi. — Essentially it gives a p.d. over all tokens (Σpi=1) as a function of all previous corpi. This is sort of what J. had in Mind in OOPS — (but he really had no v.g. idea) as how to implement it, other than "Boosts" is a few other probly Modifn. instrs.)

Black/wh.

? Discuss Reinforcement

→ We could just do a QA induction to get the S-Funct that gives the pc of any token, in view of the past Corpus. — This is, of course, the "copy successors of the past" update, à oblivious of the concept of "optimization". Sorts a "Phase I" (context) (p.d.)

Anyway: I may want to try to do that induction "by hand" (i.e. ME) - at least in the early [redacted] Typ. of TM.

More Generally (à I think I may have had this in Mind some time ago — Post assoc. w. each past part-corpus is a pd. on possl. completions (continuations) of that part-corpus. (I did write about this in recent past ---- where??). Maybe in early discussions of Difference Boss. My system à OOPS' Boosts à pms to assign pc's to tokens

Essentially it is an [redacted] pd on all poss completed corpi, à should be updated every time the corpus is augmented.

→ Note that this is a "corpus" of successful. persons [redacted] — not [Q A]sets or [A]sets.

narrowish

The hope is a move with [redacted] kind of "Context": A more General mind would also consider the hyperorder Context of the problem itself (i.e. Q) à T. type of problem the "indexes" of the Q (of any) [redacted] — the induced Σ indices, based on previous experience.

So it looks like there are several places in TM (or Phases) in which things we would use induction on past corpi. We may want to do this "by hand" in a [redacted] Young TM, but by [redacted] in advanced TM (I.e. in Phase 2 we have an extra phase of "drag").

If the IPC of the machine used is large enough, we can commit in what the "By hand" induction is have TM do it stochastically. So perhaps we should see how far we can go w. accessable IPC, à do basically a "[redacted]" Induction by hand" to bridge the "initial gap"

→ (40.00 spec)

3.6.3 428 20:07 Basta.

& row from from Middle

40 10:30 → 1
2230 → 24:30.

cal MM map | 1235A in Basta at Arrival. | 825

180 40

11 → 12.50
P. 12.25
2.15

10:39.40 : In 34.00 -.40 (Context of Tokens), I was getting a non-el. soln. of the Phase I problem — but what I was originally saying about "Context" considerations was an el. approx. — This may be less "well-defined" than the non-el problem.

03 (33.03 R) On the pc v.s. kcost problem: In 28.20 ff (on correlated-Cands) it was felt that this would not help because the write soln would get much less from assigned tok. Hrr, if ≈ all assigned cands have the same ratio of pc to 2^k (rel. complexity estimate), then it would be. OK. i.e. a kind of "renormalization"

So 28.20 ff may ≈ solve the 2^{-k} v.s. pc problem. !

Worst shape

So, the problems in worst shape (item 33.00 ff):

(2) Defining Good contexts for "scaling" problem 38.00 is approach & probly finished
(4) Analyses of Boost MOOPS 38.20 ff (first note 33.04-.08) ≈ Mutzman SSZE1 | crossover SSZee
(3) finding, expanding, fixing SAIRB TM ; see 33.26

6 (2) Context : look at non-el soln, then look at various elems.
List many kinds of Contexts. Doing so should suggest more types

1) Most General non-el. "Context" ! = PD on next Token or Entire completed corpus or portion of unknown corpus as a function of part corpus, including Problem dem (Q), Any indexing or problem-s/o induced indices.

2) Parts of 1 : PD = on known corpus or Problem dem or indices.

3) PD's on ngrams.

.1834 .175 .1846 2085

12.469 cc 10.38 Ap.03 Ap = 10.5!

0 ~SN Methodology for "Context Discovery" (by Tranor/Me). Look at situations in which I expect Context to be important, & try to devise contexts that would help get hyperp's for tokens (or larger chunks).

32 SN The Mechanics of "Search for Structs" or 37.01 ff, is a GREAT JOY!

3.8.03

» : 38.40 (on "Boost" as Mutation).

Mutation as "sort of" in GA, is somewhat different (But not much!):

In GA, we might have as input: (Cand$_i$, G_i) output D.f. on (Cand, G). ← seems to be more vicious.

are we might just have Cand$_i$ input, d.f. on Cand as output.

In G.A. we can be (slowly) looking for a good (sat. of) O^U functions — this O^U will slowly change as the population changes — so our Mut. rules change as population changes

& its ↑ in $\overline{G}$.

Note that when I say "Boost" is like a "Mutation", I mean that the entire new Cand prog, is a mutation of the "Frozen pgm" being mutated.

This Analysis of Mutation viz the GA operator O^U, can probably be applied to the problem of "Context" of 33.01 (for fighting the scaling Catastrophe)

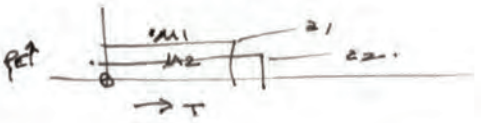
→ 42.00

SN on OOPS! The only linkage before the $1^{st}2^{nd}$ problem & TOH was that OOPS! There was no discovery of freq. of use of tokens!

Was there any (other) theory of token freqs before, say successive $1^{st}2^{nd}$ trials?

SN In the ~~conversation~~ GPD$_1$ → GPD$_2$ x theory $\frac{a}{u}$ (for mv. probs) is approx. ordering.

What is rel. probby of "beat" betw. 2 cands: a_1, u_1 , $a_2 u_2$?



if $u_1 < u_2$ this is prob of Cand$_1$ win is a_1

prob of Cand$_2$ " is $a_2(1-a_1)$

Win probby depends only on the orderings of the u_i & on values of the a_i.

If the a_i's are "small", the ordering is by a_i , indep of u's: i.e. a_1 vs. $a_2(1-a_1) = a_2 - a_1 a_2$ small.

If a_i's are large, ordering is by u_i, indep of a's. E.g. say all $a_i > p$.

I was using $\frac{a}{u}$ as rule of order (largest first). This may not be so good!

Perhaps use u ordering if a's are large, use a_i order if a's are small.

— But what is how to compare large a_i larger in w. small a_i smaller?

So use a to compare, if a are small

" u " " " " " " large

use $\frac{a}{u}$ " " " " betw. mixed a's.

What is threshold to change to $\frac{a}{u}$?

I don't feel that this analysis is so hot, but it does give a way of looking at the problem.

T. "all a_i are small" case is probably very common!

T. effect of a cand being wage = its a_i mult by a $\prod_j (1-a_j)$ that depends on i/order in u of the cand, & is a step fcn of cost order in u. The $\prod$ factor is of most import for large a_j's.

3.9.03

(42 42

20 : (Spec 41.12) : T. main present problem seems to be "scaling." Gained "context" & Boosty are 2 aspects of it — ways to deal w. it.

(of pc's of Tokens)

Can we regard the "boosted" set of tokens as a "context"? — Superficially, Yes — these sets are common to all token predictions & don't have the specificity of "contexts".

& "Boost" ~~adaptation~~ sets of tokens are common to all token predictions & are a property of the "State" of the human nervous system — so in Real life sense, they are a "Context".

07 But they don't seem much good for dealing w. "scaling", since the pc of Boosty (incl. the is the corpus size, N. — i.e. the pc of hitting a particular set of tokens via boosting is $\propto \frac{1}{N}$ — will not pc but recent. — Real pc's Normally, as N ↑, the no. of (sets of tokens) that would be helpful, also ↑. (& the no. of related problems, solved in the past). Unfortunately the any. of pc's for a trial is $\propto 2^{-k}$ not pc — which is sum of pc's of relevant (sets of tokens)'s

N.B. 29

12 Jürgen wants to use "context" (I guess) to deal w. scaling of boosting: as no. of pgms in ROM (Frozen memory) ↑.

Looking at "Boosty" as a kind of Mutation is probably much better than regarding it as "Context"!

(Successful)

Anyway: Both of boost & context can use the QA induction: In "boost" its SSZ=1 Mutation In "context," it's all previous (corpus → token) sequences as {Q, A} data set

So, boost actually does 2 things ① it ↑ pc's of Tokens (rightly "scaling") ② ~~the~~ T. pc of a particular boost & w. the corpus size, contributing to the scaling cases to arise.

So, we would like to (perhaps) use "context" to narrow down which pgms in Frozen many to Boost. So, what we want is a "Context that points at "related" (successful) pgms of the past.

Can I frame .22 - .23 as a QA induction problem (or as any kind of induction problem!)

I think I have to have the Q's = problem descr's, A's = pgms that solved those problems. However, in A's I'll be only (for Jürgen's "Boost" determinator) be interested in the frequ of tokens in each pgm.

So I want a problem to map Q's to a density distribution on Tokens

29 .12 Actually .07 - .12 may not have a "scaling" problem. As N ↑ the no of successful pgms w. token traces are useful for the present problem, also ↑. We may have "scaling problem", but of much

30 smaller magnitude than the normal scaling problem for pc's of Tokens. There is, however, the problem that the no. of possible wgts. (i.e. cardinality of the target set) ↑ w. N and is $\propto N$, so "Boosty"

32 cost is $\propto N$. — So there is a serious scaling problem, but due to the non-additivity of

34 expanded cc.

A possible way to deal w. problem of .32 - .34! If a fraction c (< 1) of the pgms are repetitive: i.e. say each pgm occurs $\frac{1}{c}$ times, then it would be well to eliminate pgms that had the same set of tokens. We might have a bunch

3.9.03

1 of 43

01:42.40 : off "clustering", & chose betw. the cluster centers.

On "Cluster" demos: Each pgm is represented as a Bag of tokens.

One way we can have a "cluster center", then demos of how each element in A cluster differs from the "center".

I could represent each pgm by a pt. in r-dimensional space: r is the card. of tokens. Each pt has integral coords: usually zero & occasionally 1, & sometimes more. See NIP/ID 4.90/D.16.02 for stuff on Clustering — from 44.00

SN on Scandpiru ID: from 836 backwards.

825: On time-varying O2 probs: I wasn't able to renumber soln. at NLP pt. Expo: 766.10 See back 2nite!

816 on scaling".

811 Gosl Theory of prob solving.

811.30 on TSP construction ← Good.

807.20: Part of proof of conv. Pgm for QATM

806: N Dim. ALP

802.02: Human backtracking / Rary revision → 800.03 → 786.20

.21: "Phases I, II, III" of TM ← this probly ≠ present idea of Phases 1,2,3.

788.12: SUMACS — 768.20: sequential coding for QA & Time series'.

772.00-.18 No. params > no. data pts. 778.00 "Curve fitting"

771.07-.08 Big unsolved problem at Mar time, NIPS 12.13 may be relevant — ± Pretty Esoteric.

765: HMC how many Corrs?

764: ANN & RANN — How to get ALP values

762.00 Montreal AAE (SOY Actor Algm Evoln), etc.: Pgms may be some as problem of Difrnt "Loss functions" ← superficially, NO! LK ALP compares to write P.D. it should be used to optimumly control "Loss function".

Refs to renews

761.17: Development of GPD — PST Grammar — from early Beginnings.

Review

753.11 Sh strategy as Godess 01/02

Scan of (NIPS 2/18/03 6600 backwards

(41): Things to put in Next revision of Report: (also see 3.01)

40: on Norvig Unival DF: Just what did Lavin show?

39: MEM & error in my ideas

30: Q · mailing list.

18:25 GA.

10.25 SOY, STEIN AAE, act

4 Clustering: Purports to be general soln. for continuous N dim vector data. → (4TH 44.00

3.01 see (41) (on next version of report).

D.16.02

ID
NIPS

CLUSTERING Duplicate

(43½ = 4

This was in response to a proof (maybe at NIPS 2002) that clustering was impossible to do in an exactly rigorous way.

0 General Clustering: Given a data set; each pt. of set has several parameters. $\vec{x}$ is m dim vector

~~Etc~~ A "cluster" is ~~away~~ a way to ~~sep~~ a) pick a pt in the space, $\vec{r}_0$ b) find a ~~scr~~ Scalar function $F(\vec{r}_i, \vec{r})$ ($\vec{r}_i$ is a data pt. $\overset{j}{\vec{r}}$ is "center of j^{th} cluster) ∋ $F(\overset{j}{\vec{r}}, \overset{j}{\vec{r}})$ = max overall second argt values. Perhaps $\sum_i F(\overset{j}{\vec{r}}, \vec{r}_i) = 1$ & $F(\cdot) \geq 0$

so $F(\overset{j}{\vec{r}}, \vec{r}_i)$ is a (kind) probability that $\vec{r}_i$ is in cluster j.

Anyway: the idea is that they is a way to dscrb. the set of points.

| How the dscrn goes, | ~~mention~~: ~~If every data pts is assoc. w. story over j & all data~~ ~~One way: Each pt:~~ If there are j clusters & n pts., each pt. can be dscrbd j ways. We want f's such that the total info in the dscrn of the functions F plus the j $\overset{j}{\vec{r}}$'s. ~~Put~~ plus the info in the data dscrns, is ~~the~~ min.

The pc of each data pt is sum of its pc's wrt. each of the j clusters:

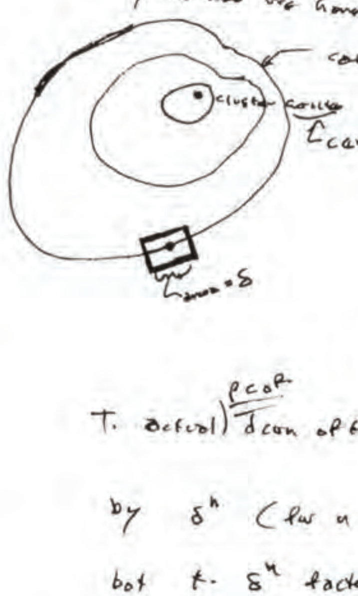
We want product of ~~the~~ pc of the n pts times (pc of ~~the~~ j centers) × pc of $F(\cdot,\cdot)$ to be max.

20 ■ We can compare it w any other clustering of those pts.

21 The category is a way to dscrb the set. ? — Yes: .22 – .40 shows how.

22 .21 is unclear! In a cluster, $\sum p_i = 1$, but how does this enable us to dscrb. each of the pts. wrt. the centr of the cluster?

Say ~~we have~~ we have a 2 dim space (m = 2)



What we want is the pc of dscrbing a data pt. so that it is within an area, A,

27 We want pc·A for small A. [(pc·A) will be ∝ A for small A.]

So what we really want is that $\iint_0^\infty F(\overset{j}{\vec{r}}, \vec{r})\, dV = 1$. (V is a volume element in (2 dim) space)

We want the (product of the) prob densitys of the data pts to be max, when multiplied by pc of the $F(\overset{j}{\vec{r}}, \cdot)$ dscrns.

0 The actual (pc of) dscrn of the data set to any accuracy can be obtained by multiplying above pc by δ^n (for n data pts & we want to know each pt to within area of δ). — but the δ^n factor is indep of the clustering method! ~~So we ... up~~ choice of clustering method does not depend on δ.

KTR 44
3.9.03

3-9-03

144 — 44

KTK
Kaltalk

≡ 14-3½

Clustering

30 : NIP/ID 9.40
D·16·02

⇒ An alternative View: Not as exact perhaps:

We have as before, a function $F(\vec{r}, \vec{r}) \geq 0$ ← center of cluster $F(\vec{r}, \vec{r}) = 0$,

$\int \cdots \int F(\vec{r}, \vec{r})\, d(\vec{r}\ \text{volume}) = 1$

↑ find.

We have a data set. We hypothesize that these data were generated by k (cluster centers),

in accordw. ■ t. ■ $F(\cdot\cdot)$ d.f. . Each center C_i, has pc of p_i of being chosen

SN Min "Clustering" problem: This ~~induction on numbers~~ BAG induction: "Two kinds of ... Induction" paper

T: General idea: We have a BAG (Continuous a/o Discrete). We express it.

Bag by a subset k "centers" & a "distance function" from the centers. (The distance function integrates to 1 for each center: i.e. Normed pc's Add.

The dem. of BAG: Each ~~bag~~ element has k (= # codes); each code gives its distance / pc.

from each Bag center. To bag element does unordered.

We want a set of k centers & a "distance" function, so that the pc of given is Max.

Each center acquires a pc — total of these pc's is 1; we assign these pc's to

Max. pc of BAG.

(Continuous & discrete distance functions can be encapsulated as "Mutations" of each Center

This suggests many possib. forms of RC) function: ∈ OOPS' boost "mutation" —

Uses pc's of ~~elements~~ tokens of center plus(low) fixed pc's for other tokens], ∈ other Mutation types

used as GA à GP], ∈ each Center is a "Grammar" that can generate any object in a BAG — probabilistic ("i") S-Grammar.

à assign it a pc: $\sum pc = 1$ — ideally, each Center grammar should be able to

express any string, but most w. very low pc]

SN Also Note 36.20, on an Advantage of ALP over "those bastards".

3.13.03

1 15 – 45

20:35.40 : On computability v.s. {universality as a predictor}

01 [redacted] CPM (Computably pc Measure): one can design a deterministic seq. w. K(n) ≈ K(n) + c,

02 Then ∀ CPM gives pc of .5 or less to each bit — so the pc error is always ≥ .5.

How probable are these "immeasurable seqs"? : ~~There is only 1 seq~~ Given a CPM dern,

there is only 1 seq. that will be as bad as .1(=.02. However, if we allow the CPM

to have pc error ≥ ε (ε < ½), for each ε, there is a larger set of seqs.

s.t. no. of seqs → ∞ as ε → 0.

It is probably possible to measure in some sense, the no. of seqs for ~~error~~ w.

pc error > ε.

0 If we have a CPM w. observed past. ms error of ε for a given sequence,

CPM we say anything about the expected future error, in terms of pc of dern

of seqs that have future ms errors of ε'.)

19 → 47.00

20 On a simple proof of the unsolvability of the Halting problem:

A version that I (seem to) understand: Say ∃ Pgm that can determine halting of a finite string.

22 1) examine all Pgms < 1000 bits long.

23 2) run all of these that halt.

[redacted] .22 – .23 is < 1000 bits long.

24 If it halts, running it → infinite loop of calling itself.

" " doesn't halt, then running it next after Pgms must halt.

This is related to: "find a halting Pgm of length n < 1000 B with longest run time"; (any pgm runs to halt, or else is a loop: not halting)

or that decides largest number

So the nmb. of the pgm seems to be:

29 1) If this (.29 – .30) Pgm halts, then it, then goes to ∞ loop.

30 [redacted] " " (#) does not halt, then stop.

Whatever the Pgm I couldn't understand! i.e. it was unclear as to what input

to the Σ pgm that determined if a string → stop or not was.

α(x) = 0 means on x → stop α(x) = 1 means ∞ loop.

33 So Pgm: 1) If α(.33 .34) = 0 ; then → ∞ loop

34 2) " " = 1 : then stop

Seems OK!

Now: Where did I go wrong? It wasn't clear to me as to what the input to α was to be.

I had not conceived of the recursive pgm, .33 – .34!

So, approximately 3 ways! (.23 – .23)R, (.25R), (.33 – .34) all about the same.

3.13.03

146 ~~46~~

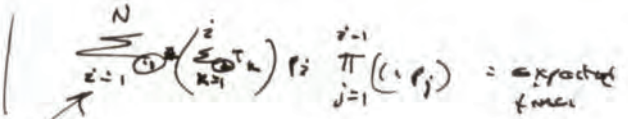
3 = 15

2.30 = 14:30

10

: T. first-G-H Thm!

Expected time to soln = 



If we change ordering of 2 adjacent i's in $\Sigma_{(1)}$

We want to compare th expressions: Only 2 $\Sigma_{(1)}$ terms of ea. result in expression change.

It would be good to include this in a report as an appendix, — tho it is an essentially trivial (tho messy) to prove. Much Bookkeeping necessary!

20

A point about th updating phase 2 update, involving h' & h. $h'(t_0)$ is th proby density of finding soln. at time t_0. This automatically says that no solns were found at $t < t_0$

Pr. that T. proby of failure to time t_0 is ? :- $\prod_{i=1}^{n} \left(1 - \frac{1}{n} p\left(\frac{i}{n} t_0\right)\right) = \prod_{i=1}^{n} \left(1 - p(i\Delta i_0)\right)$

$= \prod_{i=1}^{n} e^{-\left(p\left(\frac{i}{n} t_0\right)\cdot\Delta\right)} = \exp\left(-\sum \Delta p(i\cdot\Delta t)\right)$

$e^{-\int_0^t p(t)dt}$.

This seems wrong!

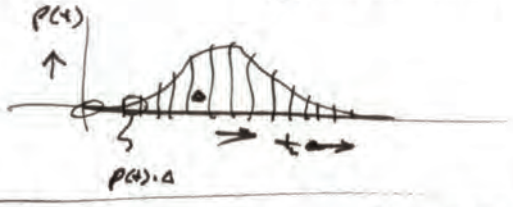
T. proby of failure at time ∞ should be $1 - \int_0^\infty p(t)dt$. (?).

those expressions are same if $\int_0^\infty p(t)dt$ is $\ll 1$.

An Question is my idea of what $p(t)$ means.

related to 2 Q's (1) what is proby of soln at time t_0?

(2) " " " no " up to time t_0?



$p(t)\cdot\Delta$

Possible conclusion: It is possible to use $h'(t) \propto h(t)$ ($+ \text{const}\cdot\delta$) : $h'(t) \propto h(\infty) - h(0)$

or $-r(t)\, e^{-\int_0^t r(t)dt}$; $e^{-\int r(t)dt}$. : In either case, the first term is the derivative of the second. — I may need "constants of integration" in both, here.

In Brian the integral starts at $t=0$ & goes to $t=\infty$,

so $h'(t)$; $\int_0^t h'(t) = h(t) - h(0)$.

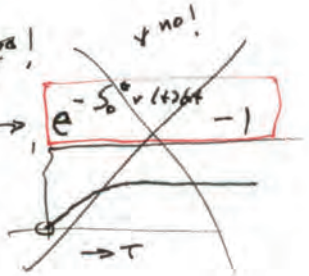
$r(t)\, e^{-\int_0^t r(t)dt}$; $\left[-e^{-\int_0^t r(t)dt}\right]_0^t = 1 - e^{-\int_0^t r(t)dt}$ → $e^{-\int_0^t r(t)dt} - 1$

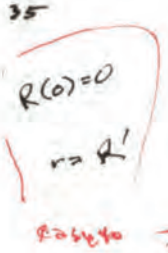
$\cdot 35R$

Proby

$h'(t) \propto h(t) - h(0)$ would be constant.

$R'(t)\, e^{-R(t)}$ | $1 - e^{-R(t)}$





3.15.03

SM

0:00 Variation on Cover's one Prod: Have a prior on strategies (linear, nonlinear...) & wt. on each of a priori. T. a priori is pf that each strat is Best?

Update by each strat having a mean divers for yield each year & then update by probty that each is "Best" (≡ max yield). Maybe not easy to update — Monte Carlo's 1 way. Perhaps have 2 or 3 outputs for each strat., & do an exhaustive search. For 3 outputs & n strats, thing is 3^n cases — so $3^{2n} \approx 10^n$ so $n=9$ for 18 strats gives 10^9 trials (a lot)

(This is a QATM phase 2 update problem!)

- A really impt. "New Idea" is the idea phase 2 update: A more specific idea as to what we really want updated! — what "Real Goal" is.

- So: Is it OK to make the pf wts. correspond to "the pf that a given strat. is the "Best" in the very considered."

- The thing that behaves much more like an updatable pd. is the curve giving pf of each possl. yield. — the pd of yield. The "phase 2 update" is a formal mathl idea of the yield curve space.

A big problem in Cover's system was the fact that just what the initial wts represented & how one could use previous & perhaps ll info, to sharpen them up. As he presents the system, the initial wts are uniform. They might represent expected "yield per yr." say, or exp(expected yield/yr) = 1 initially since exponent = 0.

We might minimize the wtd. avg of ln/(sum of ln yields of individual stocks).

45.14

0 : [redacted] : Re: Computability, accuracy etc: For revision of KolTalk – comp Journal.

APL uses a prng, to get [redacted] models: There is no "training set". All data is used for the "test set" — I think the resultant error is "unbiased"; but there is probably complex distributional nonsense. (?)

In ALP, the error is to an important extent "known", but with usual caveats. We obtain error d.f. from Entire data set. This error should be monotonic as CB↑ as is pc of data ↑. But we don't really know how bad it is! — in what sense is the error unbiased?

In the KolTalk, I said that we never have a usable upper bound on error. Well, the mean error in the [redacted] data set, for ALP gives a kind of upper bound on error — The expected error should ↓ as SSZ ↑.

On the other hands, for any CPM, (and any ALP, w. finite CB is a CPM), we know there are sequences for which that CPM will do catastrophically poorly!

1) So, there is the idea approach of .00–.04 (.09 also) : related to Conv. Thm.
2) There is [redacted] idea of .13–.14, an uncertainty of ALP.

on this subject.
131.28–.30
135.00–.40
147.00…

A reply of .00–.03 to .13–.14: The intractable seqs are unlikely. The [redacted] most extreme intractable seq. …. There is only one. Seqs that are intractable to the extent of error of ε in pc's become more numerous as ε ↓, them ↑ more as ε ↓ further!

One idea of error in approximating Univ. d.f. That as CB↑, error ↓ — but at no point do you know how large it is. [redacted] I think as CB ↑ the pc's become closer to 0 & 1.

Also ALP can often tell when one approxn. is better than another (cross-entropy sense), .22–.24 & .25 One can't get size of error, but one can know how to get error & know which ways are better than others.

However .00–.04 does give a real kind of error measurement that seems (?) to be unbiased. (Meaning of unbiased is unclear here: perhaps w.r.t. true generating body measure?

Also, the Conv. Thm applies to CB→∞ : What of CB < ∞? Then error of ALP is a CPM, & we don't know that $\frac{P_M}{n} > \alpha\, P_{PM}(M)$ (i.e. $P_M > [redacted]\, P_{PM}(M)\,M$.

The main idea I guess is that shorter codes mean one must be getting closer to the univ. d.f. & ∴ how likely that Conv. Thm. is relevant.

There is a long Seq. in Phase notes about just why shorter codes are better. — it may be that .34 was the main "final sort".

Consider the Corrollary of Conv. Thm.

ABC

spec

149.00

3.14.03
NIPS

Spec:
: 147.40 : Well, [redacted] At first Glance, t. Corrolary is not relevant. By finding Shorter codes for t. Corpus, implies we have an approxn to t. Univl d.f. Best [redacted] might have a pc for all seqns that is a "constant" times worse than M but t. constant may be < $P_M(M)$.

Note In trying to approximate t. Univl D.f., we know that as CB → ∞ we will → t. U.D.F., But this is still not t. same as knowing M. With P_M we get t. "Best-best" but not still don't know how Good it is! — Yes we do! we get, in all cases a Unbiased estimate, since our model is apriori & entire corpus ≡ "test set".

Also as CB↑, our apparent error ↓ so our real expected error will↓!
.CB-.10 looks (I mean) adequate for purposes of ALP.
A t. Conv. Thm suggests that usually it will be a sup. (perhaps t. Best possible, if all info is in applied).

Opposing ideas of knowledge of error in .05 etc.: Have is idea of how never knowing how large our err. from Univl. D.f.

In measuring error, perhaps best use KL distance, since it is concerned w. ratios — so when we get small error, its still relevant, & can't how that we still can have, ∞ "distance" to go!

General conclusions: That usually, incomputability of U.P.D is not relevant.
By using apriori precn. one cannot good approximate!
one " " " " " " " for any other P_M this way, also.

ALP, hvr., (1) Suggests other models: P.r. models.
(2) can do OSL (which MDL, MML, is most other methods can't deal w.)
(3) Tells us that we can usually do better. (But is never sure of this). } this is now from ALP & fact that Randomness is never certain.
"Better" can be deterministic!

So I'm still unclear on how to tell one is better than other. (4) If we have 2 models & one has shorter desc. than other!
Why "is" shorter one better? ALP says use wtd. means; wts are lengths of codes.
Empirically, we will often usually find that short codes methods gives smaller errors.
(tho not always: — it may have a very short model & can't more "compact error")
Well, shorter desc is closer to ALP (approxns are monotonic ↑).
& ALP is V.G. — perhaps is good as can be possible!
Is .33 as good as I can say about it? It's not bad — but....

What to do if we have 2 distinct speeds (distinct real machines) & [redacted]!
Well, this is equiv. to distinct "pre Corpora" & for distinct time senses, one has distinct weights: The disagreement can be resolved.
Unless, in some way add the 2 "precorpora" together! A Resolution!

10:49.40 : Be sure to Mention in list of desirata: Optimum tradeoff betw. complexity of model & apparent fit of model to data.
(closeness, Goodness)

.. - Under these conditions, all of the data can be used for testing and error Estimate will be unbiased.

There is an optimum tradeoff betw. complexity of model and apparent fit of model to data — the underfitting or overfitting bar

Perhaps no mention of SSE unc. v.s. model uncertainty?
(uncertainty)

07 In practical applications of the UPD, incompatibility is usually not relevant. We use approximations to the UPD that are computable and we obtain unbiased error estimates for them. Usually these errors are smaller than those obtained using other statistical methods, since we are able to use all of the data for testing. (softens this a bit — but sure it always works this way)

The incomputability of UPD implies that there are usually models of the data that are better than those we have used in our approximation to the UPD, (+bit and c) that we can never know how much better these models are. For example, if we were given a pseudo random sequence of bits, and we didn't know it was pseudo random, we would probably conclude that the probability of 0, and of 1 was about .5. If we were able to discover the pseudorandom rule, this more accurate model would give us probabilities very close to 1 / one for each prediction.

In all / empirical predictions (not only approximations to UPD), we are almost always uncertain that the best model we found that far is particularly good. It is / almost always possible that if we searched for 10 minutes longer we / would find a much better model (through the pseudo random case just mentioned / above is an extreme case).

The uncertainty about how the best model holds true for all predictions — (and uncertainty of how much better models exist.)

26 techniques for empirical data. It is a characteristic of probability itself.

(SN) Side Q: In attempting to calculate UPD: If one is interested in excess error of E_1, can one only consider a finite no. of codes? Say we have found a code for the data that's of length l; Or: we find a model of length l that gives pc of 2^{-l} to the corpus, so total code length $\tilde{<} l$ tr.? Do we only have to consider codes of the models of length $\tilde{<} l$ tr.?

Many very promising Models will take very long times to evaluate. Others will be partial recursive, so that in any finite time one can't be sure one has converged from convergence.

In 1. the talk paper, "Error" is used in 3 ways: (1) SSE error (2) expected error for a particular CPM, corpus pair (3) Deviation of CPM used from UPD ("best" model).

Models for which we have only spent enough time to evaluate partially.

20: ..... but to "Model uncertainty" ..... give poor predictions. At any point in our approximation of UPD, we will be using a weighted sum of the predictions of the best models we've found so far, and we will have a good, unbiased estimate of error in prediction, for that approximation.

At each point, however, there will be models we have not yet fully evaluated, and models which we have not spent enough time to be fully evaluated. These unevaluated models give rise to a difference between the approximation and the true value of the UPD. The incomputability of UPD means that while this difference must approach zero, as we spend more and more time evaluating models, we cannot know at any time how far our approximation derivates from the ideal UPD. This difference can sometimes be very large — as in the following example:

An example: Suppose we were given a pseudo-random seq.

Since we always have a good estimate of prediction error for each approximation, this incomputability does not in any way inhibit our use of these approximations for prediction. It does mean however that no matter how long we've spent evaluating models, there is always the very real possibility that if we spend 10 more minutes in search, that we would find a much better model. This is true of all methods of estimating probability — not just the UPD.

(SW) It is clear that there are two meanings of "error": how: One is error in prediction. One is deviation from "true" probability

Perhaps desc. 3 models of data ① M that generated data ② PM the UPD. ③ an approx to UPD ④ any CPM trying to approximate M.

partially

(Since notation of things was specific for following)

at 4.21 of Hallart

26 27 There are several types of errors involved in statistical inquiry. To see how they are related, let us consider a stochastic algorithm, M, that is able to generate stochastic sequences. M assigns a probability $P_M(x)$ to every possible sequence x — this is the probability that M will generate x.

Let $P_M(x)$ represent the probability assigned to string x by the UPD.

Let $P_M^t(x)$ be an approximation to $P_M(x)$ obtained after search time t.

(theory) $P_M(x) = P_M^{\infty}(x)$, the approximation obtained after an infinitely long search.

The convergence theorem tells us that as the sequence x becomes very long, the difference between $P_M(x)$ and $M(x)$ becomes very small. Unless we know $M(x)$ (and we rarely do) we cannot know how large the divergence is.

: $P_M^T(x)$ approaches $P_M(x)$ as T gets large, we spend more time on our approximations. While we know that $P_M^T(x)$ approaches $P_M(x)$ arbitrarily closely as $T \to \infty$, we can never know how close $P_M^T(x)$ is to $P_M(x)$. It is in this sense that $P_M(x)$ is "incomputable".

While the exact value of π, as a decimal fraction, is not computable, π is regarded as a "computable number". This is because as we make approximations to π of greater and greater accuracy, we know how much our approximations deviate from the true value of π. It is in this sense that π's "computability" differs from P_M's "incomputability". ABCDEFG

All methods of computing probability from empirical data, and similar to approximations to P_M, in that we can never know how far they are from the best possible approximation. The "incomputability" of P_M is a reflection of our inability to know its value to any desired degree of precision.

12 ...

This way to describe just what kind of errors we normally measure in bpc.

For digital sequences, how close are the digit accuracy probabilities assigned to elements of the data sequence?

For prediction of continuous quantities, how accurate are the probability densities assigned to elements of the data sequence?

1.0

One thing UPD gives is mean bpc in pc density of elements (continuous or discrete) that actually occurred. This is (like cost of coding) of the data seq. It is an "unbiased estimator" of code length (or mean code length)? The log of each element of the corpus.... Please know some distribution facts, i

20 probably Pivar sum (the log pc of code) is ~ Gaussian, so we get an unbiased estimate of mean bpc/element (or pc/symbol)

So (sadly, many) bpc/sequence is usually unbiased estimates — we can, here, know its var. (or var of $\ln P_i$).

[unclear] as to whether to use σ^2 of sequence $\times \frac{N}{N-1}$ or $\frac{N+1}{N-1}$ → probably $\frac{N+1}{N-1}$ .35

23

24 3.18.03 Another puzzle: for continuous predn! Say we want to code $x \pm \Delta x$ to

(be correct?) Will be like $\alpha + \frac{\beta}{\Delta}$. α, β "fixed costs" (pc proportional to $\frac{\Delta x}{\Delta}$? pc $\propto \Delta$

But suppose we are comparing 2 alternative codes for the $x \pm \Delta$. (for discrete coding)

Normally, we have a trade-off between cost of model & cost of corpus with the model. Here, we seem to have a (big, unstable variable) cost of corpus in terms of model.

30 Maybe "no problem": If we have 2 competing models ($mod_{1,2}$):

The cost of code of $x \pm \Delta x$ will be $\alpha_{1,2} + \frac{\beta_{1,2}}{\Delta}$? or $\alpha_{1,2} + \beta_{1,2}\left(\ln \frac{1}{\Delta}\right)$

or $\alpha_{1,2} + \gamma \ln \frac{1}{\Delta}$ (I think the part dealing w. Δ getting very small should be common to 2 models)

So maybe $\alpha_{1,2} = \log_2 \Delta \cdot \left(= \log_2\left(A_{\varepsilon,2} \cdot \frac{B_{\varepsilon,2}}{\Delta}\right)\right)$ → Model.

Model

35: 23 So perhaps "non problem"!

So if bits/symbol is a measure of precision of x approx: This is expected to go to ∞ as we approach a limit for stationary data). Here, I think what UPD gives is perhaps an unbiased estimate for $\ln p_i$ of next symbol. $\left(\frac{n+1}{n}\right)$

A test of my understanding of this would be to see how it fits linear progression.

UCRS

nice?

10:152.40 : Actually [redacted] pc of next symbol : or $E(\ln p_{n+1})$ would be nicer — More General.

Consider Linear regression: After we get a set of models, we get a PD on possible values of the next [redacted] element.

Actually, the pc of the next element is always about the source. The interesting thing is the Varc!

Of possible interest! Varc using "nohistory" v.s. varc. using UPD or UPD approx.

Say we have a bunch of x_i w. a Gaussian D.F. about 0, & Varc $= \sigma^2$.

D.F. $= \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x_i^2}{2\sigma^2}}$. to calc. pc of the pts: $pc = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \cdot e^{-\frac{\sum x_i^2}{2\sigma^2}} = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \cdot e^{-\frac{n}{2}}$

$= \left(\frac{1}{\sigma\sqrt{2\pi e}}\right)^n$. If our resolution is Δ, we multiply by a factor $\left(\frac{\Delta}{\sigma\sqrt{2\pi e}}\right)^n$.

So as $\sigma \to$, pc $\uparrow$. We are interested in the expected value of $\ln p_{n+1}$. (p_{n+1} = pc of $n+1^{st}$ symbol).

$\ln\left(\frac{\Delta}{\sigma\sqrt{2\pi e}}\right)$: $-\ln\sigma$ is the main interest : Not exactly! The $n+1^{st}$ symbol is for the next symbol.

it is for normal previous [redacted] elements.

13 A simple case: we have n, x_i values that are normally distributed about zero, but we don't know σ of M, the creator of data. Unbiased is zero. We study the n values, decide to (have) $= \frac{\sigma^2 n}{n-1}$. : The expected error for the x_{n+1} is $\sigma^2 \cdot \frac{n}{n^2}$ + right $\frac{\sigma^2}{n}$, because we think the mean is not zero but $\sim \frac{\sigma^2}{\sqrt{n}}$. — Interestingly, deviations due to error is linear corrs minding

$\frac{n^2}{n\cdot n-1} + \frac{n-1}{(n-1)\cdot n} = \frac{n^2+n-1}{n\cdot(n-1)} = \frac{n^2+n-1}{n^2-n} \approx \frac{n+1-\frac{1}{n}}{n-1} \approx 1+\frac{2}{n}$. $\frac{1+\frac{1}{n}-\frac{1}{n^2}}{1-\frac{1}{n}}$

for n Gaussian vars mean 0, Varc σ^2, the expected value of $\sum x_i^2$ is $n\sigma^2$.

2.921 mm .115" 389

156.33

1.2 1.06 .151 386 3.846 mm diam .95" = .43MM

19

20 Ex: Discuss 2 kinds of common error types! Mean $\ln p_{n+1}$; σ^2_{n+1} : Bayesian unbiased.

[redacted] AIP Theory says that whatever apparent method you use, you could probably do better.

T: Incomputability of UPD tells you that there is no way to know how much better.

The σ^2_{n+1} error: a T. prediction method gives a dist. for x_{n+1}, if $\hat{x}_{n+1}$ is the mean of the d.d. Then ~~mean square value~~ value of $(\hat{x}_{n+1} - x_{n+1})^2$ is a common measure of error (freedom.

For a discrete alphabet, one error criterion is the mean value of $-\log$ Prob assigned to x_{n+1} by the prediction method. negative log of the probability

For both of these ~~error~~ criterion, if one uses a purely apriori UPD, the

30 we can obtain an unbiased error using all of the data in our sample. In this respect the UPD gives results comparable with other prediction methods — except that they tend to be better than the average

- the entire sample is used for testing the models, whereas in purely a priori — No data is ~~used for testing~~ wasted for training.

In most statistical analyses

- ~~Usually~~, only practical recursive formulations are used for models. UPD uses the full ramps of recursive and/or partial recursive functions. approximations to

- By using a weighted sum of all of the prediction models tested, we obtain a somewhat better error prediction error than that obtained ~~arbitrarily~~ using the best single model. using the best single model found this far.

Nips

GRAMMAR of PREDICTION METHODS .34

[redacted]

: It will be noted that Rpinco mputability of Rio UPD does not affect in any way our use of its approximations for our use of approximations to it. our making predictions using approximations to it.

The incomputability merely assures us that it is likely that our approximation is not the best possible, and that it is impossible to know how much better our approximation might be if one spent more time looking for better models.

These remarks are true for any computable prediction models — i.e. one taking a finite time to evaluate for.

So we see that using UPD approximations has no deficiencies with respect to other prediction techniques, and it several advantages over other prediction techniques. and no apparent disadvantages. ← ?

Since more computable types of models are explored

Perhaps discuss extra time needed.

Even in other prediction techniques, more time is needed computation time is needed.

which it gives us access to a greater range of models than are normally used in statistical prediction

It necessarily takes more time to discover those models

In summary,

Not necy true! TSQ13: how good a model can one get w. Gram C's?

The only serious disadvantage to using the full power of UPD to prediction, is that the greater range of models considered necessarily take more time to discover. Automatically.

Re: Approx to AIP: T. obvious way would be to do a Lenat, type for rules for a corpus, starting w. a "reasonable" set of primitive insts: If these were ordinary arithmetic + various techniques to solve linear eqns, Mult/divide, add matrices, vectors, — (Maybe I'm just out of AIP.) My Guess is that for a corpus of any length, it would take too long.

Another approach might be to use a limited set of Pems — probly parameterized. Then express Corpus in terms of these Pems. This looks like Rissanen's "Stochastic Complexity".

An alternative method: Use dcy data strings as a TSQ, modifying pc's of instructions & defining new insts as one learns about TSQ. For SM one could do parallel Corpl (different stacks) & have definitions. Later begin to do "cross dependencies".

[? SN] It would be good to take a bunch of Predn. Methods & Make a Grammar for them, so they could be Extrapolated. Even via UMC's but meant a "less diffuse" version likely to work "set of Pems"

Prodn Methods are PEM's. Would know grammar for

These use of complete set of Grammar prodn methods of .34 to "learn alongs" a corpus (say SM) as a kind of TSQ. → see 192.01

NIPS

:30 : SN On unc outputs: If its a UIO w mc, and "stop" state is used to indicate end of an output, then the system has essentially 3 symbols, 0, 1, stop.

So why the more complex "prefix code" formalism?

.03 SN For any CPM, we can easily construct a deterministic seq. for which the CPM always assigns $p_i \approx .5$ to each of its bits (an "untractable seq"). Hvr. This is not the kind of error that the Conv. Thm limits. It is, the kind of error we can get an "unbiased error for our estimator" This is confusing. — To me is to be potential "reader".

To get an untractable seq. that becomes untractable after n bits —

How long is its dcrn? Say we have CPM approx to U.P.D. Its dcrn length is $\approx \log T$, T being its CB. So, the untractable seq. beginning has $n.h + b(T)$ bits + $\Theta(1)$ bcase.

.10 $\equiv b(T)$ (where h is the entropy/bit of μ, the source. That $b(T)$ (≈ lgT is bits to dcrn T) bitcost.

is needed to tell when untractability begins. Then the seq. can be untractable wrt. h,

So total dcrn = $n.h + b(\mu) + b(T) + O(1) + bc(n)$. $b(\mu)$ $b(T)$ indicator

If we just use the CPM to predict the first n bits, then use the "opposite of CPM" to predict (after n), then bc of seq is $bc(T) + bc(n) + O(1)$ indicator n, T.

.14 $bc(T) \to \infty$ as $T \to \infty$, but is a "small" ∞. (aurally. "small")

.16 .17 HU! bitcost is $\approx n$, so for long sequences, it can have $<< bc$ than $h(n)$, the bc of the dcrn of the "most timely" corpus.

.18 .16-.17 are not extremely relevant to any thing!

I probly should figure out just what "incompressibility" & "intractability" mean in view of .03-.18!

.20 Anyway, UPD does give unbiased estimate w. no "waste of training set" data. The cost in computing can be larger, hvr., since we do have to examine a lot of candidates. approx. We can, hvr, limit this & that happens as

.23 Rissanen does, by a priori choosing a (large) class of CPM's to consider.

.26 I Guess .21-.23 is the main "weakness" of using UPD, rather than "stochastic Prob"

.27 → (or perhaps "Stoc Pc" augmented by OSLrng.) ??!

The Listing of Poems in bc order may not be so hard: Much of My Prob has been on continuous prediction, hvr. Any method that does prediction then has a (MS error for some other error distribn) can be used. Trouble is, we end up doing Poem in "not pc order!"

.30 i.e. we do a set of # prodn models & we get MS error for each. We can get the ~~# smallest subset~~ size of the smallest total codes in the set, but this looks like we are not doing anything in "PC order" at all!

.34 If a "PSM" for continuous real inductive involves getting to peak pc thm, we can get there in pc order — the cc of the comb. involves time needed to find peak.

On the web, I did find a paper of a guy who did ≈ that. I think it took a long long time to run. He used various good optimization techniques. — But even so, we really don't know "pc of corpus wrt. Poem" until it's about "all done".

The idea of trying a PSM on the corpus (peak's in pc order) then obtaining pc of corpus wrt the Poem again doesn't give pc order for Comb.

130.00

3.21.03

NIPS

2½ cups H_2O to keep down Rice.

boil time → Micro at "2" for 45 min.

156

155½

: Things to Do:

1) Write Ivan Crosby

2) Finish revision of Kol Lecture: I don't need this for publication, but I do want it for my own research branch & Understanding. (Also for STAIR talk)

3) Write Query. about Differences betw. OOPS & α:

a) OOPS has less "memory" betw problem types ≈ only Boost..

b) How α has more ~~use~~, memory — i.e. direct theory of previously Solved probs. informat re's of takens.

c) How Boost is a kind of "Meta-α": Define meta-α - not - a problem type.

4) Make more "final" Corrections to IDSIA "Report".

Re: Wallace & MML: The correct way of thinking is that the data induces a P.D. on the models. By picking the "Best" (by exact PC) model, we "discard info" the amt of info discarded depends on how sharp the peak of the P.D. is.

I think this "picking the best" & pretending it is "true" is a common Error in Scientific Reasoning.....

2 Main uses of the discarded info:

1) Better prob values 2) Ready revision.

NIPS

1991 · 3.22.03

0 :154.40ff: So : Problems w. Sfuncts & Lsnchon Thesis:

I certainly know a lot of reasonable PEM's & for reals — But they do seem to suffer fr overt. difficulty of 155.27–.40 : It appears that the difficulty involves ideas that knowledge of the PC of f · code is "very sudden" — it's not a matter of of decreasing by a factor of 2 after we code a bit".

Well, Suppose we just try the Conds in PEM order. We can regard the problem as a variety of INV problem, in which a final output is either better or worse than previous guess.

What would a search for a v.g. PEM look like?

T. search is in $\approx$ $\frac{PC(\text{func})}{CC(\text{of Parameterization on Corpus})}$ order.

It certainly looks like an LSrch !

In doing Sfunctions on strings, rather than reals, I used that 2 input vmc : 137.00–138.25

In this case, the trials worked in really $\frac{PC \text{ of total code}}{CC \text{ of total code}}$. I could do a srch on these ratios & get .04–.09, which is perhaps what I had in mind before I discovered (137.00–138.25) !

Actually, .04–.09 doesn't directly apply to QA's on reals. — Tho I see how it could !

Say Q & A are vectors. We try various (parameterized) Sfuncts from Q to A. We find the params in which the "mse of prodn" is min (would be 1 way)

Examples of QA's:

	Q	A
d.g.	Name of person	photo of person analog
	photo	name of person digital
	real vector	real vector

Identification problem: Industrial process.

d.g. { Who is President / Bush? dig. ; Is 1 an integer? / Yes. }

(SN) How to get arrays of conds & all other arrays !

e.g. (for linear prodn. cond's) see (33) for v.g. ways to do linear conds.

Assume a parameterized form of desired function. Then, over a large set of problems, optimize the params. This would be somewhat (like a vicious vicious ⊙, since we must have known desired "parameters" : Hvr, we can use O.f. (old fashioned) to get the values : They need not be very exact. Also, these params can be used to give better param values (recursively — successive approx).

33 :153.19 Actually, situation is far better ! The analysis of vmc of 153.13–.19 was for n dims for linear regression. We can get unbiased vmc estimates (caused by 2 effects ① vmc of parameter $\hat{\sigma}^2_{\hat{w}} \approx \sigma^2_{observed} \times \frac{n}{n-m}$ m = no params.

② Since cond's are error, vmc of next pred is $\approx \frac{\sigma^2_m \, n}{n-m} \approx \text{cond's} \times \frac{1}{n-2m}$.

By picking most params ⇒ this is minimal we can know no. of params w.o. knowing apriori # params!

157.00

Bette James & Associates
Business Support Services - Since 1977

00 : 156.40

This generalizes to some extent to Non-linear, if it's "locally linear":

Two cases of "Not locally linear"

.05 1) we are at a local (not-global) peak — Not a peak corresponding to true M.

.06 2) we are at a point of discontinuity or ^ where no second derivative exists.

(1) & (2) are common in approx. modeling of N.L. funcs. In Neural nets we can use nice $\frac{1}{1+e^x}$ type functions.

I don't know if 153.13-.19 applies to linear "curve fitting" à la Cureton .05-.06 do not interfere), non linear curfit, tho it would seem to, from cursory analysis

→ linear curve fitting: funct = $\sum z_i F_i(x)$ wrt z_i's only, z_i.

Can the analysis of 153.13-.19 apply to other curvefitting problems: e.g. Clustering? —

In which we have to derive (parameterize) the "distance" function

NB 153.13-.19 gets var for each complete Model, so it can get relative wts: $\propto \frac{1}{\sigma^2}$.

Can we translate this into an equivalent apriori of models?

$\left(1 - \frac{2m}{k}\right)^k \approx e^{-2m}$ is equiv pc of model. — for a m coef linear model.

So e^{-2} is pc of each coef! Could this be used as a "ruthlessly" measure of pc. of continuous params? Not so simple! $n\sigma^2 \to n\sigma^2 \cdot e^{2m}$

$\frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-x^2}{2\sigma^2}}$. from 153.08 pc of comps w. vars σ^2 is $\left(\frac{1}{\epsilon\sqrt{2\pi e}}\right)^n$ (n data in comps)

$\sigma^{-n} \to \left(\sigma^2\left(1+\frac{2m}{n}\right)^{\frac{2}{n}(n)}\right) \approx \left(\sigma\left(1+\frac{m}{n}\right)\right)^{-n}$ so $\left(\frac{1}{\sigma}\right)^{-n} \to \frac{1}{\sigma^{-n}} \cdot e^{-m}$

So each param has pc = e^{-1}

If the precision of the observations of data (Σ"x") is Δ, this is $\left(\frac{\Delta}{\sigma}\right)^n \cdot \frac{1}{e^m}$.

I certainly don't feel confident about the "significance" of this e^{-m} result!

It's a statement, not about a single code but about a set of codes w. m coefs.

A different way to use this linear regression result: I twice obtained (I think in 1962!) when I integrated all possible linear models. Also integrated σ^2's I think. I may have used uniform p.d. for coefs (even tho it didn't converge). — The idea being that I could choose a "flat" a priori d.d. so that it would give the same results as "uniform" (divergent) d.d.)

Anyway, I could do the same thing w non-linear problems (having diffy of .06, but not .05) (bad conditioning) (local optima)
or the (144.00, 4.00) Clustering algorithm (assigning the "distance" function).

→ (spec. 158.00)

727 Massachusetts Avenue, Cambridge, MA 02139-0019
(617) 661-2622 / Fax 661-0802
E-mail: James678@aol.com

NIPS

$n(\sigma^2 + |r|^2)$.

> 0: 157.60 : Intro (drawn regression, kernel) case! We have a point in m-space; Plane is an n-space of cents, $\vec{r}$ running.

The n space is a rectangular space with extreme point at a distance d from it.

($d \equiv \sigma$)! The point in $(n+1)$ space is n pts in m space, so P is distance d from origin,

It is at distance $X = \left(d^2 + |\vec{r}|^2\right)^{\frac{1}{2}}$ from any pt. $\vec{r}$, in m-space.

$e^{-\frac{(x-x_1)^2}{\sigma_1^2}} \cdot e^{-\frac{(x-x_2)^2}{\sigma_2^2}}$ If 2 d.f.'s have zero first moments & sum of 2 d.f.'s has sum of second moments.

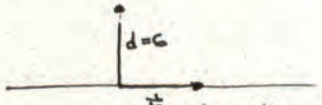
$\frac{1}{\text{wt.}} \frac{X^2}{\text{2nd moment}} = 1$ (X = distance from P.) so $\int_{-\infty}^{+\infty} X^2 \cdot \frac{1}{k^2}\, dx = \infty$. (bad!)

No! wt. = $\frac{1}{\sqrt{2\pi}\,d} e^{-\frac{X^2}{2d^2}}$ (?) so multiply by X^2 and $\int_{-\infty}^{+\infty}(\quad) dx$?

σ^2 = constant $= d^2$.

We have a bunch of pairs of form $\frac{1}{\sqrt{2\pi}\sqrt{\sigma^2+|r|^2}} e^{-\frac{(x-d)^2}{2(\sigma^2+|r|^2)}} \cdot e^{-\frac{(x+d)^2}{2(\sigma^2+|r|^2)}}$

→ second moments of $2(\sigma^2 + |r|^2)$. wt. = $\frac{1}{\sqrt{\sigma^2+|r|^2}}$



at pt $\vec{r}$, have for next product $\sigma^2 + |r|^2 = \sigma^2 + r^2$.

wt. = $\left(\frac{1}{\sqrt{(\sigma^2+r^2)2\pi\epsilon}}\right)^n$

$\propto \int_0^\infty dr \; \frac{2(\sigma^2+r^2)}{\left(\sqrt{\sigma^2+r^2}\right)^n} \cdot \left(\frac{1}{\sqrt{2\pi\epsilon}}\right)^n$ $\therefore \int_0^\infty \frac{dx}{\left(\sqrt{\sigma^2+x^2}\right)^{n-4}}$

$\int_0^\infty \frac{dx}{\left(a^2+x^2\right)^{\frac{n}{2}-2}}$ $\int_0^\infty \frac{dx}{(a^2+x^2)^k} = \frac{2}{a^{2k}} \int_0^\infty \frac{dx}{\left(1+\frac{x^2}{a^2}\right)^k} = \frac{2}{a^{2k-1}} \int_0^\infty \frac{dy}{(1+y^2)^k}$

$= f(k)$

$\frac{n}{2} - 2 = k$

$a^{-2k} f\left(\frac{n}{2}-2\right) = a^{1-n-4} f(\quad) = a^{-n-3} f\left(\frac{n}{2}-2\right)$

$\int_0^\infty \frac{dy}{(1+y^2)^k} = f(k)$

P 294 Gradshteyn: 3.249 $\int_0^\infty \frac{1}{(x^2+a^2)^n} = \frac{(2n-3)!!}{2(2n-2)!!} \frac{\pi}{a^{2n-1}}$

$(2n)!! = 2 \cdot 4 \cdot 6 \cdots 2n = n! \cdot 2^n$

$(2n+1)!! = 1 \cdot 3 \cdot 5 \cdots 2n+1$

$= \frac{(2n+1)!}{2^n n!}$

$\int_0^\infty \frac{dx}{(1+x^2)^n} = \frac{\pi}{2} \frac{(2n-3)!}{\left((2n-2)!! \cdot 2^{2n-2}\right)^2}$

$1 \cdot \frac{3}{2} \cdot \frac{5}{4} \cdot \frac{7}{6} \cdots \frac{2n+1}{2n}$

$\prod_{i=1}^{n}\left(1+\frac{1}{2i}\right) \simeq \exp \sum_{i=1}^{n}\left(\frac{1}{2i}\right)$

$\sum_{i=1}^{n} \frac{1}{i} \approx \ln n + \gamma$?

$\simeq \exp\left(\frac{1}{2}(\ln n + \gamma)\right) \simeq e^{\frac{\gamma}{2}} \cdot \left(\frac{1}{2}\ln n\right) e^{\frac{\gamma}{2}} \cdot \sqrt{n}$

See Gr & R. Bottom of p 295 for asymp. correction

$\left(\frac{\pi}{2}\right)^{\frac{1}{2}} \cdot e^{\frac{\gamma}{2}} \cdot \sqrt{n+\frac{1}{2}}$

$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{dx\,dy}{(1+x^2+y^2)^n}$

$x^2+y^2 = z^2$

$\int_0^\infty \frac{2\pi z}{(1+z^2)^n} dz$

N.B. Gr & R p 295 Bottom of page $\left(\frac{2j!!}{2j-1!!}\right)$

$\int_0^\infty \frac{z^{2m}}{(1+z^2)^n} dz$ } See Gr & R p 295 (4.25.)

Note Clustering is a BAG induction problem! What corresponds to Var (in time sense regarded)? in BAG induction

One way: if P_k functions are very narrow — In 4.3 ($\equiv$ (4 3 1/2 3 2)), the total second moments of other f functions are smaller. No: total second moment about the mean ($\equiv$ first moment).

"How: What is criterion if one introduces more (or less) cluster centers!!?

Also Note! In the $\sigma^2 \to \sigma^2 \frac{n+1}{n-1}$ discussion of the μ, σ^2 model — was this a 1 or 2 $\geq$ param model?

Nips.

0: 158.40: If it was a 2 param model, t. extrapoln to m coins would be $\sim \frac{n+m}{n-1}\sigma^2$ — quite a defeat from t. Old Maxm results of ~1962, 1972. It certainly seems like a 2 param model.

* one. Floating pt notation for numbers: (From Alex): A 32 bit string gives a set of nos. of = pc: It is a ruffly "log" distribn, but cutoff on hy & low ends. It is symmetric about $\pm x$. It is a reasonable d.f. in many Physics problems. In many cases, we know, a priori, whether it is + or −.

For other cases, t. d.f. betw. —I don't I would it be so dense— More dense than say 100 to 101, but nothing like floating pt!

I think t. moral is that for each situation, one must learn what kinds of d.f. to use. This "learning" can involve logical reasoning.

---

1 SN T. incompatibility of UPD is a real hearing, it has nothing to do w. t. applicability of UPD. The really serious problem is that t. UPD has to be "learned", that coding considerations custom its form, but not entirely as aspects

15 of its form, but essentially, much of its form has to be "learned".

T. Math community is enchanted w. t. idea that it is known "within a constant factor". This fact is usually of little value by itself. → 160.00 ...10

20

For t. p. an (positive integer) $\log^* n$ is not bad. For distributions on reals from 1 to ∞, perhaps $\log_2^* x$ is tolerable (Problem (up to base 2 is somewhat arbitrary))

Levin said it doesn't converge for $\ln^* x$. Not quite reasonable!

$\frac{1}{n}$ div, $\frac{1}{n^\alpha}$ conv. $\frac{1}{n \ln n}$ div $\frac{1}{n(\ln n)^2}$ conv? $\frac{1}{n(\log_2 n)^2}$ converges $\therefore \frac{1}{n(\ln n)^2}$ converges.

$\frac{1}{n}$ $\frac{1}{\log_2 n}$ $\frac{1}{\log_2 \log_2 n}$ diverges $\frac{1}{n}$ $\frac{1}{\log_2 n}$ $\frac{1}{(\log_2 \log_2 n)}$ conv.

$\ln(1+x) = x - \frac{1}{2}x^2 + \frac{1}{3}x^3 \cdots$

$\log_2 \log_2 n$ v.s. $\ln_e \ln n$ $\quad \log_a A = \ln A \cdot \log_a e$

1 constant $\quad \ln \log_2 n = \ln \ln 2 + \ln \log_2 e$

0 So (?) $\frac{1}{n}\frac{1}{\ln n}\cdot\frac{1}{(\ln \ln n)^2}$ div. but $\frac{1}{n}\frac{1}{\ln n}\cdot\frac{1}{(\ln \ln n + \text{constant})^2}$ conv?

If taking more than triple logs, we consider only large values of n. — small values give rise to irregularity values.

Ratio of forms: $\frac{n+1}{n}$, $\frac{\ln(n+1)}{\ln n}$, $\frac{\ln \ln(n+1)}{\ln \ln n}$.

$\frac{\ln(n+1)}{\ln n} = \frac{\ln n + \ln(1+\frac{1}{n})}{\ln n} = 1 + \frac{\ln(1+\frac{1}{n})}{\ln n} \doteq 1 + \frac{\frac{1}{n} - \frac{1}{2}\frac{1}{n^2}\cdots}{\ln n}$

" $\bar{x}^{\alpha}$: $\frac{(x+1)^\alpha}{x^\alpha} = \frac{x^\alpha + \alpha x^{\alpha-1}}{x^\alpha} = 1 + \frac{\alpha}{x}$ ≡ converges if $\alpha > 1$.

3.26.03

Nips.

0 : 159.40 : 159.11 – .15 (on necessity of long epoch) seems reasonable! So how come we have

t. ~ $\frac{n+m}{n-m}\sigma^2$ [redacted] criterion for sequencial predn of reals(s)??

→ Note Kaz 26 on "Akaike"

---

03 T. The Thrust of 159.11 – .15 is that t density function params must be 'trnd" — i.e. optimized over a large set of subsequences. There will be a hyper order of dist. that must be chosen a priori, but probably this will have a "reasonable" Bayesian that seems "obvious":

In a practical case, one could choose t apriori on density or a meta dist. on distribns of params defining densities. It might be first necessary to decide on what t ranges were ($-\infty \cdots +\infty$; $0 \cdots 1$; $0 \cdots +\infty$; $1 \cdots +\infty$; Integer or reals, cont.)

It may be that there aren't a [redacted] large no. of cases,

Consider the way one normally does statistics! We have as data, a bunch of measurements of the length of a certain object. From apriori physical statistics, we expect a normal dist. about some mean. Using no more apriori than that, we take the data fit a Gaussian to it, & get a μ & σ^2 for it.

Say we chose a uniform apriori for μ $(-\infty, +\infty)$ & σ^2 $(0,\infty)$ · (or σ? $0,\infty$)

2 the probability of the model μ, σ^2 will produce the data $\{x_i\}_1^n$ is

$$\prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} = \left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\bar{x}-\mu)^2}{2\sigma^2}} e^{-\frac{s^2}{2\sigma^2}}\right)^n$$

$\bar{x}, \left(\overline{x^2} - \bar{x}^2\right) \equiv s^2$

This is max when $s^2 = \sigma^2$ & [redacted] $\bar{x} = \mu$.

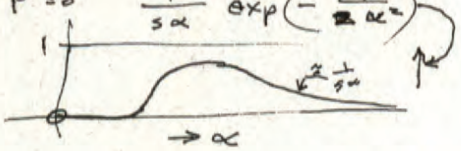
just look at $\frac{1}{\sigma}\exp\left(-\frac{(\bar{x}-\mu)^2 + s^2}{2\sigma^2}\right)$ as a func of μ & σ^2.

$= \frac{1}{\sigma}\exp -\frac{\frac{(\bar{x}-\mu)^2}{s^2} + 1}{2\frac{\sigma^2}{s^2}}$

Use new variables $\frac{\sigma^2}{s^2} = \alpha^2$ $\beta^2 = \frac{(\bar{x}-\mu)^2}{s^2}$

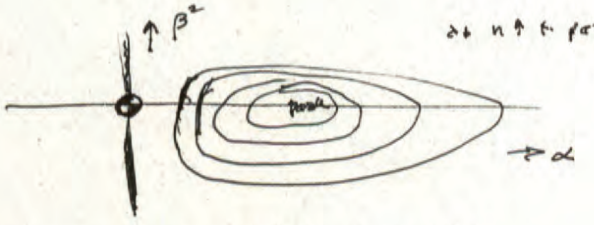
$\frac{1}{\sigma} = \frac{s}{s\alpha}$ $\left(= \frac{s}{s\alpha}\exp\left(-\frac{\beta^2+1}{2\alpha^2}\right)\right)$

for $\beta^2 \geq 0$ $\frac{1}{s\alpha}\exp\left(-\frac{1}{2\alpha^2}\right)$



for constant α; we have $\left(\exp -\frac{\beta^2}{2\alpha^2}\right) \cdot \frac{\exp\frac{1}{2\alpha^2}}{s\alpha}$

i.e. a version of width α — for tiny α very narrow for small α — but



as n ↑ the peak gets sharper & sharper. Its t-apriori of α & β^2 is assumably, $\bar{x}-\mu$ & σ^2.

impossibly unlikely for small α.

If the peak is very sharp, the exact apriori for μ & σ^2 is not important — It need only be "smooth" "smoother"

We also have: Marginal "dist" of α & β^2.

Marginal dist. for α (β^2), integrates out (α β^2)

3.26.03

Nips

SM

00 :160.40: → Note that a priori is important mainly if $\Sigma\Sigma^2$ is small.

Also: If a predn. method gives a very narrow output (small $\Sigma p_i \ln p_i$) this means it's very confident — not necy right! Hmm, if it defines "width" as $\Sigma p_i \ln p_i$ is unbiased. This may be what we want!

Re: is $\exp \Sigma p_i \ln p_i$ a probly wt.? $= \prod p_i^{p_i}$.

So Pr is the Q: Is the unbiased width for the D.F. related to the wt. of the pam? ← (i.e. its pc)

To do in directly, getting pc of pams using ideas of 160.03ff, is possible, reasonable, but often difficult. Getting $\Sigma p_i \ln p_i$ may be easier: If seems easier in linear regression/curfit. ∴ "width" of d.f.

$$\int_{-\infty}^{+\infty} \frac{1}{\sigma} e^{-\frac{x^2}{2\sigma^2}} \left(\ln\frac{1}{\sigma} + \frac{x^2}{2\sigma^2}\right) dx$$

$$\alpha \int_0^\infty z^2 e^{-\frac{z^2}{2}} \left(\frac{dx}{\sigma}\right) = dz$$

$$\ln\frac{1}{\sigma} + \alpha \cdot \frac{1}{2} \text{ const.} \to \frac{1}{\sigma} e^{\alpha x}$$

$$\int_0^\infty x^m e^{-\frac{x^2}{2}} dx = 2^{\frac{m-1}{2}} \left(\frac{m-1}{2}\right)!$$

G&K p357 top of p. note.

10 SN SM V.g.! To compute info in a strat: Usually a strat will contain several continuous params & its dsrn will involve (a few discrete) dis crete params. The continuous params can be evaluated by running the data repeatedly on "near by" pts, & getting empirical "Hessian". The discrete params may be randomly sampled ∴ ~~[illegible]~~ cost nearly. Some discrete params will ~~be optimized~~ have been empirically chosen & have associated bit costs.

~~We info~~ If there are k bits of info in a strategy (obtained this way) & its empirical yield is y, then its expected yield is $y \cdot 2^{-k}$.

~"Obtaining info in Hessian" is not clear — perhaps guess at a priori width of distribution — of ranges of the continuous params.

20

In my latest strategy (that seemed to work) there were (I think) 2 continuous params:

① "Buy" threshold & ② smoothing constant (w 3 yrs): ② was rather non-critical.

① was a bit critical w.r.t. large losses. Yield was rather discontinuous in that parameter! & gains were very inversely w. value of ① (Therewere other params (incl. no. of stocks in the "satisfactory candidate")

Re: 21: When doing lower thresholds & higher thresholds, try to find aux. properties of both that & reliability of acceptance/rejection. Since buy is an AND operation, if ↑ n cost (badly & its aposteriori ($ ↑ u cost) also new data needed to test.

Any way, it could ↑ yield a lot

30 Some "other aux properties": Correlations (+ or −) w. other stocks/indicators.

& Self correlation (obtainable if I really had "on-line" access to a "broker")

"Cure Cancer" (.18)

"i-th" approx to $P_M \equiv P_H^*$

:: London Lecture re visions: On "incomputability": What we don't have (i.e. what "incomputability" means) is knowledge of how close $P_i(x)$ is to. ~~Must~~ $P_M(x)$. Furthermore, we don't know how close $P_M(x)$ is to $M(x)$ but we (May) suspect it's a decreasing concept in given corpus.

Inference If we knew how close P_i was to M we would have idea as to whether we should spend time improving P_i. We can sort of know this from past experience, but it's a very uncertain knowledge.

However in practice (if) we don't often ask that Q. what we usually ask is "how good can I expect $P_i(x)$ to be in the future? We can answer this — given an unbiased estimate of variance or equivalent (Is it equiv.?). → see 186.16

I do have to go over this.

Another tack: If we consider only primitive recursive functions (which we normally do, anyway) we still couldn't practically estimate how close $P_i(x)$ was to $P_M(x)$. I'd like to go into the details of this: It would make meaning of "incomputability" much clearer!

---

18 SN on "Reinforcement" Cure Cancer & "Dynamic ping problems". (Machines)

We could get TM to learn to solve "in steps" on to future "cure cancer" problems.

Note: The working on "LARGE Problems" should learn to solve "cure cancer" by itself! It can be formulated as a very hard or problem w/ large (Rothstein) CB

20 Start w/ 1 then 2 then 3 ~~often~~, etc. We can look at the particular kinds of "microfunctions" that have need to be optimized & even the lots of training in that area. The no. of trials needed for optimum soln. is, I feel, excessive, but PST's can be discovered (by TM) to deal w/ it.

SN on Time Varying OZ problems. Say we have a bunch of PST's for our problem, & we have a probby of soln.... → 176.02 is a fairly Good soln.!

176.02 beats Time varying OZ # for good soln. use WON

22 SN ~~Big~~ Big report on OZ probs: I don't see how phase 2 works! → See 175.13 ~~Yes~~

If would seem that we have a pd on 2 variables; t & G (not just G)! Maybe we use t to find time limit t. Then, if we change PST's, we've used up part of T, so t remaining T, is changed! Also, if we use Lsearch, T_i T's will be different for each PST. — so it looks like we need $h'(T, G)$ a "double" probby density?

30 (3.24.03) It would be well if we could invoke t-theorem about $\frac{Pc}{cc}$ order being best.

33 Say we initially ordered them using same T_0 (i.e. one init. problem dom.) — so each PST would have a G d.p. & we could order them in "probby of weak G" order — for T_0.

34 A complex strategy: Be initially ordered as .34 using T_0 as index (first) — working "best bet". While working on it, do "updating" in II (time shares), so T, G, Pc curves

Nips

t- "Akaike factor" .26

Looks like Good explanm, understanding.

00:00 Of all PST's very change. As one works, $T_0 \to T_r$ (remaining time).

We always recordan PST's w.r.t current T_r. When a new PST become "best bet", we switch to it. We can switch back to a previously discarded PST, — retaining it work (cc) spent on it.

Another way to think about it: for each G value, have it a $\frac{PC}{cc}$ ordering among PST's, thereby crudely "optimum". Hvr, I don't see how this would help solve t. problem: perhaps: for each G value, have it a Σpc_i : t. total probly that one would solve a problem with t. next G value.

ANY way 162.27 ff: (Just how Phase 2 does OZ problems) is at present, very unclear.

10 Conceivably, a better understanding of it would enable me to outline a Soln. to t. "Time varying OZ" problem.

11 It is a problem that Must be Solved. → 165.00

"Things to do" $155\frac{1}{2}$.00

153.20 - 154.18: Derbs arrows in PEMS: whether to waves of Zeno advances is unbiased for UPD is unclear.

↑ Main Problem

151.26 - 152.12: Error is incomputability $P_M^T(x)$; $P_M(x)$, $M(x)$. } That "incomputability of UPD" is/(unimportant, is clear

They is perhaps a directly utilizable wrinkle.

20. Re: Unbiased error for $P_{nc}^T(x)$! What they means is unclear: Say we have k symbol alphabet & we are doing digital predn! Use just 2 pc for each symbol. Say in some sense, it is "unbiased": Then, if we assign real nos. to each symbol we can do this in arby ways: It would seem unlikely that all assignments would end up being unbiased! (?)

.26 Still perhaps easy way to analyse/explain/understand σ^2 of regression → $\sigma^2_{obs} \times \frac{N+m}{N-m}$:

Looks good! Say σ^2_M is true σ^2 of M. $\sigma^2_M \cdot \frac{N-m}{N}$ = σ^2_{Mc} = $\frac{\Sigma e_i^2}{N}$ is expected observed variance over past: $(<\sigma^2_M)$

Since coeffs are wrong, σ^2 for nearest x_{n+1} is $\approx \sigma^2_M \cdot \frac{N}{N-m}$ $> \sigma^2_{Mc}$: T. product is $\approx \sigma_M$ (true) $\cdot \frac{N}{N-m} \cdot \frac{N}{N-m}$

See 164.12 ff. for more exact analysis

It may be that they just Gives "Akaike's effect" — it does not give usable cost of telling how wrong coeffs are over t. form of t. regression eq. → 164.30 "real"

30 Hvr. .36 may be an aspect that is always usable for t. continuous limit of a Random Walk.

It assumes uniform a priori for coeffs, & probly uniform a priori for σ^2 (?)

To generalize from 1 parameter σ^2 to n linear coeffs: t. single parameter can be located at t. origin for t. constants, & so it generalizes easily! → 164.01

.34 The exact mechanics of how to uniform apply out coeffs getting Akaike result is unclear.

It may be simply integrating like in Maxim 1962! (see 153.00 ff for a try at it!): I think there may be a very simple way to see it by rotating Markovs in M space — which is what I was thinking about in 1962 (and off 1972). Perhaps t. idea is that all spin matrices can be

:20: 163.40 rotated so Pmy one dimensional.

:01: 163.34: σ_μ^2: observed mean var = $\frac{(n+m)}{m}\sigma_\mu^2$

predicted var for X_{n+m} is $\sigma_{dm}^2 \frac{n+m}{n}$

∴ predicted var = observed var $\cdot \frac{n}{m-n} \cdot \frac{n+m}{n} = \sigma_{obs}^2 \cdot \frac{n+m}{m-n}$.

σ_μ^2

I still have to work out the details! Consider vectors in m space; Data 2nd vectors in n space. Models are on m dimensional subspace. We try to fit model vector as close as possible to data vector. This is "projection" of data vector in to m space.

T: / true model / ($\vec{\mu}$ data) is a vector in t. m space., plus a random / isotropic Gaussian n vector:

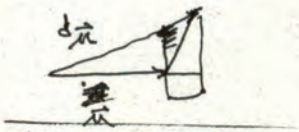
So the projection of t. ($\vec{\mu}$ data) vector, is it's correct m vector plus an n dim. isotropic vector. $\equiv \vec{\mu}$

$\vec{P}_{\vec{\mu}} = \vec{m}_{\vec{\mu}} + \vec{n}_{\vec{R}}$ (m dim, m dim, n dim) | Maybe obvious notation!

$\vec{\mu}$ = correct model in m space. $\vec{\mu}_m + \vec{R}_n = \vec{d}_{\vec{\mu}} = $ data vector. $\vec{\mu}_m + \vec{R}_m = $ projection of $\vec{d}_{\vec{\mu}}$ into m space.

Σ sq. error betw. $\vec{\mu}$ and $\vec{d}_{\vec{\mu}}$ is n ($\vec{R}$ have $\sigma^2=1$ for simplicity diagonal cov)

$|d_\mu - \mu|^2 = n$; i.e. $\vec{d}_{\vec{\mu}} - \vec{P}_{\vec{\mu}}|^2 = n-m$ ← impt. result.



This eq. gives derivation of $\sigma_{observed}$ from σ_{true}.

smaller

$(n-m)$ (n)

:19 How do we take $\underset{m}{\mu}$ and $\underset{m}{P\mu}$ into n+1 space (1 new data pt.)? They remain the same. (?)

:20 $\underset{m}{d\mu}$, however has an extra dimension.

Do $\underset{m}{\mu}$ in $\underset{m}{P\mu}$ remain the same? They still have only m dims. The new data pt. in n+1 vector is obtained from the old data n vector, by adding a small sub vector to it,

:20:164.34 Realization! For long data sequences, the contribution of the precision of the coefficients to the overall pc of the code, Becomes dominant over other coding costs.

i.e. → look at expression for bound in Sol 86.

(spec 163.11) Re: the Phase 1 OZ problem "soln": The Lsrch soln is really not so great! We have time limit $T(CB)$. If $P(*)$ is the pc of P if $P(PST_i)$ is the pc assigned to PST_i then we spend $T \cdot P(PST_i)$ on that PST_i. Note, here that $P(PST_i)$ is the probty (perhaps) that PST_i will beat G in time T, (not in time $T \cdot P(PST_i)$). T "measures" the soln. is that we spend at least $P(PST_i) \cdot T$ on the "best possl. PST."

Say we had the $P_i(T, G)$ curves for all PST_i's wrt the particular problem of interest. How much time to spend on each PST_i, $\Rightarrow$ the max G obtained from all trials was Max?

It may be (not unlikely) that the soln. is to put all time into one PST_i! — That they would have a greater expected G than distributing cc among the PST_i's "more evenly".

→ One (perhaps serious) Advantage of distributing cc among the PSTs is that we get more info about them, (to be used in future problems).

One "way out" of .09, is that for many PST's doing OZ problems, they undergo many trials. In view of these trials the lots of learning contains places the $P(G,T)$ curves of many PST's can change so we can usefully jump from one PST to another W.O. "washing" losing much cc.

So: The moral is: If I would be satisfied w. the Phase 1 soln of OZ of .00-.01, then a simple "Phase 2" version (i.e. $P(G, T_0) \equiv P_{T_0}(G)$ curves ← T: cc to read out OZ problem down) is the root is very similar to soln. for INV probs as described in §2 of "Preliminary Report".

On the other hand, the .00-.01 soln for OZ Phase 1 is not so hot & ideas of .06 - .18 suggest ways to improve both Phase 1 & Phase 2.

Improving Lsrch over OZ problems is a hardware problem. Even WON for INV probs.

(SN) On the other hand, if we use 2 or 3 param $P(T)$ curves. ( ) as we probably will, the optimum WON solns may be possl. for INV at least.

(Also note, WON solns so far do not allow learning during trials for even between trials!) → (for WON → 166.00)

WON

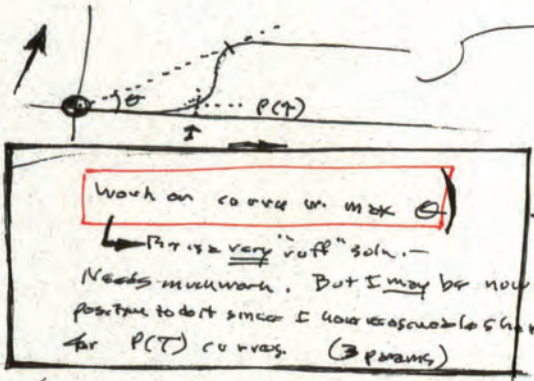
20:165:32 ! A possible WON soln. for Gaussian $P(T)$ curves: or "gamma d.f."

Gauss. : $p(T) = \frac{a}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(T-T_0)^2}{2\sigma^2}\right)$: params a, T_0, σ^2. ($T_0 \pm$ mean) — doesn't have proper origin at $T=0$.

Gamma $p(T) = a k T^n e^{-bT}$: params a, n, b. — does have kink at or almost $T=0$. finding zero or first & second inflection is easy, hrr.

↑ normalizing constant: involves $\frac{1}{n!}$ or $\frac{1}{\Gamma(n)}$.

2 tasks: which to work on to finish first (OR next):



$\int_0^T p(T)\,dT$

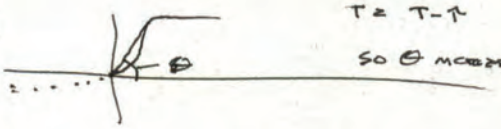
this is ≈ trivial soln. Scan 169.35–170.07 for details, proof

after moving on Past curve for time T

We have new curve $P = P(T-T)$

$T = T - T$

so ⊖ increases at first.



eventually, ⊖ hits a peak and then begins to ↓. The peak of ⊖ is when slope of (.10 ⊖) curve is max — after ⊖ is past, we jump to another PST's $P(T)$ curve

or look at it this way:

as we incrementally curve, we always have a ⊖ that gives slope of steepest line thru present point, that intersects $P(T)$ curve.

Note for Gamma d.f., $P(T) = \frac{1}{\Gamma(n)} T^n e^{-T}$

$\int_0^\infty T^n e^{-bT}\,dT$ normalize: $z = bT$: $\frac{1}{b^{n+1}} \int_0^\infty (bT)^n e^{-bT}\,d(bT) = \frac{\Gamma(n)}{b^{n+1}}$

$f(n) = T^n e^{-T}$

$f'(n) = n T^{n-1} e^{-T} - T^n e^{-T}$

$= n F(n-1) - F(n)$

$\int F(n) = n \int F(n-1) - \int F(n)$

$\frac{M_0}{a} = 1$ | a normalized function is $\frac{b^{n+1}}{\Gamma(n)} T^n e^{-bT}$.

$\frac{M_1}{a} =$ | zeroth moment is $\frac{a\,\Gamma(n)}{b^{n+1}}$

$\frac{M_2}{a} =$ | first " "

$a = M_0$

$\mu = M_1 / M_0$

$\sigma^2 = \frac{M_2}{M_0} - \mu^2 = \frac{M_2}{M_0} - \frac{M_1^2}{M_0^2} = \frac{\left(M_2 - \frac{M_1}{M_0}\right)}{M_0}$ ← linear dominant.

zeroth moment of $a T^n e^{-bT}$ is $\frac{a\,\Gamma(n)}{b^{n+1}} = M_0$

first moment is $\frac{a\,\Gamma(n+1)}{b^{n+2}} = M_1$

second moment $\frac{a\,\Gamma(n+2)}{b^{n+3}} \equiv M_2$

WON

:00 : 166.40 : Consider 2 person PCT's : 

ordering choices by $\frac{a}{u}$ is not bad.

Trouble is, if we are working in many PST's in (), we don't get any results until time u.

Say I have a bunch of PST's like Rich's. What order should I try them in? I guess that there is no gain in switching between PST's before u is spent.

My Guess is $\frac{a}{u}$ ordering. Since $a \leq 1$, we can't have a very large u w.r.t. large $\frac{a}{u}$.

09 e.g., t. expected time to solve for $\underset{u_1}{\_\_\_}|a_1$ followed by u_2 $\_\_\_|a_2$

T. first Gamb H. (hem. may prove this more exactly!

10 (a) $u_1 \cdot a_1 + (u_1+u_2)\, a_2$: $= u_1 \cdot (a_1+a_2) + u_2 a_2$ $= a_1 a_1 + u_1 a_2 + u_2 a_2$

11 (b) reverse order $u_2 a_2 + (u_1+u_2)\, a_1$ $= u_2 (a_1+a_2) + u_1 a_1$ $= u_2 a_2 + u_1 a_1 + u_2 a_1$

So $u_1 a_2$ v.s. $u_2 a_1$ or $\frac{a_1}{u_1}$ v.s. $\frac{a_2}{u_2}$ as expected;

This I'm not quite sure that [redacted] t. .10 v.s. .11 comparison is legit!

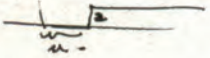
15 For Inv, probems, this suggests that [redacted] trials in $\frac{a_i'}{u_i'}$ order would be best — this or a_2 (t. Rnd param) is not zero, we might want to use $\frac{2a_i'}{u_i' + 2a_i'}$ or $\frac{a_i'}{u_i' + 3a_i'}$

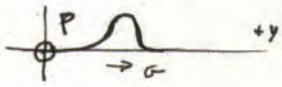
17 for [redacted] ordering is the $u_i + 3a_i$ for testing. see → 169.00 ff for much better soln!

169.00 quoting Pat "solve"!

Russ: "stochastic complexity".

T. Fung. seems to "solve" t. INV. problem!

20 what about OZ probs? I would guess Rich t. P(T,G) curves would be much distant. from 

Maybe for each T value, a 

type curve.

Is proof of .09 for min expected time of Goln? → 169.19 - .20 Pace

One strategy: for each PST, do a T_2 such that we expect some G.

We may go a bit back that.

Perhaps better way: Pick PST w. max expected G in $a_2 = T_0$ ($\equiv CG$) ($\equiv PST_0$)

As we work on this, t. remaining time is $T_0 - T$. We simultaneously ~~[redacted] expected G~~ Reevaluate "all" PST's to see which one will give best G in remaining $T_0 - T$ time.

We jump to PST that is best.

As we worked on PST_0, t. P(T,G) curves of Rich PST's changes since Rich (optimum strategy optimizer) function that maps a PSTS into {P(T,G) curves} changes as we work on PST_0.

For time being, use 165.00 -.01 for Phase 1, OZ & use 165.19 - .22 for Phase 2 OZ

The discussion of OZ is on 165.00 -.40

For INV probs; [redacted] .15 - .17 looks v.g: it's not quite clear better v.g. But 169.00 -.40 + 17000 -10 is even better! for INV space.

Can I use something like it for Phase 1 induction? → 168.25

Nips

Context.oo

9 pk.

10 : : on "Context:" in Phase 1. I had been thinking of; for each token, it would have distinct pc's in each "Context". So if Contexts were each a component of the dimension, w. k contexts, each token has an assoc'd dim vector of pc's.

It looks perhaps better to do it in reverse: for each Context, have a vector giving pc's of all tokens: A

At any rate, we have a k x L matrix: k contexts, L distinct tokens. I think I've seen in OOPS that of ~~a~~ Vectors of pc's of tokens, being stored for trial accesses: That is, the context vectors were to tokens.

More Generally it had OOPS "computing" the pc of each token, each time a new token was chosen. Sounds very general. Hmm, he only had maybe 5 instr that did token pc motion, & he only actually used 1 (boosting) in successful pgms

The "context" matrix also includes ~~css~~ info - (numerator/denominator)

If >1 context applies to a situation, it's hard to ~~add~~ add up the vectors because the css's can overlap — this Gives "double counting"!

The OOPS' "boosting" gives the equiv of a context, the system (as is) does not associate any features of the problem w. an idea as to which "q" to "boost" (a "q" is a frozen pgm). When I mentioned to him how pc ~~was~~ of (frozen pgm / particular frozen pgm) a / frozen pgm if the Machine is "Mature". Jerry suggested that the system would have to develop its own ways to "narrow down" the frozen pgm to be "Boosted".

I should go back & read my stuff on "Gened content" — just how it was defined & ~~why~~ just what it's Gorcs — "what is it trying to do" M(early) discussion of "OOPS" compared it "computation of pc's of tokens" w. my "context". At first his method looked better: but then my Gened idea of "context" began to look v.g.: try to find Prototypes.

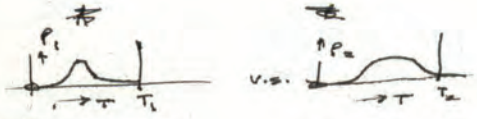
25 : 167.40 : on ~~a~~ à 172.09 / à 170.09-.07 / 167.15-.17 for Phase 1 induction: For phase 1 induction one simply lists ~~the~~ tokens in pc order. This can be somewhat modified by Laplace's rule & the discovery of sub-tokens. Sub tokens can become more useful if we have several domains of induction, w. "Indexing" on the Q's. This means that the token defs for each index can be different yet may can have common subtrees (a common regular tokens as well?).

30 Note in trying to find $O^j \Rightarrow a_0^j \prod_{j=1}^{n} O^j(A_j | Q_j)$ is max, A ~~token~~ PST is allowed to look at all of the (A_j, Q_j) pairs: Hmm, a_0^j, the pc of O^j will be indep of the data.

4.03.03

Nilps

WoN

SO: 167.17: Generalize to: 

Which order to try them?

$\int_0^{T_1} T\, p_1(T)\, dT$: assoc w. p_1 we have ~~$\frac{1}{2}$~~ expected soln. time. & $\int_0^{T_1} p_1(T)\, dT$ prob'ty of any soln

Anyway: After solving Prob one, say we have 2 tasks α_1, α_2 w. finite time cutoffs, ~~one~~ on T_1, T_2, resp. We can treat α_1 & α_2 as

$T_1 = T_2$ resp, so we could do:

α_1 t=0 to T_1 then α_2 t=0 to T_2

α_1 t=0 to T_1 then α_2 t=0 to T_2 then α_1 t=T_1 to T_1

α_1	1	2	3 4
α_2	3	4	1 3 2 4
			1 3 4 2
			3 4 1 2
6 ways:			3 1 2 4
Which is Best?			3 1 4 2

SW Sometime ago, I was considering $f_1(t) = p'(t)\, e^{-\int_0^t p'(t)\,dt}$ v.s. $f_2 = e^{-\int_0^t p'(t)\,dt}$.

$f_1 = -\frac{d}{dt} f_2$!

Alternatively $f_1 = p'$; $f_2 = p$ so $f_1 = \frac{d}{dt} f_2$ ~~Then~~ & minus sign difference, hmm. —

Maybe $E = \int_0^{T_1} f_1(t)$? $\frac{E}{T}$

So correspondence not exact.

Any way, it worked. $\int_0^T p'(t)\,dt$ ~~then~~ X, the dep variable (rather than "t")

int. $g_{1,2}$ formulation we can ~~do~~ doing $g_1 \equiv f_2$ i.e. ~~$f_1 \equiv f_2$~~

The prob'ty of success at T and failure from 0 to T

~~we have $f_2 = x$, $f_1 = \frac{dx}{dt}$, but those are backwards~~ ?

$f_1 = x' e^{-x}$, $f_2 = e^{-x}$ ∴ I was thinking $x' e^{-x}$ looked like a replacement & $\int$, but it isn't!

19 N.B. in ~~the~~ .ooff, I have to consider "Expected time to soln", but I ~~do~~ should not

20 include cases where time to soln = ∞.

$f_1(t)$ (eq. (1)) is correct for getting expected time. i.e. exp'd time = $\int_0^{T_1} T\, p'(t)\, e^{-\int_0^T p'(t)\,dt}\, dt$.

22 $p' e^{-p} = p' e^{-\int p'\,dt} = -\frac{d}{dt} e^{-\int p'\,dt} = -\frac{d}{dt} e^{-p}$ $= \int_0^T T p\, e^{-p}\, dt$.

$\rightarrow \int T p'\,dt \rightarrow \int T' f = -TF$ $\quad -\int T f'\,dt = -Tf + \int f$ $\quad | f = e^{-p}$.

Say $-f' \equiv h$.

So we want $\int T h(t)\,dt$ = min. $\quad \frac{\int_T^{T_1} h_1(t)\,dt + T_1 \int_0^T h_1(t)\,dt}{\int_0^{T_1} h_1(t)\,dt}$.

If I do Cand 1, I will win a fraction of time $= p_1$

$= \int_0^{T_1} h_1(t)\,dt$; It will take on average, time $\int_0^{T_1} T\, h_1(t)\,dt = cc_1$ | so "cjs" = $\frac{cc_1}{p_1}$.

To do Cand 1 then Cand 2 $p_{1,2} = (1-p_1)\cdot p_2$ $\quad$ cjs$_{1,2}$ $\frac{T_1(1-p_1) + cc_2}{(1-p_1)p_2} = \frac{T_1}{p_2} + \frac{cc_2}{p_2(1-p_1)}$

$cc_{1,2} = T_1 \cdot (1-p_1) + cc_2$ $\quad$ cjs$_{2,1}$ $= \frac{T_2}{p_1} + \frac{cc_1}{p_1(1-p_2)}$

cjs$_{1,2}$ ⇕ cjs$_{2,1}$ $\quad \frac{T_1(1-p_1)+cc_1}{(1-p_1)p_2} \gtrless \frac{T_2(1-p_2)+cc_2}{(1-p_2)p_1}$

Looks reasonable, simple & ~~probably~~ probably heuristic!

~~$\int_0^{T_1+T_2} T f_1' + p_2 (T - T_1)\,dt$~~

$\int_0^{T_1} T f_1'\,dt + \int_0^{T_2} (T+T_1) f_2'\,dt = \int_0^{T_1} T f_1' + \int_0^{T_2} T f_2' + T_1 f_2\Big|_0^{T_2}$

so $T_1 f_2\big|_0^{T_2}$ v.s. $T_2 f_1\big|_0^{T_1}$ or $\frac{T_1}{f_1\big|_0^{T_1}}$ v.s. $\frac{T_2}{f_2\big|_0^{T_2}}$ or $\frac{f_i\big|_0^{T_i}}{T_i}$ max.

$f'(t)$ is prob'ty density of ...

cf Secs 174.00 ff for worry about this problem

NIPS

WON: Breast Rec: .09 ± (.03 ÷ .07).

6:169 : This $\frac{f_i|_0^{T_i}}{T_i}$ looks fine!

03 Very simple "jump" algm.

In this case if $f_1 \neq f_2$!

The best way, do f_1 until past T_1 & $\frac{df_1}{dt} = \frac{F_2(T_2)}{T_2}$, then do f_2 out to $T = T_2$,

then do f_1 & f_2 simultly as to keep f_1' & f_2' the same.

This technique will work on any no. of centers. — it's particularly easy if they all have same $f'(0)$.

107 This simple form of f' is obtained only if Gamma dist. Better, since $f'(0) = 0$. → 172.00

Note now $f'(0) \equiv p'(0)\, e^{\int_0^T p'(t)dt} \equiv p'(t)\, e^{-(p(T) - p(0))}$.

So this looks like an ideal soln for INV problems!

Solve is particularly easy if f only has one inflection pt. i.e. $f'' = 0$ $\frac{d}{dt} f' = 0$

5 eg. → 172.00

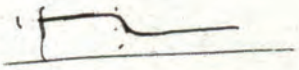
Q: I have recently assumed that $f'(t)$ had this simple shape,

but look at $-p' e^{-(p(T) - p(0))}$ it

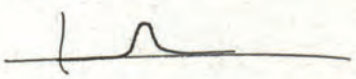
look at $e^{-\int_0^T p'\,dt}$. Then take it's derivative; But first look at $\int_0^T p'\,dt$.

so f' does look o.k. but I do have to get it into right form!

$e^{-\int_0^T p'dt}$



$= \frac{d}{dt}$



We will get TM to find optimum f', f directly, rather than by finding p' & p first (169.20 ÷ 22)

T- params μ_i & σ_i are most important. $\frac{\sigma_i}{\mu_i}$ determines (2) 1. order multiplicity. PSTs will be recruited. Say $\frac{\sigma_i}{\mu_i}$ is in correct order so $\frac{\sigma_i}{\mu_i} > \frac{\sigma_{i+1}}{\mu_{i+1}}$.

Then, when PST$_i$ has started, we will go tiredly working on all PST$_j$ & $j \geq i$.

They will all have same slope.

Actually, parallel search may be regarded as a special case of (.03 → .07): we work on all cands so that their $\frac{p'}{cc}$ is far & about the same; which = ?

Here, if f is obtained by code length directly. In .03–.07, we could vary the code length to be $(\log_2(f(t) - f(0))$, in which case we get coherently same as Lsrch.

O.K. now for 02 problems → 171.00 :

Spec 171.00

N.P WON approach to OZ probs.

10: 170.40 : **OZ problems:**

Say we have 2 PST surfaces $P_1(T,G)$, $P_2(T,G)$.

We have KB = T_0. Choose T_1, T_2, s.t. $T_1+T_2=T_0$ & then 2 resultant $P(G)$

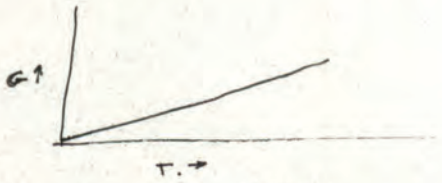
curves give best guess of peak G. ← does $\int_0^\infty dG\, P(G,T) = 1$: in which case width x ht. = const

We can do a rough & dirty soln. by using 2 params (M,a) approx.

A rough ordering is by $a_i M_i$. so maybe $M_i \cdot a_i$ is not such a good index:

Use just M_i, to start. ?? No.!

Given 2 curves at $(0,L)$, what is expected value of max?



T. probability/density of a peak at G_M is $P_1(G)\cdot\left(\int_0^G P_2(G)\,dG\right) + P_2(G)\cdot\left(\int_0^G P_1(G)\,dG\right)$

We want max expected value:

$$\int_0^\infty G\,F_1'(G)\cdot F_2(G)\,dG + \int_0^\infty G\,F_2'(G)\cdot F_1(G)\,dG \quad \leftarrow \text{maybe should be mult by } (-1).$$

$F(G) = \int_0^G F'(a)$

$d(xyz) = ?\ \left(\frac{dx}{x}+\frac{dy}{y}+\frac{dz}{z}\right)\cdot xyz$

$d(x\cdot yz) = dx\cdot yz + x\,d(yz) = dx\cdot yz + x\,dy\cdot z + x\,dz\cdot y =$

$d(xyz) = dx\,yz + dy\cdot xz + dz\cdot xy.$

$xyz = \int x'yz + \int y'xz + \int z'xy$

$G F_1 F_2\Big|_0^\infty = \int_0^\infty G F_1' F_2 + \int_0^\infty F_1 F_2 + \int G F_2 F_1'$

So $\int_0^\infty G\left(F_1' F_2 + F_2' F_1\right) = \underbrace{\left(G F_1 F_2\right)\Big|_0^\infty}_{\phi} - \int_0^\infty F_1 F_2$. unlikely!

I guess I want $\int_0^\infty G\, F_1'(G)\cdot F_2(G)\,(1-F_1(G))\,dG$

$+ \int_0^\infty G\, F_2'(G)\cdot F_1(G)(1-F_2(G))\,dG$

$= \int_0^\infty G\,(F_1'\cdot F_2) + G F_1 F_2' - \int G F_1' F_1 F_2 - \int G F_2' F_1 F_2$

$- \int_0^\infty F_1 F_2 \ (\uparrow .25)^{\infty} - \int G\left(F_1^2\right)' F_2 - \int G\left(F_2^2\right)' F_1$

My suspicion is that it's most likely that usually we will want to put all of the info into one "Apparently best" PST. We may do this until (using II updating) it is clear that it is no longer "Best bet".) see (72.20-.32 for good discussion.

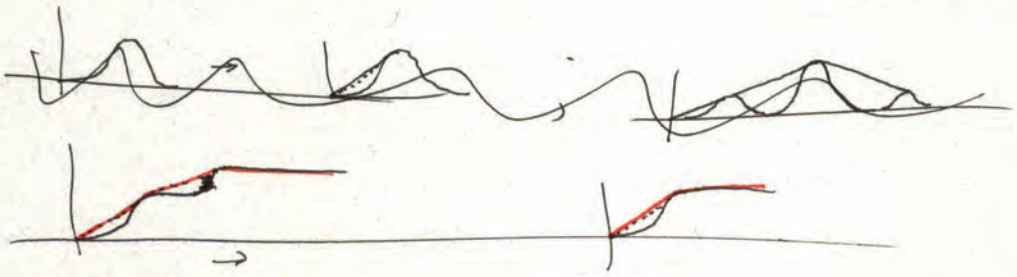
If we don't have much data or strong ideas on the $P_i(G,T)$ curves, then we may want to spend (MANY) small amt. of time on many PSTS (with process problem) to get some evidence of which might be best.

(see 172.20)

NIPS

WOPS (Inv problems)

Space
170.15
0: 170.07 : ENU probs: T search process is rather simple! first we convexify all P(T) curves:



05

→ categorize slopes & to discrete Bins.

.09 Work on Bins in slope order. Steepest slopes first.

That's it! : A min amount (Δ) is spent working on each curve; Δ >> transition cost to new curve. This Δ may cause a jump over > 1 bin. (eg. after working on 2 curve, it is put for time Δ, it is put into a new (lower) bin.

.14 Convexity means that slope is monotone (not ↑) in T.

15 While we are working on a dotted gf. line in 05, we are really "cheating", but on r. average, we will be doing the right thing. The time "between Bumps" will be small times relative to total time spent on solving problems. So R is "train size" will not be crucial. → 174.00

Space
0: 171.40 : If 171.32 "suspicion" is correct! Past Penguins work on one PST's next are highly "inter related" That's used info from one another: This info is transferred by the U updating.

The commonest "inter relation" of Opten methods is in "Trials": Each "trial" info is shared by all Opten methods that use trials (many may be managed such techniques do use "trials").

24 So we may want to do "trials" that are "common" to many PST's, (≡ OPTT's ≡ opten techniques), "common"

25 A situation in which 171.32 is certainly true! (It looks like it's always true: see .32 → .32 We have many curves G, T curves à Pen all start out (low T) w. small G's: If we distribute = time on all of them, the expected wear will be small. Say all P(G,T) curves are the same, & are increasing w. T, σ^2_G ↑ w.T.

Then if one has a large no. of cards in U w. = T's: we get sort of exact: i.e. Pack is always somewhere of ensemble of individual curves.

32 (25) → Hm, by spending all time on one card, once gets much better perf. This is true if all cards have same P(T,G) curve. If some have "better" curve, then it's a fortuitous better to pick them along. There is still the G of which P(G) curve (for a given T) is "best" — i.e. which "2" (card) to choose. Criterion for "best" is not clear — I don't see (e.g.) } ? ?
why picking curve w. greatest prob of giving highest G, should be picked

.40 So, looks like 171.35 & (.20 - .24) are most relevant to "what to do".

Space
173.00
→ 173.00

WON: OZ problems

171.35
170. 172.20-.25
172.40

Re: "Trials": In general, most OPEN methods using trials are Newtonian methods — (Gaussian)
($s \leq z_1$) or Ray uses >1 past trial to get new trial (Gen. Alg., Sim Anneal,
Hill climbing using different mix of previous pts): Nonlinear Past finding,)

Diff. opt methods use different "principles" to do their extrapolation. Most all methods are
fairly "Greedy" (little if any "look ahead").
For pts of Pris's sort (using "basis" on corpus), we can one pt(s) next given
"most promising" future trial. — Which is quite Greedy:
(what appears to be "best" trial today)

We can modify the proceed to do L.A. 1, 2, 3 ... jumps into future.

"Non-trial" optim methods: {(linear regression using solns of matrices), some ANN methods, but mainly
logical analysis'. I heard very much undet'd about this. I don't expect PM to be able
to do it until it has learned to work logical problems as part of its "Math"
training.

So: Probly t. soln for INV problems of 170.00-.40; (172.00-.15 is OK.

The treatment of OZ probs via $P_3(G,T)$ curves of 171.32-.40, 172.20-40, 173.00-.12
is a general approach that's probly correct "in spirit". — That Updating is an imp. part of t.
entire strategy).

Since Phase I QA is doing a test that weeds out normal OZ problems, (second closest to INV problem soln) I feel I must use t. recent WON approach of
for Phase I QA. The idea is that in Phase I QA we try Pris's PC order in attempts to find a code
for to corpus — which looks like a INV problem. So try to find out how t. QA problems
in INV can be mapped into INV WON type soln.

F(x) = True : find x: $h_2(\tilde{F}) \to x$

$a_0^3 \cdot O^3(\text{corpus}) = hy$ $h_3(\text{corpus}) \to O^3(\text{why } a_0^1 \cdot O^1(\text{corpus}))$

Alternatively find $p \ni M(p) = \text{corpus}$: try p's in pc order (Pris's t. 3 input univ approach to
S-induction)

.27 is it really bad at all! It could be t. best way: That all induction methods involve
finding regs in t. corpus, that have to be expressible as "short codes".

(SN) A possl. trick: Some short codes take long times to verify! A pgm that could
look at a short code for t. corpus & give good reason to believe it would
code t. corpus, but takes a long time... Something one can look at a pgm & see that
it must converge, or one can find indications while it is running on how long it will take
to Converge.

WON INV probs

(172.17) (169.35-169.40) : Worry about non-proof for INV at 169.35

I Print it to look at to do it. P_1' first then P_2' first vis. P_2' first then P_1'

This gives $\int_0^{T_1} T P_1' e^{-\int_0^T P_1'(t)dt}\,dt + \int_0^{T_2}(T+T_1)\underbrace{e^{-\int_0^{T_1} P_1' dt}}_{\text{constant factor of } = \alpha_1} P_2'(t) e^{-\int_0^T P_2' dt}\,dt$.

$= \alpha_1$ = prob of failure of P_1 at time T_1

$P_1' e^{-\int P_1'} = -\frac{d}{dt} e^{-\int P_1'}$

$F_1' = -\frac{d}{dt} f_1 \quad ; \quad F_1 = e^{-\int_0^T P_1' dt}$

$\int_0^{T_1} T f_1' dt + \alpha_1 \int (T+T_1) F_2' dt = \int_0^{T_1} T f_1' dt + \alpha_1 T_1 \underbrace{\int_0^{T_2} F_2' dt}_{F_2(T_2)-F_2(0)} + \alpha_1 \int_0^{T_2} T f_2' dt$

P_2 from $F_2 \quad = \int_0^{T_2} T f_2' dt + \alpha_2 T_2 (F_1(T_1) - f_1(0)) + \alpha_2 \int_0^{T_1} T f_1' dt$.

$\int_0^{T_1} T P_1' e^{-P_1} dt +$ 

$P_1 \equiv \int^T P_1'$

$\int_0^{T_1} T f_1' + \alpha_1 \int_0^{T_2} T f_2' + \alpha_1 T_1 \underbrace{\left(F_2(T_2) - F_2(0)\right)}_{=1}$

$\int_0^{T_2} T f_2' + \alpha_2 \int_0^{T_1} T f_1' + \alpha_2 T_2 (F_1(T_1) - f_1(0))$

$\alpha_1 = e^{-(P_1(T_1) - P_1(0))} \qquad \alpha_1 = \exp(-(P_1(T_1) - P_1(0)))$

N.B. $\int_0^T P'(t)\,dt$ must be ≥ 0 but can be > 1.

.02: $\int_0^{T_1} + P_1' e^{-\int_0^T P_1' dt}\,dt + \overbrace{e^{-\int_0^{T_1} P_1' dt}}^{\alpha_1} \left[\int_0^{T_2} (T+T_1) P_2' e^{-\int_0^T P_2' dt}\right]$ | Should $\alpha_1 = 1 - e^{-P_1(T_1)}$?

$\int_0^{T_1} T P_1' e^{-P_1} dt + e^{-P_1(T_1)} \left[\int_0^{T_2} T P_2' e^{-P_2} dt + T_1 \int_0^{T_2} P_2' e^{-P_2} dt\right]$

$\int_0^{T_1} T P_1' e^{-P_1} dt + \alpha_1 \int_0^{T_2} T P_2' e^{-P_2} dt + \alpha_1 T_1 \underbrace{\int_0^{T_2} P_2' e^{-P_2} dt}_{= -e^{-P_2}\Big|_0^{T_2}}$. $\quad P_2' e^{-P_2} = -\frac{d}{dt} e^{-P_2}$

$= 1 - e^{-P_2(T_2)}$

$\overbrace{\int_0^{T_1} T P_1' e^{-P_1} dt}^{\beta_1} + \alpha_1 \overbrace{\int_0^{T_2} T P_2' e^{-P_2} dt}^{\beta_2} + \alpha_1 T_1 - \alpha_1 T_1 \alpha_2$

$\beta_1 + \alpha_1 \beta_2 + \alpha_1 T_1 - \alpha_1 \alpha_2 T_1$

$\beta_2 + \alpha_2 \beta_1 + \alpha_2 T_2 - \alpha_1 \alpha_2 T_2$ $\Big\}$ vs. $\beta_1 + \alpha_1 \beta_2 + \alpha_1 T_1 (1-\alpha_2)$

$\beta_2 + \alpha_2 \beta_1 + \alpha_2 T_2 (1-\alpha_1)$

$\beta_1 - \alpha_2 \beta_1 + \alpha_1 T_1 (1-\alpha_2) = (1-\alpha_2)(\beta_1 + \alpha_1 T_1)$

$\beta_2 - \alpha_1 \beta_2 + \alpha_2 T_2 (1-\alpha_1) = (1-\alpha_1)(\beta_2 + \alpha_2 T_2)$

If $\frac{(1-\alpha_1)}{(1-\alpha_2)} > 0$ } true

$\frac{\frac{\beta_1}{\alpha_1} + T_1}{\frac{1}{\alpha_1} - 1}$

$\frac{\beta_1 + \alpha_1 T_1}{1-\alpha_1}$ vs. $\frac{\beta_2 + \alpha_2 T_2}{1-\alpha_2}$

169.37-.40

$F_i(T_i) \equiv \alpha_i$ or $1-\alpha_i$

This looks like it complicated WON solns I got in '90!

$\alpha_i = e^{-\int_0^{T_i} P_i'(t)dt} = e^{-P_i(T_i)}$

$\beta_i = \int_0^{T_i} T P_i' e^{-P_i} dt$.

$1-\alpha_i$ is the prob of success at time T_i.

β_i is the partial expected soln time at T_i.

On the other hand, the max $\frac{P}{T}$ slope result of 169.35-.40 seems quite reasonable.

See if I can show it is correct!

space

→ 175.00

4/31

NIPS

WON: [INV: .00 ; OZ: .19] Summary

00:174.40: reading 169.35-.40;

definitions of f_i: From 168.10-.11 some confusion!

$f_i' \equiv +p_i'(t)\, e^{-\int_0^t p_i'(t)dt}$; $F_i = -e^{-\int_0^t p_i'(t)dt}\Big|_0^T = 1 - e^{-\int_0^T p_i'(t)dt} = 1-\alpha_i$ 174.15

Look at (174.30R)! $1-\alpha_i \equiv \gamma_i$ $\alpha_i \equiv 1-\gamma_i$ $\frac{\beta_i + (1-\gamma_i)T_i}{\gamma_i} = \frac{\beta_i}{\gamma_i} + \frac{1}{\gamma_i} + T_i$,

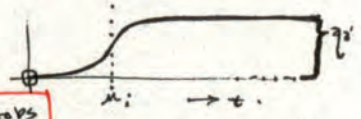
$\gamma_i \equiv e^{-\int_0^T p_i'(t)dt}$.

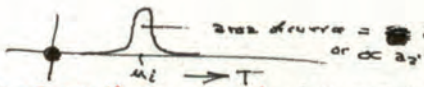
I do see what 174.30R, but I suspect the reasoning is somehow wrong – Rule (.08) is more likely to be right: SEE 176.23 -.40

.08 HVR: The Arg. of 167.10 -.17 gives good reason for $\approx \frac{\dot{e}_i'}{\dot{m}_i'}$ as a viable approx for ordering.

$z_i \equiv \int_0^\infty p_i' e^{-\int_0^T p_i'dt}dt$. I think: it's the probability of success at time ∞.

This closes updating

.10 

for p_i' = 

T. Gen of .04 is fine is not much different from the $\frac{z_i}{\dot{m}_i}$ "soln."

.11 for INV probs T. method of 170.03 -.07 is probly correct. → SEE 176.25 ff for Modify: Probly more correct: It assumes the successes of the PST's are independent

& that the curves are "f" curves $F(t) = p'(t)\exp(-\int_0^T p'(t)dt$; $F = -\exp(-\int_0^T p'(t)dt)$

When doing Updating during Srch, After each update, can recurse (172.00)

Then necessary return to Bins (172.09) review of past work on each. — Rem can have

work on best bin that has content of 1 or more PST's.

.19 For .OZ probs: If we do "Update during Srch; (Note: 17#.32 -.40, 172.00 -.40 ff, 173.00)

Choose single PST w. best "expected" yield for remaining Times & variables; work on it until update time.

.20 Then Do update then loop to .20 → this need not be best decision criterion {SEE 176.12 -.20}

.21 The update should be able to discover "correlations" (eg. when certain PST's do badly,

.22 .23 certain others will also/ get worse curves/types assigned to them.

Routine .19-.21 is ≈ for INV also, except that after Update, as we will

obtain jump from one PST to another as we move along a P curve & we jump

.27 into a different "Bin" (172.09).

So .08 -.27 may be a Adequate good start for phase 2 INV & OZ. Could it also be final "Phase 2" Soln?

.30 Many Variable Problem

For SFunct Search in Phase 1 Q.A. 137.00-138.29 (Important unrec. — using Jaynes formalism)

Also of much important "context" 168.00-.20 is a good recent discussion.

Try to find older, older discussions of Context, & mechanics of how they work; Then inducing

meaning (Just how does Ray modify a prop, & just when a prop'd Ray Modify)

Another big Q: How does QHTM turn Bernoulli seq. & types to ys!?

ONE BIG Trouble w. Lsrch is that it doesn't correctly give cc credit for giving ll codes —

which is one way to realize Bayes seq. (≡ Laplace's rule) by ALP.

See how to prove that "$2^K \geq$ ALP within a constant" works — try to fix Lsrch so it

can take advantage of ll codes.

4.6.3 → WON

Nips

Time Varying OZ:

There is a previous mention of this problem on ID 825...

4.25

§2.6, 55. may 59.7 plastic tray

E: 1 9 2 5

§4.2.5

: WON: Perhaps go back to working AND nets.

I had a proof/criterion for all AND/OR nets. — to show it was a meaningful problem.

OZ: Time Varying OZ. Say our Gain is $G(x) \cdot F(t)$: F is a known & function of T.

To solve this: we have to $P_i(G, T)$ curves for each P & T for static Gain, $G(x)$.

When modify these curves $\tilde{P}_i \ni (G, T) = P_i \cdot (G \cdot F(t), T)$.

The shapes of the $\tilde{P}_i$ curves will be quite different from the usual $P_i(G,T)$ curves because of G necessarily & eventually w. by T.

For each P & T_i at each T, we will have a different $P(G)$ curve. We want to select the "Best" curve.

Using the W. hyest "expected value" of G is not write unless G has been "Linearized". If the D.P. for $P(G)$ is "narrow", Expected value is O.K. —

Otherwise, it's not clear.

Actually, the "linearization of G" is part of the problem/dcm. One has to be able to tell whether how much better one G_1 is than another G_2.

Otherwise, it's impossible to choose between $P(G)$ curves.

So, anyway, one simply picks out the $\tilde{P}_i$ and T that have the "Best" $\tilde{P}_i(G,T)$ curve

$\equiv P_i(G \cdot F(t), T)$ curve.

Meanwhile, once is updating, & it is clear that a new i', T should be chosen, one jumps to the new i', T.

Essentially, the "Linearization" soln. of .15 means that I must be able to decide between "Betterness" of .3 of G^{of} pc = .3, G = 10 v.s. pc = .6 G = 5 or .6 G = 6 or .6 G = 4

4.7.03: $\int_0^T T G' e^{-G} dt = \int_0^T \blacksquare e^{-G} dt - T e^{-GT}$

So 174.300

$$\frac{\int_0^T \blacksquare e^{-P_i} dt - T e^{-P_i \blacksquare} + T e^{-P_i \blacksquare}}{1 - e^{-P_i}} = \frac{\int_0^T e^{-P_i} dt}{1 - e^{-P_i}}$$

we want Min

(this is probably a very old idea & very well known!)

= Probability of success at time T.

$\int_0^T e^{-P_i} dt = J(t)$

So: $\frac{J(t)}{1 - J'(t)}$ $S(1 \cdot e^{-P}) = T \cdot S \cdot e^{-P}$

We want Max $\frac{1 - e^{-P_i}}{\int_0^T e^{-P_i} dt} \approx T$ for small T.

, so $\approx \frac{1 - e^{-P}}{T}$ for small T

(.36B) which is G+FT! soln.!

$1 - e^{-P} \approx P(t)$ if $P(t)$ is small.

T. corresponding to the "Convexifying w. stalling" is to find the smallest T at which $\frac{1 - e^{-P}}{\int_0^T e^{-P} dt}$ is Max.

Do this for all Conds. Pick cond for which this is max, do it first — Have a loop until Gain = Gain of next best Cond. Then do next best cond. Then work along both, so as to keep Gains =. When Gain = the initial Gain of 3rd best cond, do 3rd best on till we reach that max Gain pt, Then work on all 3 conds on ll. etc.

4.8.03 WON : Also Rev 10

NIPS

.01, 8000 bits
800 k bits of symbols.

Expo!

00 : Do Presentation of Thm 3 parts ① "review of Univl. Dd. & its properties
② Soln of QA problem (Phase 1 soln) (2.5) Lsrch? Do before QA
.02 ③ WON solns for LNV, OZ, time varying OZ.

SN T. Soln of time varying OZ is also soln. for usu-time varying OZ? ⇒ (see 175.19ff)

SM Re: "Incomputable ALP" Lecture: One can often tell if one Model or Model set is closer to P_μ than another. Can well as tell for "good" each model set is for predn (of next symbol). So perhaps Machine Mind.

N.B. Post 179.30 result ≡ 176.3 result ≡ "very old WON result" Suggests that Algebra is OK. — T. first 2 weeks of 3 month distrib. usage.

10 STATE of TM!

1) General mode of Phase 1: Phase 2 operation seems like Good Idea
Phase 2 via WON (175.00–176.40) will probably work fine (including "updates during srch")

2) Input unit for QA problems seems good to ≈ start on.
2) Getting it to work Boon sup. type problems — remains to be done
b) How to introduce Context! for Lsrch in QA problem. See 168.00–40 This is Good.
There is much earlier work on "Context": I should read some of it; it does have interpt. ideas! ↳ 183.00
16
3) TSQ needs to be written.

~~I don't know what else~~ 2 parts reading must work:
Context 2b); TSeq 3) : Context may need more work: I'm not sure of
20 state of the problem.

TSQ may be o.n.: I could just use fairly large machine & try to do ≈ Algebra test.
for Advanced work, May Not Book that Ramanujan Learned Math from. RAMANUJIN

More immediately: 1) Finish off Kol Lectures
2) Put out next edition of "Report": Add to any version: WON Solns., also
Various indices to my notes ~~etc~~ to explain various parts of the "Report".

30

Rec! Solns to WON for AND, OR nets:
31 1) For OR nets! Re: "Soln" is used for LNV & OZ probs.
T. solns used assumes the P's are "INDIP" (certainly not true)
We can use define a set of tasks that are indep, that somehow we average the dependent tasks?
Unlikely In general, prob. No, because the space of code-pieces that probs is a very big dimensional space, & the set of "indep" tasks seems (as vectors in Hilbert space) seems like much less info)

1) List of solvd, unsolvd problems + refs to work on Phase probs
2) find a date list. try to update it.

Anyway! Some ideas I had on how to get this ≈ representation (175.00)

.0.177 Ideas: ① ~~Clumping~~ Clustering cands into sets & using "representative" cands for each set as indiv cands, w. hvly correlated other members of each cluster

② Cross corr matrix betw cands.

③ Updating during Srch, If the current cand begins to look bad, update rest matrix note that cands similar to cand should also begin to look BAD

Anyway, the idea of using best trial for cands then doing updating, seconds use. way to deal w the problem, but I'm not certain that is the best overall is the correct. Updating seems hvy, detect corrctns. very well, so, it sounds like

.08 a very useful idea

SN To what extent can I use my current "soln" of the OZ problem to solve "Cure Cancer"? As I have no body to present "soln" to OZ posits the indiv cands, which are true for regular OZ Continuity & Interrupt not for "Cure Cancer"; The "update betw trials" (or during trials) idea helps w. OZ; would it help w. "cure cancer"? Updating during trials is perhaps more relevant.

SN For WON problem: for OR nets; Criterion of soln is:

~~Best~~ a) The tech required will solve ~~each cand~~ nets fraction of time, F. F is maximized.

b) Of the time that a soln is obtained, Expected time to soln is Min.

My soln. satisfied both extrema.

For WON in general, these 2 criteria can be used for solns, ~~& take as any~~ that soln always exists satisfying those to both cands, is a theorem.

20 This prob of soln's value is easy to compute! For real tasks, have will be a pdf of completion at t=∞. From those values, it is easy to compute pc of soln

22 at T=∞ for each time unit. → → 180.22

Re: OR problems (2 tasks, say). If $p_i(t)$ is pc density of soln at time t (w.o. restriction that this is smallest T for soln.) then if we do 2 tasks in order, the new P function will be (if task 1 is first) $P_1(t) + P_2(t-T_1)$ — T_1 being end time for task 1. However, I have no picture of how a $P_i(T)$ could occur — i.e. how can a possible soln. occur at > 1 T point? — It could for separate cands, (tasks), but not for a single task! — Unless the single task is, say, composed of (many dependent (or even dependent) subtasks. (Req. independence cuts dependencies & necessary (reduces previous task

So the prob func for a single task will always be one for first time solns so $\int_0^\infty p(t) \le 1$.

For 2 tasks, we have the normal $\sum p_i$ ("first time pc") for tasks followed by $(1 - \int_0^{T_1} p_1)$

.35 $P_1(T_1) \cdot P_2(t - T_1)$. doing this & integrating gives $\int T P_1(T) + (1 - \int_0^{T_1} p_1) \int (T + T_1) P_2(t)$ → 179.00 → 179.11

~~$= \int T p_1(t) + p(T_1) \cdot T_1 \cdot \int p_2(t) + p_1(T_1) \cdot \int T p_2(t)$~~

$\int T p_1 + \int T p_2 + T_1 \int p_2 - \int p_1 \cdot \int T p_2 - T_1 \int$

V.S. 5

10:178.40 : $S^{T_1} T\rho_1 + (1 - S^{T}\rho_1)\, S(T+T_1)\rho_2 = S^{T_1}_{T}\rho_1 + T_1 S\rho_2 + ST\rho_2 - S\rho_1 \cdot ST\rho_2 - T_1 S\rho_1 S\rho_2$

$S T\rho_1 + S T\rho_2 \;|\; + T_1 S\rho_2 - T_1 S\rho_1 S\rho_2 + ST\rho_1 - S\rho_1 \cdot ST\rho_2$

$T_1 S\rho_2 (1 - S\rho_1) + \cdots$

$T_1 S\rho_2 = S\rho_1 \cdot ST\rho_2 - T_1 S\rho_1 S\rho_2 \;|\; T_1 (S\rho_2 - S\rho_1 S\rho_2) - S\rho_1 ST\rho_2$

$= T_1 S\rho_2 (1 - S\rho_1) - S\rho_1 \cdot ST\rho_2$

$= S\rho_2 [\, T_1 (1 - S\rho_1) - S\rho_1 \cdot$

$S\rho_1 \equiv A_1 \;|\; S\rho_2 \equiv A_2 \quad ST\rho_1 \equiv B_1 \quad ST\rho_2 \equiv B_2 \;|\; \int_0^{T_1} T\rho_1 + \int_0^{T_1} S\rho_1 = T\rho_1 \Big|_0^{T_1}$

$B_1 + S A_1 = T_1 \rho(T_1)$

11 : 179.35 → $B_1 + (1 - A_1)(B_2 + T_1 A_2) = B_1 + (1 - A_1)(B_2 + T_1 A_2)$

$B_1 + B_2 + T_1 A_2 - A_1 B_2 - T_1 A_1 A_2 =$ $T_1 A_2 (1 - A_1) - A_1 B_2$

v.s. $B_2 + B_1 + T_2 A_1 - A_2 B_1 - T_2 A_1 A_2$ $\quad T_2 A_1 (1 - A_2) - A_2 B_1$

$A_2 (T_1 (1 - A_1) + B_1)$

$T_1 A_2 (1 - A_1) + A_2 B_1$

$T_2 A_1 (1 - A_2) + A_1 B_2$

$A_1 (T_2 (1 - A_2) + B_2)$

$\frac{T_1 (1 - A_1) + B_1}{A_1}$ is Gone. $= \left[T_1 \left(1 - \int_0^{T_1} \rho_1(t)\,dt\right) + \int_0^{T_1} T\rho_1(t)\,dt \right]$

$= \frac{T_1}{A_1} - T_1 + \frac{B_1}{A_1} = \frac{T_1 + B_1}{A_1} - T_1$ $\quad \int_0^{T_1} \rho(t)\,dt$

$= \frac{T_1 (1 + \rho_1^{(1)})}{\rho_1^{(0)}} + \frac{\rho_1^{(1)}}{\rho_1^{(0)}}$

$\rho_1^{(1)} = \frac{2\pi}{\text{moment}}$

$\int_{T_1}$ strike $d(te)$ $\quad (T\rho)' = T'\rho + T\rho' \quad T\rho = S\rho + T S\rho'$

$T\rho = S\rho + T\rho$

$\frac{T_1 + B_1}{A_1} - T_1 = \frac{(T + T_1)\rho_1}{S\rho}$ Gone $\frac{T_1 + \int (T + T_1)\rho_2}{S\rho} = \frac{T_1 + T_1 \rho_1^{(1)} + T_1 \rho_1^{(0)}}{\rho_1^{(0)}} = \frac{T_1 (1 + \rho_1^{(0)})}{\rho_1^{(0)}}$

$+ \frac{\rho_1^{(1)}}{\rho_1^{(0)}}$

$\rho_1^{(0)} = \frac{2\pi}{\text{moment}}$

· Say $\rho_i \to \rho' e^{-\rho}$

$S\rho_i =$

$\frac{T_1 (1 - S\rho' e^{-\rho}) + ST\rho' e^{-\rho}}{S\rho' e^{-\rho}} = \frac{T_1 e^{-\rho(T_1)} + ST\rho' e^{-\rho}}{1 - e^{-\rho(T_1)}}$

$S\rho' e^{-\rho} = -e^{-\rho}\Big|_0^{T_1} = 1 - e^{-\rho(T_1)}$

$= S\left(T (e^{-\rho})'\right)$

$S T (e^{-\rho})' + S(-e^{-\rho}) = -T_1 e^{-\rho}$

$\frac{S e^{-\rho}}{1 - e^{-\rho}} = \frac{1}{e^{\rho} - 1}$

$+ S T\rho' e^{-\rho} + S e^{-\rho} = T_1 e^{-\rho}$

which is 176.30 ☹

$ST\rho' e^{-\rho} + T e^{-\rho} = S e^{-\rho}$

$\frac{T_1 + ST\rho - T S\rho}{S\rho}$ $\quad ST\rho = S\rho + ST\rho'$

00: : Re: The use of "WON" to replace Lsrch in parts of TM. : This seems like a very impt decision!

While WON is in a sense "optimum", it is only as good as the $P_i(T)$ curves.

Similarly Lsrch is only as good as the assumed. But Guidance is.

Lsrch seems to have the advantage in that we can visualize an upper bnd (determined by) on the cost of discovering a "best" soln (≡ downcost of that soln)

For WON: we will try PST's in some reasonable order (which is what Lsrch does)

Actually: from WON 23.00 (4.11.03) $F_k = p_k G\left(\frac{I}{p_k}\right)$ ($p_k \equiv$ pc of PST_k) $G(x)$ $1 - e^{-x}$.

If we Lsrch is to be optimum: we can easily get this if we assume $\alpha = p_k$ or αp_k and the params of F_k corresp are $a = p_k$; $\mu = \frac{1}{p_k}$. $\sigma^2 = p_k$ or αp_k

10: Turns out that we don't have to choose betw. WON & Lsrch!

From 179.26.30: Define $F(t) \equiv e^{-p(t)}$

Then, "Updating" has to determine each $p_k(t)$ (for a particular problem/PST_k)

If can use some standard form for f_k & determine suitable "moments" or a, μ, σ^2.

for the desired f_k. If we used any "reasonable" form of $G(a, \mu, \sigma^2, T)$,

Then, if the update uses $a_k = pc_k$; $\mu_k = pc_k$ (G is dilated/expanded in both G & T directions equally)

Then (σ^2 is not relevant) Lsrch is optimum; Time spent on PST_k is $\propto pc_k$.

So Essentially, "Updating" can determine if it wants to use Lsrch or Not — Lsrch is a special case of WON (OR). So WON will automatically use Lsrch if ⓐ Updating is smart, ⓑ Lsrch is Basic, ⓒ we tell Updating to pc_k of each PST_k = pc_k

We DO have to tell "Update pgm" what the pc_k of PST_k is, hvr. → 188.17

SN Rigorous analysis of "OR" Nets: AND/OR nets. 178.22

In the case of OR tasks problem: Say we have 2 tasks, w. assoc F_i curves.

If $F_i(\infty)$ is prob of soln by $t = \infty$, Then $1 - (1 - F_1(\infty))(1 - F_2(\infty))$ is prob of any soln by $t = \infty$

This is indep of order in which we do trials.

If we do Task I first, then Task 2, then Task 1 again, Then of the α fraction of successful trials,

we will have an Expected soln. time. We want the ordering of tasks that minimizes the Expected time.

This is what was (?) done in 169.35 — Then more accurately in 174.30; Then accuracy Recalculated and simplified in 179.30

For any AND-OR net of finite size, having arbitrarily complex serial, || (sub)tasking:

We can easily compute the prob of soln of any and set, i.e. any or set., & computer science ∃ combinatorics

to get prob of soln. of entire net at $t = \infty$. One definition of pc of soln: one spends

1 msec on each task, by timesharing until all tasks of some selected net is solved. A certain fraction of successful trial

sets will solve it. This α is indep of order of trials. For each algorithm that orders trials

on tasks & parts of tasks, the α will be the same but on the successful runs, we will have a

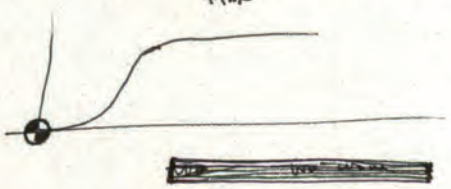
unique value of expected completion time. We want the ordering ⇒ that ∃ time is min. space

↳ Maybe NO! → 181.00

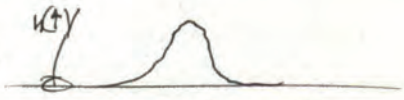
4.12.03

Nips Won

00 (Space 180.40): A possl. counter Arg. to 180.31-.40: That very long 'solns can occur & make "Expected completion time → ∞ (Is Rei's possl.?).



T. density of solns is



We want min $\frac{\int_0^\infty T\,h(t)\,dt}{\int_0^\infty h(t)\,dt}$ =

Well, we can [redacted] describing h(t) ∋

58 & zeroth moment exists, but not & first moment (i.e. first moment = ∞) e.g. $\int \frac{dx}{(1+x^2)} > \infty$ $\int \frac{dx\,x}{(1+x^2)} = \infty$

(Also zeroth & first may exist, but not & second. Moment Gauss. à fin. & all moments exist) ← Rei's Pris is best route we have.

3 → But note 181.13-.17

If we ask T on to optimize & first 3 moments for our [redacted] data sets, it will very probly be able to find good finite values.

13 I f h(t) is of form $P_0 e^{-P(t)}$ can [redacted] (.08) Go Bucs?

Say P' is bounded, and ≥ 0. This ∫ function would seem to have all moments!

15 i.e. $\int_0^\infty t^n P'(t)\, e^{-\int_0^t P'(t)\,dt}$ is bounded for all P'(t). | $\int_0^\infty P'(t)\,dt$ need not converge.

So it is likely (I Rei) that for all finite AND/OR nets a definition of "Optimum soln" exists.

17 :180.21: On Lsrch V.S. WON for INV problems! Note also Rei's Pris soln. satisfies GHT1 for small T. (176.36R) i.e. it is max $\frac{P_i}{T_i}$ ordering for small T's.

171.32-.40

20 On Lsrch V.S. Won for OZ probs! The won soln. is 175.19-.23; Also Note 172.00-.40 173.00

Will this ever be ≥ or ≡ Lsrch soln? — which spends time & share on each OZ PST$_i$ or PC$_i$'.

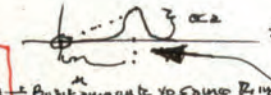
Won picks "Best" PST$_i$ & works on it [redacted] until "update time" we may switch PST's after update time: (The "T remaining" will change after each update).

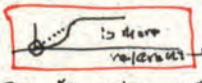
Actually, for Both INV & OZ probs, I should be comparing Lsch & WON after update has occurred, when I have a $P_i(T)$ or $P_i(T,G)$ for each PST$_i$

In Rei's cases, Lsrch gives ∼ a = $\int_0^\infty p'$ ordering = (mean a zero moment) or $\frac{\mu}{a} = \left(\frac{\int_0^\infty T p'}{\int_0^\infty p'}\right)^2$ ordering = $\frac{\text{first moment}}{\text{zero moment}}$.

While a ordering for Lsrch is [redacted] about rite (I wd. det for exp culling of "Good" PSTs)

$\frac{\mu}{a}$ [redacted] simply orders PSTs - it doesn't yet assign pc's to them — as are needed in Lsrch.

For [redacted] F' of form  , $\frac{a}{\mu}$ is initial slope for Won "ordering"

Maybe F = ∫p' =  + Best amount to [redacted] Rei's

(last few Months)

There is a way to order possibly by using only [redacted] $[a_i, \mu_i]$ info. (Detailed analysis within [redacted])

Of course $\frac{a}{\mu}$ is [redacted] of uniform order for GHT1. It is exact if $\sigma^2 = 0$.

I Rei nt. Rei's order used by Won if T is small (176.36R)

So for INV probs Rei's should be comparable or better than Lsrch.

My impression is that if we have 2 parms & or f' dif. giving à à la Rei's makes ≈ ideal GHT1 Srch possl. — & Lsrch ≈ ideal GHT1 Srch also — but $\frac{a_i}{\mu_i}$ order may be better.

Nips

↳ concurrent / simultaneous

10: 181.40 : So WON such wins ● ll updates looks fairly good & is likely to be better than Leech — It is certainly closer to it.

For OZ probs, WON snn seems v.g. — probably usually better than Leech, because won tends to spend more time on very promising PST's — while Leech distributes CB over many PST's — even if only one is Rit to be v.p. [see 175.19–.21 for discussion of OZ]

Other remarks on WON for OZ! 175.22–.23, 176.12–.17 (on linearization of G),

The discussn. of "crowding" between Cands for INV, is relevant to OZ as well (177.31–178.08)

→ One possl advantage of Leech: that by trying many distinct cands, "update" has more data to work on.

For My version of "T. report" I want to write a bibliog. review of WON for BR, AND & OZ. Also, ■ actual expository writing on: main ideas & arguments as to why it ought to work — why it may not work & ideas on "How to Improve it".

20

SN $ functs: 3 realizn methods: 1) 3 imputune 2) Laws rule, like AZ41 3) The "Quote" expressions in AZ141 that express S functs can be described by a Lispish (functional) Lang. [This last is not quite clear! — It may well be that Lisp automatically does this! It can defn a string (= pfun) then execute that string (eppm).

SN In OZ trials I almost never have "failures": A PST will usually get some G: 1° Occasionally, however, there is not enuf time for a true trial, & G = −∞. This −∞ can be one of the G values & update can use this info for learning. I'm not yet sure how the probabilities should be mixed w density of G i.e. pc of G=−∞ v.s. pc of G= finite. Actually, we have the same problem in INV problems! (?) No: At cc=T the PST either succeeds, or fails to succeed. Its the GPD function $O^j(\cdot)$. Actually the O^j function could look at

30

The situations in OZ & INV do correspond. — In INV, the $O^j(T)$ corresp gives densities for success at T, & interpreted density values for success at times > the time cutoff. h'(t) | T₀ prob of no completion up to T₀ is $1 - \int_0^{T_0} h'(\tau)\,d\tau$: This assumes h'(t) vanishes no later than T_0.

Similarly $O^j(T, G)$ gives a prob density for T, G±Δ, and a prob that G=−∞ for T.

We multiply prob densities by actual probabilities. Do we want to accord same (logarithmic) wt. to both of them? → .35

Nips 18.33–.36 criticizes & updates optimizn of O^j on basis of SOP: hmm, see this [Nips 18.36 ... to 19.06; 18.36 ...]

35 → It's 0.9! say we had n_d prob densities & n_p probs for our corpus. for ∫ density width Δ, the actual prob of the model is $\left(\prod_{i=1}^{n_d} p_d^i\right) \cdot \Delta^{n_d} \cdot \prod_{i=1}^{n_p} p_p^i := \alpha$!

α is a regular probability. What we want to maximize is $\frac{\alpha}{\Delta^{n_d}}$. The $\frac{1}{\Delta^{n_d}}$ factor is indep of what models we try, so it's not relevant to the maximization process. So it's o.k. — We are not "adding apples to oranges".

Nips

>0! 177.16: "Context" : In induction problems, we always have how to solve a problem, no matter how many examples we have worked in the past. If we had enough time, Lsrch would be the best way to go, to code for the entire corpus. Always we are always short of time. The best way to code a corpus in very short time is to identify code. (It's a very lazy, coder, bird!) Another trick is to use caches (with Levin limits) to the w.s freq. w. with they occured in the past coded corpus — which is the usual AZ141 way. There are other tricks, but may all depend on ideas about saving time & having a too small CB.

How is "Context" relevant to the above? Well, in various past contexts certain coding tricks have been "useful."

How does this idea accord w. J's "successful" (?) use of "context" via the "Boosting" instruction?

EN LZ compress uses OSL! (one shot Lrng)

Mainly, AZ141 uses ideas that during past have been used in the past, are likely to be compress in future, as well. Are all "regs" expressable as "definitions"?

→ It is my impression. That all types of regs that one observes/invents, are obtained initially by ≈ Lsrch (or a process w. the same efficiency as Lsrch — w. perh a factor of 2x "depth" of concept). So another simple schema for inductive coding ought to work ~ optimumly! i.e. for each new problem, search a fairly large amt. ("oversearch" is meaningless at this pt., since the CB is part of the problem descrn. .... The problem of what CB to use for induction — & updating, is unclear: The 50% soln. is meaning full for updating Inv. probs, but it's not clear what CB to give for OZ problems originally — & for (non-update) OZ problems — in which case the "stop soln" is meaningless/arb!)

I suspect that we will need "hyer level Lrng" to decide how much cc to spend on some OZ problems (in which the CB is not given). Chess, w. 40 moves in this, total, is a mild kind of problem of this sort! CB for each search has to estimated by TM so as to optimize some overall Gore.

In .13~.21, it would seem that "Context" is irrelevant. — yet w.o. (of course!) Context consideration, we seem to have a serious Scaling problem — that pc's of concs ↓ as corpus size ↑. It may be that we intuitively (in "context") are content to overcome the "Scaling" problem.

One duffy in my thinking: I think of a "new problem" as involving not only more than finding some new concs, that satisfy a certain constraint (possibly a "crazy Gorcish" constraint — an "OZ" problem, like).

→184.00

Mips

BACKTRACKING : How to do it! .25 -.30 ; When to do it, ~~à human-options be-prudent~~ -.31ff ; How much time to spend on Backtracking: 185.15-.25

00 : 183.40 Hvr. QA problems à t. Update problem don't seem to be like Pack — in that we are looking for a global soln. to "all problems Pass for".

One might look at in this way: one has a sequence of problems: Pr. k+1 problem is to find a good model ($\exists O^k$) for $[Q_i A_i]_{i=1}^{k}$. TM uses its ~~power~~ experience w.r.t. first k problems to solve k+1st. — This is what "OOPS" does.

It's also what I wrote about in trying to find a good function that goes from Input of all past k problems à their solns, to the prob distribution of the relative frequencies of the k+1st problem (Q_{k+1} given) and its soln

---

10 **In General on "Context"**: Whenever one discovers/defines a new Concept, it is necessary to know **Under** what circumstances that Conc is usable. This is "context" à it is about as important as knowing the inventing/discovering of the conc. itself. When a conc. is first defined/discovered, its SSZ is small à so the SSZ for its context is also small — perhaps

→ see (183.14: LZ coding!)

OSL will have to be used, initially, for employing/exercising/trying the contexts that occur w.r.t **newly** defined conc. Also, we want negative data for each conc., when it "fit" but was inappropriate: for to-date, my present methods can work when only positive data is available

There are a standard set of context types that will be initially explored!

Note 185.26! →

.10 ff **may** be a difft approach/defn of "context" than my previous approach.

It seems to have a difft theoretical backend. Although we Only try to code the corpus, using concepts that have been useful in the past, on this corpus. It is a kind of "Sequential Coding"! (little if any "Backtrack"). ← This is a serious Criticism. ! → 185.26 w.r.t. Backtracking, one can get "Badly stuck"

20

A possl. way to do "Backtracking": Say we were doing OSL à we look for things in past that would give proofs of/ very recent past.... } I don't see how OSL could be Modified to Help

24

25 I think I know how to do Backtracking: One normally has several alternatives ("branches") codes for the corpus — using difft defns. To continue the corpus, one will chose one branch à try to continue, using its defns à pc's of concs. If "things don't work out well" in continuing on that branch, one does "backtracking" à chooses another (initially lower pc) branch to continue.

30

31 While .25 — .30 does tell how to do backtracking, it does not tell Under what circumstances to do it! In the Sci Community Backtracking (≡ "Recovery revision") is done infrequently — ∴ No it is occasionally necessary — It is usually a "big deal" à causes much fighting tearing of hair, and Homonim attacks ect. The idea is, that the predictions don't seem to be as accurate as we expect that they can be. What is our Criterion for this? In physics, we expect proofs to be "exact" within precision of instruments, à we continue to ↑ accuracy of instruments. In Chemistry, we

: often can't control things exactly, ~~be~~ in which cases we settle for less precision
In biochem, & biology, we know that we don't know even all of the conditions that affected an experiment & we expect even less precision

In Biological experiments, we have relatively little control & the experimental results are not of much accuracy & often are not reproducible (except at low precision)

As one ~~unders~~ begins to understand the "Domain" one begins to know what the uncontrolable/unknown variables are — & estimate how important they are. —
So as to get better idea as to how accurate one's predns should be!

E.g. say ~~Temp~~ pressure Temp is a variable that we couldn't formerly control or measure. —
So it caused much variance in chem/phys results. Then we are able, not to measure it, but to be able to estimate when 2 temps were about the same.
So we now expect 2 experiments w same Temp to give same results —
or to both be in accord w "theory".

In General, TM will learn when to expect hy accuracy & when to expect low accuracy in predn. This will be by looking at past & seeing under what conds, by accuracy was obtained. This ~~predn of~~ Expectation of hy accuracy will be wrong many times, but it will be enuf to tell us how much time to spend on Backtracking. → Note 189.00

Is .15-.20 an adequate Soln. to the problem?

T. backtrack problem is: Should I spend time T looking for extension of old code or Backtrack? I.e. — which is more efficient use of my time, T?

(= concepts) Well, the program is mindfull of what I used to do whenever I discard ~~discarded~~ something
novel. I would spend a lot of time trying to see just what situations in the future, that conc would be relevant. (like 184.10 → ±.20) → Also Note 186.12 -.15!

Also I tried to see if that conc. could simplify (∴ shorter coding) problems (was "solved by") of the past — problems that had not used as part of the sample that gave conc.
This suggests that normally, I do not use my entire corpus (commonly true! Particularly note. This is the "Encyclopedia problem"!) So perhaps, in the Encyc, I begin to look for more cases in the past. However, my not using the complete corpus for every predn. could be because it was too time consuming. I suspect that for any predn, I usually select out what I think is the relevant data of the past, & make predns. on that basis. After I find a good conc. for the latest predn., then its structure may suggest other areas in the corpus to look

4. 18.03
Nip>

.16-.35 [How ↓ of σ corresponds directly to ↑ in mean density of predns ⇒ ↑ of pc of symbols that occurs] [This is discussed in Ravasios to keep tech talk]

DO: 185.40 for relevant data that conco might help code.

{183.15 to 186.00} → Does this Give fairly complete approach to TM — or at least to "Phase 1"?

T. idea of "sequential coding w. backtrack" is attractive.

→ Anyway try to write this up as exactly as possl. : I don't want to forget it.
Also try writing up re. nature TM Phase I in as much detail as possl. so I can discover main project bottlenecks

.N Ouchhhy to deal w. pc v.s. 2^{-n} disparity in Level! If stochastic systems are properly defnd, it may well be that they are about the same. Using tokens of various pc's as in A2141, may help.... but can it solve the problem completely?

.12 :185.25 Re: 185.26 ff: essentially what I did was create a set of obs for "contexts" in which conco would be likely to be usable. There I'd have one or more obs for each conc I discovered (or for all tokens")

15 The system would then be a bit like Selfridge's "Pandemonium"

.16 4.19.03 On ↓ of variance v.s. ↑ mean b cost of next [expected] symbol.

.17 If we assume (for "rare" problems) a uniform spread of the variable being predicted,

.20 then ↓ of σ by a factor of k (i.e. $\sigma \to \frac{\sigma}{k}$) [NO! Expected value ($\sum_{\text{all pc}} (-\ln pc) \cdot pc$) of next datum ↓ by factor of k exactly] [Expt. value of pc of next symbol ↑ by factor of k] see .24-.27

.21 Two things of interest: (1) The "uniform spread" doesn't mean that its uniform from $-\infty$ to $+\infty$ It does mean that the spread doesn't vary much

.23 in the "region of interest" → 187.20 for exact forecasting results.

.24 (2) That we use Gaussian coding for $[x_i]_{i=1}^{n}$ $x_i = \bar{x}_i + \Delta_i$ $(\Delta_i \equiv x_i - \bar{x})$

So pc of sequence $= \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\Delta_i^2}{2\sigma^2}\right) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \left(\exp\left(-\frac{\sum \Delta_i^2}{2\sigma^2}\right)\right)^n$

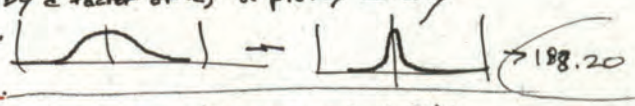
.27 $= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \left(\exp -\left(\frac{n\sigma^2}{2\sigma^2}\right)\right) = \left(\frac{1}{\sigma\sqrt{2\pi e}}\right)^n$ since $\sum \Delta_i^2 = n\sigma^2$. ($\exp -\left(\frac{n\sigma^2}{2\sigma^2}\right) = \frac{1}{\sqrt{e}}$)

So ∴ expected value of pc of next symbol is $\propto \frac{1}{\sigma}$

30 NB The demo of .24-.27 does not depend on x_i having a Gaussian Df. i.e. it probly works if σ^2 is the variance of any d.f. that has a variance.

First assume .21-.23 mean. If we narrow the d.f. by a factor of k, the probby density at the "usual" data pt. will ↑ by a factor of k.

35 But in general, if $\sigma_1 = k\sigma_2$, the probby density height ↑ by $\frac{\sigma_1}{\sigma_2}$.



36 Re: Context" : I think an important part of my thinking about TM involved my expectations of what future problems would be. It was important to have ideas as to what I was aiming at, when I acquired very early orientation. That was done by "Looking" at the unsolved problems of science from a "popular" pt. of view. → Not very detailed or quantitative. →187.00 spa

t.1g.03
NIPS

(Spec. 186.40): 186.36 - to was, I think, overall impt. in RTM, since I was mainly interested in "future world". (Horizon Issue). Since I'm not working on RTM just now, this may be irrelevant to present problem. But it is an impt. part of the way a Scientist normally thinks. More a part of early education, than later ... later the Scientist really knows many of the main problems in his field of expertise.

When a Scientist proposes a radically new theory: To 1 the [illegible] of the new ones used, he shows how those ones can be used to better codes data of the past.... This may or may not involve back tracking. For other scientists it probly involves backtracking. But this Scientist may have obtained the "New View" w.o. back tracking — i.e. he may never have really accepted the current view that he wants to (demolish / discard).

The main things that QATM will Not do: <(??)

1) Solve "Cure Cancer": Tho I think it could be taught to solve probs like that by teaching it first hal, then haz, ect. (Cure Cancer is a RTM-type problem)

2) RTM (see 13 ≈ 6(3)) → [Any Large znA problem can get TM to act like RTM]

3) Propose a/o do "Experiments" (An "experiment" is one used to gain info — not directly related to problem soln.) → (cf "info. is to be used "later")

Propose that we design a more accurate instrument to support new regions of Universe to look at.

Q: What about EncycTM? This is an impt. kind of TM problem. T. "Encyc" can be RW, in which case, TM would do "experiments". Even w.o. Encyc being RW — if Encyc is text, TM's indexing of the Encyc would be used to ↑ precision of future predns. — so this would be an "Experiment".

On the other hand, if one has a large OE problem, part of the r.e. PST Could very well be "purely mopping" — only indirectly related to the Optim.

So I'm not at all sure about the above. E.g. to "Cure Cancer", if TM really understood what the problem was, it may very well find a PST that would look like the "Cure Cancer" routine that I've described. "Cure Cancer" really is a Time limited OE problem — but TM must have access to RW. Its criterion of "cure" is that when TM accesses RW news or Google requests, it finds that Cancer has been cured — that ways have been found to get almost all cancer patients to get well.

QATM, INV, OE, Theorem predn, Bay induction & recall special modes of behavior of TM: using a common EPD. Is Encyc TM an other of those "special Modes"? — if so, just how does it work?

I did write a fair amt. on Encyc.TM.: well there is this larger data base (Encyc), that TM has access to. Say we want to do prediction on seq. a (a is in Encyc. Universe) One way is to find short codes for a, using access to Encyc.

Nips

:00: Or, we may have a seq. of QA's à a final Q to answer.
TM has to learn how to use t. Encyc. — so that TM doesn't have to read t. whole Encyc to find useful info. TM could answer Q's by giving addresses in Encyc that are relevant & then doing some operations on those references.
We'd probably have to use some kind to TSQ to teach TM how to use Encyc.

.05 Find Previous Work on this Encyc idea. ... I don't want to repeat old work.

So: I want to find (at least) 2 things
1) Discussions of Encyc problems — (187.19 — 189.06) — ID 585.21, .36 ← V.G.!
2) Discussions of old Contexts. (Also 184.10–186.00 seems like a very good discussion of Context, 193.15–194.09 — 186.12–.15
Earlier sources may be also useful.

:11 (Encyc) cont.: In One version of this problem: the "A" is always a direct Quote from the Enc text.
Or, it may Quote > 1 part of Encyc.: At a hyer level, TM will use t. Encyc along w. other info — perhaps statistical info from Encyc &/or other sources. At the most general level, it will be solving Oz, Inv, QA ... problems à use refs to Encyc as part of t. Code for a soln.

:?? Res: Finding Older Work on Encyc: (≡ Enc): Alternatively: Declare the problem carefully &; this will suggest Soln. /// Note that "indexing" of Texts can be part of "Updating time".
Only major t. Enc problem arose when I had a large corpus à a problem that was related to a small part of it.
In theory, I could code the whole corpus to solve the problem, but this would be wasteful of cc. (It would be OK, if CR = ∞, hm ☺)

:20: 186.35: Re 186.21–.23 (186.21–.23) We can assume a uniform apriori for all from −R to +R in this case.
For · R ≫ 6 (say of 6's) The results will not depend much on R. and results are indep of R as R→∞. This argument about R→∞ works OK. for this particular problem —

.27 but not merely for many other problems!

SN In using WON w. Il updating for both INV à OZ problems, the system doesn't try much variety of PST's: The result is that the basic data for update is poor à Updating itself is poor.
To remedy this: Spend ½ (or ⅓? — ⅓ on won, ⅔ on update.) of time shared on regular Lsnch trials

:? SN NB Re Context cues: cues? are part of t. code of the Corpus. They modify pc's of other cues.
They are what V. talks about when he says t. system compares t. pc's of the most token ("oops").
If we allow Rosa & context cues to have adequate arguments (like "type of problem", à even "recency" ....) then they will be general contexts.
→ Is there any way that Context can relate to cc? — essentially a non-"pc of corpus" type of consideration?

4.20.03
NIPS

Back Track : reasons to Do :
.00 is summary.

0:185.20 : On what to do "Theory Revision": [195.36 - 256] stuff was all based on idea that one seems to be. — Better precision possible. — induced from previous experience.

→ so I motivate the Backtrack ① Better precision would be very valuable scientifically or pragmatically. imp → ② ③ ID [195.20-.23]: One is trying to take a sub-corpus by extending a code for a prior corpus. At what point do one abandon this code & Backtrack to a different code for a smaller corpus. If this is the case, then ⑤

off. system) can have 2 aspects: ⓐ Production via approxn to Univ. D.f. ⓑ considerations of finite C.B.

More on "Context" (183.00 ff ....) It may be that "context" (as well as other aspects

It may be that in phase 1 these aspects of TM's behavior are "glossed over", but in Phase 2, they are clearly separated.

There is still so much Uncertainty in my mind about "context": about what role it is a legit part of the approach — a legit prediction-oriented "regularity" .125 V.S. a heuristic device to enable more rapid (i.e. less cc) coding of the corpus. This Q may be somewhat linked w/ the ideas of .05-.10.

My impression is that .11-.125 is probly correct. Consider 185.26, 184.10, 186.12

Essentially, the "obs" of 186.12 were obtained using induction on situations in the past that were assigned, similar to the situation in which the "obs" were to be invented. Any other method of getting such "obs" & their assoc pc assignments to trial codes ("Tokens") would not be legit. There is no other way to discover these obs other than induction on the past.

However, in the large, there seems to be danger of "double counting" because we are looking at various objects (some of them rather complex) that may contain other of the objects: specifically; we look at the corpus $[Q_i, A_i]_{i=1}^{k}$ for each k & the associated O^j that were found for each k value. — What we are doing is trying to predict or get a good d.f. for the $k+1^{st}$ O^j. This last is (in a sense) equiv. to obtaining p.d.'s for each "Token" used in constructing Cond. O^j's [i.e. If one has a d.f. for O^j's for each "k" corpus (a "k+1" corpus) then one can obtain from it, by purely Mathematical means, a D.f. for Tokens & each step of the construction of cond. O^j's.

.24 was an early Goal of QATM: To be able to directly "update" O^j by looking at the appropriate historical data.

From large, one might get idea that this solves the QATM, except for Maybe the "Overcoming Ugly" problem (.20) — But not so. Phase 2 is needed, because it understands "Optimization" & this "Phase 1" simply trys "what worked well in the past".

On the other hand O^j is for unordered data & the problem of .16 seems (like ordered data!) seems

Also: There are known that are purely "cc reducing" — like using Matrix inversion S.W. to find a good set of linear regression coeffs → 190.00 Spec

~~TB~~ THINGS TO DO! .09

◀ NIPS ▶ ...

(space 189.901)

In 189.39: If I ■ devise a ~~[illegible]~~ pgmg trick, the resultant PST has a smaller cc so it's ~~[illegible]~~ more likely to be run in Lsrch ~~[illegible]~~ and it will be [illegible] be expeditiously in WoW.

The resultant PC obtained for the Oi, Hvr, will be indip of just how Perf Oi was obtained.

Hvr, Phase I QA does not use PST's.

T. idea of 182.10 ff was that I would not do ~~[illegible]~~ body in Phase I by using Lsrch on a suitable SUMAC (W. apriori augmented by Backtracking).

189.31 – 190.03 suggest reasons why this may not work.

I want to do 6 things!

1) Write fix to kol-folk paper

2) Write Fogtchript paper ■: Review of convergence Arms — w. perhaps a section (drawing section?) on their significance

3) Patch up IDSIA report.

4) Write Title, Abstract for DIMACS talk.

Before this, Hvr, I want to write a Bookmark" telling where I am & what I want to do when I finish .10 → .03

5) Write Comments on OPPS for J.

6) Outline TM: See what needs to be done w. perhaps a section à perhaps start on Phase I symp. (e.g. see Nips 5.00)

7) Write Dimacs Talk.

8) Write or view, at least 200 pp or so

Lsrch for Encyc problem

Contains

186.16 on Ce ↓ vis. Pg.g.

163.26

163.22 "Ahaiwa factor"

183.15 – 186.00 ← Early mono of. idea

Also Note 188.30

1) Book mark about 182.10 ff (note .04). Presumably use of a SUMAC (w. scribble Backtrack provisions) would be a good approach. See 184.29 for how to backtrack, 184.31 ff on when to backtrack. A Q is whether Context" is included in Pgg approach. See also 611.35 for discussion: it refers to ID 768.20 → 40

189.11 ff suggest it is included ---- except for the "Double Counting" possy of 189.20

Also 189.11 context appears to be sequential (as any Sonar) & QATM is in a certain sense Not sequencial, since $O^j(A_i|Q_k)$ is order indep of i's).

So I want to go back to 182.10 .... 190.04 to see just what goes here: It would be nice if I could use a good SUMAC for QATM. It would order the i's for O^j cands properly, but would not be able to incorporate cc saving hours into the approach.

Such hours might be Externally (by Me) implemented.

2) The "Ency." problem: I have forgotten my Scini to this problem. (187.19 : 188.11 (185.33 : try to find old references to "how to solve it" (I think I send back from srched Nips 189.49 → 190.00 Nips to ID 585.21, .35 is very Good!

3) What T. presents TM "Quit Do" on 187.12 I made a list, but it seems that it is wrong! — Any TM w. "BIG Problems" can eventually do whatever a RTM can do. Go into this more Carefully! possible cands: 1) try to design hyper rat tcl instrument. 2) Suggest New Rapidly Universe to Explore. ← TM could [illegible] be, but only if it were given examples of the [illegible]?

4) 186.16 on How prediction by Maxing expected pc of next symbol is active. to Minimize or for several pred's. also 188.20

more exactly min or → min expected pc of next symbol.

$\int_{-\infty}^{+\infty} (\exp pc(x))^2 dx = \int_{-\infty}^{+\infty} p(x)\,dx$

Σ = expected value of p

But this is not a sum $\in$ face) = Σ simplex"

5) I really want to go over Nips from 1 to 199: Do a review: put impt. ideas discussed in a INDEX of some sort, so I can find them again. "Context" is probably Pro Hope — on ID. (~ 800 pp!)

6) Find old Refs to Context & its most General kind: ① simple unc. ② "adequate"?

7) Discussed 5 rounds of Bayesian [illegible]

4.22.03
NIPS

0 : 190.40 : First finish kol-Lecture revisions; put on net, try to arrange talk at MIT.
MIT Talk will be expanded version of Dimacs talk.

~ 151.00 ff is a revision of kol talk. 151.27 — 152.12 revisions is inserted at 4.21 of kol talk.
Also ~ 162:00 ff à 177.00

Re: list of features on p3 of talk: 1) Our explicit use of spirit enables us to use very small size primacy.
2) ~~Discuss over/under fitting~~ Restrictive If concept e.g. resolve the over/under fitting problems
3) It is sometimes possible to ~~use all of the data for the test set and not at~~ avoid discovery of a training set à use all the data for the test set. (I will discuss this later (perhaps) (a.k.a.!).)

→ 194.30

SN In J's "oops" à Koza's GP, They create a definitions à status "of random" ("sort of" "for J"). This is done in AZ141 this corresponds (to some extent) to my looking for common sub trees. ~~This~~ (last) is rather time consuming (I guess) à perhaps by using J's a/o Koza's methods, I might do it, yet ~~still~~ speed things up a/o have a shorter pgm (i.e. shorter, simpler for me to write)
↑ I was initially a fanboy of Koza's doing this!
A big difference: I don't have cost of "making a definition" unless I "use it. Concept > 1 time. J & Koza always have this extra cost.
→ Hw. Check to see That is really do it (it) in AZ141 !
It may well be that AZ141 does it sooner than J & Koza do
At any rate, Dissp ~~x~~ 3 kinds of copys should be compared à maybe merged (if this is usefully possible.) I will (probly) be using ~~x~~ AZ141 type of p.d. ~~for~~ to get Phase I to (re 5 Results: (in addition to the 3 importance.)

For real no. prediction: A run-run method to give the d.funct the "center" of d.f. plus a a2 "for corrections". This is sort of "done on 3 importance; the R input is the "corrections" for the sum of the "corrections".

There are (perhaps) MANY ways to get Sa Results: 1 common way:
F(Q, D) F is a d.funct of Q, D; But D is a s.funct on "no input" - essentially, D is a D.f. T. s.funct, D has a kind of "Usage-type cost", for K(·,·) that ~~because~~ it may be regarded as the sum of the parallel codes it implies.

27 In the case of the Update problem, say we always had Phase 2 or 3 param "f" curves
28 The problem is to get functions Real map from strings a/o reals to triplets of reals (u, a, ε).
.27-.28 is perhaps the main s-function problem.

Nimps

15 30 45

00 : "Last items" : → 216.00 for Mann account list. (preliminary)

01 1) How I was able to get TM to generate new PST's as part of the normal "Update process". along w. Mann Mac's Faraby problems

[redacted] Expansion of a Trainer [redacted] furnished "PST enumerator" or a set of "factored" PST's.

Might be used. 154.34 mentions this || 220.00 is a recent formulation.

2) Ency. (Encyc) ···· how did I deal w. this problem? → ID 585.21, 585.36 ← (This is pretty Good.)

3) List (pages entries) about "GA" & other impt. topics.

4) Discussns of "S-Function types" @ 137.00 — 138.29 is a v.g recent discn. of "3 types" of "time" types. AZ 141 is a subRoutine type.

5) "Cure Cancer": first soln on ID [redacted] 19.26-.40 : I think Rec'd a simpler soln. later.

6) Early Gauzing of idea of "context" for updates ID 141.19 ff has a bit. also ID 141.30 : 217.00 is recent discussn, remembrance.

0: Recently, (186.16) I discussed goal of narrowing of continuous d.f. of pred. as means providing ↑ of mean PC of predicted symbol (as did discrete preds),

Say $M(X)$ is the generator of the data, $P(X)$ is the predicted d.f.

we want some large $\int M(x) \ln P(x)\,dx$ ($\hookrightarrow \sum_i M(x_i) \ln P(x_i)$ for discrete case)

$\sum_i P(x_i)(\ln M(x_i) + \ln P(x_i)/M(x_i))$

$\sum_i M(x_i) \ln P(x_i)/M(x_i)$

The both are so. because $\sum \ln(P(x_i)-M(x_i))^2$

→ Is there any reason to believe that this or any other measure (like [redacted] or σ^2) is "Unbiased"? If it is unbiased, it will probably be so solely because the P(t) d.f. is purely a priori. I think it is possible to have an a priori that does give systematically spurious low σ's. Consider linear regression. Consider the a priori that looks at the training data, selects 3 coefs for best fit & gives a central point w. σ^2 = observed σ^2 of past.

However, perhaps the use of a priori gives an empirical (type based assumption of error) correction or "Inductive" Part is unbiased.

Superficially, the Conv. Thm says that error estimate (say $\overline{(P(i)-M(i))^2}$) ↓ as n ↑. ← large corpus.

In general, in induction systems, expected error ↓ w. larger corpus, so empirical measure of past, is always an upper bound on error for next symbol (or "X" values) — Furthermore, in the case of a continuous stream regression, true σ^2 goes as $\sigma_0^2\left(\frac{n+k}{n-k}\right)$ — for "large n"

— "R" concern which the main error is due to be certain parameter reflection thing

193.00

00:19240 discrete params that desrb $P(\vec{x})$. It assumes also that both M & P have k continuous params.

If we have the wrong discrete params but the right no. of continuous params, I think we get the same kind of convergence of σ^2 but it's a larger σ^2. The discrete params define a k dim space and we pick the closest point in this space that we can, to the "correct" point. (Not "closest" — but a local min of error). Actually, maybe we are constrained to a k-1 dim space: the intersection of two k dim. spaces. Analogy: 2 planes (2 spaces) intersect in a line (a 1 space) — if 2 spaces are imbedded in a common 3 space.

The above considers how this is OK for M being a differentiable function, but not if there are discontinuities in the function or its partial derivatives. (⌐ is not smooth.) P should be a differentiable function.

Perhaps M can have non-differentiable points, if P is only a differentiable function.

If .12 is true, then by using differentiable P, we can get e.g. estimates of error in next x_i.

Well, say the forgetting was O.K. for the continuous params in the model, P — that it would give a good value of σ^2 for the next x_i.

What about discrete params? — can we make good estimates of how expected P of next x_i varies as n ↑?

If the forgetting, I assumed that the params were selected to min (say rms) error in the past.

Even if I'm doing discrete params, I can still have k continuous params.

Say a binary seq. in each bit has a P.L. of a k param function of all previous bits. —

Could we use an algt like 186.16 ff to get expected value of next symbol?

A possible approach: Do "Max pc" analysis for the σ^2 situation of next x_i.

However, say $\vec{x}$ (n bits) is observed. We want a space of k dim models: we pick a model (pt in k space) that gives max pc to $\vec{x}$; Say $\vec{r}$ is a k vector in model space. $P(\vec{x}_0)$ is a func $F(\vec{r})$ we pick $\vec{r}_0$ s.t. $P(\vec{x}_0, \vec{r}_0)$ is max (i.e. $\vec{x}_0$ fixed, pick $\vec{r}_0$ to maximize P. Given that $\vec{r}_0$ is max $\vec{x}_0$ also max? I would guess, usually not for any $\vec{r}_0$ it's easy to compute the $\vec{x}_0$ w. max $P(\vec{x}_0, \vec{r}_0)$ (May not! —) → what I was thinking, was given $\vec{r}_0$ first choose most likely x_1, then given x_1, $\vec{r}_0$ pick most likely x_2, ect. — This is probly not the best way!

My (strong) Guess: If $\vec{p}$ is the mean (in probly) of observed symbols; w. k params & n (binary) symbols, $\vec{p} \cdot \frac{n}{n+k}$ is expected (infpl) value of next symbol. However, P(·) itself will give a pd. for next symbol. Say P_0 & P_1 are P's for 0 & 1 then expected (wrt P(·)) log pc of next symbol is $p_0 \log p_0 + p_1 \log p_1$

Since $\vec{p}$ is <0 $\cdot \vec{p} \frac{n+k}{n-k}$ is more appropriate, i.e. ↑ $\vec{p}$ is ↓ $\vec{p}$ & ↓ less ≠ $e^{\vec{p}}$.

$\vec{p} \sqrt{\frac{n+k}{n-k}}$ is more reasonable, however:
see 194.25

NIPS

20 : 193.40 : A possl. formulzn! We have a function from $\vec{x}$ to (0,1) interval. ($\vec{x}$: continuous param; $\vec{r}$: k vector)

21 $F(\vec{r}, \vec{x}_i)$: We want $\vec{r} \ni \sum_{i=1}^{n} F(\vec{r}, \vec{x}_i)$ a max, & we accept such a $\vec{r} = \vec{r}_0$.

(Say, to start, that $\vec{r}_0$ is not at a "wall" of its permissible space)

[redacted] exp(F) = probly. So this expression maximizes "likelihood" ≡ PC.

There is a "True" value of $\vec{r} = \vec{r}_M$; that generated the data probabilistically.

Or another way to look at it: We generate n probabilities, $p_{\vec{x}_i}$: We Montecarlo, use $\vec{p}_M$ to generate a binary string of n bits, = $\vec{x}_M$.

We then take: $P(\vec{r}, i)\Big|_{i=1}^{n}$ function of $i = 1 \cdots n$ and try to find $\vec{r}$ ⇒

10 $\prod_i P(\vec{r}, i, X_{\vec{x}}^{i})$ is max & is an "input config to P" & $X_{\vec{x}}^{i}$ is the i^{th} component of $\vec{x}_M$

$P(\vec{r}, i, x^i)$ is the probly assigned to x^i w. "input $\vec{x}$ = (to P)"

12 So this has a probly value for each i". How much does it differ from

I'm not doing well on this problem! Define it & put it on "Stack" (#ID 611.38) #14

Discussn starts on 192.20, goes to 194.12.

T. ideas sheets w. Akaike factor (86.16-.35; 187.20) $\sigma^2 \to \sigma^2_{observed} \frac{n+k}{n-k}$.

186.16 is a very outline of a proof of it; probably it would be well to make a careful proof of it before

18 going on to the present problem, which is: → We have an n bit seq $\vec{x}$ generated by p.d. M.

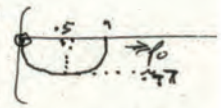
k param

. We have a function of the previous prefix, that gives a value pc for each bit. This funct

. has been optimized wrt those k params, to the pc of the corpus is max.

21 . What is the expected value of the ln p of the next symbol? "Is it an unbiased" estimate?

↳ we don't want $p_0 \ln p_0 + \beta_1 + \ln p_1$? = (≡ $p_0 \ln p_0 + (1-p_0)\ln(1-p_0)$ ≠ function of p_0 alone.



→ A (new) possl. approach: Consider the continuous (Akaike) case is divided into a discrete (areal). See how "Solomonoff" version works, then → 2 levels (≡ 0, 1).

earlier work on this problem: 152.24 — 153.19; 156.32 - to 157.00 - to, 158.00 - to, 159 no ff.

?: 193.37 N.B. 193.37 is probly wrong: $\sigma_{ob} \to \sigma_{ob}\sqrt{\frac{n+k}{n-k}}$: probly of what symbol is of $\frac{1}{\sigma_{true}}$.

$\sigma_{est} \approx \sigma_{ob}\left(\frac{n+k}{n-k}\right) \approx \sigma_{ob}\frac{n+k}{n}$

"Back tracking" is perhaps a special case of "[illegible] (trying/evaluating) a new Model".

0 : 69.09 152.00 +.12 is also a useful direct "copy"

260 : 150.07 ff looks like good copy! , 157.00 ff, Trouble is, I have the original "folk" as written, & 150.07 - 192.12 (+162.00 +191.05 -.08 (& discussn. of 95.10 - 196.13 on "failed probabilities".) — But many don't fit all fit together!

150.07 - .26 looks v.g.! : So perhaps do this: for each section I've written, write a short-summary of its [illegible] & use "Listing" of them. Then see how I could fit them together or make a [illegible] part of the Paper from these parts.

SUMACS .00

30: : **SUMACS** : ID 378.00 Develop Sumac Scheme, but fear it would not work because eventually, the initially finite no of O^i counts would go to 0 as the no. of QAs ↑. If we do "backtracking", this will not occur.

Note also, that "Backtracking" a complex corpus need not be so difficult. We can always do "soft coding", in which we make a lossy code for the corpus & then add in corrections.

A main difficulty here is that this does not give codes in the "pc order" — which is of importance in using the Esrch to look for induction codes (i.e. O^i's).

0: On assigning pc's to Nine reactor, Net defenses, Generic catastrophe, space shuttle crash. Is this related to incomputability of UPD? While the reason for the real incomputability of .10 are v. ideational to those for incomputability of UPD, (that all models have not been fully evaluated ... many models are p.r.), .10 is not the same as UPD.

In .10 we devise theoretical models w. pc's that we may or may not have enough time to evaluate.

But these seem to be different from the kind of thing we make models for UPD. In UPD, we are trying to get good models (i.e. short codes) for a known (& slightly unknown) corpus. In .10, we take "known" (i.e. by pc.) scientific laws & try to use them for extrapolation.

18 One way to think about .10 (In .10 we have no empirical data, so we are working only on the apriori. This will be the apriori on sequences of new events that may or may not lead to the failure whose pc we want to evaluate. In the case of "Reactor failure", we do have empirical pc's of various event types that help evaluate our needed apriori. These empirical pc's may have been obtained from this or from other reactors.

10 Tho it may be necessary to expand the detail in .15ff, I think that it is essentially what we do when we use **UPD** for predn. In the R.W., & ~~doubtless~~ there will be many possible conds. to evaluate. The set of "eligible conds" is not always clearly deduced. or Lack of sci knowledge: e.g. Bad sleep Cycles on people Monitoring Reactor

26 Sometimes a Theorist "Eligible conds" are not evaluated, for "political reasons". The actual empirical failures of reactors have been by far the most part, because "ineligible conds" (conds not considered).

A cond can be ineligible because (a) Very low expected apriori. (b) Relatively impossible.

10 (c) takes too long to evaluate (d) Simply wasn't Part of (related to) the one too distant from (a). Common Cause or Concurrent Unexpected failure (Correlation of events not considered to be correlated.

So, It seems clear that .10 is a case of UPD, but that the "instruction set" or the rules for creating conds &/or assigning pc's to them may not be clearly deduced — also, (.26)

There is another way to evaluate pc of reactor failures: We have the large corpus of reactor histories — not many failures, but many malfunctions & partial failures, each.

from this corpus

4.26.03

NIPS.....

:0 : 195.40 : An interesting Q: Int. problems of 195.00, The incomputability of the UPD is very relevant.—
Yet it is not in most prediction! — Why not? How do these situations differ?

Well, in 195.10; We either know no previous corpus on which to test our models or [t. corpus is too small, that's it. so SSZ error is > t. upper bnd on permissable failure probty

Hwr, t. forgg argt doesn't consider P_M^T approaching $P_M^\infty (\equiv P_M)$ or $P_M \to M$. at all!

Maybe we consider $P_M^T \to P_M$, and that P_M is "as good as we can hope to get".

Actually, in those failure cases, t. problem is always to show that $\sum pc < \epsilon$, and this is necessarily impossl. We have to show that there are no codes w. pc's of $\sim \epsilon$.

0 & those of the kind must be (usually) $<<\epsilon$.

This is opposite from t. usual problem of prediction, in which we want to find codes of as high pc as possl. (Wells seems to want same thing in 195.10, but we want to show this "Max $\sum pc$"

3 is $< \epsilon$.)

That discussn of 195.10 – 196.13 is good & useful, I still feel that I don't have a complete Understanding of what's going on. T. predn. for t. items of 195.10 seem to be quite distant from normal predn using UPD.

20 : 194.90: T. page cs of now: ① 3.26 SSZ uncertainty ② 3.31 Modal uncertainty: how it arises;

So: read th. paper's notes carefully, then outline what I want to write then (+ a) write it.

③ How this "incomputability" is distnt from inc. of $\sqrt{2}$ (Th. by Preparata)

④ Errors of unk. size due to invisible regrs in corpus (If we have an acceptable SSZ, we do have a bound on Pe error — But th. bound will not be w. the SSZ.

⑤ complete ↔ incomputable

⑥ ~~Always~~ examples: always a better model may be possl.

⑦ certain Scientists have confidence in low error estimates for system failures.

150.07 ① Incomputability of UPD usually irrelevant to applicn.

② What incomputability means in practice: in a selected Models. Uncertainty of "error" size.

e.g. Pseudo Random seq. uncertainty that we have best model — & how much better error procedure could be —

③ Part of this difficulty arises in duplicity of using UPD's approxns for predn. — it is true in all

10 151.00 at any pt. we will have a set of models, : Uncertainty is incomputability arises from encapsulated models.

(see 197.15)

4.27.03
NIPS

00 : **Statistical Estimation as a form of ART** :

In a good estimate, the scientist will bring to bear all of his experience in the area of the estimate. (This) estimate will be a reflection of that experience interacting w. the Data.

We can do analogy to Painting, in which the Estimate stands by itself & is not modified by the user. Or with performance art such as Music in which we may have a trained statistician to "interpret" the original estimate, to modify it to fit a particular problem.

The range is an extreme case of subjectivity of stat. estimates. In many cases, there would be much agreement among statisticians as to what a poly should be used. But in many important cases, no such agreement exists, and the estimation becomes an Art — The user must decide (to use) judge which one (he) likes.

0 Similar to use of "experts" in decision making. The user may want a "Second Opinion"

5: (Spec 196.90) :..... Two kinds of Uncertainty: ① uncertainty of prediction ② Model uncertainty.

S/M predn: [redacted] mins error. [redacted] in value of [redacted] share. (or we cannot [redacted] one [redacted] obliterated.

[redacted] We then [redacted] try many new models and find one w. [redacted] rms error, and [redacted]

[redacted] We suspect that there are better models to be found, but [redacted] at any point in the search for better models, we can never be certain that the best model we've found, is the best available, and we can't [redacted] devise any useful upper bound on how far we are from the best model possible.

In using approximations to UPD for practical prediction, we make successive approximations by considering more and more models, and by evaluating models with more and more accuracy. At any point in this process, we can stop and use a weighted mean of all of the models we've considered, as a grand prediction model. We can estimate the expected precision of this Grand approximation using cross validation — often with more economy of data than is normally used in model evaluation.

0 [redacted] estimate in any absolute way, is how far our current model. We know that by considering a large enough set of models, we will eventually get arbitrarily closely to the UPD. The uncomputability of the UPD means that at no point in this approximation process can we ever [redacted] know a useful bound on how close we are to the ideal UPD.

This limitation on our knowing how far our model is from optimum, applies to all methods of probability prediction and probability [redacted] evaluation — not only to [redacted] approximations to UPD, but to

428.03
NIPS

>D: 197.40: From the foregoing, it would seem that the incomputability of UPD would have no practical impact on our use of it or any other model for prediction.

Not quite True! I will mention two situations in which the incomputability of UPD is of immediate importance.

One is in intelligent systems — human or machine. We have several probabilities in different domains that must be evaluated in a certain total time, T. If we were able to know how close to optimum we were in our probability approximations, we could spend more time on evaluations that were further from optimum. The incomputability of UPD assures us that we can't do this. Malignantly allocate time to each problem or any optimum way.

Another problem is the evaluation of failure probability of very large complex systems, such as a nuclear reactor, a national defense system, the launching of a space shuttle, or rules designed to insure the safety of genetic engineering.

In many systems of this kind, the system whose failure probability we are determining, has not yet been built; or it has been tested in too few ways for too small a sample size for this data to be used directly in the evaluation of extremely small probabilities.

Instead, the probabilities are evaluated "a priori" using test data on parts of the complex system. Various models for failure are proposed and sequences of events that might lead to failure are proposed and evaluated. Strictly speaking, there can be no "incomputability", since (24)

FN In the RW systems like .11–.12 The relevants to occurs have to take a limited amt. of time — seq. of

So there can be no "Halting problem" — There can be no "infinite loops".

Ltr: (21) each test in sequence can only take a known, finite time — yet for practical purposes, the number of possibilities and the logic time needed to evaluate some of them, make the computation effectively incomputable from a practical standpoint

4.30.03

N1B

26:50
27:01
27:23
27:45
28:07
28:29

28:51
29:12

+1

Dinner Title: Use of the Universal distribution for Machine Learning ~~The Universal Distribution~~

00 : On Hol (talk/paper)!

01 ~~Discuss~~ May years ago the Pythagoreans discovered that $2^2 = \left(\frac{2}{1}\right)^2$ was unsolvable (there could be 2 interpretations: $\sqrt{2}$ was "incomputable" ~~many many years~~ ago, it took several centuries to get a good understanding of this problem — but even today we (I) don't think we've ~~really~~ good understood, approximate here ~~understand~~ understood.

02 In case of UPD, we know that UPD is incomputable, but we can make approximations, & we know of an approximation procedure that comes arbitrarily close to it. Final platonic ideal still evades us. The difference in the two situations is that for $\sqrt{2}$ we know a useful upper bound on how large the error is: for the UPD, we ~~don't even know~~ don't know what we cannot even know a useful upper bound on how much our approximation deviates from the ideal UPD.

03 Fortunately, for most applications, we don't need this information. What we usually want is ~~to know~~ not how much our approximation deviates from the ideal prediction, but rather an estimate of how accurate our approximation is — for practical purposes. This latter can ~~be obtained~~ usually be obtained if we have a reasonable sample size, ~~this~~ is easily obtained by cross validation, but often ~~we can do~~ better than cross validation: ~~since we can use a zero training set and~~ If the approximation we use is entirely a priori (derived before the data was known), we can ~~use zero training data, dispense with the~~ use all of the data for testing — since none is needed for "training" the model. → 200.00

22 Are there any situations in which the incomputability of the UPD is relevant? (I can think of one and one half situations; comment.)

One important situation is theory revision in science or Machine learning. A common criterion for theory revision or "Backfitting" is that the present theory does not predict as accurately as it could. In modern Physics, we expect that all theories to be as accurate as the precision of our instruments allows — ~~so~~ "when to revise" is not a difficult problem, so it is usually not difficult to decide when revision is needed.

In most other sciences (and in parts of Physics) the criterion for revision is not as clear, and knowledge of how close we are to the Universal Distribution (probably the best we could do with available data) would help make the decision.

Ups

(Spec 199.22) The the incomputability of the UPD is usually not relevant to the problems in practical prediction, it is of interest in the philosophy of science.

Some scientists are repeatedly disturbed by the need to revise their understanding of their sciences. They look forward to a final theory that will put an end to all revisions.

~~and the revisions~~. The incomputability of the UPD assures us that this never happens. With any amount of data and finite computing time, we can never know that we have found the best, final theory.

~~For many of us, this is not a cause for alarm~~ ~~Many~~ ~~this~~ does not at all disturbed by this state of affairs but find it to be a neverending source of joy in discovery.

For many of us, this incomputability assures us that science will continue to be a never ending source of joy in discovery. ? or

Big but a feature!

~~Before Cantor's diagonalization argument~~

In ancient Greece, ~~they~~ the square root of two

Let me explain

~~By Pythagoreans discovered that there was no finite integer~~ did not exist the square root of two could not be expressed as the ratio of two integers. ~~This number was "incomputable"~~

~~for them. Using the best, approximations were used~~ and eventually, it was understood ~~that the approximations could be arbitrarily close to the "true" (uncomputable) value.~~

It took the mathematical community several centuries to get a good understanding of this problem, but well before that approximations were made and used. None of the approximations ~~In a very real sense, the square root two was incomputable~~ was actually $\sqrt{2}$, but they got arbitrarily close.

Big Q: Does using purely Apri Model give "unbiased estimate" on testing?

It would seem that if err^2 was unbiased than usually err - would be not unbiased!

Now, if one uses rms error, i.e. $e^{-\frac{(x-\bar{x})^2}{2\sigma^2}}$ error distr. prob distribution, one may get unbiased error for σ^2 (or σ?) — For any power law error $e^{-\frac{|x-\bar{x}|^k}{\sigma}}$ is e.t. for p.d. —

TL value of σ is irrelevant to "fitting". The ~~true~~ true expected error will be the mean of something like $(X-\bar{X})^k$, & it may be that this will be "unbiased".

What does "Unbiased" mean? —— One possy: that if one uses "true μ" for source of a large bunch of corpi — that the average error will be the actual error.

→ This is apparently trivially true. It is true of any error estimation method.

NO! In 32 I was thinking of observed error rather than predicted error in predicted variable

However, in the case of 6 (linear predn of a real), it may be that if we sum over all possible "parameter vectors" in spaces we will get some kind of unbiased error.