08.04.03

Nips

50 How Rev2 differs from Rev1: (to be put at website! Also in letter to Correspondents)

A table of contents ~~added~~ has been added, ~~have~~ giving a description of each section.

The updating technique has been modified to include information on trials that have failed.

An advanced ~~[illegible]~~ search scheme is described that appears to be ~~[illegible]~~ significantly better than Lsearch. in important systems

Some short explanations have been expanded and various ~~[illegible]~~ small errors have been corrected.

10 Another Nice Expo for Rev2: Before we explain the new update/srch scheme: Describe it in general terms: From previous experience with various PST's on various problems, we ~~decide~~ find that ~~[illegible]~~ Fa is the most promising PST for the present problem, ~~[illegible]~~ Gn. We apply ~~that [illegible] to the present present problem~~ Fa to Pro for ~~an interval of time~~ a short time, then ~~[illegible]~~ in view of its degree of success, or lack of success, we revise our estimate as to ~~[illegible]~~ which is the best PST for ~~[illegible]~~ ~~[illegible]~~ Gn. ~~[illegible]~~ We then ~~to~~ apply this "~~[illegible]~~ Revised PST" (which ~~may~~ can or may not be the same as Fa) to ~~[illegible]~~ Gn for ~~an interval of time~~ a short time — This process is repeated until we solve the problem or our time runs out ~~our time runs out~~.

Insert before 13.16

But first, given a corpus of historical data on the results of various PST's working

20 on various problems, how can we ~~estimate~~ find the PST that is most promising for the present problem?

This alternation between use of ~~application of a~~ the best PST and revision of our ~~estimate of~~ choice of the best PST, continues until we solve the problem or run out of time.

13.16 → "Consider the set of quadruples . . ."

At the end of Section 3 (on OZ probs)

27 It should be noted that ~~the~~ time limited optimization problems are only well defined when the G(x) function ~~has been~~ is "linear"

29

30 It will be noted that if x maximizes G(x), then x will also maximize H(G(x)), if H is a monotonically increasing function. In general, however,

$$\int_0^\infty G\, P_0(G)\, dG = \bar{G}_0 \quad ; \quad \int_0^\infty H(G)\, P_0(G)\, dH$$

35 If $\int_0^\infty G\, P_0(G)\, dG > \int_0^\infty G\, P_1(G)\, dG$ $\qquad$ $\int_0^\infty G\,(P_0(G)-P_1(G))\,dG > 0$

then $\int_0^\infty H(G)\, P_0(G)\, dH > \int_0^\infty H(G)\, P_1(G)\, dG$ need not be true

$\int_0^\infty H(G)\,(P_0(G)-P_1(G))\,dH > 0$ need not be true.

$$dG = dH\cdot\frac{dG}{dH}, \qquad dH = dG\cdot\frac{dH}{dG}$$

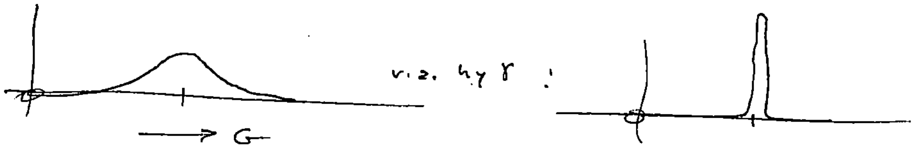
40 $\int_0^\infty H(G)\,(P_0(G)-P_1(G))\,\frac{dH}{dG}\,dG = \int_0^\infty H(G)\,\frac{dH}{dG}\,(P_0(G)-P_1(G))\,dG$

8.04.03

N.ps

10 : Say we ~~[illegible]~~ modify G by a monotonic ↑ funct: $\alpha(G) \cdot$ ~~[illegible]~~ G^γ

G is from 0 to ∞.



Drop this for a while.

Going back to OZ probs ($\approx$ 289.00–06) ~~[illegible]~~ and 289.22–40

Assuming Linearized G; we pick F_l w. h that the best expected G: 289.30.
This is a single pick — we then just run that F_l for t. specified time. We will not necessarily get a v. g. "G" value, "but on t average" for these OZ problems, we will get ≈ t. "expected ~~[illegible]~~ G's".

Say we have a OZ problems we really need to optimize it on t. eventually We mite divide up t. time into 10, say: work on t. most promising PST for $\frac{t}{10}$; Then do a reoptimization of 288.35 then work on PST that looks most promising, etc.
If we spend ½ time on ↗ & ½ time on optzn prms.
Instead of $\frac{t}{10}$; we can timeshare betw. 288.35 and various F_l's.

So, (except for the Linearization postulate) t. prog. would perhaps solve OZ probs.
In both Env & OZ, hvr, this "advanced update" will not work unless TM is able to get good models of O^2 in 288.35 & in t. corresp. ~~[illegible]~~ for ENV.

20 So do explain that ⓐ Normally x "Trainer" will decide when to switch to WON. & ⓑ That ~~the~~ best way to decide is to do same problems via WON & via Lsrch & see which WON seems to be doing better.

A hyly educated TM will usually be able to chose ~~[illegible]~~ very appropriate PST for most OZ problems, & so it wouldn't be much jumping from one PST to another. It may be wise to spend less time updating in such situations.

Tho, in general, updating is "self-improvement" in t. larger sense, & so it would be worth while to spend 50% of available time on it.

30 In doing Inv probs each cand could take a fairly small amt. of time . . (Not necly! — We are still trying PST's, & a PST has to look at t. problem & try to generate a soln . . . they can take a lot of time! — In fact, a PST could be a search (or Lsrch) & take as much time as we'll give it!

Nips

.00: 288.40 : ~~possibly G_j, t_j will~~ [illegible] ~~find an x with $G_j(x) = G^{j,l}$ within time t~~

~~O^i expects~~ [illegible] That F_l will obtain an x / in time t_j such that $G_j(x) = G^{j,l}$

Defining ~~[illegible]~~ [illegible] $h^{i}_{j,l}(G^{j,l}) = O^i(G^{j,l} | \tilde{G}_j, t_j, F_l)$ } Then j,l product is overall experience past optimization problem solutions obtained using the set of $[F_l]$

.06 We want a PST, F_l such that the expected value of G is as large as possible.

Q: about t. GHT! Fact "expected value" of G implies that G must have been "Linearized"!

"It will be the pt. at which weight may be of succ ess per unit time expended.
t such that
h(t)/t larx is largest

generally max $h(t)/t$ is t. Gorc

.10 $\boxed{SN}$ 1) Write about "when to switch from Lsch to "WON". : Mainly involves [illegible] } one could use both & switch when won was ahead.

being able to find good O^i functions

2) perhaps mention that max "$h'(t)/t$" is / measured Gorc, not $2/\mu$ (from GHT1).

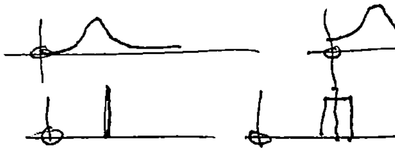
3) Is t. WON method I'm using actually optimum or near optimum? It seems very "Greedy" i.e. ⓐ work on "Best $h'(t)/t$" to peak; Then recalculate Gorcs, loop to ⓐ

I think Best $h'(t)/t$ is best initial (Greedy) strategy.

4) It would be nice to have pictures of k' cands

a) Ask Grace b) Maple prob can do it!

5) possibly include Concept Net!

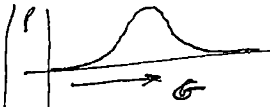


6) Perhaps Make big Zip file for all of my Downloadable papers! so people could

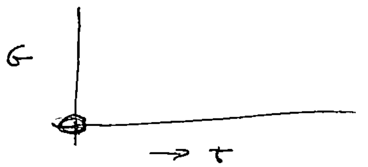
20 Download them in one fell swoop!

---

22 Actually, Prob density will be a funct of F_l, $\tilde{G}_j(\cdot)$ [illegible], t_j and G.

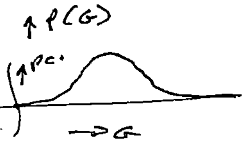


Prob density of G obtainable in time T.

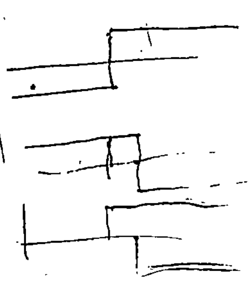


a plot such as this for each F_l.

Say we use time varying Gorc.



For each T value we have a prob profile of G!



To select t. "best" of these.

30 — If G has been "linearized" is $\int_0^\infty G\, pc(G)\, dG$ t. rite Gorc?

"Linearization" means pc' = po of G; is equivt to $\frac{po}{2}$ of 2G.

i.e. pc·G is what we are interested in. So t. "Σ" = "∫" pc·G dG is

what we want to maxz.

If G is not linearized, t. optzn problem is "not well defined". I really need more

info before I can give a good soln. In t. absence of knowledge of Linearization,

I can act as if G was linearized — which is not good, but may be t.

best I can do. I would recommend getting Lineariza. info.

Perhaps Title of section: ~~[illegible]~~ Conditional Probability Distributions.

(285.22 spec.) (282.00-.23): Start out by discussing d. funcs, s-funcs:

There is only one way to express D-funcs. (~~[illegible]~~ Then

(well no: say $x = f(t)$, $y = f(t)$ implies $y(x)$ but can be many valued ∴ not a function.

For $-functs there are ~~many~~ several ways to express them;

1)

AZ (alpha)

~~[illegible]~~ Can ~~[illegible]~~ assign pc's to any funct of k variables.

How different from Cond'l. Pd ?

$P(x,y)$ v.s. $P(y|x)$

?

$P(\vec{x} \to y)$

SN AZ can assign a pc to any funct of a vector (k vars, k scalars) ($y = F(x)$) ← AZ assigns pc to this. This is a ~~[illegible]~~ pd that is fixed by choice of primitives & formalisms of pc assignment. I want to be able to try various possl. pd's on all $y = F(x)$'s. By adding just 1 more input, ("essentially random" R") we get a (perhaps) universal s-funct. Consider ~~[illegible]~~ scalar funct of a scalar. $y = F_i(x)$. We have in AZ, a fixed pd on all such F_i. Here, in $y = F_i(x, R)$ the function depends on R and the pc of y depends on the (AZ assigned) pc of F_i mult by $2^{-|R|}$.

In AZ, the legal R (in fact, any input) must be a prefix set, so TM can tell where that input ends. A "stop symbol" is a common way to end — but ~~[illegible]~~ does this really give the pd that I want?

SN Looking at §2.1 " ~~[illegible] problems~~ improved updating and search techniques " (in IDSIA report)

It starts out looking for a Lsearch soln. — Also ~~[illegible]~~ Section §3 does same.

Try this: On p 14, after eq(7) $a_0^i \prod(h) \prod(1-h)$

"When we find a suitable . . . . at time t."

SN Section on "Updating" new end! Spending ½ time on updating & ½ time searching is perhaps not so clear in WON! — looking this!

26 This O^i becomes part of the updated GCPD (General Conditional Probability distribution) and can be used to guide Lsearch.

(insert at 14.20) ~~[illegible]~~ While it is, indeed, possible to run an Lsearch in this way, we will describe a search technique that ~~[illegible]~~ seems to be much faster ~~[illegible]~~ ~~[illegible]~~ than Lsearch.

Continue on: p 14 Given a problem, $(G_n, \S)$ . . .

---

32 Changes needed in §3.1 (OZ probs) [change Name of §2.1

(insert) We want to find O^i's such that

35 $$a_0^i \prod_{j,l} O^i(G^{j,l} \mid \tilde{G}_j, t_j, F_l)$$

is as large as possible:

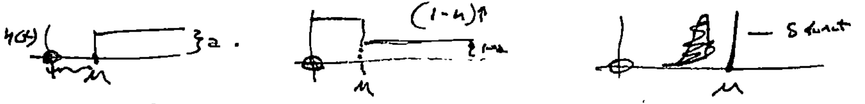
(G_j, t_j) describes the j^{th} optimization problem, to find, in time t_j, an x such that $\tilde{G}_j(x)$ is as large as possible. (in view of O^i)

$O^i(G^{j,l} \mid \tilde{G}_j, t_j, F_l)$ is the probability density ~~[illegible]~~ ~~[illegible]~~

8·02·03
NiB

90 : I want to discuss how s-functs are to be realized in QATM!
Have a new section on "Stochastic Functions"
Do give examps of a 3 param ∩ h'(t) curve as a way to get a s-funct from a 3 output d-funct. In section 2.1 A ~~very good~~ much improved update technique,
~~[illegible]~~ Refer to this section for examples.

SN: Since I really don't use the σ in h'(x) of the update function (§ 2.1 of report)
Why not use σ = 0

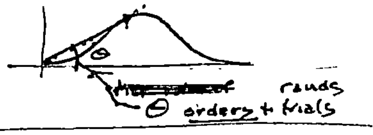


would I get φ pc for some cases?
I would! For h' I would almost allways get ~~[illegible]~~ p=φ except for when $t^{true}=\mu$ at which point density=∞.
So it looks like I will have to model σ, even tho I don't care about its value — it does contribute much to "Goodness of fit".
13

So while Eq(1) p5 is correct, ~~[illegible]~~ Appendix A tells how to get sums of d functs, not the stuck of eq(1).
We can start our TSQ w. d-functs (MTM TSQ), but eventually we will want to work on s-functs. At one time I thot that it was necessary to get TM to do products of s functs — ~~[illegible]~~ so it could do updating. But recently its become clear that u.d. updating can be done w. ~~[illegible]~~ pure d-functs. I still don't have a ~~clear~~ good idea how to construct vector outputs of functions, using economical codes, so all vector components functions share some costs. → 288.00
23

N.B. GHT-I ~~assumes~~ (that $\frac{pc}{cc}$ ordering is optimum) assumes the pc's are uncorrelated — i.e. that knowing one trial failed, ~~[illegible]~~ leaves other pc's invariant. The ~~ordering~~
~~[illegible]~~ rec optimization of the $\sum_0^2 TTO^i(QA)$ takes advantage of correlns betw the cands.

In WON/GHTI, the ordering of the h's is by max values of
0 $\tan(\theta)=\frac{h'(t)}{t}$ .. Each h' has a t value so this is Max.
the largest θ for a cand, becomes its "ordering index"



If we then define trial to ~~[illegible]~~ extend out to that max θ pt. and stop, then $\frac{pc}{cc}$ obtained is exactly right, for the GHTI! The only part that causes it to fail, is the correlns of the pc's of the cands.

We say "suggests" because the Gambling house theorem assumes the ~~[illegible]~~ success probabilities of the trials are uncorrelated — i.e. when one trial fails, ~~the~~ the ~~[illegible]~~ success probabilities of other ~~cands~~ trials do not change. In our update scheme, we will take advantage of these correlations ~~[illegible]~~ to speed up our search.

Rev2add2.tex ← Title of file.

Transcript of 284.06—285.9

Section 3.6.copy

00 Still How to chose among them? There are two derivatives in a h'(t) function

First consider $a = \int_0^\infty h'(t)dt$ This is the probability that the associated PST, F_ℓ, will (can) cover solve the problem of interest. We want a ~~[illegible]~~ to be as large as possible.

Next consider $\mu = \int_0^\infty t\, h'(t)dt \, / \, a$ "divided by" ⊙ ~~[illegible]~~ for those F_ℓ that eventually do solve the problem, this is the expected time to get that solution. We want μ to be as small as possible.

In the present case, The first Gambling House Theorem

0 {$\fn " At a certain gambling house there is ..... p_k/t_k gives least expected time to win" ⊙No (Sol 86 section 3.2)} suggests that we will minimize expected total solution time if we schedule our F_ℓ trials so that ~~the~~ their

13 associated $h'(t)$'s are in a/μ order: largest values first. ~~We say "suggests" because a/μ is not exactly the same as probability of success divided by~~

5 ~~time to complete a trial.~~ Replace by .37 –.40

Obtaining a set of PSTs of high a/μ value is a time limited optimization problem that is solvable by the techniques of section 3.

0 To solve our original Inversion problem, we first try the PST of largest a/μ. If it has not been solved by time μ, we reoptimize ~~eqn~~ equation (282.21) using the additional information that the present PST has failed up to time μ. If, after this reoptimization (continuing to work on) this PST still ~~[illegible]~~ gives the best a/μ, we continue working on it ~~[illegible]~~ ~~until latest μ — If the~~ — otherwise we switch to a more promising PST.

It will be noted that the foregoing technique is not at all, "Lsearch" — In fact, it seems to overcome a serious deficiency of Lsearch. If there ~~[illegible]~~ are

0 many trials that are identical or nearly identical, but which ~~[illegible]~~ have different codes of the same length, Lsearch will test all of them — which is quite ~~[illegible]~~ wasteful. The technique just described will usually test only one of them ~~[illegible]~~ ~~[illegible]~~ when a candidate is abandoned because it ~~[illegible]~~ no longer looks promising, usually candidates that are identical or very similar to it, will ~~[illegible]~~ also be abandoned.

37 We say "suggests", because the Gambling House Theorem assumes that the success probabilities of the trials are uncorrelated — that when one trial fails, the probabilities of success of all other trial do not change. In the update schemes we will describe how to take advantage of existing correlations to speed up our search

1

Insert at .13–.15

8.01.03

NIPS.

[illegible]
transcription
of 18~~4~~½.06–185.01

00 because it no longer looks promising, candidates that are identical to or very similar to it, will usually also be abandoned.

.01

02 [SN] To get h, h', one way is to get the 3 vector a, μ, σ as a d-function of ~~E~~ G_n, S_n, F_e. (presumably, we know how to do "d-fcns")

For computational purposes, h & h' need not match exactly — just so they ≈ match a, μ, σ. We would like $h(0)=0$, and $h(x)$ should be ↑ funct of x. We might use same functional form all the time ~~x~~ say $h \sim x^n e^{-x}$, and use TLU to get values for both h' & h.

For h, ~ tanh and $\frac{1}{1+e^x}$ or other squashing functions from ANN can be used.

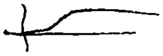
0 Because of the form of h, would it be good to use ANN for models? — perhaps use feed forward or more economic (other exp. [illegible]) methods. Since the models will not be/need not be very accurate, we can afford very approx. models — but we have to be careful that the errors of the models do not screw up the results! — so we have to find just what aspects of the models are critical, & which are not.

Other imp't topics:
~~2.~~ (at least)
–2 ways to generate S-functions,

Perhaps mention using the 2 param $h'(t) = a\delta(t-\mu)$. We may try fitting these 2 params as a separate problem, then try to find them as a function of F_e, G_m, S_n.

10

Maybe not such a good idea! See 287.00–.13!

A FET ~~[illegible]~~ transistor could give  as output — starting as square law, and ultimately saturating.

.22 → 288.00

284½.34–.35 isn't exactly correct: try: If continuing work on the present PST still gives best M/d, we continue working on it — otherwise switch to a more promising PST.

30

8·01·03

Nips

0 : Expo following lines of 13.28–.29

"When we find a suitable ~~#~~ O^2, then for new problem $\tilde{G}_n, S_n$ and arbitrary F_e, ~~[illegible] and new prob~~
we can find the probability $h^{2'}_{e,n}(t)$ that F_e will solve that problem ~~time~~ at time t.

Then 14.11 ff: Given a problem (G_n) and a good O^2 function, there are an infinite number of ~~[illegible]~~ F_e functions ~~then~~ for which ~~[illegible]~~ O^2 can obtain associated $h'(t)$ functions. } This is already in paper! found stuff just before pt.

06 ~~Then~~ How to choose among them? There are two desiderata in a $h'(t)$ function:

first consider ~~[illegible]~~ $a = \int_0^\infty h'(t)dt$, the probability that the associated ~~[illegible] problem will ever~~ PST, F_e, will ever solve the problem of interest. We want this to be as large as possible

) Next consider ~~[illegible]~~ $\mu = \int_0^\infty t\, h'(t)dt$ / ~~[illegible] as small as possible.~~ ~~[illegible]~~ when ~~there is~~ F_e does ~~[illegible]~~ a solution, μ is the expected amount of time to ~~get~~ that solution. We want μ to be as small as possible.

~~In the present case, we will argue that minimizing μ/a is a very good criterion for how good a $h'(t)$ is.~~
~~Consider the first Gambling house theorem (Sol 86 ~~#~~ section 3.2, p479)~~

In the present case, the ~~[illegible]~~ first Gambling house theorem {fin. copy / part of Sol 86 section 3.2 ~~then give reference to Sol 86 section 3.2~~} ~~[illegible]~~ suggests that we will

-0 minimize our expected total solution time if we schedule our ~~[illegible]~~ F_e trials so that the associated $h'(t)$'s are in a/μ order: largest value first. We say "suggests" because a/μ is not ~~exactly~~ even the same as the ~~[illegible]~~ probability of success divided by the time to complete ~~[illegible]~~ a trial.

~~[illegible]~~

~~[illegible]~~ Obtaining a set of ~~[illegible]~~ PST's of high a/μ values is a problem that can be solved by the time limited optimization technique of ~~[illegible]~~ section 3.

~~This set of good PST's will be~~ To solve our original inversion ~~[illegible]~~ problem, we first ~~[illegible]~~ try the PST of maximum a/μ. ~~We use this~~

~~PST until the problem is solved, or until μ time has elapsed ~~[illegible]~~ which~~

5 ~~PST / for μ to time~~ If the problem has not ~~[illegible]~~ been solved by time, μ, we ~~[illegible]~~ reoptimize equation (2.82.21) using ~~[illegible]~~ the additional information that ~~[illegible]~~ ~~[illegible]~~ our present PST ~~[illegible]~~ failed at time μ.

34 If ~~[illegible]~~ continuing to work on our present PST ~~is~~ still ~~has~~ gives the best a/μ, we continue

35 working on it — otherwise we switch to a more promising PST.

It will be noted that the foregoing technique is not at all Lsearch. It seems ~~to~~ able to overcome a serious deficiency of Lsearch. If there are many trials that are identical or almost identical, but which have different coding, Lsearch will test all of them — which is quite wasteful. ~~We~~ The method just described will usually test only one of them. This is because when ~~one~~ work on a candidate is abandoned

2.31.03

Nips

: A rewriting of 282.07 - .29 in more readable form

Note that ~~Though~~ (13.D) uses only information from successful (rem) ~~problem~~ attempts to work problems. This expression ~~rems~~ must ~~[illegible]~~ be modified to include information from failures as well.

Befor explaining how to do this — some simplification of notation:

Let ~~[illegible]~~ $h^{i\prime}_{j,l} = O^{i}(t^{jl} \| (\tilde{G}_j, \overset{\text{i.e.}}{s_j}, F_l))$

This is the probability density (according to O^i) that PST, F_l will solve the problem $(\tilde{G}_j, s_j)$ at time t^{jl}.

Let $h^{i}_{j,l}(t) = \int_0^t h^{i}_{jl}(t^{jl})\, dt^{jl}$ (according to O^i)

This is the probability that by time t, F_l will have solved the problem $(\tilde{G}_j, s_j)$.

$1 - h^{i}_{m,k}(t)^{mk}$ is then the probability (according to O^i) that F_m has failed to solve $(\tilde{G}_k, s_k)$ by time t.

Consider the expression:

power! put space between ' and / — so / is above "l"

$$a_0^{i} \prod_{j,l} h^{i\prime}_{jl}(t^{jl}) \prod_{m,k} ~~[illegible]~~ (1 - h^{i}_{m,k}(t^{mk})) \qquad \text{(eq 282.21)}$$

The first product is over j,l pairs in which various F_ls have been successful (rem) at times t^{jl}.

The second product is over m,k pairs in which F_k has failed (rem) to solve problem m, by time t^{mk}

We want to find a O^i such that (282.21) is as largeas possible — the O^i that makesmost likely, the observed successes and failures.

\widetilde{G}

7.30.03

N.ips

.0: 282.29: A possl approach: to discuss option of $\frac{n}{2}$: then do won; then go back & do Lsrch as Alternative way. Discuss General trouble w. Lsrch due to "many ~ conds" problem.

N.B. on pp13 & 14 find out how to put space ~~[illegible]~~ between i and $/$ in $h^{i/}_{m,l}(t^{n,n})$

nips02.tex ← 5min (later ~~add~~) more recent

vs nips02new.tex

Having obtained a suitable O^2, ~~then we at least~~ we will ~~[illegible]~~ discuss two ways ~~[illegible]~~ that the updating can proceed.

SN It may be best to get F_e's in ~~[illegible]~~ $\frac{n}{2}$ order first, then branch into 2 methods.

Another possy. is to do won only: Do analysis purely in terms of ~~[illegible]~~ n & S_n & S_{tn} & S_{t^2n}.

Then give $w = ax^n e^{-bx}$ as example & give its ~~[illegible]~~ a, n, d^2.

~~so w~~ Discuss eq. (3.1 (say) w. O^2 giving a, n, b as functions of F_a, & G_n; S_n.

Perhaps appendix w. outline of proof of "GHT#1."

SN ~~B~~ In discussn of Grammar v.s. 3 I U models of universe / S. Functs:

T. ~~Gramm~~ "Grammar" model uses functions that have (occasionally), limited Domain: i.e. there are certain inputs for which they "have no opinion" (these may be Partial recursive ~~[illegible]~~ opinions — so one often doesn't know! for sure then whether t. machine has an opinion or not), Anyway: Having no opinion is not at all t. same as having pc = 0 for all outputs w. that input. — The Exact interpretation is to be obtained by looking at t. proof that t. grammar & 3 I U models are equivt.

In generally ~~[illegible]~~ functions that have zero output for certain inputs can be "outlawed" — so if a funct does have zero outputs we regard it as showing "no opinion" about those inputs (this, of course, is subject to "change of opinion" as CB13 ↑)

p1 ~~drop~~ n cov unmodular

2 extralines

2:4 xtra

3 6 extra

4 eq(4)+2 lines

5 8 extra

6 9 + spaces

7 14 lines extra

8 14 + many extra

9 20 + lines

2.29.02

NIPS

a, ℓ, j : i, ~~[illegible]~~ m, n.
ℓ, i, j ' ~~[illegible]~~

.00 : 279.40 : P15 of report: for "eq. 13.1" ($\simeq$ equ 5.5) write : deduce this

$$a_0^{i} \prod_{j,\ell} O^{i}(t^{\ell j} | (\tilde{G}_j, s_j, F_\ell(\cdot|\cdot))) = a_0^{j} \prod_{j,\ell} h^{i'}_{\ell,j} \quad \text{eq (13.1)}$$

After 2 lines following eq (13.1) . . . . . ~~[illegible]~~ $(\tilde{G}_j, s_j, F_\ell)$, A_j Pairs. ← ≅ line 13.27

~~Let us simplify~~

.07 Though (13.1) uses only information ~~[illegible]~~ from *successful* problem solutions, ~~[illegible]~~ ~~[illegible]~~ we can ~~also we can~~ modify ~~[illegible]~~ it to include information ~~[illegible]~~ from ~~unsuccessful~~ failures ~~[illegible]~~ as well. Before explaining how this is done we will simplify the notation a bit.

.0 Let $h^{i'}_{j,\ell} = O^{i}(t^{j\ell} | (\tilde{G}_j, s_j, f_\ell))$

This is the probability density that PST, F_ℓ will solve the problem $(\tilde{G}_j, s_j)$ at time $t^{j\ell}$.

Let $h^{i}_{j,\ell}(t) = \int_0^t h^{i'}_{j,\ell}(t^{j\ell})\, dt^{j\ell}$

This is the probability that by time t, F_ℓ will have solved the problem $\tilde{G}_n, s_n$.

$1 - h^{i}_{m,n}(t)$ will then be the probability that F_m has *failed* to solve $\tilde{G}_n, s_n$ by time t.

~~We want to maximize the mo~~ Consider the ~~product~~ expression

.0
.21 $$a_0^{i} \prod_{j,\ell} h^{i'}_{j\ell}(t^{j\ell}) \prod_{m,n} (1 - h_{m,n}(t^{mn})) \qquad \text{eq } (282.21)$$

The first product ~~[illegible]~~ is over ~~[illegible]~~ j, ℓ pairs for which various F_ℓ have been *successful* in solving problems at times $t^{j\ell}$. The second product ~~(over m,n)~~ is over m, n pairs for which F_m has ~~not~~ *failed* to solve problem m ~~to solve the m th problem~~ by time t^{mn}.

We want to find a O^{i} such that 282.21 is as large as possible — i.e. the O^{i} that makes most likely the observed ~~[illegible]~~ successes and failures. → 284.00 spec. → 285.00 ← Grace has.

.28
.29

Next correct the details of 13.28 — 14.23. on a, μ : Also give expression for σ^2.

.0

M_0 ~~[illegible]~~ $\int_0^\infty \alpha x^n e^{-ax} dy = \alpha\, n!\, a^{-n-1}$

M_1 $\int_0^\infty \alpha x^{n+1} e^{-ax} = \alpha (n+1)!\, a^{-n-2}$

M_2 $\int_0^\infty \alpha x^{n+1} e^{-ax} = \alpha (n+2)!\, a^{-n-3}$

M_0 : $\int_0^\infty \alpha (ax)^n e^{-ax} dax = \alpha\, n!$

$\frac{M_1}{M_0} = \boxed{\frac{n+1}{a} = \mu}$

$\sigma^2 = \frac{M_2}{M_0} - \left(\frac{M_1}{M_0}\right)^2$

$= \frac{(n+2)(n+1)}{a \cdot a} - \frac{(n+1)(n+1)}{a \cdot a}$

$= \frac{1}{a^2} \cdot (n+1) = \frac{(n+1)}{a^2}$

$\boxed{\sigma^2 = \frac{n+1}{a^2}} = \frac{\mu}{a}$

$\sigma = \frac{\sqrt{n+1}}{a}$

$\frac{M_2}{M_0} = \frac{(n+2)(n+1)}{a^2}$

$\boxed{\text{wo want } \frac{\mu}{M_0}}$

Big error! ⇒ 330.00

Normed Γ distribn: $\boxed{\frac{1}{a^{n+1} \cdot n!} x^n e^{-ax}}$

$\alpha x^n e^{-ax}$ slope = $\alpha(n x^{n-1} e^{-ax} - x^n a e^{-ax}) = \alpha \frac{x^n}{x} e^{-ax}$

$\mu + \sigma = \frac{n+1+\sqrt{n+1}}{a}$

$(n-1) - xa = 0$

$x = \frac{n-1}{a}$

this is slope of

its this is just the value of x ($\equiv t$) at the max pt.

7.~~[illegible]~~ 29 03

MX : Machine XLtn.

Nips

: Using ~~Language~~ Grammar Models that are "Nonparametric" — or just Grammar Models that Scale (upto SSE.

7.29.03
NIPS

STATISTICAL Machine XLTN

20 : On Mach xltn; statistical. ~~[illegible]~~ Franz Joseph Och: As he is doing it, he puts in a bunch of ~~[illegible]~~ sentence pairs from L_1 & L_2. (Q_i, A_i) & then is expected to get the new Q_j. — So it is a QA problem. — But as it is, it probably doesn't do incremental lrng. — ~~Also, probably t. statistics model~~ But there may be some "incremental lrng" by successive improvement of models.

Anyway, this looks like it might be very useful for getting ideas for the more ~~General~~ general QA problem. (Not necy linguistic; ~~[illegible]~~ t. solns. seem to be based on statistical distributions, "Grammars" quite different from those in

0 NL) Also, i'd want t. system to learn a second lang (text pair) more rapidly, after it has lrng to xlt a first pair. (Also, perhaps, t. lang pairs could share a "lang. in common".)

I have a bunch of Och's papers in D:\PS\M Translation-statistical. I didn't copy all of his, hvr. I downloaded 1 pgm! he has 2 more that I didn't download.

20 ((Q! Can I use his pgms (w. minor modifn) to learn Algebra & go to ~~[illegible]~~ Phase 2?))

I had it printed up: Put into "Grammar" files

22 2 papers on Allignment: I started w. shorter paper (both 2002): They use HMM (hidden Markov Models) — like in Voice Recogn. — ~~those~~ (I downloaded a paper on relation of MT to V recog.).

An Impt. Characteristic of MT is that words in Q correspond to ~~[illegible]~~ words in A. Certainly this is not true in many other kinds of QA problems.

~~[illegible]~~ ~~[illegible]~~

What about genzn of mapping from (word seq) to (word seq)?

30 It may be that t. allignment problem(s) of .ZZ are critical to understanding what they are doing.

Another impt. difference betw. my QA TM & this MT lrng pgm is that I expect good results from small SSZ. — Much smaller than this pgm uses.

Other than t. word to word mapping & his use of excessive SSZ, what are t. xforms that are learned? Could they be genzd so I could use them in ~~[illegible]~~ much more General problems!?

→ Is his enormous SSZ necy? The enormous SSZ may be due to using

0: 275.17 : $\boxed{\text{oops}}$ — criticism — : ~~the~~ reading the first few pp: I think he's over hyping how good it is in optimizing / Modifying its pc's: In "Optimized search over Search methods"

In fact he only has 5 pc modifn. methods; it is by no means clear that the system using them (or any other similar ones I can think of) is at all "optimum" in any way. Of these 5 methods only ~~two~~ one, boosting, has been found useful.

That update routine of ~~t~~ time limits (Time dependent Pi's for) p_i is actually $\boxed{\text{not bad}}$. It multiplies Σcc by between 2 & e each round". — Tell J. this!

Actually, my original analysis was wrong: Say T_i is time spent by on cand in pc $= p_i$

so $T_i = (1+T_i)p_i$ This is conditional termination of trial.

$T_i = p_i + p_i T_i$; $(1-p_i)T_i = p_i$ ∴ $T_i = \frac{p_i}{1-p_i}$; $1+T_i = \frac{1-p_i+p_i}{1-p_i} = \frac{1}{1-p_i}$

$\prod_{i=1}^{n}(1+T_i) = \prod_{i=1}^{n}\frac{1}{1-p_i}$ ∴ if n is large & all $p_i = \frac{1}{n}$ then $\left(\frac{1}{1-\frac{1}{n}}\right)^n = \left(\frac{n}{n-1}\right)^n = \left(\frac{n-1}{n}\right)^{-n} = \left(1-\frac{1}{n}\right)^{-n} \approx e$

Hm if $p_1 = \frac{1}{2}$, $p_2 = \frac{1}{2}$ $\Pi = \left(\frac{1}{1-\frac{1}{2}}\right)^2 = 4$ So it looks like factor is e minimum,

& can be $\gg e$ say $p_1 = 1-\epsilon$, $p_2 = \epsilon$ $\frac{1}{\epsilon}\cdot\frac{1}{1-\epsilon} \approx \frac{1}{\epsilon}$ ∴ arbtly large.

So, if one of the p_i is very close to 1, factor can be very large.

So, ~~e~~ factor of 3 is optimum, a factor $\gg 3$ is bad.

My impression is that usually all of the p_i will be $\ll 1$ so the factor will be $\approx e$.

~~2 papers on "Alignment": I started reading the shorter one.~~

~~They use HMM (Hidden Markov Models) like in Voice Recogn (Recall a paper I copied on relation of M.T. to Voc. Recogn.)~~

I have found $\boxed{\text{Nips 28.06 – .28}}$: (A discn on $\boxed{\text{How to use failures}}$). Into Integrating this into §2.1 (Much Improved Updating Technique) may be difficult. I could just add it as a footnote or appendix. Best rewrite that part of the section!

As for the WON method; I could just add it as an extra section, saying that it will probably work, but that I am not sure it will work for all problems, ~~as~~ I have not yet discarded the Lsrch method. Is there any point in this report where I discuss GHT #1 ($\frac{p_i}{cc}$ is optimum ordering) Perhaps not: So I will prob'y have to introduce GHT 1.

But perhaps easy to approximate: i.e. Q is how much accuracy do I need?

In discussion of || updates, note that the d.f.'s of other PST's will not change much at all

T gets $> \mu$ — since no soln. yet for $T < \mu$, is not at all surprising. $\prod_{k=1}^{r}\left(1 - a_k\int_0^T x^{n_k} e^{-b_k x}\,dx\right)$ can be too consuming to compute!

μ ← mean of cutround PST distribn.

If I use $x^n e^{-bx}$, the products for the successful cases, will be simple, but the product for the failures will be quite complex!

what is $\int_0^x x^n e^{-x}$ See ~~G&R 92: 232 ff~~ $= x^n \cdot e^{-x} \cdot \sum_{k=0}^{n} -\frac{1}{x^k}\frac{n!}{(n-k)!}$ (n ← ∞ also works!) $= \boxed{\frac{x^n e^{-x}}{n!}\sum_0^\infty\left(-\frac{1}{x^k}\cdot\frac{1}{(n-k)!}\right)}$

Add extra section on the WON method! Give heur Args about GHT#1, etc. discuss "|| updating", but mention that I have included both methods because I am not yet sure that WON will work! Args for its effectiveness are "good", but ~~...~~ I don't yet ~~feel~~ certain that it will work

Spec 282.00

7.27.03

NIPS RP (report).

"Pathways to Intelligence" on How to write TSQ's

Refs to revisions: 263.00 - 40 ; 265.00 - 40

Comments by J. of int:

1.10.03 ① on the "sections". ② He says OOPS will improve its "basic" specificity 35c4 lines — but it is unclear (in present OOPS model) how that ~~could~~ would occur! — that is 33 ff. entirely Q of how OOPS is to do Metalearning.

He mentions "OOPS RL" (Reinf. learning) — tries to Maximize future reward.

Comments by J. on ~~[illegible]~~ second version of report:

1) in 11.27.02 Letter to J: remarks 3 & 4 are of import. — (J. didn't remember ~~[illegible]~~ 4 first remainder of my remark #3, so I guess I didn't put it into/report

See if my remarks 3 & 4 can be incorporated into report — perhaps

2) subscripts) But see that they have not all really been included — since those remarks were before first revision.

A Good Approach! 1) List sections, etc, that I have to write & write them.

~~2~~ Make tentative table of contents. 2) Make list of ~~[illegible]~~ corrections, Additions I want,

— Write them & insert them. || 263.00 ff more or less does this.

Sections 1) Table of contents w. summaries

2)

7.25.03

Nips

RV

0 ;277.00 I want to do 2 things in next month: ① Review state of TM, record & discount goals. ← This is pretty to interest Grad. Students

??!! (See Sol89 on this) RV ② Finish up — or make next revision of ~~Report~~ Report. → ≡ RP

The following text will (alternately) discuss work on these 2 problems.

03 RV: See §6 of report: pp 18, 19, 20 : [redacted]

Also see Sol89 for "present state," what needs to be done, etc.

Some might kinds of TSQs that might be tried,

1) ~~E~~ Elementary Algebra

2) Symbolic Integration: Try to learn techniques of present symb. Int. "expert systems"

3) ~~Start~~ Start w. a well developed Expert system (in Literature)

Try to devise TSQ to learn to do what ES does

10 4) Learn English about Algebra — Then gradually move to other subjects that TM knows about: a/o have it learn via data in "English" (or formalized subset of English).

~~Note~~ NOTE I. writing ~~of~~ of TSQ's has 2 goals: ① Teach TM ② Teach TRAINER how to write TSQ's: How to teach TM. Maybe discuss "Hints", Scenarios TSQ's, "Ideal CJS at each point in Life of TM.

16 See 253.04–.40 for discusn of Series of Talks. To be given at MIT, say,

18 5) ~~Try to Show how this~~ ~~Discuss~~ List main problem in ML research & show ~~how~~ how this Approach addresses them. → (I think ∃ previous note on this) ! → See 253.27–.40 ! Quite different!

20 Main probs: ① Great variety of probs solvable. ③ ② Adequate induction Algms. (M) Compatibility Subjectivity This Defines the ML problem N.B.

④ ③ Adequate srch algms.

order ① ④ The "educate from baby to college: Solve problem, kids Ego 'start over'" approach is bad.

24 Solved by ① Inc. lrng ② Great variety of probs solvd ③ Good trans for lrng. algms.

30

Super Prostate Formula
1.800.318-5723

7990+595 3 pills/day
60 days worth of pills
7.26.03 Late of purchase

Cust Serv 1.800.789.3438
929 117 order

via 7–10 ~~Bus Days~~ Business Days.
60 day MBG from day of receipt of pills:
Phone Cust Service
say wk. Before return of empty Bottles.

July 20, start.

NIPS

.00 : So we have t. set of functions of 275.34 – .40 : ~~we~~ Since this set of funcs ~~[illegible]~~ decrbs t. corpus, we want to minimize its b cost, so we devise a grammar (S), that assigns pc's to t. (d-) functions we use.

T. system would work just as well if we had S-funcs in line (.02), – ~~B~~ in some cases, it mite be easier to work probs in that form.

.05 — Which brings us back to t. System of 271.00 – .20, that we wanted to show was Universal. (fine)

→ T. ~~Forgo~~ ~~soft~~ gives a good understanding of Section 1 of Reports where I had Recognition functions (R) & stochastic operators (R(?)). Recently I'd been it's very General(.). – Also its mainly of it is a very AH method, but its not at all AH : a way to go from a formalism for d-funcs to one for S-funcs.

.09

0 : 267.40, 268.27, 272.16 — On ANL again: I just wanted to get some CJS's for learning up to general Polish notation evaln. ~~B~~ After this, ~~[illegible]~~ get CJS for k rather than just 3 binary operators;

Then have it learn 1 & 3 arg. operators.

An alternative TSQ would learn 3,7,+; then 3,7+8+, then 3,7+8+5+ etc.

Then maybe 3,7,–, then 3,7–5+ etc.

The solns to these problems could be unique & easy for me to discover if I use only d-induction.

If I allow S induction, there are more poss TSQ's & more possl. solns.

A general soln is Do(if num then push, if $+_s$ then to, if $-_s$ then $-$op, if $\times_s$ then $\times$op) until $\#_s$

4 · 1   4 · 2   4 · 3   4 ← cost from menu 5

4   5 : pc of call

.10 - - - - - - - - - - - -

So $rc \approx 4^5 \cdot 1 \cdot 2 \cdot 3 \cdot 5 = 4^4 \cdot 5! = 256 \cdot 120 = 27,120k$

probably more! ← → So $27,120 \times 16 = \boxed{433,920k.}$ for final soln for arbg expression only 3 binary functions.

do ~~S~~ S+ ; S+ $\overset{4}{\to}$ S+ S+ $\overset{4}{\to}$ S+ S+ S+ $\overset{4\ 5+}{\to}$ S+ S+ S+ S+ ~~S~~ $4 \cdot 4 \cdot 4 \cdot 4^4 \cdot 1235$

num + – ×

push 4 4 4

Do I have all details correct? If num then push : 4 5 4 rc = 80 ~~[illegible] ; rc = 4~~

problem is "6" soln is "push 6 on stack." = "push" — put soln in store as a possib call.

~~then~~ Actually, t. soln is just "push".

for t. problem 6,8, t. soln is "push, push"

No! The grammar can't just do unconditional "push" so – is a poss soln. "push" is not.

For problem 6,8 $+_s$; soln is push, push top.

At first, content of store will be "St"'s only (statements) — i.e. solns. to problems.

Be sure to run "trial" funct. n. just "+ & –", then +, –, ×, then + – × ÷

7.23.03

NIPS

› : I'm not so sure 274.30–.40 is of any value! ☺. → Woops! It looks O.K. now → (.34)

Perhaps wait until I have found s-functs that are useful & I want to convert them from Gramm to 3IU or from 3IU to Gramm. — e.g. 272.09–.16

.03 → [SN] OOPS uses incremental lrng only [redacted] via boostg : It does not use definitions shared between problems. If true, it's a very serious criticism of OOPS!

Hvr. OOPS does have memory of previous solns. to problems & ~~I~~ I think J. expects to be ~~[illegible]~~ "editing" these ~~[illegible]~~ to use in trial solns to new problems. —

But this is not saying much! The only thing he's done so far is boostg plus a ~~[illegible]~~ insb. to store & use "patterns" of token use. So he has not really solved the problem

› of how to do "transfer lrng" — except, perhaps, between problems in same "set"

— like for the grammar problems, he did try concatenating on to old successful fcns.

→ But did he retain [redacted] tokens defined from previous problem solns? (even in same "set of probs" (i.e. all in the "same type of grammar problem)

I have to ~~r~~read the OOPS paper again. Many things are unclear as to how it works! — But in general, his facilities for (a) Recalculating pc's (b) use of info in previous solved problems; is Minimal. →279.00

.17

18 Review: I want to do 2 main things now: (1) "finish up" report. (2) Defn of state of the "project": what has been done, what needs be done, what ~~can~~ can be

0 — worked on now ~~[illegible]~~ as a "well defined sub-project" for PhD work, say

21 for David Lindsay, ~~[illegible]~~ & MIT prospects | & Long term v.s. short term goals → 277.00

22: 274.22 (SN) N.B. In order to allow universality, the individual d-functs ~~[illegible]~~ probably must be able to say "no output" for certain inputs. I'm not sure of this, but the proof of 274.12–.26 demands that the functs have that capability. If all d-functs have legal outputs for all inputs I don't know if universality is possl. or impossl. In my earlier analysis of this problem, I may have assumed output ~~for~~ for every input & perhaps proved that universality was impossl. & (not realizing that this was the crippling assumption).

.29 My treatment of "Recognition functions" ("R functs") in my present "DSIA Report" (Section 1, pp 6, 7 ~~(...Maybe 8)~~ Maybe p.8 also) — P( )

.30 — Does find a way to create ~~[illegible]~~ s functs that have output for only certain inputs, but the

31 technique would work just as well if P( ) were ~~d-functs rather than~~ s functs. could be a ~~[illegible]~~ useful way to think

32: 272.16 In fact, the Analysis of "R functs" in (.29 R.f.)

about. 272.09

.34: (.00) ~~[illegible]~~ Well, this makes 274.~~30~~–.40 more ~~[illegible]~~! (.22–.31). The idea is that one has those d-functs — each one has

› been ~~[illegible]~~ designed to work only on certain "kinds" of problems. (The functions themselves tell which kinds of problems). Each d funct has a wt. The resultant (potentially universal) s-funct. is a normalized ~~[illegible]~~ probability wtd sum of all these d-functs. For a given Q, it has to be normalized — it can be 1 usually; because of (.39–.40) — not nearly <1 — since the sum of all pc's need not be 1 — ~~(it's usually <1)~~ — certainly not.

39 Q: do the sum of all wts of all of the functions have to converge? In 274.19–.22 they don't — they

40 are 2^{-i} for $i=1,2,3\ldots\infty$ — they diverge rapidly ☹.

7.23.03

Nips MATH → SN

$\boxed{\text{Gramm} \longleftrightarrow \text{3·Input Umc}}$ equivce

Proof is: .12, .26 | Also see 271.21–.24

00 : : $\boxed{\text{SN}}$ If, say ordinary arith. were inconsi: this would have no serious effect on R W. T. inconsistency would be obscure & would be very rarely relevant. We would still use Math in t. usual ways, because we know that t. inconsistency is of "depth 1000", say. But in general, Math is just a formalism that happens to correlate well w. events in R W. If Math were inconsi: — we would still know how to use it to successfully predict events in R W.

.08 : 223.40 More details on t. "Many ϕ_j model" Each ϕ_j is a Baysian hypoth. ~~It~~ each may produce no output for certain Q's. The predictions for a given Q_i are renormed to take into account t. ϕ's that are "silent" for that Q_i.

0 Maybe diffnt ~~from~~ .08; Baysian hypoths have each a pri, and a pc of data. T. ϕ's on t. other hand, give pc=1 to each of its outputs if that hypoth is true.

.12 O.k. 271.21–.24 is exactly correct: From input #1 of 3IU & each R input, define a d funct on t. Q input. For a given R, there may be no output for certain Q's — furthermore, since t. functs are partial rec. we can be uncertain as to whether there is a legal output or not.

For each Q there can be a diffnt prefix set for its R's that give output.

.18 If, for a given R, t. machine asks for another "R bit", then "that R" has no output",

.19 In lines .12–.19, for each #1 input to 3IU, we get this set of functions

.20 on Q: One function for every value of R .... T. set of R values for which t. functs are defined do not constitute a prefix set. This set of R's is all finite binary strings. Various subsets of course, do constitute "prefix sets". For a given Q, t. set of R's that give legal outputs, form a prefix set. → 275.22 →

22 From .12–.22 It seems clear that any 3IU s. function can map to a set of "partial" d-functs (that have outputs for some Q's but often not for others).

26 → To go the other way: It would seem obviously possl. — The model of (.08–.10) with each "Bayesian" hypoth only having pc of 1 or ϕ for each possl. output. Prob 1 as a "d" function & having certain output, or occasionally no output. Since this model does define a s-funct & 3IU can define any s-functs then clearly

0 this model is simulatable by 3IU.

31 But there is another method of simulation that's of interest, because its more "direct".

Suppose have ϕ_j s.a. w. pc's a_j resp. (a_j are not usually int. powers of 2 — tho we can fix it so they are), by expressing a a_j as t. sum of int. powers of 2, we break up t. associated ϕ_j into a larger set of identical ϕ_j w. diffnt (power of 2) wts. So for each Q we have a set of pc's that are powers of 2; for all diffnt outputs for that Q $\Sigma\, 2^{-(\ell_i)} \le 1$, so we can (by inverse of Kraft inequality) assign t. ℓ_i to strings so they constitute a prefix set.

or $\Sigma\, n_i\, 2^{-i} \le 1$ n_i = no of pc's w. wt. 2^{-i}.

This enables us to assign R values to each Q_i, A_i pair. → 275.00, but more details, 275.34 and 275.22–.31)

2.23.03

Frank Tipler : Physics of Immortality

NIPS

.00 (spec 272.40) : Actually, I'm not so certain about t. objectiveness of $\overset{272}{27\cancel{2}}$.36 – .40 : it may not exist at all,

.01 & if it does exist, it means only, that ~~#~~ 3IU → Gram isn't exact inverse (usually) ~~€~~ of Gramm → 3IU

Apart from t. doubts of 272.36 – 271.01 ; It looks like AZ & OOPS are both (potentially) universal s. functs.

To use them as s functs may not be so feasible for t. QA problem:

.05 We have to get a ~~#~~ single d funct that gets all of t. A's "correct" . . . . which is → a BUG, since t. same Q_i can have several difrnt A_i's.

This ↑ seems to invalidate ~~t.~~ t. equivc. of "Gramm" to 3IU. Maybe related

10 to t. "diffy" of 272.36 – 273.01 in Grammars

11 ~~[illegible]~~ .06 is wrong. For each ~~Q~~ A_i, we ~~=~~ sometimes are lucky & usually (but not always) have t. same ϕ_j. In general (t. Best (hyest pc) ϕ_j will give a hy A_i for most Q's: but for some Q's it will give t. "wrong" answer or simply have no output at all (i.e. stop before output or loop w.o. output)

For each Q, we want at least one of t. present array of ϕ_j's to have t. "rite" answer.

[NB: 272.09 – .16

So we want a set of ϕ_j's such that .... (See 271.00 – .15 for how it's done) ~~[illegible]~~ T. ~~[illegible]~~ hyest pc ϕ_j gets most of t. A's rite, T. next hyest pc ϕ_j gets most of t. rest, rite, ~~[illegible]~~ etc., etc.

20 T. forgo is Mindful of $\boxed{272.09 - .16}$ d. functs

21

.22 : 272.16 → In 272.09 – .16 I have ll ~~codes~~ of t. corpus — ($Q_i \to A_i$ functs); ~~I [illegible]~~ Initially, these ϕ_j are informationally indep. — they each code a sequence, but do not share codes. We then decide to code the set ~~of~~ ϕ_j's as an s-grammar, to minz. t. entire cost of them, so ~~they~~ this set of ϕ_j's can be used for predn., as 271.06 – .20. T. present idea tells me just what I want to optimize — just what I want short codes for — because I have a top Goal for t. entire procedure .... how to get max pc of t. QA corpus = $\boxed{\sum_j a_0^j \prod_{i=1}^{n} O^j(A_i|Q_i)}$

30 It looks like Gramm & $\overset{}{\underset{3IU}{3I\cancel{U}}}$ are interconvertable, but I'm still not sure!

Another diffy: in ~~3IU~~, if an output has a pc of >.5 & close to 1, say, t only way it can do this, is to have many codes. (to get as close as $1-(\frac{1}{2})^k$, one needs at least k codes.

It could have many pc's close to 1 — say essentially d-predn.

→ I had better write up t. proofs before I forget them! — Also tell how to deal w. various apparent Anomalies.

: First note: In "proof" of 271.21 – .24 : For each possl. value of k: Certain Q_i's will have output; others will have no output. Each Q can have a difrnt {particular set of k's}

(~~This has to be true! Say t. rite A for Q_i has pc=.5~~

274.08

7/23/03
NIPS

$\boxed{\exists\,\text{IU}\cdot 29}$ defn ≡ ∃ input UMC : (s-function)

.00 : ~~271~~.40 : Re: ANL : what I've been trying to implement: "A method of ANL in which I make trials. ~~[illegible]~~ Some trials will be of ~~some~~ some value in coding.
I want to be able to use these ~~[illegible]~~ "somewhat useful" objects to
.03 obtain new trials of hyper expected yields. (.09) ↙ seems to be of value here: it is a more formalized version of .00ff
e.g. "Push, push x_{op}" is of some value, so I should keep it in memory;
just how to use it to help create new cands, is unclear (≡ the problem of $\underset{\sim\sim}{S_{0}56}$ ☺).

.09 (.03)↗ I may want to do several ‖ codings of the corpus: Each one uses different trials — has different successes, has different ~~[illegible]~~ Definitions. The ~~[illegible]~~ defns have pc's that are peculiar ~~[illegible]~~ to each ‖ code. If we find a conc. that seems v.g in ~~one~~ ‖ codes we will usually try it in ~~[illegible]~~ the other ‖ codes as well.
Basically, what I'm aiming for in .09 ff ~~[illegible]~~ is to find different ways to devise different kinds of Regularities: (the treatment of ‖ ϕ_j codes at 272.24 – 273.2) (273.11 – .21 in particular) is very relevant here

→ See $\boxed{269.23-.40}$

Also note 275.22

.16

.17 HA! I had forgotten how I got into the problem of 270.08 (the universality of the parallel s. functs).... It was from 269.23: I had an s-grammar giving P.d. on strings that were
.19 d. functions (Q→A). So this would answer the Q of how I got the PC's of the set of d functs

.20 This leads to the (impt. interesting) Q of $\boxed{\text{are the } \overset{\text{oops}}{\text{(AZ)}} \text{ systems universal?}}$ (say they have Turing – complete set of insts

Also (the unclear) Q of "universal in a useful way".

.24 Anyway, the "BIG Break thru" is 271.21 – .24: That any ~~[illegible]~~ P.D. that gives > 0 pc to every poss. d. funct (Q→A) must be a universal s funct in the sense that its output pc is ~~only~~ no more than a constant less than the pc of any finitely describable s funct.

→ It would be good if I had easy ways to switch betw. these 2 ways to represent s-functs

Def .29 ⇒ Say I have an s-lang for d-funcs: say it is CFG. To get it into ∃ input UMP (∃ IU) form

Since I have a P.d. on a set of objects, it is possible to
.30 derive a set of R string assignments to implement that P.D. (Willis did it for a finite set of d-funcs. Elias, Gacs & Leung (1978) did it for a countable no. of d-functs. [last comma] I think
So, the Grammar defn plus a "free" program to xfm pc's into pgm. lengths, gives the desired ∃IU. So the 2 methods of defining the final s-funct are exactly
.35 equivalent in power of defn. ~~[illegible]~~ This is a very formal way of doing it — may not be practically achievable, hvr.

.36 ,,,, To go from 3IU to s-lang over d functs: (271.21 – .24) does it, but not exactly.
For each R value, we get a d funct over all Q inputs. But In general, R_1 and R_2 may give same
.39 outputs for Q_1, but usually they will not give same outputs for some ~~[illegible]~~ other Q_2..
This is different from ~~[illegible]~~ Grammar → 3IU of .30 – .35 → 273.00 spec.

7.22.03

Nips

50: 270.40 ; Say I had a large finite set of k (cardinality) d-functs, & a $[Q_i A_i]$ corpus.

Say, for each of t. $Q_i A_i$, at least one function would do it. — So if $[\phi_j]$ are t. set of functions, we can represent t. set $[Q_i, A_i]$ $i=1 \mid n$, by a set of n ϕ_j functs that do t. correct mapping. In some cases, > one ϕ_j will work so we have pairs or triplets (or whatever) for several $[Q_i, A_i]$

.08 If we only have one ϕ_j for each Q,A pair, we have a Ramsey of n symbols, & we can assign pc's for t. ϕ_j by Lap's rule (or ~~based~~ Laps' rule).

0 ← If we have >1 ϕ_j ~~for~~ for some QA's, we can ~~adjust~~ make t pc of t. relevant QA's be sum of t. pc's of t. ϕ_j's that work. We can then assign pc's to t. ϕ_j's so that pc of. corpus is max.. (I dont know if this problem

.15 — is hard or easy.) —— That this is correct is evidenced by t. proof of (.21 – .24)

If $k > n$, then it would seem that we are not doing much induction.

If $k \ll n$ then we might be doing good induction. We can compute entropy of corpus. It is minimal when we have most wts on ~~[illegible]~~ only a few Q_j's.

.20

.21 A possi. way t. system works: Consider t. 3 input umc model of a undl. pd. {3 IU} We can make a set of ϕ_j's from it! For each value of R (for all $[Q_i A_i]$ taken together, t. same R value.), we have a unique ϕ_j. The wt. is $2^{-|R|}$.

.24 → That's it! T. 2 systems are identical! GREAT! It simply proves that t. 2 systems are Equiv. → see 272.24 for a useful interpretation.

Hvr, for t. 3 input Umc model, we use a single ϕ_j that we d c u b corresponding to $R = \Lambda$ (null). Each value of R enables t. dern of a new ϕ_j, and $|R|$ is it's bcost, over that of "$R=\Lambda$". (This bcost is t. first input to t. ~~—~~ 3 input UMC.)

So t. set of ϕ_j's that simulate t. 3 input umc have relatively small total costs.

.30

Easy to go from 3 input umc to set of $[\phi_j]$'s: But can we go the other way?

Say we have ~~a~~ set of 3 ϕ_j's push, push $\begin{pmatrix}+\\-\\\times\end{pmatrix}$ op. The bcost of coding ~~[illegible]~~ all 3 is not so big! push, push is common, then t. 3 diffrnt ops. This total bcost is needed for t. corpus code of .00 – .20.. If each of t. Q_j's is improved — say by "If t_3 then push push x op". — This has a hy pc of working (≡ Getting rite Answer)

A Big problem is usually: How to get 'by pc's of t. set of S-functs used?

This is easy to do in t. 3 input umc. One way it is done in t. 11 S-functs is by 272.17: Use an S Grammar to generate t. strings that represent t. $[\phi_j]$.

7/22/03
NIBS

.0 This lists is quite diff. in spirit from what I remember about Searb ANL!

It may be that we wouldn't get as rapid ↑ of recost of solns (scaling) as we did w. t. old Searb. solns, s/o that "context" will be broken in a more natural way, "Modifying t. Grammar" can allow anything (universal Grammars).

Essentially, this growing S. Grammar represents t. ~~add~~ wtd. set, of stoch operators [O^j] → see 272.09-.16

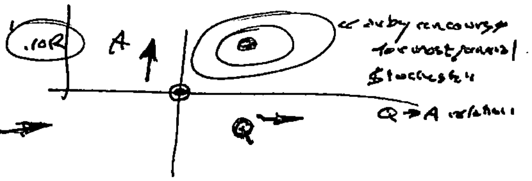
~~Any~~ T. ~~An~~ ~~alternative~~ A's's of t. grammar are d functions that map Q to A.

.08 → So we have a P.D. on functs that map Q to A. Can this be universal? — i.e.

09 Can all s functs be represented this way?

10 Genl. s funct: for each Q we can have an arb. d.f. on all A's. .10R

Look at Q & A as x & y (2 dims) ⟶

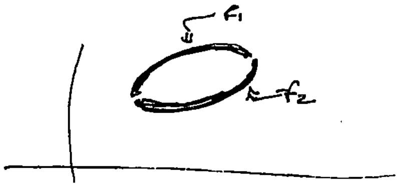


Now consider all possl. d functs relating Q to A.

Consider each contour of constant pc on .10R to be a functional relation betw. Q & A. So .10R would then express a Pd on that set of functions (≡ contours).

There is a diffy here, hvr: In the Graph of .10R, T. contours make A a 2 valued funct of Q — i.e. not really a "function".

20 More general, for a given Q_0 there may be ≫1 A that have same pc.



T. foregg. discn is about a single contour — i.e. I think many different functs betw. Q & A can have t. same probty, so both f_1 & f_2 could have same pc.

So maybe all stoch functs can be represented that way:

26 — But (even if it were universal) is it a good way? Actually it is unclear that t. "older way" represented by .10R, is particularly good!

Perhaps Try using it for some easy/hard Corpi & see how well it works w.r.t. this Question.

30 Another BIG freedom I haven't used in t. P.d. on d. functs betw. Q & A, is addition of pcs by being on >1 d function! — (This may not be so useful from pt. of view of L srch)!

Perhaps .26 is t. main Q: Are either methods Good for t. kinds of probs I'll want to be solving?

→ I have worked on (.09 .08) (t. Universality Q) some time ago. (at ASTIA perhaps) or even before.)

One problem is: if I did have a d.f. over all d functs: what would it cost (recost)?

271.00

Nips

20 : Note "t. Stack" (in Sean's ANN) as part of "Context": It summarizes impt parts of recent history

~~SN (I very probly wrote this b4 (w/o recordg it) in QATM)~~

03 : 268.32 ; Unclear as to how serious this is: While "update" is ~~not~~ not exactly a QA induction problem, it can use cones used by QA. ~~[illegible]~~

35 .... [Formally, any sequential predn. problem can be regarded as a single QA,
.06 but that t. A's possl. need not be finite: i.e. t. machine doesn't have to "stop" (ever)
.07 after printing t. part of A to be extrapolated.

It would seem, that .05 does tie sequential predn into QATM in a
10 useful way. It tells how t. cones for QATM are lined to those of t. sequential predn. problem. In t. more general version of QATM, .06-.07 is t. way we evaluate t. individual $O^j(A_i|Q_i)$'s

So saying "Updating is just another QA problem" is a bit deceptive, but only because ~~it need~~ t. statement needs expansion to be properly understood. Anyway — for me its O.k., & I now understand some things that were probly confusing me in t. ~~[illegible]~~ past.

One of t. things is re: Jess' remark about updating being a regular problem of QATM! That this recursive idea could give it meaninglessness. BUT, it is O.k. if there are other
20 Q A's in t. corpus — T. more there is, t. less chance for "error" due to "self definition" (or "self optimization"). — In general, t. result of this self optimization will depend much on t. Apri info .... but much less so, if there are a lot of other probs in t. corpus.

23 [It is easy to get a stochastic QA from [redacted] Paul's Grammar of 6.7.90 Paul 14.03]!
We define a Q: e.g. "[illegible]" → 3, 7, $+_s$

The "machine" is t. current s. "CFG" (of 90 p14.03-.12) that generates pems. So it gives a s. lang. as output, for any input, like. [Actually, this is a peculiar "s-funct" we have an s grammar that gives pd on strings (≡ pems) → pd on accept strings ~~or~~ pems we have t. Question of 270.08-.09

After we have gotten several pems ~~[illegible]~~ in t. corpus, that solve problems (i.e. Got high pc's for "correct" A), we use these examples to modify t. grammar
30 so it gives even more pc to t. corpus. We can do this by making new (Tokens) defns, & modifying continuous params of both old & new ~~[illegible]~~ tokens

An exact understanding of this is 274.12-.22 275.22-276.05

→ At first these "successfull examples" are given new names, ~~[illegible]~~ & assoc. pc's. — ~~I~~ say $\alpha_1, \alpha_2, \alpha_3$ are 3 of these I solns, i.e. ~~[illegible]~~

α_1: push push top, α_2: push push -op, etc.

We augment Paul's Grammar by $St \xrightarrow{pc_1} \alpha_1$, $\xrightarrow{pc_2} \alpha_2$, $\xrightarrow{pc_3} \alpha_3$, since this gives each example in corpus a pc of $\frac{1}{3}$, (which is >> t. amt of info in "3" & "7" (3 & 7 could be 16 bit random nos (or 8 bit — for speed)).

~~So~~ As t. corpus ↑ (& as our CB ↑) we continue to modify t. grammar to get higher pc for t. corpus.

7/20/03

Nips

.00 : I've been running thru this a bit too rapidly, missing impt. points.

1) Num is not a terminal , so is Cond → Num, $+_s$, $-_s$, $\times_s$, # ← cond.

presumably one could have a operator that recognizes that some thing is a number.

2) we can have difnt parts of memory for strings, conds, ops, etc : This would be rc of storage.

3) 66.04 discusses "The final ANL" form

ibid 67.25 is a difnt formulation. (73.20

4) 4.36 ff in ANL (08/84)

SN Actually putting every problem soln found in "memory" (≡ strings) w. equal wt. does not seem like a good idea! Perhaps (at first) just solve a lot of problems, w.o. using "calls". Then look for regys in solns. T. regys will (at first), be t. solns to t. 3 arg. operators, w.o. "conditions". (like "push, push top")

So we solve all these problems, & in doing so, we have all those little "pgm fragments" that solve t. problems. At first, we don't know which fragment to use in a problem but we do have a proby distribn. over fragments. (say 4 fragments).

T. "sub grammar" has as push, push is one word & of $\begin{pmatrix} +op \\ -op \\ \times op \end{pmatrix}$ are t. other 3 "words" "fragments".

.00 Using these 4 words (& a pc. on them) plus t. other possl. words, TM discovers t. longer fragments that have hyer pc. like:

push, push if $+_s$ then +op else $\begin{pmatrix} -op \\ \times op \end{pmatrix}$ ← 2 pgms. That work $\frac{2}{3}$ of time

So → gets defined.

push, push if $+_s$ then +op else if $-_s$ then −op else ×op. ← always works

.27

28 NB In induction, t. functional forms can be (& usually are) order indep.

So funct is of form $\prod_i O^j(A_i|Q_i)$.

.00 But approxns to O^j can be & are dependant on order in which data is processed.

((So t. update problem is then significantly difrnt from t. "first order" ("external") problems that TM works on! → 269.00 (so not such a big deal!) ⇒!))

32

SN An alternate Approach to t. problem (in Lsrch) of many good cands having about same pc ("of being best") & being hyly correlated.

If we use for our "guiding" P.D., "t. pc that t. G of a cand will be > G0", this will be o.k. — but then t. pc's are not normzd in a useful way (i.e. Σ pc. need not be bounded), so we don't get a useful CJS (which is t. main Goal of Lsrch!)

7/19/03

NLB

: On looking thru SHARB notes for work on ANL: details of implementation & CJS's of solns.

1) 90P 96.30 Orgn. w. Peter Borgmann.
" " 98.10 " " "

(8/5/90) → 90P 96.25 Orgn w. W. : Answer write up ANL w. CJS's, ~~[illegible]~~ list of difficulties, expected directions. So maybe ANL ufinh was borderline!

2) OSA notes. 90P 63.33 & ibid: 103.19ff, 111.21ff

3) AH! 90P 14.03 : ~~[illegible]~~ W's Grammar
Also ibid 44.25 (?)

Some mentions of "rcosts" ibid. 11.25, 11.20 | ~~[illegible]~~ 14.20, .25 80 : 80 | 15.30 (1280) | 16.09 (320) | 17.26 (320)

21.29	50.625k	(or only ~~[illegible]~~ 47.25k ~~[illegible]~~
.25	79.608k	
.05	3.136k	

T. Prog. is a 17 symbol Grammar. (ibid 14.12) The it has "Call", its not clear how subtrns are (defined / called). How it stores successful pgms & then accesses them later.

T. Grammar ibid (90P) 14.03—.12 seems ok. (but no ÷ — I don't see why: simply because ÷ϕ gives trouble, is not enuf! It could just cause "end of trial".

Anyway, T. first problem tsq is "~~[illegible]~~ f nu. "17" say:

tr. soln. is "If num then push" 3 symbols.
1 2 3

~~[illegible]~~ rc's = 4, 5, 4 : ~~[illegible]~~ rc = 80

This become subr #1

To do addition first st → st ; st. (rc=4) first st → call 1 (rc=4, rc=1) } 4×4×80 = 1280 = rc for addition.

st → $\underbrace{\text{If}\ \overset{5}{+_s}\ \text{then}}_{4}\ \overset{4}{+_{op}}$ so 80

~~Same rc for subtracting mult.~~

rc for subtraction is mult by 2 because 2 possl calls.
" mult " " " 3 3 " "

So 1280 × 2 subtraction
1280 × 3 multiplication

Notation is Rev. Polish. So say it got 7, 3, + as input:

It ~~[illegible]~~ pushes 7, push 3, $+_{op}$.

~~[illegible]~~ So total rc for add would be $\overset{\text{call}(\#1)}{4}$, ~~[illegible]~~4, 80 = 1280

for subtraction its 1280 × 2 × 2 (2 calls.)
for mult its 1280 × 3 × 3

or, we may have (push, push) as a single concept. Much simpler!

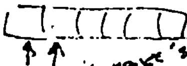
3, 4, 5, add_5, mult_5. I don't remember how I got t. "general soln"."

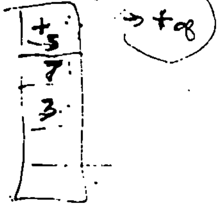
Perhaps main idea: "If there are k consecutive nos. on t. stack & a k input function is symbolized, then create associated operator."

6.7.90 Paul   This is a dup of a dup ! of a dup   ⊭ 5

see NIPS 266   DUP   14

7/19/02 for refs





dynamics associates

Paul's Grammar w. defns:

"st = ? statement"? yes: it would just go to make 'st' key

Maybe unnecy – consider ⟹ st transfer (mod)

This enables a statement to consist of several statements – Ignore a multi-line prgm.

103 st ⟶ If cond then ~~op~~ op else.

⟶ call (number of ~~given~~ defns)

⟶ st ; st   Is this? This?

⟶ do st untill cond {enddo}

~~⟶ [see 17.21 for loop inst.]~~ ⟶ ~~loop : st : If cond endloop~~ end

$\frac{1}{4}$ $\frac{1}{6}$ $\frac{1}{4}$

.08 cond ⟹ Num ; $+_s$; $-_s$; X_s | # :

.09 op ⟹ Push | $+_{op}$, $-_{op}$, X_{op},

.10 Num ⟶ $\{0, 1\}^*$ finite (binary string.)

is ; a concat symbol? .... prob: Then I don't think it needs to be an of "17" : its "16"

$[0, 1]^*$ 15 (untill is unnecy).

12 Alphabet = Σ = Σ (If, ~~then~~, ;), do, ~~until~~ enddo, Num, $+_s$, $-_s$, X_s, #

No! (If, then, else) is one word!

Push, $+_{op}$, $-_{op}$, X_{op}, 0, 1)   17 symbols.

Looks like 14 symbols.

[IF, call, DO, NUM, $+_s$, $-_s$, X_s, #, Push, $+_{op}$, $-_{op}$, X_{op}, 0, 1]   5   15 symbols

Perhaps don't need so its "14".

(No division!) ← Actually no trouble adding division: ÷ φ is no problem.

first try sep. soln. :

0 If Num then push. (pc = $\frac{1}{4 \cdot 5 \cdot 4}$ = 80)   from input list

This prgm ends up w. number on to tos. which is what we wanted.

"If" examines input list;

"Push" ~~then~~ then puts. object on input list at pointer position, onto stack, increments pointer index.

If num then push : "Num" is not a terminal. ∴ illegal. Σ ?

But still, let's examine ~~this~~ i.e. of Grammar :

So "Num" is not a terminal, but a language – (≡ set of strings).

5 If $+_s$ then $+_{op}$ pc = ~~96~~ 80   (~~96~~)⁻¹

$\frac{1}{4}$ $\frac{1}{6}$ $\frac{1}{4}$ = ~~96~~⁻¹ 80⁻¹

≡ α_+

Def If we want our grammar to take defns : we have ~~a~~ special symbols ~~for~~ Between (bd) defns. & at the end of all defns., ~~the end of all defs is " ed "~~

~~between defs ; " bd ".~~

If a grammar ~~[illegible]~~ begins w. "ed", Then there are no defns.

~~If After the last defn, we don't use "bd", we use "ed".~~

Grammar ⟶ ~~[illegible]~~ Def, statement

Def ~~⟹ # [illegible] statement [illegible]~~

⟶ statement ~~Def~~   st bd Def

⟶ bd

bd

$(st\ bd)^*$

~~statement~~

7·19·03

Nigs

: Probly best way to do t. conc. net ε language ε vocab ε insts:
write t. TSQ: Then make conc. net.! But conc net is, at first, rather vague:
It is expressive of a vague human mind.

At this point, we can take several possl. paths!

1) Make conc net less vague

2) " " " actually not vague, but exact.

3) If I try to devise lang that has enuf "looseness" in it to implement t. Conc. net as **S-functions!** Then as time goes on, new concs are discovered ε many of t. S functs become closer to d-functs! This looks like a quite difnt approach from what I've had in mind in t. past! — In t. past, I worked on t. Conc. net to "make it exact" ε then devised a lang/inst. set to realize it.

These are 2 (seemingly) essentially difnt approaches to Lrng Algebra! One purely/deterministic, certain, exact — the other very uncertain, fuzzy, probabilistically.

Either way is possl. perhaps my BoR of them!

In t. deterministic path, I eventually have to have it learn to work problems w. S-functs as solns. (So TM would get t. pt. where it would decide which kinds of problems were likely to have d-solns, v.s. which needed S-solns

Also, Remember ~~Making~~ Making / large Conc. net ε making TSQ's to go from one pt. to another (no need starting out "Bottom").
for each pt in t. net, write several solns. — see which are most easily implemented.

N.ps

spac 26340

1) Perhaps Make Comments on OOPS: ⓐ how "factor of 2000" is a funct of "73".

ⓑf How OOPS can't do probilistic induction, but can be easily modified to do it.

→ ā why we don't it to do it

ⓒg How the system can do ~~Boom~~ induction — (its doing it now for the a priori)

ⓗ ⓓ "Boost" as a kind of Mutation

→ ⓔⓘ "Boost" seems different from AZ141. Just how is it different? can we fix it? — in General ~~form~~ is it "Good enuf" i.e. is it universal? This may be a very old Q that I successfully answered (a few times!). we must invoke the a priori value for p when sample size is ϕ — so "modified Laplace's rule" ← what's this talking about?

PC

12 Main problems of TM:

1) ~~Good set of instr~~ Good initial language (good set of concs)

a) Good set of insts, concs.

b) Good form of Lang (Forth?)

2) Good way to ~~E~~ represent s-functs & P.d. on S funcs.

a) I have a 2 input Unc.

b) Modify (Forth (≈ OOPS) to get this.

c) ~~How to~~ to Do AZ-type (Born) pc's. — How to do this on second level? I. e. P.D. over AZ-type langs.

3) Early Search routines:

a) Lsrch (See ref. on ~~[illegible]~~ N.ps 133 for a way to do search over ~~[illegible]~~ TSQ for Stmrs.)

2.6 b) On "Continuation" v.s. "Backtrack": ~~how much~~ when & how much to Backtrack.

→ Design of Langs etc. so that 2.6 is feasible.

4) Design of TSQ:

a) Simple Algebra: Chose lang (1) & Design search routines (3) so that these probs

10 can be solved.

M/ ~~[illegible]~~

5) Context: Used to modify pc's of tokens.

I think it ~~would~~ would be desirable to do (4, 1, 3) together: Interactively. As it is, I ~~think~~ think I can devise a concept ~~net~~ for 1 var. Algebra. (Also ~~some~~ w. some elements of large CJs).

7.15.03

Nips

(.17) Math : P ≠ NP : undecidable

GOOD!

5c [SN] On "Undefining" in view of more data. Say I define a ngm early in life. Later, t. corpus is ∋ t. defn. ~~would~~ is no longer a compressor. Hvr., this defn has been used as components of other definitions, so undefining it would undefine several other defns that have, in view of t. larger (subsequent) ~~corp~~ corpus be useful compressors. It would be a BIG BACKTRACK

Well, t. thing to do is to retain all defns, but ~~u~~ update their pc's. There is a "chain rule" involved since "higher order" defns have pc's that are dependant on t. pc's of ~~[illegible]~~ ~~earlier defined~~ ~~[illegible]~~ ngms.

.10 Updating pc's involves retaining all ~~old ones~~ defns but updating their pc's in t. temporal order in which t. ~~[illegible]~~ ngms were defined. So one can do this for t. entire set of defns every once in a while: Th. top level defns pc are updated more frequently

.11 Updating in OOPS: ~~[illegible]~~ Does it consist only of freezing old solns, & "acknowledgement" of fns that have been defined in soln of last problem?

→ It's really not clear in my mind as to just what old solns. are accessable (as frozen nts) to a new problem-to-be-solved.

17 [SN] [Math:] It could very well be that t. P v.s. NP problem is undecidable. It does involve behavior at n=∞ only. i.e. we are concerned w. behavior at arbly large values of n, only. Whether a particular problem is P or not could be undecidable. Seems related to Chaitin's work on t. unsolvability of certain probs in Number Theory.

If ~~[illegible]~~ P=NP (all probs were poly) then t. Q. of whether an arby prob was P or NP could be solved! (i.e. it's always P). So if it's impossl. to tell if certain probs are P or NP, then "P" can't be true.

20 Updating OOPS seems very easy (trivial!). Why is Alpha update so hard? It's a matter of ~~definition~~ terminology! in Alpha, we spend much time on update, but prediction is ~~[illegible]~~ always very fast. In OOPS, we spend much time on predn. & there is ≈ ∅ update. The sum of t. cost of update + predn. is what is of interest & they are both big in Alpha & OOPS.

Alpha ≡ (Algorithmic Probability Heuristic Algorithm) ☺.

27 : [262.12] → Oneway! Say we have a bunch of density vectors that are all desireable (i.e. ~~[illegible]~~ they are assoc w. ~~[illegible]~~ problems that are "close" to t. prob. we want to solve). We may simply want to "OR" them together. If there is a subset of these vectors that are close together — then OR them. ~~As~~ If we can

30 partition the vector space into ~~[illegible]~~ CLUSTERS — do that and (OR each ~~[illegible]~~ or t. "center" of each) Cluster as a "boosting" vector. (Remember these vectors are BAG representations, so we have to be careful in OR'ing.)

7/13/03

Nips

: Report Revision:

1) Look at ~~my~~ letters (from) J: —(7.1.03) | After lifting Go thru J's comments.

Several impt items Also other Letters. | ~~(5) see 7/2/03: "things to do" lines 22 ff~~

2) Table of contents:

3) How to learn from failure.

4) How to design O^j ~~m~~ functions.
2 ways: ① 3 input univ | 272.24 & forg. discussn.
② Grammar | to 276.09

5) I said the lang used by OOPS is a kind of stack (not stochastic!) ~~stack~~ lang. — He wants me to emphasize

"the lang used by the current OOPS pilot implementation is a kind of stack lang." | I don't really see this!
→ But in fact, I can only make comments on the Paper as is, not on ~~uncertain future~~ expected future developments

6) Use of ~~[illegible]~~ WON instead of elaborate Lsych.

⑦ A section on what has been worked out & what needs to be worked out | This is closely related to RV: 277.03 – .24
→ 265.12 ff
is V.Good
Essentially RV is ⑦ + ⑧

⑧ A new introduction ~~as~~ a/o Abstract: Explaining what Mech Lrng ~~prob~~ problem was: What were major sub-problems, What solns I'm using & why I think they are V.G. | See 277.03 – .24 + 253.27 – .40

9) Have more detailed discussn of OOPS. in § 6
e.g. $\left(\frac{n_a}{(1.\cdots)}\right)^4 \cdot \left(1+\frac{1}{n_a}\right)^3$ effect.

Also give some of the criticisms that I wrote J. about in my letters to him.

10) Also Note OOPS doesn't solve Sfunct / Predn problems — only d funct

11) Main differences betw Phase 1 & OOPS.

~~[illegible] 7/2/03 [illegible]~~

a) Lispish v.s. Forthish langs.

b) Ph. 1 tries to do / univ. S functs. ; OOPS does only d-functs

impt in philosophy of research → c) Ph 1 trys a tsq of easish problems. OOPS starts on hard probs. This may be for political, rather than Scientific Reasons.

d) Ph ① uses Genzd context to deal w/ scaling : How OOPS deals w. ~~aspect of~~ scaling is unclear | →265.00

e) OOPS calculates total pc's . . . How to [illegible]

7/13/03

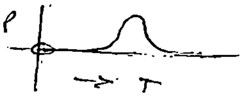
N.B.

00:261.40: Actually 261.36 differs much from what OOPS did, — what boostq did. Boostq picks a single problem at a time (plus/uniform background density) — If one could use 2 or more "Boostq's", t. result would be of much lower a prio.

The previous discussn (see 261.18 on "Dot products") reveals a serious diffy in ~Lsrch: From t. dot products, we will get a lot of ~~[illegible]~~ density vectors that are promising, but similar to each other. I would like to use them in ||, by adding together t. pc's of various tokens occuring (jointly). — t. "WON" approach. Unfortly Lsrch doesn't do parallel addition. Instead, one way is to choose t. single best looking density vector. If it doesn't work after a while, choose another ~~[illegible]~~ hy pc density vector that is not too hyly correlated w. t. first, continue trials for a while. Of unsuccessful try density vector ~~[illegible]~~ relatively uncorrelated w. earlier trials, etc. → 264.27

This problem was dealt w. in ~~ordinary~~ ordinary Lsrch by using WON as || updating system. So we chose "Best looking" PST. As we work on it unsuccessfully its value as "Best" ~~best~~" decays: As does other PST's that are "hyly correlated" to it. When These ~~[illegible]~~ ranks get to be no longer #1, we jump to t. new #1 ... which will be a PST that will tend to be dissimilar from first trial (because PST's that were ~ to t. first trial will have their "probtys of being best" ↓).

In ~~a~~ Won system, the pobty of being best will be $\frac{(0^{th}\text{ moment})^2}{1^{st}\text{ moment}}$

The moments being of t. 1P  function for t. (PST, problem) pair.

26:258.28 Note on $\left(\frac{k}{11}\right)^4$ formula: ~~t.~~ value of pc/ of soln of T. of Hanoi for search

W.o. ~~any~~ preprocessing:

$$\left(\frac{1+k}{7\cdot k}\right)^3\left(\frac{1}{7\cdot k}\right)^4 = \frac{1}{7}^3\cdot\left(1+\frac{1}{k}\right)^3\cdot\frac{1}{7^4}\cdot\frac{1}{k^4} \approx 7^{-7}\cdot k^{-4}$$

$$= \frac{1}{7^7}\cdot\frac{1}{k^4}\cdot\left(1+\frac{1}{k}\right)^3$$

After trng

$$\left(\frac{1+k}{7\cdot k}\right)^3\left(\frac{1+k}{12\cdot k}\right)^7 = \frac{1}{7^3}\cdot\frac{1}{12^7}\cdot\left(1+\frac{1}{k}\right)^{10}$$

$7^3\cdot 12^7 = 1.229\times 10^{10}$

so

$$\frac{\text{w. trng}}{\text{w.o. trng}} = \left(1+\frac{1}{k}\right)^7\frac{7^4}{12^7}\cdot k^4 = \left(1+\frac{1}{k}\right)^7\left(\frac{k}{11.053}\right)^4$$

So, pc of soln. after Boost is ≈ indep of n_Φ,

But pc of soln w.o. boost is $\propto \left(\frac{n_\Phi}{11}\right)^4$

7/13/03

Nips ABCDEFG—#9
ABCDEFG—#8

0 problems may not occur in an ordering so that "recency" can be used. T. only justifn. of using "recency" is that it may help TM "Get off t. End" much faster (easier for trainer).

In t. work since ~~the~~ start of IDSIA, I've been thinking of OATM in a ~~Very~~ General way, — not thinking of TSQ's much (if at all). If might be that if I had a conc. net, ā a ~~[illegible]~~ TSQ, that I could train TM using a rather simple TSQ such ppm — Then find suitable "contexts" to speed it up.

Wrt. Context for oops ā t. "density vectors" of 259.12 : If a new problem is ~~[illegible]~~ judged to be "similar" to an older solved problem, then this ~~could be~~ "similarity equivalence set" (fuzzy set) can be a kind of context for use of t. density vector of t. older "solved" problems.

So we have to have a way to "decide how "similar" 2 problems are — to warrant using ~~#~~ "density vectors" (≡ "Genzd boosts")

In OOPS (Initial version) boost has no "context" ā prob'y of use of boosts is indp. of t. nature of t. problem being solved.

18 To study "closeness" of problem pairs; look at t. "inner dot product" of ~~[illegible]~~ their
19 density vectors — "how many" insts they have in common. If a ~~[illegible]~~ new
0 problem P_n is "similar" to P_1, P_2 ā P_7, then we might try t. ~~sum~~ wtd mean of
.21 t. density vectors of $P_{1,2,7}$ ~~⊕~~ for solving P_n. In addition to these 3 density
vectors, there is also some "similarity" to t. density vector consisting of all tokens ~~[illegible]~~ to date.

.23 Also a density vector assoc. w. t. initial according to (tabula rasa) inst. set of

From t. studies of .18 –.19 we will get distances betw. various problems. Then we will try to find characteristics of problems that are able to predict which probs are close to which. This is actually a "normal" inductive inference problem — but dif'lt because I don't have any Good ideas on how to solve it ....

0 Tho I do know t. corpus (It is t. set of ~~[illegible]~~ pairs of solvd probs ā t. dot products of their density vectors,

So if d_{ij} is t distance ~~from prob~~ betw 2 problems, ā $\vec{V}_i$ is t. density vector of problem i Then for new problem, j Try density vector

36 $\sum_i d_{ij} \vec{V}_i$. The d_{ij} may have to modified, normzd, or whatever

Also add in t. "background" density vectors of .21 –.23

Nips

259.02 Spec

00.259.40 → on "context") say wais hourly the O^j's of lig. [redacted] $G = \gamma = \sum_0^j \prod_{i=1}^{n} O^j(A_i | Q_i)$

N.B. — t. ~~data~~ QA data is "unordered" only in t

Sense that t. models used are of t. form 260.00 R (i.e. product of pc's).

In general, our Ing Algm will be sequencial & will depend much on t. order in which t. QA's are presented to TM.

Say we have some O^j's for i = 1 to n (.00 R)? We can use some regy observed in these O^j to try to Construct O^j's that will work for i = 1 to n+1.

Juergen's Experience w. OOPS is perhaps instructive. He tried 2 strategies in ll to do i = 1 ··· n+1 : t. first was by concating onto t. O^j for i = 1 ·· n ; ~~[illegible]~~

0 t. second was to start at t. begining & try t. find a O^j for i = 1 ··· n+1.

I thot that t. second was extreme back tracking, & that another way would be to backtrack by only one QA. : then try 2 QA's back, ect.

One could spend a certn. amnt. of time w. ~~[illegible]~~ ao success at each backtrack level, befor going back one more.

A Big Q in .00 : There are 2 goals involved in t. search for O^j : One is to find hy value of γ in .00 (say ≪.00)

.18 The other is to find hy γ in short time; or Max γ for Given time.

~~[illegible]~~ L srch (its approxnsly first GHT) deal w. .18.

I. pure induction problem of Max γ (= G) of .00 would seem to admit of no heurists —

0 T. pc's of t. various tokens don't change much in searching O^j, since ~~[illegible]~~ t. code for O^j usually isn't long enuf. (?)

So, t. use of Context can use experience of "t past" as if we were doing a sequential coding problem. Contexts useful in t. past, become useful now. Contexts are a kind of regularity & ~~[illegible]~~ defining them is logit ⇔ they ↓ code cost (~~resul~~ beast).

.26 On one hand, there are t. pc's assigned to tokens via AZ. ~~[illegible]~~ — This is t. G value of an OZ problem. T. havrs for t. OZ problem : Could they (temporarily)

.28 assign difrnt pc's to tokens than AZ, t. "True" Gore?

Simple L srch seems to ignore .26 – .28 (But note that it is by no means t. "ultimate OZ prob solver".

0 Also Note: T "Recognition functions" used in t. early QA problem in "t. Report". (I think Section 1). This is a somewhat difrnt. approach to coding. — Perhaps Similar to human Methods

In SOL 89, I was thinking in terms of learning Conc. nots — which is ok.···· ~~but t.~~ but I thot conc. nots would be adequate : They are only a first, important step. Getting pc's of concs to a reasonable level, is an essential "next step" "Context" is one great aid for this.

"Recency" could be used ~~[illegible]~~ as "a context" for early learning — but in RW,

7/12/03

Nips

.0 : 258.40 : So three problems:

1) what is exact justfn & mechanics of "Context" dependent token pc's.?

02 In particular, for various kinds ("levels") of context, just exactly what is t. corpus? → 260.00

2) In OOps: >1 Q about "Boostq", a) Boost looks like a particular kind of "context" — is it?
If so it gives a joint pd on successive new tokens — so it is diffrnt from my "context" —
Can I genz. my "context" to include this "joint P.D.")

b) A general Q. about "instructions that modify pc's of tokens". J. says that "in general, insts can compute p.c. of next token". Just how legit. is this? In t. case of boostq, it

0 seems to have some epistl. justfn., but for arby insts that change pc's of tokens . . . .
Unclear how reasonable this is.

He may have called them "patterns" patterns

12 I think that ~~J~~ J. was thinking of various ~~the~~ "density vectors" A density vector is a set of pc's for all of t. tokens. One can combine them in various ways ① linear combination ② multiply 2 vectors together (& perhaps normalize). Thus t. "Boost" density vector set was only one he included in OOps.
Presumably for various problem domains, one could have different "density vectors".
— They give a kind of apriopd for each problem domain.

0 3) There was the Q of how to generate/formalize/define a Universal P.d. over S functions: AZ does it for d functions (in a manner of speaking),

(Solution) One formalism for univl. dfn. on S functs: any 3 input UMC defines an S funct. T. first input (#1) describes t. function. T. next input (#2) ~~des~~ is for t. input; t. third (#3) is the "R" (random) input.
— As we have normal UMC machine derived prbly-types of output.
So, random input on input #1 gives a univl. df. on S functs. T. longer t. prgm, t. less pc. Each input is a self limiting input that decides when ~~it~~ that input should stop & go on to t. next input.

Yellow Paper page.

0 There is a ~~ref~~ ref. to a detaild ~~e~~ Lynch over such UMCs in t. NIPS (133.—)

Another way to describe an impl. set of S-functs, is t. Bern. formalism:
Simply describe each object & tell what its pc is. To ~~obtain~~ discover sfuncts of this kind:
One way is formal langs. Bernoulli & Finite state (~~#~~ HMM) langs, stochastic CF Grammars,
Stochastic Context sensitive Grammers .... any other kinds of stoch Grammers,
Also BBN's (~~Bayesian~~ Bayesian ~~Belief~~ Belief Nets) — (257.00)

So .20 (3) doesn't look like a big theoretical problem.

2/11/03

N.B

Gospers Summation trick. Σ is like S t. inverse of a "difference" function.

For every integration trick, there is an analogous Summation trick. e.g. "integration by parts"

10 : As is AZ is a univl/dif. over d funcs. ~~However~~ We ~~want~~ need a univl. prob ~~fd~~ Pd. over s funcs, hvr.

Seems to me that I solved this problem within last yr or 2! It was regarded as a serious theoretical problem at that time. Also much unsuccessful work before that.

.04

.05 As related to OOPS! It doesn't look ~~[illegible]~~ like his system is near universal w.r.t. modifying .Bs

(SIU) Benefit of ~~[illegible]~~ ≈1000 from use of "Boost"; how this factor depends on

$Q_n (=73)$ ∴ no of ~~[illegible]~~ tokens ~~[illegible]~~ g started with.

.30

$Q_n \equiv k$:

$$\alpha = \left(\frac{1+k}{7 \cdot k}\right)^3 \cdot \left(\frac{1+k}{12 \cdot k}\right)^7 = 9.322 \times 10^{-11} \text{ for } k=73$$

$$\beta = \left(\frac{1+k}{7 \times k}\right)^3 \left(\frac{1}{7 \cdot k}\right)^4 = 4.539 \times 10^{-14}$$

$$\text{ratio} = \frac{\alpha}{\beta} = \left(\frac{1+k}{12k}\right)^7 \cdot (7 \cdot k)^4 = \frac{(1+k)^7}{(12k)^7} \cdot 7^4 \cdot k^4$$

for $k \gg 1$ $\approx \frac{k^7}{12^7 \cdot k^7} \cdot 7^4 \cdot k^4 \approx \frac{7^4}{12^7} \cdot k^4$

$\approx \frac{k^4}{15000}$ | $11^4 \approx 15000$

$12^7 \approx 35.83M$

$7^4 \approx 2{,}401$

$\frac{12^7}{7^4} \approx \underline{15k}$

So gain = ≈ 1 for $k=11$ otherwise $= \frac{k^4}{15000}$

$= \left(\frac{k}{11}\right)^4 \cdot \approx 2000$ for $k=73$.

$\left(\frac{k}{11}\right)^4$ should be mult. by $\left(1+\frac{1}{k}\right)^3 \approx e^{\frac{3}{k}}$ } $e^{\frac{3}{k}} \approx 1 + 3/k$ $e^{\frac{3}{k}} = 1 + \frac{3}{k} + \frac{1}{2}\frac{9}{k^2} + \frac{27}{6k^3}$

so $\approx \left(\frac{k}{11}\right)^4 \cdot e^{\frac{3}{k}} \approx \left(\frac{k}{11}\right)^4 \cdot \left(1+\frac{3}{k}\right)$.

$(1+s)^n = 1 + ns + \frac{n(n-1)}{2} s^2 \ldots$ $1 + \frac{3}{k} + \frac{4.5}{k^2} + \frac{4.5}{k^3}$

More exactly $\boxed{\left(\frac{k_Q}{11.0527}\right)^4 \cdot \left(1+\frac{1}{k_Q}\right)^3}$

$\left(1+\frac{1}{k}\right)^3 = 1 + \frac{3}{k} + \frac{3}{k^2} + \frac{1}{k^3}$

28

(262.26)

$\left(1+\frac{1}{11}\right)^3 = 1.298 \sim 1.3$

10 (:05) Perhaps it would be best to think about .00 ~~[illegible]~~ before considering .05?

In .05 OOPS uses "pc modifying insts" — they modify pc's of new-tokens-to-be-created.

Boost is like my "Context" dependent pc's of newly ~~[illegible]~~ created tokens.

In my "Context", it ~~[illegible]~~ is a regular "regularity" that is true of a certain corpus [we will discuss just what this corpus is, later].

Hm. Boost is a bit different from "Context" in the usual sense, in that "Boost" fits a set of tokens to work together — to make a kind of "Anagram" (= rearrangement of letters) of a previous soln.

7.8.03
Nips

Practical aspects of WON : .00ff from(254.00ff)

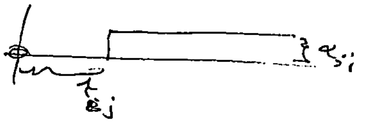
(20–.22) Discussn of Non-El problem of optimizing GPD — including inventing new PSTs

.10 : ≠254.40: Actually, this problem may be far easier than the picture of 254.00–.40!

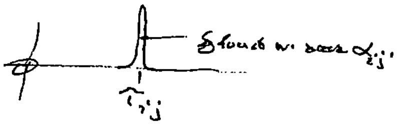
To start off, TM finds (by optzn) a funct F (~~[illegible]~~ PST_i, Prob_j) that is able to look at (PST_i, Prob_j) and obtain a $P_{ij}(T)$ for soln (for Inv. probs).

Usually (maybe always), I'm only interested in the Zeroth & first moments of $P_{ij}(T)$ —

So TM could use the standard P(T) model:



(of



)

(I'm not certain whether σ^2 of the P_{ij} dfn is needed if we want to include empirical info on failures).

14 So we end up w. $F_1(PST_i, Prob_j) = T_{ij}$; $F_2(PST_i, Prob_j) = \alpha_{ij}$

So the problem then is, w. Prob_j fixed, to find PST_i ∋ $\frac{\alpha_{ij}^2}{T_{ij}}$ is max

16 We have then new funct ~~[illegible]~~ $\tilde{F}(PST_i, Prob_j) = \frac{F_2^2}{F_1} = \frac{\alpha_{ij}^2}{T_{ij}} = \tilde{F}(i,j)$

.7 an OZ problem We want to find a i ∋ $\tilde{F}(i,j)$ is max. Depending on how smart TM is in
.9. the area at the time, i can range ~~[illegible]~~ over known i, or (if TM has suitable
21 Good Grammar for PSTs) it can try to invent a new PST that is particularly
22 Good for $Prob_j$

DOES .20–.22 constitute a non-El. soln. to GPD update problem?

Anyway say we use .00–.22 to pick the "Best" PST to try*; We work Prob_j using that PST. From the data coming out of this work (first of all, a data pt. on Failure of the PST for Prob_j — with T increasing: Second,

D doing the calculations involved during trying to solve Prob_j, Various other regys. may be noticed, resulting in modified choices of code cost for Various aspects of $\tilde{F}(i,j)$ (.14–.17), which can also change $\tilde{F}(i,j)$ for this i & other i's — so that the present PST_i is no longer "best guess" — So we switch to a new "Best" PST_i.

N.ps

"Learning English". [(Naive → Real) Physics]

I had Idea of TM learning Algebra, then learn to discuss Algebra in simplified English. To graduate to more complex & to less "simple grammar" English.

Several ways: ① Stick w. "Algebra World"

② Get TM to lrn Simple "Physics" by learning to play "Ping pong". This last can be in graduated degrees of sophistication. To start off, "balls" are points & no air resistance. Bouncing is perfect, there is no net. No "opponent" — only a "reflecting" wall. TM is given exact coords of ball at all times.

Then difficulties are slowly introduced. Air resistance, Net, balls of >0 diam, eventually, an ~~opponents~~ opponent of gradually increasing skill & speed.

Eventually, I may want ~~to~~ it to learn to play a game w. a Real "opponent" that is ~ to TM & get TM to model the opponent ~~[illegible]~~ & realize that the opponent is "~" to itself.

In general it is probly possl to teach TM to play various "games" that become more & more like R.W. — Including Entities that are like People, So it would be then useful to Get TM to understand English discussion about "T. Games". There may be actual/existing video games that TM could learn to play.

21 A Quite different ~~of~~ approach is used in Statistical Machine Xltn (which they does "Q.A.ing". To some extent "Language understanding" is key to Xlt ~~[illegible]~~ between internal & External (Nat lang). lang. (See D:\PS\ MachTranslation-statistical for many papers on this.)

7/6/03

N1B ~~Probabilistic~~ ~~Boolean~~ Bayesian Nets (Bayesian Belief Nets) BBNs

10 ! What ~~Boolean~~ Bayesian ~~[illegible]~~ nets may be!

Say we have discrete, finite alphabet of possl. variable values:

A ~~module~~ I,O, module will have as input, a set of variables. For that module, this will induce a p.d. on each of its outputs. For each input config. (n binary inputs → 2^n input configs) it will have a p.d. ~~[illegible]~~ on its set of output configs (if there are

.07 m binary outputs 2^m configs) —— So, for each of the 2^n input configs, it will

.08 have a list of 2^m pc's (that sum to 1) for each possl. output config.

Some simplifns:

0 outputs

1) only one binary output 2) a single ~~[illegible]~~ radix r output.

3) .07–.08 are a single, radix 2^m output.

4) several outputs, but p.d.'s of them are indep. (equivt. to several indep modules in ||).

inputs:

same as 1, 2, 3 for outputs:

relations of input to output: 1) Most general is in form of a list of probls.

2) I can't think of other common forms

I, O. modules can form a net by connecting ~~[illegible]~~ outputs of 1 or more

20 modules to inputs of one or more modules. This usually results in many intermediate & final values of variables being correlated.

So: This Gives a way to get fairly complex ~~[illegible]~~ [illegible] s fcnts for discrete variables.

For continuous Vars, one could also define "Modules" w. defined I-O. characteristics, but it would be more diff. (i.e. in ~~Discrete~~ Discrete case, it is known that a complete (often realizable) dcn of the problistic I/O ~~b~~behavior of a modules is ~~[illegible]~~ always possl.)

30

In the discrete case, perhaps simplified ~~[illegible]~~ modules & nets of modules are being devised so that their behaviors can be easily analysed!

In particular: How to (usefully) constraint. no. & nature of params defining a module or net, so that Learning can occur w. a reasonable ssz! For most nets, modules, data types, the problem ~~a~~ may be practically unsolvable → unreasonable cc.!

7/15/03   Practical Aspects of WON:00   254

N.ps

$M_1 = \int dt \cdot t \frac{p}{t} = T$

$M_0 = \int dt \cdot \frac{p}{t} = p$ prob of soln. ever

.00 : SN in t./WON approach to Searching for good PST's I ~~orde~~ used for INV probs: (modified)

$M_{0,1}$ = first & second moments of prob of soln. v.s. time:

.02 I think I ordered trials by $\boxed{\frac{M_1}{(M_0)^2}}$ ← smallest first. "p curves"

.03 It would seem that I could get curves w. very good (.02) but very small prob!

NO! t. p curves are obtained by selecting them for "good fit" to corpus of past problem solving trials. Mets it M_0 & M_1 are not considered in the "curve fitting" routine. After curve fitting ~~I shall~~ put them in .02 order. L

.09 Hvr, if we integrate t. 2 sub-problems of curve fitting & .02 maxim L (≡ "Non-el")

.0 we end up w. diffy of .03. A logic Q: How much time to spend on fitting p curve .02R vs. time spent searching for ~~best~~ best (.02L)

.12 Say we have a certain CB. How much time to spend opting (.02 R) v.s. opting (.02L) ...

One general type of soln. is to spend ½ of time on each part. — But actually, One of t. 2 on .02 (R or L) may take much less time than t. other!

A standard method might be to pick a PST; spend time T_1 on getting good .02R curve. Then get $\frac{M_1}{M_0^2}$ for that curve; Repeat, keeping track of best $\frac{M}{M_0^2}$ values thus far.

.0 Note that t. curves of .02R will prob. be parameterized so that $\frac{M_1}{M_2^2}$ will be very easy to compute. — so t. Q then becomes: "If we have CB_0

.21 to look for a good PST, what value to use for T_1?" "We assume T_1 is

.22 main cc involved in (.02 L, R)

So this turns out to be a ~~interesting~~ Serious problem: 257.00 -.22 may be an adequate soln

.27 One somewhat Non-el goal, is to find/invent a PST that ~~will be~~ "Best"

.28 for t. present problem. (As in ≈ .09)

In .12, we have elzd t. problem & we ask ~~how~~ about how best to do that elzn.

.30 In t. Non-El prob. of .27-.28, we may be able to proceed since we may have a large enough SSZ of PST trials

One way to deal w. t. problem: Essentially, one is given CB_0 (.21), to choose good T_1 (.22).

Note that CB_0 is for t./current problem ~~soln~~ soln, so TM must have some idea of how ~~to~~ long t. PST's will take for t. current problem.

Actually, t. "problem" is rather complex: The idea is to chose/generate a v.g. PST for t. present problem. It involves t. [redacted] generation/modifn. of PST's, ~~etc~~ etc.

257.00

7-5.03
Nips

1) Re: OOPS! If J's method of assigning pc's to pgms is really Universal, then I can use it as a way to derb. S functs. He says he can get any possl. P.D. this way, but I don't (yet) ~~see~~ How!

2) As is, I have 2 ways to derb. S functs.
  a) 3 input umc (perhaps improved by using methods of oops)
  b) Ganef Born Seq. — perhaps A2(4) of "Appendix A".
I'm not sure how "Universal" this is, but it seems to cover most common Methods of ~~Getting~~ S functs.

Starting See .30

9 3) For MIT talk: Start out by stating why I think this is a good system, i.e.

0. a) It uses Univ. D.F. (explain that "Universal" in most contexts doesn't mean Turing Universal or "Kusp" universal)
  b) " Problem w. incomputability
  c) " " w. Subjectivity of a priori.
  d) How to apply .... Lsrch. (near optimality of Lsrch | Best BLIND srch): t. GHT on optimality of $\frac{p_i}{t_i}$ ordering.
  e) Limitation to INV problems. So we can solve all inv, oz probs.
  f) How OZ probs solvd by Lsrch
  g) Updating GPD (note "C" on why this is mandatory for any Good system. ~~So we c~~ This enables us to Learn to solve INV, OZ prob.
  h) Updating as an induction prob.: update as OZ problem.
  i) WON problem: how it improves on Lsrch — How simultaneous ~~up~~ Prob. solving & update enable "Non-Blind" srch.

27 In ③ I want to List major problems of Machine Lrng. & explain how α solves them.
So BIG Q: what are "Major probs of Mach Lrng?
1) Universality 2) efficient srch. 3) Actual Lrng. 4) Updating (all systems must have updating).

0 or: Start w.:
In steady state System can solve a certain set of problems w. certain Measure of "Goodness". After solving one or more problems, t. system is "updated" so that it's future performance is improved in one or more of follg ways
  1) More kinds probs solvable
  old probs solved ② faster or ② w. better solns
} I.E. "improvement".
All ML learning systems do these things to some extent. How is our system better?
  Then. somehow bring in some of ideas of .10 ff. → See 2.77.18 ~~After~~ continuation of this.

This looks like a good way to Start an introduction to α

6/27/03

Three kinds of probabilistic Induction: Universal distributions and convergence theorems.

c : 251.40 section 4 : Incomputability of Universal Distributions.

Abstract: We will describe three kinds ~~of 3~~ probabilistic induction, ~~[illegible]~~ and give ~~Solution~~ general solutions to each kind with ~~and~~ associated convergence theorems to show they ~~will~~ tend to give good probability estimates.

The first kind ~~[illegible]~~ extrapolates ~~[illegible]~~ a sequence of strings and/or numbers.

The second ~~[illegible]~~ extrapolates an unordered set of strings and/or numbers.

The third extrapolates an unordered set of ~~[illegible]~~ ordered pairs of elements ~~[illegible]~~ that may be strings and/or numbers. Given the first part of ~~a~~ pair, to ~~get~~ a probability distribution over the corresponding

11 second part of the pair → .25

~~For each solution a corresponding~~ convergence theorem is given, showing that as sample size grows, the error in probability estimate decreases rapidly.

~~This problem these kinds of induction problems are very general and~~

17 :.29 ~~[illegible]~~ → The solutions given are very general and/ may cover ~~[illegible]~~ all kinds of induction problems. Time series prediction, Grammar discovery (for formal or natural languages), curve fitting, the Identification problem and the categorization problem, are a few of the kinds of problems amenable to the methods described. End.

~~[illegible] For each of the ~~3~~ three kinds of problems, a universal distribution is given that solves it, and a corresponding~~

29 :.11 → Each of the three kinds of problems is solved using an associated universal distribution. In each case a corresponding convergence theorem is given, ~~[illegible]~~ showing that as sample size grows, the expected error in probability

29 estimate ~~decreases~~ decreases rapidly. → .17

0

.0 : : Exports list. from ~249.12

~249.12 (1) Dimacs 1.22–.28; 1.29–.32) do as one, though 1.22–.32
DM1.tex DM2.Tex

250.19 (2) Sol 99 256.27–.37 ~~[illegible]~~ S99–1.Tex

247.30 (3) (IDSIA Report.) BSI eq(1) to 5.27 S02–1.Tex
Appendix B complete S02–2.Tex

09 : 247.40 If we set $Q_i = \Lambda$ ($i=1\ldots n$) in equ (B)

0 : It ~~[illegible]~~ becomes clear that the equation (A) for induction on unordered sets is a special case of Operator induction, and that the convergence theorem ~~[illegible]~~ (B) holds for ~~unordered~~ eq (A) as well. This also assures convergence of the Operator induction technique of section 2.1.

Is there any ~~other~~ advantage in using equ (D) rather than eq(B) for operator induction? D ← § 2.1

eq(D) exploits regularities in the set $[Q_i, A_i]_n$ ~~eq (A) exploits regularities in A_i as a function of Q_i~~. It includes regularities in the set $[A_i]$ — which we discard. So it would seem that we are doing more work than is necessary. In eq (B) we only find regularities in ~~[illegible]~~ functions relating A_i to B_i. ~~Superficially, this m~~ ~~[illegible]~~ Such regularities may be easier to find than ~~[illegible] the more complex~~ regularities in the more complex object $[Q_i, A_i]$. ~~However~~ In general, however, the convergence of either of the techniques will depend critically on just what problem is being solved

252.00

NIPS

00 :249.40 The result is that the probability errors for the normalized measure, P'_M can converge much more rapidly than those for the semimeasure, P_M.

Gacs (~~Gacs~~ ibid) also shows that the corollary corresponding to eqs 5 and 6 holds if $P_2(\cdot)$ is an unnormalized semimeasure.

Marcus Hutter (Hut) shows that these results hold if we use alphabets with ~~more than~~ any finite number of symbols.

In the forgoing convergence theorems the total squared probability difference is used as loss function. The proofs of the theorems also show ~~[illegible]~~ the same convergence for the Kullback-Lieber loss function (which is greater than or equal to the square loss function). 10

Hutter (ibid) considers more general loss functions and shows that the use of the universal distribution gives losses that converge rapidly toward the smallest values that they could have.

Section 2 Induction on Unordered Sets ~~[illegible]~~ [1]

Subsection 2.1 The problem and ~~solution~~ a Solution.

~~as~~ $\boxed{\text{Copy Sol 99 256.27 - .37}}$

0 A solution using a universal distribution is obtained by assuming that the data was generated by some unknown stochastic distribution on all possible single finite strings, $\mu(D_n)$

The Universal distribution is a weighted sum of all finitely describable ~~describable~~ probability measures and semimeasures ~~on unordered sets of strings~~ on finite strings.

$$P_M([D_n]) = \sum_j \alpha_j \prod_{n=1}^{h\ \text{which.}} P_j(D_n) \qquad (8)\ \text{eq}$$

0 h is the number of strings in the set $[D_n]$

α_j is the weight of the i^{th} probability distribution on finite strings

$\alpha_j = 2^{-|a_j|}$, where a_j is the shortest description of $P_j(\cdot)$ ~~[illegible]~~ and $|a_j|$ is the number of bits in a_j

The M index of P_M indicates that the functions P_j are to be described with ~~respect to~~ reference to machine, M. Since M is universal, it can be used to describe any describable function.

The probability assigned by μ to $[D_n]$ is

$$\mu([D_n]) = \prod^h \mu(D_n) \qquad (9)\ \text{equ}$$

→247.00

[1] F.N. section 2.1 follows the discussion of Sol 99.98 256 and 258

6/25/03

>8 To start off, we will normalize P_M to create P_M': ∫

249.1 – .15 P_M' . . .

~~The particular method of normalization~~ ~~[illegible]~~ It is not difficult to show that this method of normalization ~~[illegible]~~ maximizes ~~[illegible]~~ for all ~~[illegible]~~ $P_M'(x)/P_M(x)$ ~~[illegible]~~ for all x. Later ~~we will see~~ it will become clear that this condition also ~~gives least expected~~ ~~[illegible]~~ ~~least expected error in probability estimate~~ ~~a kind of minimal error in prediction~~ ~~leads us to expect minimal errors for P_M' to take minimal prediction errors.~~ gives least expected ~~[illegible]~~ prediction error for P_M'.

12 : from ~~[illegible]~~ Printout 2.40 Just how accurate are the predictions of P_M'?

or $\boxed{\text{DIMACS } 1.22-.28}$ but modify by using $\mu \to P_M'$, $P_0 \to P_0'$.

or $$\underset{\mu}{E} \sum_{m=1}^{n} \left(P_M'(a_{m+1}=1 \mid a_1, a_2 \cdots a_m) - \mu(a_{m+1}=1 \mid a_1, a_2 \cdots a_m)\right)^2 < -\tfrac{1}{2} \ln \overset{\text{u.c.}}{P_0'} \qquad (4)$$

or $\boxed{\text{DIMACS } 1.29-.32}$ ~~[illegible]~~)

The truth of eq (4) hinges on the fact that if μ is a computable probability measure ~~[illegible]~~ then there exists a positive constant P_0' such that ~~where~~

$$\frac{P_M'(x)}{\mu(x)} > P_0'$$

and that while P_0' will depend on $\mu(\cdot)$ and $P_M'(\cdot)$, it will be independent of x.

Eq (4) can be usefully generalized so that if

- P_1 and P_2 are any normalized measures on x
- $x(n)$ is a string of length n.
- $$\frac{P_2(x(n))}{P_1(x(n))} > \alpha(n) > 0 \qquad (5)$$

where $\alpha(n)$ is a function of $P_1(\cdot)$, ~~[illegible]~~ $P_2(\cdot)$ and n, but not of x

Then $$\underset{P_1}{E} \sum_{m=1}^{n} \left(P_2 \qquad - P_1 \qquad\right)^2 < -\tfrac{1}{2} \ln \alpha(n) \qquad (6)$$

~~The proof of the convergence theorem of eq (4) given in Sol E78 is for the~~

The Convergence Theorem of eq(4) ~~[illegible]~~ is true if P_M' is a ~~un~~normalized universal measure. Peter Gacs (Gak 97 93) has shown it to be true for the ~~un~~normalized semimeasure, P_M, but the associated convergence constant $-\frac{1}{2}\ln P_0$ is much larger than the corresponding constant, $-\ln P_0'$ for P_M'. ~~[illegible]~~ The difference between them is $-\frac{1}{2}\ln \frac{P_M'}{P_M}$ ~~[illegible]~~ $\frac{P_M'}{P_M}$ is the value of the normalization ~~[illegible]~~ factor for very large n. We have selected a normalization technique to make it as large as possible.

→250.00

Nips

00:247.40: stuff to ~~bottom of 241~~ is perhaps readable; i.e. 241.36 ; 242.15: revise

01:241.36 We are using a/typical normalized ~~[illegible]~~ universal distribution, $P'_M(x)$. Minimal normalization conditions are $P_M'(\Lambda)=1$; $P_M'(x0)+P_M'(x1)=P_M'(x)$

There are many normalization techniques that satisfy these constraints. It can be shown that an additional constraint ~~[illegible]~~ leading to minimal expected probability error is that the probability ratios of the normalized and unnormalized distributions remain the same: ~~[illegible]~~

$$P_M(x0)/P_M(x1) = P_M'(x0)/P_M'(x1).$$

This gives the recursion relations:

.10 ~~$P_M'(x0) = P_M'(x)\frac{P_M(x0)}{P_M(x0)+P_M(x1)}$; $P_M'(x1)=P_M'(x)\frac{P_M(x1)}{P_M}$~~

11 $$P_M'(x0)=\frac{P'_M(x)}{P_M(x0)+P_M(x1)}\cdot P_M(x0) \qquad P_M'(x1)=\frac{P'_M(x)}{P_M(x0)+P_M(x1)}\cdot P_M(x1).$$

15 with the initial condition $P_M'(\Lambda)=1$. → 242.00

~~To reduce expected prediction error in probability,~~ we will ~~be~~ using a particular normalized ~~[illegible]~~ version of the universal distribution.

.17 ~~[illegible]~~ Because certain of the codes, s_k do not result in useful output (i.e. the computer prints out part of x, but continues to calculate without printing any further.) the resultant probability distribution is not a true measure, but a ~~[illegible]~~ *semimeasure* — usually italics

20 i.e. usually $P_M(x0)+P_M(x1) \leq P_M(x)$.

~~By suitably [illegible]~~ ~~It can be shown that~~

We will normalize P_M ~~[illegible]~~ to P_M' so that

23 $$P_M'(x0)+P_M'(x1)=P_M(x).$$

The additional constraint, $P_M'(\Lambda)=1$, assures us that ~~[illegible]~~ the probabilities of all strings of a given length sum to one.

Later, it will become clear that the larger P_M' is, the less expected error it has. There are many normalization methods that satisfy these two constraints. To obtain as large P_M' as possible, we add the constraint.

30 There are ~~[illegible]~~ at least 2 ways to use (1) for prediction:

$$~~[illegible]~~ \; P(x0|x) = P_M(x0)/P_M(x): \quad P(x0) = P_M'(x1)/P_M(x) \qquad (3)$$

$$P(x1|x) = P_M(x1)/(P_M(x0)+P_M(x1)) \quad P(x0) = P_M(x1)/(P_M(x0)+P_M(x1)) \qquad (4)$$

eq(3)

6/25/03

Nips

2~~50~~.40
50: 2~~45.40~~

• Suppose that $[D_n]$ $n = 1 \cdots h$ is a set of h strings generated by some unknown stochastic device. What is the probability that ~~[illegible]~~ our universal distribution ~~[illegible]~~ assigns to D_{n+1}, a ~~[illegible]~~ (possibly) new string?

It is just $P_M([D_n] \cup D_{n+1}) / P_M([D_n])$. (10) eq.

omit: Any ~~[illegible]~~ function that can assign a probability to any finite string can also be used to assign ~~[illegible]~~ a probability to each bit of ~~that~~ a string, conditional on the preceding bits of that string.

~~How accurate are these probabilities?~~
~~These probabilities~~ For a suitable ~~[illegible]~~ set of strings, $[D_n]$ these probabilities ~~[illegible]~~ can be very close to those assigned by μ, the true generator of $[D_n]$

In section 3, we will discuss Operator Induction and prove ~~[illegible]~~ an associated convergence theorem ~~[illegible]~~ that implies a convergence theorem for induction on unordered sets.

Section 3 Operator Induction.

In the Operator Induction problem, ~~[illegible]~~ as described in the ~~[illegible]~~ introduction, we are given an unordered set of n string and a number pairs, $[Q_i, A_i]$ ~~[illegible]~~ Given a new Q_{n+1}, what is the probability distribution over all possible A_{n+1}?

We will ~~describe two~~ give two solutions. ~~[illegible]~~ ~~The first~~ [Section 3.1] In the first, we consider this to be an extrapolation of unordered finite strings, D_i, in which $D_i = Q_i, A_i$.

~~D_i [illegible]~~

eq. 8 is used to obtain a probability distribution on all unordered sets of Q_i, A_i pairs and eq (10) gives us a probability distribution over ~~[illegible]~~ ~~[illegible]~~ (Q_{n+1}, A_i), — i.e. $P(Q_{n+1}, A_i)$ for all possible A_i.

~~Then~~ $P(A_i) = P(Q_{n+1}, A_i) / \sum_i P(Q_{n+1}, A_i)$ (11) eq.

Section 3.2

The second solution to the Operator problem is

5o2th.tex §5 of Report (eq (8) of report) to 15.27 then jump to Appendix B.) This includes "Appx B shows" Modify this to "we will show this".

50 2-2.tex our par §2.3 (AppxB) of report. Then through end of appx B. Some Modifying.

Leave 3 or 4 blank lines — I will fill it in.

I can take care of this part myself.

~~To~~ Follow this w. discussion of how Bay induction is special case of

so since our first soln of Op induction problem used Bay induction — we can be certain that it, too, converges rapidly

241.36 → 24~~3.~~15
243.15 → ~~[illegible]~~, 243.20
243.20 → 242.00
~~241.40 [illegible]~~
~~[illegible]~~
242.40 → 243.25
243.25 → 243.40
243.40 → 245.00
245.40 → 247.00

251.09
~~248.00~~

N.ps

00 .24.40: Actually, we really chose cands on basis of max $f_n(x(n))$ — for x(n) being the corpus.
So this $f_n(x(n))$ is perhaps an approxn of $\underset{u}{E}\, f_n(x(n))$. (i.e. x(n) are assume is a "typical" u output.)
or maybe $\underset{u}{E}\, \ln f_n(x(n))$.

.03 So after all is said & done we select f_n's ∋ $P(f_n) \cdot f_n(x(n))$ is large & sum w. $P(f_n)$ wts. Simple old Bayes; But we realize that ① $P(f_n)$ is impt. ② it's much better if $P(f_n)$ is truely & carefully Updated!

05

So, indep of all the conversion stuff, we still use same methods to estimate error

(The Alp does enable us to do away w. the training set)

10 The Cost of eliminating the Trg set is that we have to consider many more models — So their apriors ↓! so we may not be getting such a BIG Bargain!

But the main Idea is that all the careful analysis about accuracy, etc. is not so important as $\boxed{.03 - .05}$: Just to use good standard Bayes w. good updated apriori. A possl. turn from ~~ALP is to~~ try to achieve a priori "Universal" — that one can get ideas on how to get the A priori from the existing Scientific language of the Domain of inquiry.

20 $\boxed{\text{SN}}$ On Things that Converge "with probability one": Unfortly, this tells nothing about How fast they converge! Such theorems are often based on other theorems about "Converging w. pc=1"

23 It would be possl. to put many theorems in forms giving rates of convergence, if one had a large enuf BASIS of Thms on quantitative Convergence Rates. Unfortly, many theorems in statistics are of "pc=1" character — this is like in Biology, discovering "A influences B" gets a nobel prize, but finding out just how much "A influences B" is usualy not looked into at all — certainly no nobel prize.... Quantitative Confirmation is usualy regarded as Hackwork — 10 "non creative".

In .23 ff this work would involve ~~f~~ devising a large enuf basis of thms on Convergence rates. One would also have to know lots of "inequalities". Maybe a good PHD thesis for someone! — Or maybe a much Bigger job!

6/24/03

N.p.

o: 243.40 In the foregoing ~~convergence~~ convergence theorems, we have used the total squared error of probability as a loss function to be minimized.

Hutter has shown that the universal distributions also converge rapidly when more general loss functions are used ~~(Hutt~~ → Optimality of Universal Bayesian Sequence prediction for General Loss and Alphabet

Tech Report IDSIA - 02-02 18 Feb 2002

Give web site.

or: General loss bounds for Universal ~~Sequence Prediction~~ Sequence Prediction

Tech report IDSIA 03-01 10 April 2001

only 8 pages

This also has proof But is a paper

I could expand this — explaining ~~what happened~~. Just what he proved.

usable for Both Hutter references.

Hutter considers more general loss functions and shows that use of the universal distribution gives ~~a~~ loss functions that converges rapidly toward their smallest possible value. (Hut)

perhaps a footnote. — perhaps refers to sub section "2.1"

Section 2 Induction on Unordered Sets.

~~In this sec we will~~ ~~De~~ The beginning of this section 2.1 follows the discussion of Sol 99 pp 256 and 258

Sol 99 256.27—.37

Suppose we have an unordered set of n finite strings of symbols. . . . . [2, 3 ~~[illegible]~~ refs. ~~pp 240-251~~ (omit?)]

A solution using a universal distribution is obtained by assuming ~~[illegible]~~ ~~a set of probability distributions~~ ~~$P_j(D_n)$~~ ~~on~~ all possible strings: that the data was generated by some unknown ~~[illegible]~~, finitely describable probability distribution on all possible finite ~~[illegible]~~ strings, ~~$\mu(D_n)$~~ $\mu(D_n)$.

The universal distribution on unordered sets of strings, $P_M([D_n])$ ~~$P_M(D_n)$~~ kind of is a weighted sum of all finitely describable probability measures and semimeasures. ~~Eq~~

$$P_M([D_n]) = \sum_j \alpha_j \prod_{n=1}^{h} P_j(D_n) \qquad (9)$$

h is the number of strings in the set $[D_n]$

α_j is the ~~weight~~ WEIGHT of the jth probability ~~[illegible]~~ distribution on finite strings.

$\alpha_j = 2^{-|a_j|}$, where a_j is ~~[illegible]~~ the shortest description of $P_j(\cdot)$ and $|a_j|$ is the length of that description.

The M index of P_M indicates that the ~~descriptions~~ functions are to be described with respect to Machine, M.

Since M is normally universal, it can describe any finitely describable function.

The probability assigned by μ to the set of strings, $[D_n]$ is

$$\mu([D_n]) = \prod_{n=1}^{h} \mu(D_n).$$

Maybe not include

Since P_M includes μ as one of its component probability distributions

$$P_M([D_n]) > \text{~~[illegible]~~} \; W_\mu \, \mu([D_n]).$$

i.e.

Here ~~[illegible]~~ W_μ is the ~~[illegible]~~ weight of μ in P_M.

Since we don't know μ, we don't know W_μ.

247.00

6/24/03
Nips

10 $\boxed{\text{SN}}$ On radix ≠2 for ALP & Conv. theorem: For ALP, there is no diffy in formulating t. distribn. For t. conv. theorem, showing its true for t. KL distance is easy via Gacs's proof. Showing

02 $\Sigma(\text{error})^2 <$ KL dis for radix >2 is diff't.

Hvr one could code radix 3 as 00, 01, ~~■~~10, with 11 never used. We then just use our binary formulation of ALP, (possibly removing 11, when it occurs & renormalizing t. corresponding, 00, 10/01. I suspect this would give t. right universal dist. Could one use the Conv. theorem for binary strings to show that these ternary pc's also converged properly?)

a b are 2 bits in a pair coding 0,1,2'. $\underset{00}{\alpha}\ \underset{01}{\beta}\ \underset{10}{\gamma}$ are t. 3 ternary values.

~~[illegible]~~ ~~[illegible]~~ $p(a=0) = p_\alpha + p_\beta$. p'_α, p'_β & p'_γ are corresponding pc's for $\underline{\mu}$

0 $p(a=1 = p_\gamma$

$\beta(b=0) = p_\alpha + p_\gamma$

$\beta(b=1) = p_\beta$

for t. ternary error we know that $\Sigma\ (p_\alpha - p'_\alpha + p_\beta - p'_\beta)^2 + (p_\gamma - p'_\gamma)^2 + (p_\alpha - p'_\alpha + p_\gamma - p'_\gamma)^2 + (p_\beta - p'_\beta)^2$

converges: Q: does $(p_\alpha - p'_\alpha)^2 + (p_\beta - p'_\beta)^2 + (p_\gamma - p'_\gamma)^2$ converge?

call $p_\alpha - p'_\alpha$ Δ_α is $(\Delta\alpha + \Delta\beta)^2 + (\Delta\alpha + \Delta\gamma)^2 \geq (\Delta\alpha)^2$?

$2\Delta\alpha^2 + \Delta\beta^2 + \Delta\gamma^2 + 2\Delta\alpha\Delta\beta + 2\Delta\alpha\Delta\gamma \geq (\Delta\alpha)^2$ $\Delta\alpha \to \alpha$: $-1 \leq \alpha \leq +1$

is $\alpha^2 + \beta^2 + \gamma^2 + 2\alpha\beta + 2\alpha\gamma \geq 0$?

0 $(\alpha+\beta)^2 + \gamma^2 + 2\alpha\gamma$

I think I have this t. wrong way, but either way its wrong!

is $(\alpha+\beta)^2 + (\alpha+\gamma)^2 \geq \alpha^2$. $\underline{\underline{\text{No}}}$. say $\alpha = 1$, ~~[illegible]~~ $\beta = -1$, $\gamma = -1$

$0 + 0 \ \ldots\ 1^2$.

$(\alpha+\beta)^2 + (\alpha+\gamma)^2$ v.s. α^2

So it looks like it may have to be proved by showing .02 is true: which is diff'lt. Hutter says he did it but its diff'lt: I haven't checked his proof, but I have shown it to be very likely for radix 3 (& maybe 4) by Montecarlo trials.

28 $\boxed{\text{SN}}$ Say we are using a set of functs $\mathcal{M}$ to try to find t. best one to predict our data.

0 μ t. true generator, is not in $\mathcal{M}$. But for each value of $\underline{\underline{n}}$ (t. no. of bits in t. data we're using)

There is a function $f_n \ni \frac{f_n(\overset{\text{data of length } n}{x(n)})}{\mu(x(n))}$ ← pc assigned by fn is max.

(Even if μ is in $\mathcal{M}$, it may have a small wt., so that fn in this case will get more wt. than t. μ that is within $\mathcal{M}$.)

Hutter treats t. case of μ not in $\mathcal{M}$ in one of his papers (maybe t. 44 pp. one!)

He considers the funct $\underline{\text{in}}$ $\mathcal{M}$ that has ~~t.~~ closest KL distance from t. true μ (or something like that).

In my formulation, $\mathcal{M}$ will vary w. T, t. $\underline{\text{CB}}$. T. ~~no~~ value of n may vary (for a given T) over different functs. in $\mathcal{M}$ — causing trouble. Hvr, if n varies between bounds, we can't compare them very well (but we can approximately compare them).

246.00

6/24/05

NIPS

→ T. Discussion in S78 § VI on Cover's measure, is easy to read. — T. Discussions of interest & Lemmas & proofs are easy to understand.

30 : 242.90 SN In Sol 78 I showed that Cover's Extension Complexity Never differed from my Method of Normalization by a factor that → ∞ as n → ∞, (but very slowly!)
Since Extension probly is based on a countable no. of continuations of the corpus, & ALP is based on an uncountable no. of continuations, ... this may explain it .... but look into this! Essentially, Cover used a semimeasure, like Gacs version of the Convergence theorem.
(SN) Perhaps include Gacs's "Sequential Predn. via Semis." & explain why "Gone.""

→ The proof

0 : 242.40 : ~~Theorem VI Eq(2) has been~~ The proof of eq(2) in Sol 78 is for ~~[illegible]~~ a ~~[illegible]~~ P_M that ~~is~~ has been normalized so that

Omit

$P_M(0)+P_M(1)=1$ and $P_M(x0)+P_M(x1)=P_M(x)$. ← This is a ~~very~~ general normzn & not what I need

The

---

15 : 241.36 To obtain minimal error in probability, we normalize P_M to P'_M using the constraints:

$$P'_M(0)+P'(1)=1; \quad P'_M(x0)+P'_M(x1)=P'(x); \quad P'_M(x1)/P'_M(x0)=P_M(x1)/P_M(x0). \quad (2) \leftarrow eq$$

→ Probly should give normzn eqs (shown in LIV or Sol 78

~~For this normalized P'M~~

~~[illegible]~~ To use P'_M for prediction: ~~[illegible]~~!

0 — $$P(x1|x) = P'_M(x1)/P'_M(x) \quad (3) \leftarrow eq$$ → 242.00

maybe begin $P'_M(x1)$ in terms of $P'_M(x)$, $P(x1)$ $P(x0)$

→ 242.00

~~The convergence Result (Sol 78 § p 426) assures us that P'M gives very good predictions.~~

---

25 : 242.40 The proof of ~~eq~~ the convergence theorem of eq (4) given in (Sol 78) is for the ~~[illegible]~~ normalized ~~P'M measure~~ universal measure P'_M. Peter Gacs ~~by~~ (Gac 97 93) has shown that it is also true for the unnormalized semimeasure, P_M, but the convergence constant, ~~[illegible]~~ $-k \ln P_0$ is much larger than the corresponding constant ~~for~~ $-\frac{1}{2} \ln P'_0$, for P'_M.
The difference between the two will be about half the natural log of the normalization factor for large n.

0 The result is that the probability errors for the normalized measure P'_M converge much more rapidly than those for ~~the~~ the semimeasure, P_M.

Gacs also ~~[illegible]~~ shows (ibid) that the corollary corresponding to ~~eq (5)~~ eqs 5 and 6 holds if $P_2(\cdot)$ is ~~[illegible]~~ an (unnormalized) semimeasure.

Marcus Hutter has shown (Hut) that these results hold if we ~~[illegible]~~ use ~~[illegible]~~ alphabets with more than 2 symbols.

Tech Report IDSIA 07-01 ~~Feb~~ 2001
Convergence and Error Bounds for Universal Prediction of Nonbinary Sequences.

Stac. → 245.00

6/23/03

NIPS

248.15

00: ~~243.20~~

The convergence theorem (Sol78 p426) assures us that this technique gets good probability values.

The way it works:

[Dimacs 1.22 - .28] suppose you had a device, μ generating . . . ~~[illegible]~~ . . . generator of the data

Then put in

$$\underset{\mu}{E} \sum_{m=1}^{n} \left(P_M(x_{m+1}=1 \mid x_1,x_2\cdots x_m) - \mu(x_{m+1}=1 \mid x_1,x_2\cdots x_m)\right)^2 < -\tfrac{1}{2}\ln p_0'$$ (4) eq

Dimacs 1.29 to 0.32 ~~p_0~~ is dependent on ~~the probability that . . . to order~~ and of ρs : . ∴ . . . very good probability estimates. ~~one of the universal generator~~.

~~An useful term corrolary to this theorem~~:

The proof of this theorem ~~is based on the idea that for any [illegible] any [illegible]~~ hinges ~~based~~ on the fact that

if μ is a computable probability measure ~~[illegible]~~ (or more generally, any finitely describable probability measure) then ~~theorem~~

$$\frac{P_M(x)}{\mu(x)} \geqslant p_0'$$ ~~(5)~~ eq

~~There exists a constant, $p_0 > 0$, so that~~

~~for all values of x~~ p_0' is a positive constant whose value depends on ~~μ and P_M~~ the structures of μ and of P_M, but is independent of X.

Omit

Equation 4 can be usefully generalized ~~[illegible]~~, so that ~~[illegible]~~ for any two ~~probabil~~ probability ~~distributions~~ measures P_M and μ ~~[illegible]~~ not necessarily universal or computable, P_1 and P_2

If $\frac{P_M(x(n))}{\mu(x(n))} > \alpha(n)$ $\frac{P_2(x(n))}{P_1(x(n))} > \alpha(n)$ P_1 normal

~~[illegible]~~ ~~[illegible]~~ n being the number of bits in X

Then equation (2) holds with error bound $-\frac{1}{2}\ln\alpha(n)$.

(perhaps ~~simplify~~ use P_1 and P_2 instead of P_M & μ — and rewrite eq (2)) (5) eq

~~Not~~ to Gacs

use Balls •

~~The~~ Equation ~~4~~ can be usefully generalized so that if • P_1 is any normalized probability ~~distribution~~ measure, ~~and~~ • P_2 is any other probability ~~distribution~~ measure or semi-measure, ~~[illegible]~~ ~~[illegible]~~ • x(n) is any string of length n ~~[illegible]~~

• ~~[illegible]~~ ~~[illegible]~~ $$\frac{P_2(x(n))}{P_1(x(n))} > \alpha(n) > 0.$$ (7) eq

where α(n) ~~[illegible]~~ is ~~[illegible]~~ a function of $P_2(\cdot)$ and $P_1(\cdot)$ ~~[illegible]~~ and n, but not of x —

then $\underset{P_1}{E}$ $< -\frac{1}{2}\ln\alpha(n)$. (8) equ

see on → 243.25

6/23/03

NIPS

Possibly ommit section 4.

10:238.40 : T. forg. ~~[illegible]~~ 238.00-40 can be ~~[illegible]~~ long abstract or short introduction.

At the end of the Intro.

Section 1 deals with ~~the~~ sequencial prediction and its universal distribution. We discuss generalizations of ~~Generalizations~~ of the problem ~~[illegible]~~ that are amenable to the same general solution.

~~A convergence theorem for the normalized) Universal distribution for sequencial pred~~

This is followed by a convergence theorem for the normalized distribution and some more recent generalizations of it.

Section 2 ~~describes~~ ~~[illegible]~~ deals with the extrapolation of ~~[illegible]~~ a set of unordered strings and/or numbers, and gives an associated convergence theorem.

0 Section 3 deals with Operator induction, and gives the associated convergence theorem.

Section 4 ~~Discusses the incomputability of~~ Discusses the ~~the~~ incomputability of the universal distribution and ~~[illegible]~~ its (limited) ~~[illegible]~~ bearing on practical implementation. ~~We also g~~ ~~We give a quantitative estimate of the frequency with which~~ ~~the incompit~~ We discuss the "halting problem" and the frequency with which it has a bearing on the computability of the universal distribution.

Section 1 Sequencial prediction

The universal distribution for sequencial prediction is a probability distribution on strings that is obtained by assuming the strings were the output of a universal machine with random input. We will at first consider only universal Turing machines with binary unidirectional input and output tapes and an infinite bidirectional work tape. ~~Though~~ Most of these conditions can be relaxed using more general kinds of machines and input output mechanisms.

20

~~To obtain the probability of a finite string x~~ How do we use this definition to get an expression for the probability of a particular finite string, x?

We will ~~[illegible]~~ regard the probability of a finite string x ~~to be the sum of the probabilities of all finite or infinite strings that have x as prefix.~~ Let [S_k] be the set of all binary programs for our machine, M such that M(S_k) ~~[illegible]~~ gives an output with x as prefix. To prevent double counting we have the additional constraint on the set [S_k] : If we drop the last bit of the string S_k, the resultant program will not have output with x as prefix. With this condition the probability of x becomes the sum of the probabilities of ~~[illegible]~~ all of its programs;

30

$$P_M(x) = \sum_k 2^{-|S_k|}$$

(S_k) is the number of bits in S_k and $2^{-|S_k|}$ is the probability of an input that has S_k as prefix. ~~[illegible] normalized P_M(x)~~ → 243.36

13[illegible]

To use this distribution for prediction:

0 mmit

$$P_M(x1|x) = P_M(x1) / (P_M(x0)+P_M(x1)).$$

$\sum$ x0 ~~[illegible]~~ being the binary string x followed by the bit, 0. [illegible]

This gives us the probability of a 1 following the string x.

Nips

CFG discovery

GREEDY

30.239.40: we will have several choices of defns to make. Rather than chose the single "Best" one (the one w. max pc), chose defns w. ~~pc~~ wts ∝ their pc's. ← (NOT so GREEDY)

Try to design my System so it is ergodic w. not ~~too~~ small pc of going into all possl. states — essentially, this means that it can't get "stuck". On the other hand, we want the System to spend almost all of its time in v.g. states.

Hvr, one idea of Universality is that it has an ∞ of states & that their pc's can be varied. On the other hand in any ~~a~~ U.D.F. no state gets pc = 0!

In Gerry Wolff's early system he would periodically reparse it. If some bad defns were made, (perhaps) reparsing would make them unimportant.

It may be that in ~~the~~ ~~a~~ (e.g. English, that parsing rules are usually of pc close to 1 — which means that usually/likely using our parse will be adequate.

On the other hand, in more general systems in which I will ~~be~~ using (Generalized Induction), this will not be true, ~~&~~ & the ideas of 237.30ff may be ~~a~~ best.

Anyway, it would seem that there are 2 essentially diff't Approaches here:

1) 237.30ff: ~~to~~ try various defns, pc's etc. sequentially branching various parse trials — ~~Then~~ Doing many trials for entire system (parse/definitions) .... predict based on wtd. combination of the Best ones.

2) ~~Re~~ Do various defns but repeatedly reparse after every (or every few) defns.

2) would seem better if it is likely that there are only a few v.g. ~~parses~~. parses a/o (set of definitions).

6/23/03
Tv.ps

CFG Discovery

.00 : 237.40: Say ab occured 10 times & the best way to parse it would be do 3 of ab→C & leave 7 as ab. So we randomly chose ab→C w. pc = $\frac{3}{10}$ to obtain a parse. We then increment the corpus (15→16, say) and look for new defns. If none are found ⋕ inc corpus size & search: loop until new defn. is found, then randomly prune as before.

.06 We run thru the entire corpus this way using random choices. Ideally, we would go thru the entire corpus this way, many times, & use a wtd sum of the resultant Grammars, for predn. ~~[illegible]~~ What kind of D.F. we'd get on the wts of the Grammars/parses is unclear. Will it be a few hi wt. or a broad set of lo wt? Will we have a Gaussian D.F. on ~~[illegible]~~ ln wt.? (most likely).

.10

.11 ——— An alternative method: very more rough, (perhaps): Do corpus length 20, Select out best 100 grammars & continue with them to 30 symbol corpus: retain best 100 grammars & continue etc.

Another way (in the spirit of .11) Go thru the entire corpus as .00–.06. Next go thru the entire corpus again (random choices), but as soon as (if ever), we get far behind ~~the best run~~ any point in the best run thus far we abort the trial, & backtrack (don't necly go back to zero corpus size), back to a point at which "things aren't so bad". We work on a particular branch/tree for a certain amt of time: if we don't seem to be making enuf progress per unit

.20 time, then we may go back to the root (zero corpus size) & start over.

For certain kinds of stochastic processes (ergodic is one kind, perhaps), if we go thru this branching process, we will almost always end up in the hi pc region. Whether the "process" of .237.30 ff is of this kind ... I don't know. In Ergodic Processes we are always ~~[illegible]~~ a finite distance (pc-wise) from ~~[illegible]~~ any other state, so for a long run, the pc distance betw runs, will be "Negligible".

.28 [It is conceivable that in certain kinds of processes, one could branch into regions of state space

.29 that were quite diffrnt Statisticly from others.

30 While ~~a~~ long English text maybe Ergotic, my method of analysing it may not be. My impressn is

31 that decisions made early in the process cannot be "undone" —— so if one makes a bad mistake in defining something, certain symbols will be permanently removed from the corpus, so one couldn't ~~[illegible]~~ ever use them "properly" later ← Tho may be not so extreme since normally one does not completely "parse away" any particular symbol — one only reduces its probability (which may be bad enuf)

A way to ~~[illegible]~~ perhaps avoid .31 ff. ~~[illegible]~~ At each pt. in the process,

240.00

6/22/03

NIPS

CFG (i maybe CDG) Discovery

CFG Discovery IMPT!

.30 ff looks VERY GOOD! — perhaps Great Breakthru!

.00 : This is not a new idea, but I was sort of forgotten it — I think I had some of these ideas in Sol 56 ("an ind. inf machine") ↓ Now: From Corpus, using methods of NZ 141 (latest methods: old Z14 didn't work).

~~@~~ derive 2gms from Combine 2gms to get ngms, etc.

Next get ngms sets: Some ways: b a the set of all b's ∋ ~~ab~~ ba is likely to be ~~[illegible]~~ useful ngm. or a c set of all c's ∋ ac is used ngm.

Other ways to derive ngsets from ngms. ~~[illegible]~~ various environments: perhaps context dependence.

After we have ngms & ngsets, we can try making new ngsets formed by

.10 ① Concat ② Boolean combinations (usually "AND"). If we use any other combin rules we may end up w. a non cfg (But that's o.k.)

Then test recursive rules like A = AB ; A = BA, A = BAC, etc.

We will test such rules ⇔ we see an instance of it: i.e. a_1 and $a_1 b_1$ are common ngms ~~and [illegible]~~ a_1 is in ngset A; b_1 is in ngset B.

.14 So we try ~~[illegible]~~ A = AB

.15 → It may be that rules & heurs of this kind are the only efficient way to ~~[illegible]~~ discover CFG's. ← LIKELY!

It seems clear that w. adequate ssz, longer and longer ngms will be found, & they will be expressible by grammar rules, & the heuristics for trying Grammar rules will be that the rule works for ~~[illegible]~~ 1 (or a small ssz).

.20 We can "tune" this heuristic by balancing ssz in heur, w. cost of ~~[illegible]~~ trying the heur,

.21 & "yield" of the heur.

~~OK: Back to 235.30 .40! In them: start w. abstract of Sol 99.~~

SN on CFGs: I had steered clear of doing pc = 1.0 parses. The result was many possible parses — lots of work to keep track.

I could try 100% replacement of ab ~~[illegible]~~ G say. Maybe not always nice, but easy to go to "next level"

Another possy is less greedy parsing: Do 2 levels at a time — so I have some "Look ahead" for opportunitys at "next level",

.30 Alternatively, ~~set~~ after a new defn has been made, select a set of random parses & see which can be best compressed. (Actually all parses would have to be investigated)

.32 Actually, .30 gives the true picture! ~~[illegible]~~ Each parse is part of a highly branching stochastic process! — So we just pick the parses at random & continue along each one. We trace thru as many as ~~[illegible]~~ we can ... obtaining many grammars ... then we pick a set of the grammars of highest pc.

.32 should also be used in the more complex parsings resulting from use the recursive ~~[illegible]~~ Grammar rules discovered via .00 – .18 (+.15 – .21)

This .30 ff looks VERY GOOD! The way it works: we do the first ~~[illegible]~~ symbols of the corpus: try to find good ~~[illegible]~~ 2gms. For each 2gm we have a parsing decision: → 239.00

6/22/03
NIPS

00:236.40 Problems in ~~inductive inference~~ probabilistic induction are of two general kinds: In the first, we are given a linearly ordered sequence of symbols to extrapolate. ~~Although~~ A ~~[illegible]~~ very general solution to this problem is based on the universal probability distribution, and there has been much work done in ~~approximating it~~ finding good approximations to it [lit ref]. It has been ~~proved~~ shown that for long sequences, the errors in probability estimates converge rapidly toward zero.

~~Though the second problem is a~~ In the second kind of problem, we want to extrapolate an (unordered) [illegible] sequence of ~~strings~~ finite strings and/or numbers. ~~Though~~ ~~[illegible]~~ A universal distribution has been defined that solves this problem (Sol 99). We will give a convergence theorem that shows ~~that~~ it to give small errors as the number of examples increases — just as with sequential prediction.

~~[illegible]~~ In Operator induction we have an unordered sequence of ordered pairs of elements (Q_i, A_i) (that may be strings and/or numbers). Given a new Q_i, to obtain the probability distribution over possible A_i's. The Q's can be ~~[illegible]~~ questions in some formal or natural language, ~~[illegible]~~ the A's can be ~~associated~~ associated answers. The A's can be inputs to some unknown stochastic device and ~~the Q's~~ A's can be outputs (The Identification problem). The Q's can be descriptions of Mushrooms, the A's can ~~be~~ tell if they are edible or poisonous (The categorization problem).

The Q's can be number and the A's can be exact or noisy value of some unknown function of those numbers (The curve fitting problem).

We will give two solutions to this problem based on universal distributions, and give associated convergence theorems that suggest their precision in prediction.

~~[illegible]~~ All universal distributions are, of necessity, incomputable. We will ~~[illegible]~~ show how the practical application of these distributions is usually not affected by their incomputability.

The incomputability ~~is~~ of the universal distributions is associated with the unsolvability of the "halting problem". We will show that ~~[illegible]~~ for ~~[illegible]~~ most prediction problems ~~only a fair amount of data~~ ~~[illegible]~~ The occurrence of a halting problem ~~occurs with [illegible] rapidly~~ becomes very unlikely as the size of our data set increases ~~decreasing frequency, that rapidly decreases with sample size.~~

omit? (While this fact is of theoretical interest it is very rarely of importance in practical prediction.

We will show that the frequency of occurrence of halting problems, ~~rapidly~~ approaches zero rapidly, ~~as the size of the data set grows~~ with the growth of the data set.

237,
239–240
on Grammar
discovery
→ 241.01

00 : (SN) to EΣ pc07 conv. theorem May mean that ~~for~~ μ = CPM, one can get an arbtly small/knowable ~~error by~~ doing so; But ~~probably that~~ tho this is a kind of "computability" it is not practical computability: E.g. if we limited our models to prim. rec. functs — They would give "computable" ALP, but not ~~practically~~ computable ALP.

If μ ≡ Bern seq. w p ≡ .5 This should give interesting ~~statements~~ about halting probly of an arbty UMC! To simulate "μ", just copy input onto output — so it should have a fairly short descn. The descn can be 1 bit in a UMC as so: for any input: prefix ∅, rest of input is copyd to output. for prefix 1, machine acts like ~~regular~~ U10, UMC, for rest of input.

10 SN It maybe that my/proof of conv. ~~Q~~ QA induction in "Appendix B" (pp 23–25) of Report is wrong!

The idea of proof: that the seq $A_1 s A_2 s \cdots A_n s$ (s is marker (space) betw A's.) constitutes a seq. to which the "corollary" (convergence criteria) applies —— is correct; but K has 2 parts:

1) descn of the operator M (which is indep of ~~[illegible]~~ [Q_i])

2) descn of the ordered ~~[illegible]~~ sequence, $[Q_i]^{ord}$:

The ~~[illegible]~~ descn length of 2), grows with n. This Fact is relevant to the last ¶ of Appendix B on p25.

0

Unclear as to whether this is a Bug or not! It looks like a Bug: i.e. The descn $[A_i]^{ord}$, the descn of M is inadequate. But is it legal to assume the $[Q_i]^{ord}$ is "free"?

.19 The Q is can we obtain a genzn of $75 TB cor'l that includes "free Info"?

20 Well, as the $[Q_i]$ sequence changes, the length of the M that can get a particular $\prod P(A_i/Q_i)$ for that sequence, changes.

Whether the $[Q_i]$ are "free" or not, is irrelevant, hvr, the Q is "Does the corollary of (.19) apply?"

Well, not knowing ~~the~~ $[Q_i]$, The U.P.D. generates a M that gets the descn length of the $[A_i]$, which is some constant. How the constant will grow as $[Q_i]$ grows, is unclear.

If the Q_i's remain in "same Domain", its possl. that the const will not ~~grow~~ much (if at all).

A Similar problem occures when we use the "$\prod P^j(Q_i, A_i)$" format induction. (235.02) If we have a certain P^j ~~that~~ works well w. the known $[Q_i, A_i]$ set, — then if we change the character of our Q's much, the P^j will change — & it will probably need a much longer descn. So it would be well to give both conv. thms & perhaps explain the deficiencies of each of them.

Qu.: Back to 235.30–.40 Introdn: maybe start w. ~~abstract of~~ Sol 99? Probablistic induction problems are of Two kinds: . . . . [1,3,4,5,6,9,10]. Two general solutions to the second kind have been described. Maximum UPD Sol 99. Based on associated UPDs. ~~While~~ Though a convergence theorem has been proved for sequential induction, Showi

Mention UPD's.

6/20/03
NIPS

~~[illegible]~~ CIAP : Expo @ .20

~~235~~
235

So t. Q's, just how to present this?

If P(A|Q) is normed already, / ~~Pass~~ $\overset{\text{Max?}}{\sum_j} \sum_0^j \prod_i p^j(A_i|Q_i)$ seems to work.

c82 Otherwise | Maxe $\sum_0^j \prod_i p^j(A_i, Q_i)$. It is not clear that these ~~2~~ methods Maxe to same thing — i.e. that ~~they~~ get t. same soln! We have conv. thms for both, but difrnt thms converge to difrnt constants.

What I can do, Give t. 'conv. thm. for Bags: Then say it will be proved as a special case ~~of Op induction~~ a theorem we will prove later.

For Op induction: 2 ways to state it.

SN can ~~t~~ t. op induction of t. latest QATM be regarded as Encyc. induction of BAG.

T. ~~[illegible]~~ encyc is one "Given" & t ~~[illegible]~~ Bag on use into encyc.

In t. latest QATM, 1. ~~[illegible]~~ Encyc is t. set of Q's. (t encyc can be expanded into in addition to t. set of Q's. — except that t. A's are somewhat ~~[illegible]~~ tied to t. ~~[illegible]~~ associated Q_i's!

So: We will present two somewhat different ~~solutions to~~ solutions ~~[illegible]~~ for Operator induction. ~~The problem~~ This kind of induction is very common, and comes in many guises. We will ~~try~~ discuss the ~~[illegible]~~ relationships between the two solutions.

SN on CIAP: A ~~[illegible]~~ young, inexperienced conscious machine is like a Baby Tiger; cute as hell and lots of fun to play with — but ~~[illegible]~~ as it grows up, it becomes increasingly ~~unmanageable~~ ~~[illegible]~~ unpredictable ~~[illegible]~~ unmanageable, and ~~[illegible]~~ eventually, (very) Deadly. / very dangerous / Very life threatening

For Operator induction, we will present two different solutions to this very important problem.

Give the conv. theorem for Bag induction, then say that proof will be a corollary of another ~~theorem about~~ convergence theorem for a more general kind of induction. — Particular conv. theorem in Sol 78 form; ~~but simpler theorems~~ say derived in recent Lectures

In introduction: / ~~Revs.~~ review Sol 89 + ~~conv. seq. conv theorem~~ the methods for — seq'l induction, / bag induction ~~discussed in~~ describe in Sol 89.

This could be Abstract. 2 methods of Bag induction given.

Conv. theorems for seq induction are well known. & various generalizations to ... each, by Gacs, Hutter.

~~We will discuss~~ Here we discuss Op. induction. Give 2 models ~~&~~ ~~and~~ & their convergence theorems.

— and an assoc conv. theorem for Bag induction

Then > expansion of this: Incomputability due to unsolv. of Halting prob. ("T. halting problem is usually not a problem" Open problem: If seq. has for it a.r.w. CE = To; will ~~[illegible]~~ p(a) converge to CE = To?

For Brief abstract: We will ~~draw~~ give conv. theorems for ~~the~~ Bag induction:

Then ~~2~~ 2 solutions to Op induction problem & ~~2~~ their ~~[illegible]~~ assoc. conv. theorems / or also conv. theorems,

Discussion of applicability of these results to practical induction &

Some theorems on the ~~[illegible]~~ infrequency of / unlikeliness of incomputability.

6/20/03

Nips

140/90 — 154 lbs w. pants. walked; no shoes but sox — 5' 11"

β 20:47

.0: : What I want in Introdn.: Dcrn of problem, review of previous work.

[~~Re~~ are could bound on seql. predn! ~~Gru Ga~~ Martin ~~&~~ my conv. Theorem & Gacs, Hutter's ~~[illegible]~~ extensions of it.

Then Derb Sol 99 results. Tell what Ray left out (conv. Thm).

That the soln. to the O operator induction problem was suggested, here will give that soln. explicitly, & give a different ~~soln~~, perhaps more practicable soln as well as the assoc. conv. Theorem.

~~Remark~~ for the most part the forgoing results are all proved theoretical and all conv. d.f.'s are incomputable

We will discuss ~~[illegible]~~ ~~of~~ of ~~these~~ bearing (or lack of bearing) incomp results on practical ~~[illegible]~~ predn. 10

~~Maybe ∈ — section ( ) discusses has some interesting results~~

~~In all cases the incomp~~ 2 30.30 is a nice intro to the Σ prev result.

Later, I may want to discuss diffrnce betw $P(A,Q)$ & $p(A|Q)$

$P(A,Q) = P(Q)\cdot p(A|Q)$: But how is $P(A,Q)$ related to $P(A|Q)$?

$P(A) = \int_Q P(A,Q)$; $P(Q) = \int_A P(A,Q)$ $P(A|Q) \overset{?}{=} \propto P(A,Q_0)$ perhaps normalized

so maybe $\dfrac{P(A,Q_0)}{\int dA\, P(A,Q_0)} = P(A|Q_0)$

thr, Using the a priori one to get $P(A|Q)$, we have to normalize wrt A ~~anyway~~! — tho in some cases (perhaps) it would already be normzd. 20

So send wm. Black copy of 60/64 papers.

So it would seem. But one could do QATM 2 ways:

→ ① Indirectly by d.f. on (Q_i, A_i) pairs. $P(A_i|Q_i) = P(A_i|Q_i) / \sum_j (A_i^j | Q_i)$

→ ② Directly by conditional d.f. · $O^j(A_i|Q_i)$ but usually we have to do $O^j(A_i|Q_i) \Rightarrow O^j(A_i|Q_i)/\sum_j O^j(A_i^j|Q_i)$

Though ~~Be~~ Operator induction ~~as~~ can be ~~[illegible]~~ looked upon as a kind of BAG induction problem, we will treat it ~~[illegible]~~ as a special kind of problem — having its own convergence theorem.

Ordinary BAG induction can be treated as a special case of Operator induction.

We will present 2 ~~[illegible]~~ kinds of solutions to the Op induction problem.

30 Sometimes $P(Q,A)$ is easier to model.

" $p(A/Q)$ " " " "

32 Conjecture: for every random no. x (say on (0,1)) there is an interval ∈ ∋ all ·nos. on interval $x \pm \epsilon$

33 have k cost > some integer, R. For small enough ∈, R is a simple function of ∈.

→ This is related to integrating continuous pc functions

Perhaps show 32-.33 for rational nos. in $x \pm \epsilon$.

We can consider, say π, & rational nos. close to π. On interval $\pi \pm \epsilon$, all rationals must have denominator $> f(\epsilon)$. (~~[illegible]~~ f is some ~~[illegible]~~ increasing function of $\frac{1}{\epsilon}$.

(could show that $f(\epsilon)$ is diffrnt for $\pi \pm \epsilon$ v.s. $\sqrt{2} \pm \epsilon$.

6/17/03

00:231.40. SN Actually, Gac's proof has t. same kind of corollary as Sol(78) T3:

I think Gac's proof also proves Sol 78T3, ~~as~~ if it is dependent only on $\frac{\mu(x)}{\rho'(x)} > e^{-k}$ ρ can be a/2 semi measure (it includes computable measures) I think t. only property of ~~P~~ P_M ~~and~~ that Gacs uses is that probby ratio inequality In fact (p331 LIV97) Corr 5.2.1 does

06 just that: (if $\mu(x) > e^{-k(n)} \rho(x)$: $\sum_{i=1}^{n} S_i \le k(n)$ ~~[illegible]~~

μ & ρ can be any 2 semi measures satisfying (.06L).

Any normalized semi measure is a measure, so it works for ~~[illegible]~~ normed measures — except that -k(n) is smaller for normalized ~~[illegible]~~ semimeasures.

0 I could just write a shorter paper giving Bay & Operator induction & ~~theorems for~~ prove conv. theorems for Op mth & say t. other proof is a corollary. So perhaps do that first, then if I have time, I can add in stuff on sequential predn. & "Section 5" on P(U) convergence.

Desn of Big problem & Soln

~~Introduction~~ or abstract We have described two kinds of probabilistic induction: In ~~one~~ the first we have a linearly ~~ordered~~ sequence of symbols that must be extrapolated. In the second we extrapolated an unordered set of finite strings. ~~The Methods [illegible] There has been much successful work published on the first~~ For the first type of problem,

-0 there is a well known solution ~~involving a~~ using a universal probability distribution ~~s~~ ~~We~~ ~~obtain~~ There is an associated convergence theorem that assures us that the probability estimates made will converge rapidly to the correct values as the ~~[illegible]~~ length of the symbol string increases.

~~The present paper will give an associated~~ ~~[illegible]~~ for the second type of problem we presented a ~~[illegible]~~ kind of universal distribution ~~to~~ that would solve the problem. The present paper gives a convergence theorem for that solution.

We ~~[illegible]~~ also ~~deal~~ discuss ~~[illegible]~~ Operator induction ~~[illegible]~~

10 ~~and~~ its associated ~~a~~ universal distribution, and its convergence theorem.

31 SN On use of sequential trails ~~to~~ (w. backtracking) for predn.

In sequential trials, we try to adapt a pgm by adding onto its end.

A better way might be to try "General Modifn" of t. pgm. — ≈ "Mutation": how efficient this is, is unclear! (What I meant in 31/ by "sequential trials" was that t. trial functions ~~[illegible]~~ all used t. same prefix function. J's "Oops" does this rather nicely: his progs (never tried as that prefix ~~[illegible]~~ for, are able to request additional bits if they like).

20 : SN It would seem that t. discrete & continuous univ. d.f.'s would be very similar.

.01 The discrete d.f. assigns a pc to each finite string.

.02 T. contin. " " " " infinite (or infinite+finite) string.

If we use .01 for sequential predn, we get Cover's "extension Complexity".

Th. BIG Difference seems to ~~be~~ ~~be~~ in implementation.

If we use .02 for predicting we can use a UIO device so we can tell when a partial code is appropriate (whether t corpus is a "prefix" of it).

In .01 we often don't use a UIO machine to describe "finite objects" — The objects can have their bits written in any order. I got into this diffy when I tried to use "extension complexity" to get probys using that "dimensionality" estimator pgm for SM predn.

For any string, it gave that the string was obtained by a linear process (?±) (The BDS pgm Brock, Dechert, Scheinkman) (I don't know where t pgm is, but 7.9.93 is a date on notes): I think t pgm was able to assign a pc to any string, — but I wasn't able to get it to do proper "sequential predn" — in which I'd get pcs of part objects as I moved along in t object. It may have been that pc was a rather "global" function of t finite string. — Tho its possl that now (10 yrs later) I could devise a sequential pgm.

A stranger ~~[illegible]~~ in t SM case above, was that the prediction pc's were not very strong for t. kind of predn. I was doing! (I.e. single seq. predn). Tho I did get some good looking "yield estimates"! 0 i.e. if t actual corpus was a factor k less probable than t "default corpus" than one should be able to get a brokerage free yield of $\frac{1}{k}$ for that sequence — I think I got some attractive yields!

If we forced the discrete d.f. to use a UIO machine to ~~[illegible]~~ simulate them, it would work O.K.

It may be possl to express t. BDS pgm in an "incremental way", so t. ~~value of~~ pc of X was found, then t pc of X1 was found using ~~a~~ a modification of t. value for X & ~~of t pgm~~ modifns of parts of t. "trace" of t. calcn of t. pc of X

We want t. calcn of X1 to be a minimal addition (in pc) to t. calcn of X.

20

For §2 (Bays) use notation of Sol ($\overset{03}{99}$) 2 kinds of prob ind.

✓ Stick close to Sol 78 notation, terminology.

Section 1: Sequencial Prediction.[1]

~~[illegible] (strings of numbers)~~

We ~~[illegible]~~ are given a data sequence of n binary symbols x(n), (0 and 1), and ~~[illegible]~~ are required to find the probability distribution over the next symbol.

We assume that the ~~[illegible]~~ sequence x(n) was generated by ~~the~~ a ~~[illegible]~~ stochastic source, P(x), that is able to assign a ~~conditional~~ probability to ~~[illegible]~~ any finite string.

P(·) is ~~a~~ a sequencially normalized probability in the sense that

~~P(x0)+P(x1)=P(x), and P(0)+P(1)=1~~

$$(P(x0)+P(x1))/P(x)=1 \quad \text{and} \quad P(0)+P(1)=1 \qquad (1)$$

We also assume that P has a "finite description". Later we will ~~express~~ explain more exactly, just what this means.

~~The universal distribution for sequencial prediction is obtained by assuming that the given data was obtained as the output of~~

The Universal distribution for sequencial prediction of binary strings, is the probability distribution on

Actually a Univl dist. has been obtained for at most ~~a~~ countable alphabet. ~~The~~ Sequencial prediction ~~of~~ of an arby seq. of objects, has not been dealt w.

Perhaps as a corrolary (Modify) of QA induction?

In paper "2 kinds of prob ind." I have unc.w. [illegible] binary strings. This would be ok for a finite binary strings being "alphabet", so ternary Alphabet

Claims re: QATMs

3) uses Univ dist. — a Best ind. system

1) Incremental How induction based on past data

2) ~~The~~ optimum ~~trans~~ for learning

4) Super wrapper: can completely replace soft if just done by data

Univ. B.F.
5) can do osc MDL only

[1] footnote: ~~[illegible]~~ The ~~discussion~~ explanations and notation of Section 1 follow the ~~[illegible]~~ discussion of Sol (78). ~~The universal distribution normalized Universal Distribution is Discussed in~~ (Li v 97) pp 272–314 ~~358~~ ~~[illegible]~~ discusses sequencial prediction using the semimeasure ~~universal distribution~~, ~~and has some [illegible] discussions of the normalized universal distribution.~~ pp 315–334 discusses inductive reasoning ~~[illegible]~~ and use of the normalized universal distribution. Section 5 of the present paper ~~explains why our particular kind of normalization was used and why it gives better predictions than the unnormalized semimeasure.~~ gives some recent results that illuminate the use of the normalized universal distribution for ~~[illegible]~~ prediction

Possibly a subsection of Section 1?

: • one or more ~~[illegible]~~ members of a population of Genetic Algorithm trials. ~~[illegible]~~ Mutation corresponds to predictions using n=1; Generalized Crossover corresponds to predictions using ~~[illegible]~~ n=2.

~~If the data elements are from a language, I~~

Section 3 deals with the third kind of problem ~~is~~: operator induction. We are given an unordered set of n ~~ordered~~ pairs of elements, Q_i, A_i. We are then given the n+1th Q ~~object~~ element, Q_{n+1} and are asked to ~~[illegible]~~ give a probability distribution ~~[illegible]~~ for the n+1th ~~[illegible]~~ element A_{n+1}. [redacted] We assume that the data was generated by some unknown, but finitely describable stochastic algorithm that assigns a probability to each A_i based on its corresponding Q_i.

- The Q's ~~[illegible]~~ and A's might be questions and answers in a formal or natural language.

~~The [illegible] might be [illegible]~~

- The Q's might be the inputs and the A's the output of some unknown stochastic device (but not the same ones) — in which case Operator induction is close to "System identification".
- ~~The~~ Q_i's might be a list of characteristics of a type of mushroom and the corresponding A_i would tell if it were poisonous or not. ~~Induction [illegible] of this sort corresponds to~~ ~~[illegible] problems~~ This is usually regarded as a ~~[illegible]~~ classification problem.

For ~~[illegible]~~ each of the three prediction problems we will show that the associated universal distribution converges rapidly to the generating distribution — i.e. its probability assignments to the predicted elements ~~[illegible]~~ become very close to those of the generating distribution as the number of data elements, n increases.

While this would seem to give an ideal, almost perfect solution to most induction problems, the road is not very direct since all universal distributions are formally incomputable. Section 4 will ~~[illegible]~~ show that the incomputability of the universal distributions does not ~~[illegible]~~ limit them from use in ~~predictions~~ obtaining accurate predictions.

omit Section 5 will ~~[illegible]~~ give some ~~recent~~ results on the very rapid rate of convergence of the "undefined" output of universal distributions.

The incomputability of universal distributions is associated with our inability to know, in general, whether a particular universal machine will eventually ~~[illegible]~~ stop ~~[illegible]~~ (Turing's "Halting Problem"). Section 5 gives some new results showing that as the sample size, n grows, the probability of a universal distribution being undefined because of the halting problem, approaches ~~zero~~ ~~very~~ rapidly.

6/12/03

Nips

.10: • Insights for .225.175: The Q,A pairs can be questions and answers (not necessarily correct) ~~[illegible]~~ ~~[illegible]~~, or the inputs and outputs of some unknown stochastic operator (The "identification problem") or ~~they might be~~ Q might be a list of characteristics of mushrooms and A could be the label "poisonous or edible" (The categorization problem). Q is a problem description and A is an algorithm that successfully solves the problem. (They might be "input" Q problem A = prob. soln)

~~a~~ A better way: just list the possibys in separate lines w. balls "•".

Initially Give section nos. in which each kind of prob is treated (§'s 1, 2, 3)

.07 Int. abstract (intro do tell about discussions of incomputability & how the univl. d.f.'s are actually approximated. How much of this I will want to write, is unclear, at present.

It is possl. that I may not want to spend so much time on .07 ##.

For discussn of (Multidim. Gen. n. of sequential prediction.) I could ~~[illegible]~~ Gen. ~~to~~ "prefix set" — or just a "prefix". Define "M strings" (Multidim strings) —— M strings need not have data in all of its "cells". We can have large finite alphabet ~~or~~ or ~~[illegible]~~ countable alphabet.

X_1 is a prefix of X_2, if: If a cell of X_1 ~~[illegible]~~ has a certain symbol, then the corresponding cell of X_2 has the same symbol. X_2 may have ~~additional~~ symbols in cells with X_1 has no symbols.

Use a UMC w. no "stop" state, so no erasure for output cells.

No point to use anything but a unidirectional random input tape.

"unmarked "Section"

So we resume with the NIPS Paper, 228.02: Title: Three Universal distributions and Their Convergence Theorems.

(Introduction) We will describe three kinds of induction problems, and will associate with each, a kind of universal probability distribution that can be used to solve it. Each such distribution has a convergence theorem that ~~gives bounds~~ quantifies the precision of these induction techniques.

In section 1 ~~[illegible]~~ deals with The first kind of ~~induction~~ problem: sequence extrapolation. We are given a data sequence of n ~~objects~~ elements — they may be strings and/or numbers —. The problem is to give a probability distribution on ~~the possible~~ the unknown, n+1th ~~object in~~ element. ~~the sequence~~ We assume the data was generated by an unknown stochastic source that is describable by a finite sequence of symbols. ¶ The sequence might be: the daily values of a ~~[illegible]~~ stock or Bond ~~financial instrument security~~, a set of meteorological parameters for a sequence of days at New York City, pictures of a particular person taken yearly, ~~the values of an unknown mathematical function~~.

"Ball" type o list

Section 2 deals with the second kind of problem: ~~Called "Bag induction" or Grammatical~~ Variously called "Bag induction" or "Grammatical Induction". We are given an unordered set of n data ~~objects~~ elements and are required to find a probability distribution over the ~~[illegible]~~ unknown n+1th ~~object~~ element. ~~Again~~ Again we assume the data to be generated by an unknown but finitely describable stochastic source.

The set of objects might be:

- A set of acceptable sentences in some ~~[illegible]~~ formal or Natural language.

Nips

00 : SN Q: how to divide up into better. Intro & rest of report. e.g. Induction in Sequence extrapoln., how much is in "Intro" v.s. how much in "Section 1"?

02 Restart: We will ~~[illegible]~~ describe three kinds of induction problems ~~[illegible]~~ — how they are solved using Universal distributions and give theorems affirming the accuracy of these methods.

The first kind of induction problem is sequential extrapolation. We are given a data sequence of n objects (these are strings and/or numbers), and we must give a probability distribution on the ~~[illegible]~~ unknown n+1th object. We assume the sequence was generated

10 by some unknown stochastic source ~~[illegible]~~ that is ~~[illegible]~~ it describable by a finite sequence of [redacted] symbols.

15 → see .20

In the second kind of problem, we are given as data an unordered set of n objects, and we ~~[illegible]~~ must give a probability distribution over the n+1th possible object. We assume the data to be generated by an unknown stochastic source of finite description → .25

.135 → "Operator induction"

.14 In the third kind of problem, we are given an unordered data set of n (~~[illegible]~~ object pairs, $Q_i A_i$ We are then given the n+1th "Q" object Q_{n+1} and we must give a probability distribution for the n+1th "A" A_{n+1} object.

.175 → ~~[illegible]~~ We assume the data was generated by some ~~[illegible]~~ unknown stochastic algorithm, that assigns a probability to each possible object A, based on its associated Q object.

20 (Insert for .115) The sequence of objects can be the daily values of a stock, or ~~[illegible]~~ a set of meteorological parameters, or ~~[illegible]~~ pictures of a particular person taken yearly ~~[illegible]~~, ~~[illegible]~~ or the ~~[illegible]~~ values of some unknown mathematical function ~~for successive integral values of its [illegible] argument.~~

25 Insert for .135 The set of objects might be a ~~[illegible]~~ set of acceptable Sentences in some natural ~~[illegible]~~ or ~~[illegible]~~ formal language, a set of pictures of people ← not good example; usually contains noncriminals also. [illegible].

problem ⊗ → known to be criminals, } I wrote list of Grammar discovery probs "BAGs", w. only positive ~~[illegible]~~ data. [Actually Need for Gramm induction is rare: OA is much commoner.] → What about Mutations, Crossovers? for GA? ssz=1, ssz=2.

Good idea! When I'm writing & I get stuck at a point for a while, write down just what the problem is, & continue on. Keep at all times a list of unsolved "problems" & work on one when I have an idea about it. I think this will save time in writing

30 (& in other prob solving as well! ~~working out [illegible] what the problem is, often solves it!~~

SN Re: GA: If I were able to get a set of ~~[illegible]~~ cands w. pd. over expected G, and time needed to create & test, we could do more intelligent GA! Also do it w. ssz = 2 or 3 or more → "many". We should try Cands in approx $\frac{\Delta G}{\Delta T}$ order. (Woops! This assumes "linearized G", … so we need to linearize G "first".)

SN in .10 & .14 perhaps mention "self delimiting" & ~~[illegible]~~ "prefix" ideas. } Not yet! I'm just describing the problem, not the soln!

Nips

20.mb

~~Introdn.~~ ↓ Maybe abstract

Those given by the

We describe three kinds of prediction problems; with each is associated a universal probability distribution that enables the solutions of problems of that type. ~~Associated with~~ Each of the distributions has a convergence theorem that sharply limits the difference between the probabilities as given by the universal distribution and the unknown probability distribution that generated the data set.

All universal distributions must be incomputable. This incomputability ~~is caused by the unknown~~ arises ~~when various inputs to universal~~ because we cannot know whether certain inputs to universal machines will cause the machine to eventually stop or not .... (the unsolvability of the Halting problem). We will show that as the sequences of data grow longer, the probability of not halting converges very rapidly toward zero.

~~It is fortunate that~~ ~~Fortunately~~, the incomputability of universal distributions ~~usually has no~~ ~~little effect on their use in practical prediction~~ rarely inhibits their ~~does not inhibit their successful~~ use for practical prediction.

Introduction:

We will describe three kinds of universal probability distributions ~~used for prediction~~ and their uses for prediction.

The first, which is dealt with in Section 1, has been called (L&V '97) "The Continuous universal distribution" It is the probability distribution induced on the finite and infinite output strings of a universal computer (e.g. a universal Turing machine) having random input strings.

If $P_M(x)$ is the universal probability induced on finite binary strings x, using Machine M as reference machine, then the conditional probability (knowing x) that x will be followed by the symbol, 1, is

$$P(x1|x) = \frac{P_M(x1)}{P_M(x1) + P_M(x0)}$$

$x1$ being the concatenation of x and 1

If we have a finite string, x, n bits long, that has been generated by a stochastic algorithm having a finite description, then this stochastic algorithm will ~~associate~~ give a probability for each bit of x. Similarly the universal distribution will give a probability for each bit of x. The first convergence theorem tells us that the expected value (with respect to the generating algorithm) of the sum of the squares of the differences in bit probabilities for the stochastic vs. the universal distribution, will be bounded by a constant. Since this sum is independent of n, the length of x, it is clear that this difference must decrease more rapidly than $\frac{1}{n}$.

[redacted]

6/8/03

NIPS

Three Universal, Inductive Distributions and their Convergence Theorems,

Conv. Thms Paper: .00

0 : 202.40 : Title: Convergence Theorems for Universal {Induction {techniques, models}, Probability Distributions}

.01 [≈SN] Is t. Operator Induction problem Covered by "BAG induction"?

At first Glance: "Yes": In Bag induction, we ~~are~~ are given part of an object (this "part" can be ~~Ξ~~ Λ(null)) — to get PD over rest of object.

The Q_i A_i's can be regarded as parts of t. object Z_i ($Z_i \doteq Q_i, A_i$).

.08 Z_i (usually will / can) have ~~E~~ punctuation, say comma, betw. Q_i & A_i.

0 Furthermore, sequence predn. (& continuous distribn) can be regarded as a special case of bag induction, in which one or more of t. objects are allowed to be infinitely long.

Data can be whole or (~~usually~~ always for infinite objects), part(s) of objects.

Variations on a Theme Suggested by Wallace. A unified theory of prediction'.

The QATM (op induction) is a special Case of BAG induction, it does Seem different: T. solution I'm using for QA seems to take a different form from ~~that~~ suggested by BAG Induction. i.e. we feed in ~~Q~~ Q_{n+1} & we directly get a d.f. over A's or hyps pc's first.

In Bag induction, we usually get a D.f. over t. set of "next objects"'s

20 T. Difference isn't very much, hvr, because in QATM we often have part of a A_i to complete — & probly in BAG induction, given part of ~~a~~ a BAG element, we can ~~ask~~ ask for P.D. on t. rest of t. ~~el~~ element.

This "completion" can give a soln. to t. sequential predn. problem.

There may be some differences in QA v.s. BAG: In QA, we are given t. Q's: In BAG, we have to code t. Q's as well as A's:

27 → In QATM there is no info in t. Q's. In BAG, Q's have as much info as A's.

I think I thot about .27 a lot before doing present QATM Model.

Superficially, it would seem that in QATM corpus, Q & A are symmetrical: that one has as much "info" about t. Q's as about t. A's. 0 Anyway t. discussion of .01 – .08, on how QA is exactly covered by BAG induction — seems Rite! Hvr in QA induction, T. TM would not have any idea as to what to do if given a null Question, or partial Question. BAG TM would be able to deal w. that, [redacted] hvr.

2) It uses the universal distribution (or — more exactly, approximations to it) (Lsearch) to do induction. As I have mentioned, this is a very good, very general method of induction.

3) To search for models and to search for good, relevant problem solving techniques, we started by using Levin's universal search algorithm. in ~~in~~ what I call Phase 1 — For the ~~conditional~~ information state that we start out with, this is close to the most efficient ~~[illegible]~~ ~~[illegible]~~ search technique possible.

In a later stages of learning, which I call Phase 2 ~~we have different kinds of information available, but through~~ [illegible] it acquires different kinds of information and ~~it~~ uses a ~~[illegible]~~ search technique that is more ~~better than Lsearch, for that kind of information~~ matched to that kind of information. I expect it to be much better than Lsearch.

4) At all stages of its learning, the searches are guided by a conditional probability distribution.
~~The~~ Rapid success of the search is ~~critically depen~~ much dependant on ~~the~~ how good that probability distribution is. The system ~~[illegible]~~ executes the ~~updating~~ modification, the "~~[illegible]~~ "updating" of this ~~probabi~~ guiding probability distribution as one of it normal induction problems. It's a real "Bootstraps" process, so that the method of improving this guiding probability distribution, improves itself as the distribution improves.

5) The system is able to relate any ~~[illegible]~~ problem domain to any other problem domain. If the problems in the two domains have common elements — structural, analogic, or whatever, the system is able to use this commonality for induction.
It ~~[illegible]~~ is an idealized kind of "transfer learning".

21: 2.24.39: insert; ~~Also~~ In inversion problems,

"My main Goal . . . B' 5.36 – .38 :
~~Some early papers I wrote I wrote some papers~~ In '86 and '89 I gave a preliminary design for ~~[illegible]~~ a system designed to learn to ~~to~~ solve difficult problems. The present talk is a progress report on that system.
To pique your interest, I will list a few ~~[illegible]~~ of it's more exciting properties ~~of the system~~.

("insert"

Here's how it works: ~~B~~ B' 6.34 – 7.04.

Then discussion of a few details of the system.

Printup the talk up to here ← give talk
see how much time I have left.

Try to get entire talk printed up in Big type.

Check on Slides :

The main thing needing retyping is Seminar

Also ~~10~~ 15 lines from A .05 – .22

5.27.03

NIPS

For example, we have to design a ~~car~~ having ~~certain~~ specifications. We want the cost of the car to be small, but if ~~the design is completed after to~~ it takes too long to complete the design, its value ~~to~~ to the ~~[illegible]~~ corporation will decrease in a known way, because of emerging ~~[illegible]~~ innovations of competitors.

For a long time I was unable to obtain a useful solution to the time varying G problem, but ~~about~~ a few months ago / I found ~~obtained~~ what looks like a useful solution.

~~At the risk of [illegible] 89, I [illegible] wasn't~~

Induction problems can be of at least three kinds:

1. The extrapolation of an/ordered sequence of symbols and/or numbers: #Time series# extrapolation.
2. The extrapolation of an unordered set of finite ## objects## — these are strings and/or numbers,
3. The extrapolation of ~~[illegible]~~ a set of unordered pairs of objects.

SN ~~[illegible] 3 [illegible] is clearly a case of 2; 2 is clearly a case of 3:~~ If thought of in this way, it may be that the conv. theorems will say what I want them to say. —

But the conv. theorem for 3 is significantly different from that for 2! The Q's in 3 ~~enter into the coding cost in a gut~~ don't enter into the coding cost — the they would if ~~2 is the~~ (3) is regarded as a special case of (2).

Though (3) is a special case of (2) I have found it more useful to consider (2) to be a special case of (3).

~~→~~ 'My ~~[illegible]~~ main goal (B'5.36–.38) Strong A.I.

~~The approach I will describe~~ It is clear that we have to get an enormous amount of information into a machine before it will be very intelligent. One approach, by Lenat, ~~[illegible]~~ is to carefully program "common sense" into the machine, so it can eventually learn by reading books and/or ~~[illegible]~~ the internet. My own approach has been ~~to teach the machine how to do inductive inference, then give it a sequence of problems of increasing difficulty~~ more like the way I believe humans ~~[illegible]~~ acquire ~~[illegible]~~ knowledge — ~~by~~ by / inductive inference on a suitably designed training sequence.

There is a ~~trade off between~~ ~~the amount of information in the apriori distribution and the amount of information in the training sequence~~ the case wi

~~[illegible]~~ To motivate ~~your~~ interest in ~~the rest of the talk~~ and perhaps to get people to actually ~~[illegible]~~ ~~to~~ read the report, I will list a few claims for the system.

1) It is able to/learn to solve both inversion problems and time limited optimization problems. These problem types cover almost all, if not all problems in science and engineering.

Inversion problems ~~are~~ are
Time limited optimization } see B'6.26 & 6.28. the P and NP problems of Computational Complexity Theory.
include all kinds of inductive inference, and surface reconstruction. All problems

5.27.03
Nips

~~222~~

can't find 222
does it exist?

Well defined problems;

Discuss Inv probs & OZ probs: Well defined probs are Inv. probs.
OZ are 2 cases; we don't know we have best soln, but we have no. that tells us how good we are.

0 : 221.40 : Claims list: 219. 28R tf: (3 claims) 1), 2), 3),
looks like way humans work.
is very general prob. solver.
was v.g. induction method.
induction method.

4) ~~It's~~ V.g. transfer lrng.

List kinds of induction problems it can solve; sequential time series
~~Bin~~ induction: Languages (Natural & artificial), clustering, categorization probs
if several clusters and data are given it becomes a QA:
Q.A. induction: ~~#~~ Operator induction, System identifn, Categorization

Another talk would be to say that sol 86, 89, ~~94~~ Paul reported earlier work on this system —
This talk is a progress report — since then.

The system I will describe is one that I ~~[illegible]~~ first wrote about in 1956 ··· —
It was called "An inductive inference Machine".
~~[illegible]~~ ~~There were important deficiencies in that~~ Its success was limited by my incomplete understanding of the inductive inference process.

After 1960, I mainly worked on how to understand and apply the universal ~~dist~~ distribution to practical induction.

Not well written →

In 1986 and 1989 I wrote ~~[illegible]~~ papers describing ~~my~~ ~~[illegible]~~ my progress in this area. ~~a~~ ~~[illegible]~~ It had evolved into a ~~[illegible]~~ system to ~~[illegible]~~ learn to solve ~~very~~ general problems. In those papers I was concerned only with two kinds of problems: INversion problems and ~~[illegible]~~ time limited optimization problems.

Inversion problems are ~~[illegible]~~ "well defined problems" ~~of [illegible]~~ the P and the NP problems of computational complexity theory. —

They include solving equations, ~~[illegible]~~ constraint satisfaction problems, symbolic integration, theorem proving and ~~[illegible]~~ traveling salesman problems.

~~[illegible]~~ Time limited optimization problems include all kinds of inductive inference problems, surface reconstruction, ~~research planning, [illegible]~~ devising scientific theories/ ~~etc~~ research planning, ~~[illegible]~~ ~~etc~~ act. Designing an automobile ~~[illegible]~~ in 6 months, satisfying certain requirements and having minimum cost is ~~another~~ an example.

30 ~~Real optimization problems~~ Most practical ~~Tmp~~ problems in science and engineering are time limited optimization problems. They are of several types: New line

• In general we are given a function G(·) that maps strings and/or numbers into real numbers. The problem is to find an argument X, for G(·) such that G(X) is maximized — and we only have known time T to find this X.

~~Some [illegible] kinds of this problem~~ ~~Variants~~ variations on this theme:

New line
• There is ~~uncertain~~ " " G( ). New line
• G(X) takes a long time to evaluate, so we want to use a few ~~[illegible]~~ carefully selected trials for X.

• There is no time limit but G is a known decreasing function of computation time T, as well as of X.

5.25.03

Nips

Continues (in a sense) discussion of "T. measure of $P(U_i)>0$" (.00)

contd 222

0:220.40: 1 Superficially, it would seem that any M likely to be a model of a real world phenomenon, would be an ordinary semimeasure and be Normal. (i.e. not needing normalization). Does this make it a CPM ?

A quick reply would be that partial recursive p.d.'s are then pc≠0: but still, Approximations to part rec. functions could be good models of processes that take a long time to generate (in RW). — i.e. lots of cc. — Not nearly ∞, but large enough to be beyond the computing capacity we have. Could be in Geology & Economics (ESM).

P30 1015 1130 1835 310 9P.

SN On the "hyper chaoticness" of a gas of molecules: I found that the uncertainty in one position component of a molecule increased by a 100 to 1000 factor for each collision.

Hvr, how does this face up w. Liouville's theorem, that keeps the volume of a region of state-space constant as the system evolves? Maybe related to the manifold containing the system evolving into an extremely "fractalish" shape!

(218.00-.07) : Introduction or B' 1.02-.05 & just → Intro

a) B' 1.20-.29 — what is the UPD & how is it used to predict.

20 : 219.40 → A 1.05-.32 ~ how good it is: Convergence.

b) ■ B' 3.37-4.29 Incomputability

c) ■ B' 4.31-5.33 subjectivity

d) ■ B' 6.22-7.04 Roughly how TM works.

219.28 R ff "CLAIMS." Also: kinds of problems solved. : also kinds of induction problems solved.

Maybe "claims" before end....

e) C - 6.04-11.42 how TM works: Details (Explain about "Phase 1" & "Phase 2"

f) OOPS

I ~~mean this~~ expect this model to be developed into a rather complete prob solving system — for just about all kinds of well defined problems.

In reality ..: I have described the evolution of the sprfd in a person as he solves problems in his life — an ongoing "updating of the corpus".

The system I will describe this is a uses a ~~...~~ model ~~...~~ for its cognitive ~~...~~ development during ~~...~~ its active existence.

Claims can start out w. claims for UofPD; then claims for TM, ~~...~~ — omit earlier claims of B' — put them in here.

Then d) B' 6.22-7.04, ruffly how TM works, then some details from C then OOPS.

Another possy is to leave all claims about UPD where they are now (3.28 ff in B'), then have special section for claims for TM — later.

5.25.03 220

Nips

$\boxed{\text{(Augmented) PST Update}}$ .00

(Continued Normalzn .20)

10 : "$\boxed{\text{PST update}}$" This includes $\boxed{\text{both}}$ updating of ~~.~~ G distributions of old, ~~&~~ PSTs & t. generation of new PSTs.

I had a good idea on how to do this, but I've forgotten it & lost reference.

I can perhaps write out a kind of "Theoretical" "Formal" soln:

06 Given all past (prob$_i$, Pst$_j$, T, $\binom{\text{Fail}}{\text{Solve}}$) quads, to extrapolate to

07 ~~PST~~ (prob$_k$, Pst$_j$, T, Solve,) = probab of or $P(T_{\text{solve}} \mid \text{Pst}_j, \text{Prob}_k)$
↳ a specific problem; ranging over all problems, (like "A" in QA problem solns)

context 217.00
Pst update 220.00

192.00 – 194.25

Just as in solns to QA probs, I want to know Pst$_j$ "values" that are particularly good:

09 i.e. " $\dfrac{\text{~~harmonic~~ zeroth moment of "T" d.f.}}{(\text{first } " \;\; " \;\; " \;\; ")^2}$ = max $\left(\approx \dfrac{\text{prob of soln}}{\text{Time of soln}}\right)$ = max.

0

So .09 is "what I'm looking for".

Next of importance: Say trainer inserts a bunch of PST's into t. system for "consideration". How should TM best use this info? I assumed that TM should "factor" t. set of PST's, & make a grammar for them — (that Trainer could help by (partially perhaps) prefactoring t. PST set.)

A possl. way for this to work, Trainer gives TM a small set of (perhaps factored) PST's. TM ~~[illegible]~~ tries them on its set of problems, so it can get data for (.0~~6~~). The "factoring" of t. set of PST's is used to help get solns (or trials) for [redacted] (.06–.07)

.0 : 215.40 $\boxed{\text{On Normalization:}}$

What Levin probly showed: While any Universal semi measure > c · (any other ~~[illegible]~~ finitely describable semi measure), some semi measures have ~~[illegible]~~ normz consts that → ∞ as n → ∞, so ~~[illegible]~~ their Normd forms would not be dominated by a univl ~~[illegible]~~ semi measure.

For there to be no mult. dominant univl. measure, this means that every univl. ~~[illegible]~~ semi measure must have at least 1 seq. for which t. normz const → ∞ w. n.

~~Also, default these seqs must be difent. for difent universal measures.~~

On t. other hand, if we assume that t. seq. being predicted is very long (tho we only know t. 1st 1000 bits or so) then μ is unlikely to have a normz const that diverges, since then $\Sigma p_i^2(v)$ diverges ∴ probly $\Sigma\, p_i(v)$ also diverges — which means it

0

is very unlikely that this seq. did not terminate! In fact, $\prod(1-p(a_i))$ is t. probty that the sequence did continue — & it would seem to "rapidly" (?) → 0 if t. normz constant $\left(\prod(1-p(a_i))\right)^{-1} \to \infty$. ⑥

So, it is likely that μ does not have an infinite normzn constant & ∴ that normalized ~~[illegible]~~ μ is mult dominated by any universal semi measure. If μ is not finitely normalizable, it is likely that it would have stopped (if ~~[illegible]~~ we have a long data seq).

? What if μ is a normed semi measure w. an infinite normzn. const.? — Then no ~~[illegible]~~ Universal measure . . . (?) Much Confusion about this!

221.80

On t. other hand, note that $\prod_i(1-p(a_i))$ is a posterior probability, not a probability ratio, so perhaps it is of no significance!

5.23.03

NIPS

0: 218.40: Perhaps better approach to writing the talk/paper:

Write it up ruffly, w.o. neccessarily good "turns of phrase". ~~[illegible]~~ then mark the parts that need rewriting, & work on them. This way, I will quickly have a usable talk, & I can slowly make corrections if needed.

→ Start out by telling that ~~[illegible]~~ the talk is = T. Univ. Dist. & Mach Learng — ~~[illegible]~~

213.21 – .27 is main outline: + "A Prog. report" . . . (much longer) } at my website http://world.std.com/rjs/...

Other things written: Ⓐ ① Nips talk
Ⓑ ② kol. talk Ⓑ' kol paper
Ⓒ ③ summary at R. Holloway,

Present plan: ~~[illegible]~~ Intro: Accuracy, incomputability, subjectivity
~~[illegible]~~

0

Intro: 218.00 – .07

Discussn of what is UPD?: B 1.15 – .30 or B' 1.20 – .29.

30 Accuracy ~~[illegible]~~ T. conv. thm & its recent genzns: A 1.05 – .32 | B' isn't so good — needs a little rather vague

Incomputability: B' 3.37 – 4.29

Subjectivity: B' 4.31 – 5.33

36

In the next part, I want an introduction that will motivate their listening to ~~the~~ some of the tech details that will follow — & also motivate

40 their reading of the Report.

So there will be a connecting discussion "Rosetafe" betw. B' ~~[illegible]~~ 5.33 and B' 6.22 ~~[illegible]~~ → 7.04 ~~[illegible]~~ which is the/a dem of "lites nontechnical" ~~[illegible]~~ TM.

I will probably want to include stuff from Ⓒ summary in this ~~section~~ latter section → 6.04 – 11.42 entire summary

W.r.t. "summary", I is assumed that almost all of audience heard the ~~[illegible]~~ "Kol Lecture", the previous day.

So perhaps for "Rosetafe", use mainly the Kol lecture B' 6.22 – 7.04

28 perhaps modified to emphasize the idea of "the evolution of the approach" .28.40

Actually the discussion in B' (6.22 – 7.04) is quite short & might ~~[illegible]~~ in its entirety, be used in the talk — perhaps augmented by the

"Claims" of .28.40 ff.

Be sure to mention "Well Defined Problems"

Expand "Claims" by giving Examples. ← The "CLAIMS" ~~[illegible]~~ section

is a very impt. part of the Lecture..

So spend a fair amt. of time on "Claims".

Also discuss "OOPS" — a partial realization of part of Phase 1 (i.e. it's ~~only~~ only deterministic "predn")

If we make a code for a corpus and that code is not "random" then we can recode it and compress it.

Q: if a code is not the shortest code for a corpus, must it be non-random? — must it have regs(s)?

If this is true, then coding & recoding will always work!

Counterexample: The code is A followed by B.

A is a true min code of corpus — when A is executed, the corpus is printed & machine stops.

A is Random. B is any other random ~~[illegible]~~ seq. So code for A B is random & is a code for the corpus — that is not at all ~~[illegible]~~ minimal

.28.40 ① Uses best model of induction that we have.

② Uses best model of human ~~[illegible]~~ problem solving that we have.

③ It is a fairly General probsolver.

~~I~~ At first I ~~[illegible]~~ thought that there were some kinds of probs to which it could not ~~be~~ applied — but as time went on, I found ways to apply these methods to their solns or approxl. solns

5-23-03
NIPS

SELECTIVE BACKTRACKING : .31

D:(Spec 213.27): Title: T. Univl. D.P. & M.L.

I'm going to talk about two things: First, the Universal ~~[illegible]~~ Probability Distribution. and some of its properties.

Then I will ~~[illegible]~~ tell how to ~~apply the universal distribution to~~ use this distribution ~~[illegible]~~ for a very general ~~kind of Machine learning~~, very powerful ~~kind of~~ "Machine learning.

BZ
X8 ~~[illegible]~~ Three critical properties of the Universal distribution are

1) Its accuracy
2) Incomputability
3) Subj[illegible]tivity.

~~Then~~ When I first discovered the distribution, the main question was whether it worked at all. There were various heuristic discussions, and approximate applications that suggested it ~~[illegible]~~ would give good results

The universe

List ways that ALPHA is better than other systems.

Re: "Boost" 2 scaling factors: 1) In mature machine, pc's of tokens will be about 1/1000 so "factor of 1000". is much smaller (always <1) 2) In a real system, pc of "Boost" ~~[illegible]~~ instructions is much lower due to no. of successful, frozen 'funs'.

D: .. Expo: (special): While the incomputability of the Universal Distribution usually has no ~~practical~~ ~~implications~~ ~~implications in~~ relevance to practical induction, it does have something to say about Science as a continuing adventure: ~~"Many Scien~~
It "Many Scientists . . .

~~[illegible]~~ We can never be sure that if we spent 10 more minutes searching for a better theory, we ~~we~~ would not find a much, much better one than the ~~one~~ best we have now.

~~This~~ This is not at all bad ~~for~~ ~~those~~ it means that nature will ~~[illegible]~~ For those of us ~~of~~ of an adventurous state of mind continue to be a neverending source of joy in discovery — That the peak of Scientific ~~[illegible]~~ invention will never end!

D

31 On "BACKTRACKING" I have been assuming that backtrack can always be characterized by the "No. of problems Back" one Goes. Not so! A better way would be to "selectively backtrack", to violate the sequencial order in which the TSQ has been given. For early TM, this is not such a good idea! The TSQ ordering contains much useful heuristic info. For a more mature TM, the TSQ is not being so carefully constructed (& may be largely selected by the TM itself) — in which case, "Selective b'king" is in order

5.22.03

Nips

CONTEXT: .03

.0: ~~[illegible]~~ :::: FN If we restricted our UPD to prim rec. funcs, would it be "computable"? If so, then if we had a convergence thrm, we could falsify it w. an "infeasible funct" — would such a funct be prim. rec? — superficially, it would seem so / i.e. T. generation of such a function would seem to be very "straightforward". Hvr., it does involve deciding if a certain real no. is > or ≤ ½ — which can be undecidable in some cases — i.e. ~~[illegible]~~ normally UMCs can't compute discontinuous functions. All we have to do is compute first bit of binary expansion of pc. If one could do this, t. function is partial recursive (not primitive recursive). We also have to decide if t. zeroth bit, i.e. (a bit of binary pts) is 0 or 1.

.03 FN Context: "Discn of context" in 138.00 (also 133.02, 140.16–.32) may be adequate start. Context is used in Phase 1 (perhaps only phase 1) - type of reasoning — i.e. t. apriori pd is for "similar problems in t. past" — not for optzn.

.06 It is possible that in t. search in which context is introduced, I "theoretically" use t. same apriori pd as ever, but because of t. ~~[illegible]~~ search technique effectively imposes an additional bias (of an uncertain nature) on t. resultant apriori pd!

A remembrance of my ~~idea~~ (More recent) ideas about "context":

1) A somewhat General idea of context: We are doing trial codes to derb a corpus: T. pc of a Token ~~[illegible]~~ can be indep of context. (This would be a Bern. model for t. set of tokens) — Or it could depend on all of t. tokens thus far, or on f(n) of t. tokens thus far, or on ~~[illegible]~~ ~~[illegible]~~ them a/o on a larger context — t. "nature of t. problem being solved" (ranging from ~~[illegible]~~ a very general or "OZ problem" (e.g. "Chemistry") to a narrow "verbatum" dcrn of t. current dcrn of t. problem (e.g. "Chemistry") to a narrow "verbatum" dcrn of t. current problem.

More generally, ~~[illegible]~~ this context could ~~[illegible]~~ help control not only t. pc of t. next token, but t. pc d.f. on t. entire sequence of tokens to follow.

.0 FN If $\sum p(U_i) < \infty$ then $p(U_i)$ is normalizable! Is this at all a useful way to look at $p(U_i)$? T. normed $p(U_i)$ would be t. relative pc's. $\sum p(U_i)$ would be t. total no. of "hits".

(continued): IN AZ(4), t. only context used was freqs of tokens, ~~[illegible]~~ on t. ~~[illegible]~~ previous present partial code. — Also, we could define new tokens as we coded along. More generally, we could .251 augment t. AZ language by including any new kinds of "Legit" regys that we've found (Legit means t. regy reduced total pc).

.25 It may be that context is only relevant to ~~[illegible]~~ devising "induction codes" i.e. {Short codes for a corps} — Not relevant to devising conditional p.d. over PST's (for a particular problem)?

.0 Each new kind of (productive) regy found in t. "corpus up to now" may need a modifn. of t. language .251 used to derb. t. corpus. So we have a "Metalang." that tells us how we are allowed to do this.

Is this a "TM₂-like" idea (i.e. t. Metalang)?

.4

.1 From t. foregoing (limited) analysis: Contexual regs are always a kind of "re coding": a finding of regys in a code. One then attempts ~~[illegible]~~ re-re-coding & re⁽⁴⁾-coding.

Hvr., ~~[illegible]~~ t. args. of .00ff should be written more Carefully, so that a person not familiar w. t. ~~[illegible]~~ proposed system, could understand — (i.e. Me of Now + 1yr!)

NEEDS REVIEW!

5.22.03

Nips **Rev**

ID 612.00 on "Anytime" Problems,
190.09 : "Things To Do" (Re: TM).

Also Note "STACK" of ID 611.00 ff

0: : A list of some impt. ideas I may have lost: Previous list 192.00 [redacted]

1) [redacted] Conds under which {WON (OR) Net} result ≡ Lsch.: What functional form G(·) must take.. Was simpler equation. ⋮ One easy presentation: Give ϕ, 1, 2 moments as function of cpty. of G distribution.

2) **Discussn of "Context"**: Just what it is: its Genens, & how it is to be used for "Updating". What is the Updating Genc & how does "Context" info help optz it? 133.02 is ref to Context, also {139.00} ← (This is a good idea!), 149.16 – .32: (Note 217.03)

3) I got a good understanding of the Gencs for discrete & continuous predn: just how narrowing of the width of continuous d.f. was equivalent to ~~f~~ pc density of soln. (If the choice involved was at tip and of "f.d.f.") ← usually true. 192.20 – 194.25 is early discussn. It was put on the Stack! ID 611.38 (#4) || Perhaps adequate soln! 186.16 – .35 187.20 – .23 || 186.70 – .35 is mem. agent (192.20 – 194.25)

4): Anthropic Principal: 213.24R ; 214.00 – 215.10

5) 190.09 THINGS TO DO (Re: TM). I scanned from 216 back to 190. ... continue scanning: ~~try to~~ Make index of ~~...~~ impt. ideas.

5.21.03

NIPS

www.cursor.org

Antropic Principals.

10: 214.40: Actually, 214.32 is maybe t. key to t. "Antropic Principal." That t. low pc. of any particular kind of world, — be it epistemological ("initial" a priori) or t. constants of t. universe — that this low pc is not important. Any state of t. Universe is of very low pc. T. only interesting things are not pc's but ratios of pc's (214.20) — which are used for prediction & for assignments of pc's to theories (or possible "causes").

So it would ~~seem~~ that t. Anthropic principal is not of interest: it does not give preds. — it is not "falsifyable". That this is true for Epist ~~[illegible]~~ as well as Physical Anthropic Principals

08: 214.26: "Using t. a priori hypoth to Evaluate itself": It May be that there is $\boxed{\text{No Alternative}}$!

09 So t. "Anth principal" answers t. Q "why is t. ~~[illegible]~~ pc of t. universe so small?": This Q should not be asked, since ratios of pc's not pc's are of interest.

0

15: 211.40: A simple proof of ~~Σ~~ both $\sum u_i$ & $\sum u_i^2$:

1) $-\ln(1-u) > u - \frac{u^2}{2}$ by 211.19-.22

2) $-\sum_i \ln(1-u_i)$ ~~[illegible]~~ $< -\ln p_\mu$ by 211.02

.19 3) from (1 & 2) $\sum_i \left(u - \frac{u^2}{2}\right) < -\ln p_\mu$. Well! not such a breakthru! I still have

Note! no E_μ needed!

0

to know that $E_\mu \sum \frac{u^2}{2} < -\ln p_\mu$ in order to get ~~[illegible]~~ useful info out of .19.

.19 says $\sum u - \sum \frac{u^2}{2} < -\ln p_\mu$; but from that alone, $\sum u$ & $\sum \frac{u^2}{2}$ could be arbly large. If I know $\sum \frac{u^2}{2}$ was $< \frac{1}{2} \sum u$ that would be enough to prove $\sum \frac{u_i^2}{2}$ & $\sum u$ are both $< -\ln p_\mu$ or ($< -2\ln p_\mu$.) — but I do seem to need ~~[illegible]~~ more info than just .19 (& t. fact that $\sum u_i$ converges so that ∴ $\sum u^2$ converges) that $\sum \frac{u^2}{2} < -\ln p_\mu$ is sufficient:

If I could show that $\sum \frac{u^2}{2} < -\ln p_\mu$ (No E_μ) then this implies (by .19) that $\sum u_i < -\ln p_\mu$ — a much stronger result — But it seems unlikely to be true. e.g. say t. sequence to be extrapolated was a random seq., & μ was some simple deterministic seq. ... like 1^{000}. In this case, I would guess that ~~Σ~~ $p(u_i)$ would be ~~[illegible]~~ constant — i.e. indep of i & neither $\sum p^2(u_i)$ or $\sum p(u_i)$ would converge. In this case, hwvr, $\sum p(u_i)$ has to converge — it always converges. So $p(u_i)$ must ↓ w. i, it can't be constant.

0

Drop this for t. while!

→ 220.20

5.19.03
NIPS

B 18:42

.00 213.37R spec "Anthropic principal"! We ask: "Why is 113+217 → 330?" An explanation might show how to obtain this sum from a table of addition and carry rule". → we then predict that this method of using tables & carry rule will work on other addition problems, like "43+91", etc.

So: In Math, the tautology, "An explanation" can be a recoding into a more useful form that may reduce cc of computation

In general, having an "Explanation" in Math, does help solve future problems — & Math problems always involve "cc".

So, I may need to Extend my idea of what an "explain" is. The stuff on this of 213.24R ff & 214.00ff may be very impt.

Hmm! The Anthropic principal in Physics, → suggests a principal in Epistemology (w.r.t. apriori).

0 This linkage is a kind of "productive" thing since they are very similar "structurally" (i.e. "Analogous"). On the other hand, if both principals are "content free" it is of not much interest that they are "linked".

in .00 the explanation of "113+217 → 330" suggested a general scheme of doing addition — The only reason it wasn't "predictive" was that all addition (& all true Math) is tautology.

Drop this for a while. I can mention the Anthropic Principal at DIMACS talk, but don't need to go into its meaningfulness (tho I may figure out something by then).

18 5.20.03 Re: Anth Princ: (In Physics): It could be used to justify any theory, no matter how unlikely! (including the "God Hypothesis"). What we are interested in, is not
0 → absolute probabilities but relative probys of various hypotheses. → 215.09
In particular, at an earlier state of Physics, chemistry, we would have an enormous list of essentially "Ad Hoch" constants that define the Universe — The P.C. of that set would be very low — much
23 lower than that of the present small set of universal constants.

24 Re (.20)R (Rel pc's): we predict on the basis of rel. aprobys of Models:
Hvr. in the case of our evolution any Model of "aprip", we are using the hypoth to
26 evaluate itself. (!?) ← [This seems ~ Hume's objection to usual Method of induction.] → 215.08

So it may be that .18 ff says that the Anth. Principal in Physics is not needed, & doesn't help anything. Hvr. .24-.26 is bothersome: the applicn of this line of reasoning
0 — to the aprip. Model itself!

(.20)R is related to the "Unlikelyness of Biochemical Evolution ever producing
32 "MAN" or any similar creature. A MAIN Counter Argt. ∠.32R: Low PC means nothing: Ratios of PC's are what's important!

If ~~[illegible] .24-.26 [illegible]~~ .18-.23 is correct (note diffy of .24-.26)
Then this may solve the Main Problem of Epistemology: "Where does Aprip come from?".
Except, perhaps for .24.26 & .26R in particular.

A vague argt. for UPD: That we are born with this aprip "built in" & particularly "things that worked well in past are likely to work well in future, if ~~[illegible]~~ the models have hy "aprip." i.e. we have this aprip for models "built into us"
→ 215.00

NIPs [redacted]

.00 (spec 212.33): In discussing a need for apri info in order to learn to walk, talk, understand speech:
— Also mention people are apri able to breath air (fish can't do this) — eat certain kinds of food. (mainly milk). They have a set of sense organs that are apri known to be useful in figure in which they will live. Other creatures have senses that are more or less sensitive in different modalities (color, intensities, types of odors ...).

That the apri'd for learning is [redacted] part of the design of the organism for a certain expected environment — just as is having facilities to breath air to metabolize certain kinds of molecules, to be able to perceive certain sense modalities with certain sensitivities. Sense in U.V., I.R.; electric fields, magnetic fields, ...

.10 Inherited info is called instinct: Our apri ability to learn certain things very easily is "instinct".

.11 Perhaps mention: for many statisticians, subjectivity is to be avoided. He wants something that the Statistical Community Agrees on. / Using subjective info may involve extracting it from his client — a difficult and imprecise operation. → while it may be agreed that properly used subjective info can only help a statistical inquiry

I'm my own case, I am both the statistician and the client — so this

.18 problem does not occure.

.20 So: Just do the London talk w. follg modifns:

.21 1) Shorter explann of "incomputability" write: $P_M^T \to P_M \to \omega$
Give exposition that I now have on Website (of London talk)

2) In the discn. of subjectivity: Go into it more; 212.15 — .33; 213.00 — .10 (maybe 213.11—18)

.24 So how impt. a role of apriP is: That I ~~[illegible]~~ to allow this in my System. Perhaps Anthropic Principal.

3) To Sachunon Alpha: 210.14 — .18 → Also 212.06 — how QA covers sequencial, bag, Grammar discovery categorizing; Oney, model identification.

.27 So I can just use the ~~[illegible]~~ web version of London talk as a guide to do .21—.27

→ 218.00

.24 Main problem w. Anth Princ: It doesn't seem to be predictive / "falsifyable". Is Mathematics "predictive" — or "Falsifyable"? Math is always "Trivial" Tautologic if correct. (Carnap: "Math statements are Analytic". Math would "explain" Nothing only because what it is meant to "explain" needs "no explanation".

.30

.37 .37R → 214.00

→ 217.00

5/14/03

NIPS

Backtracking Approach: .34 (needs Work!)

(Spec)
00: 210.18: SN Perhaps main reason I don't like writing this is that exposition is such an "Ill defined Problem"!

A possl path: Discuss ~~[illegible]~~ UPD: "Incomputability"

Subjectivity: How apr evolves ~~[illegible]~~ animals, man, (apr of individual man evolves during life)

Anthropic Principal of Epistemology

Big Q is: How is ~~[illegible]~~, updating done? (How QATM is like Google)

.06 — 2 phases: ① the QA problem. This covers ⓐ sequential predn ⓑ Bay induction (hypotheses)

ⓒ categorization, Identification problem, ~~[illegible]~~ surface reconstruction. ⓓ (Engr Implementation) (Info about implementn. Google)

0 Uses of Lsrch for induction. ⓔ can it do "clustering"?

So: explain how it does ~~upd~~ Lsrch & updating.

Then explain phase 2.

Perhaps do explain about "incomputability" (also irrelevance of semi-incomputability)

So start out w. ~~a more~~ the way the London Lecture started, but more sophisticated;

15 More emphasis on the importance of "updating" (evolution of apr pd): One can do induction, prob. solving w.o. it, but no real progress

One can get Good results w.o. updating, by coding large/diverse Corpus, but this would probly Cost far too much.

Perhaps So what I may do, is give a picture of how lrng evolves in Man — how it must

20 evolve in Man (& in Life) ← (The "Pt lrng.")

One can start out w. much less apr info than a child (but it must be some minimum of info on how to lrn) Then if its small, we have to put more in the TSQ (which is usually good, but it means more trg. time & more TSQ has to be written — ... There is always this tradeoff betw. amount of apri info & info in the TSQ.

~~And~~ A way to look at "evolution of Aprpd": One can start w. apr & No data ("neonate" ~~to~~ tabula rasa.) Then do ALP on entire large corpus leading to present problem (enormous srch ~~[illegible]~~ time) or start w. ~~[illegible]~~ do small problem, update aprpd, do next problem, etc. If we spends enuf

30 time searching, we will get same result as befor. — we less srch time, not as good a soln, but acceptable srch time. (perhaps mention "Backtracking" to make evoln. of

.33 ~~[illegible]~~ apri" More realistic. If one stores many possl good codes, "backtracking" is faster (spec 213.00)

34 SN BACKTRACKING: HOWTO! Say I have 3 codes $\alpha_{1,2,3}$ for corpus up to n — of lengths α_1 2, α_2 2+3, α_3 2+4. : For coding out to n+1, try Continuations of all 3 α_i, but give relative wts 1, 2^{-3}, 2^{-4}

This can ~~[illegible]~~ perhaps be generalized to backtracking further back. ← Not obvious! Maybe False! Needs work! Hui: looks like its worth spending time on!

5.13.02

N ips

~~Expect~~ $\boxed{\underset{\mu}{E} \sum f(U_i) < -2\ln p_\mu}$ → (.09)

__Bounden__ $E \sum f(U_i)$

By .15-.19;

00 : 208.40' : ~~Similarly~~; $\ln(1-U) < -U + \frac{U^2}{2}$ ∴ $-\ln(1-U) > U - \frac{U^2}{2}$

$\sum -\ln(1-U) > \sum U_i - \frac{U^2}{2}$

$\sum U_i - \sum \frac{U^2}{2} < \sum -\ln(1-U)$

$\frac{p_\mu}{\prod(1-U_i)} < 1$

normaliz const: $\frac{1}{\prod_i (1-U_i)} < \frac{1}{p_\mu}$ ∴ $\prod_i (1-U_i) > p_\mu$ ← $\sum_i \ln(1-U_i) > \ln p_\mu$; $\sum -\ln(1-U_i) < -\ln p_\mu$

$\sum U_i - \frac{U_i^2}{2} < \sum -\ln(1-U_i) < -\ln p_\mu$ so $\sum U_i - \frac{U_i^2}{2} < -\ln p_\mu$ ∴ $E \sum U_i - E \frac{U_i^2}{2} < -\ln p_\mu$

also $E \sum \frac{U_i^2}{2} < -\ln p_\mu$ → __$E \sum U_i < -2\ln p_\mu$__

.09 So Theorem: $\boxed{\underset{\mu}{E} \sum U_i < -2\ln p_\mu.}$

What __intuitive Signific.__ is this? $\sum_{i=1}^{n} \frac{1}{i} \sim \ln n + \gamma$ ← Euler's const.

$-\ln \frac{1}{n} + \gamma$

if $p_\mu \approx \frac{1}{n}$, this would do it!

.10 old theorem. $\boxed{\underset{\mu}{E} \sum \frac{U_i^2}{2} < -2\ln p_\mu.}$

m latter to __Prod U.__ are < 1,

so $E \sum U_i$ & $E \sum U_i^2$ both have same upper bnd of $-2\ln p_\mu$.

These 2 Thms can be readily genzd to p_μ being a function of n, t. sequence length

.15 To show (.00) $\ln(1-U) < -U + \frac{U^2}{2}$; $f(U) \equiv \ln(1-U) + U - \frac{U^2}{2}$

$f(0) = 0$ $f'(U) = \frac{-1}{1-U} + 1 - U = \frac{-1 + (1-U)^2}{1-U} = \frac{-2U + U^2}{1-U} = \frac{U(U-2)}{1-U} = f'(U)$

$f' = 0$ at $U = 0$. Consider $0 \le U < 1$ $f' \leq 0$. ∴ for $0 \le U < 1$, $f(U) \leq 0$,

.19 since $f(0) = 0$, $f'(0) = 0$ & $f'(U) \leq 0$ in that interval. (i.e., it can only ↓), it must be ≤ 0 on that interval

20 ~~Shorter~~ summarize: $f(U) \equiv \ln(1-U) + U - \frac{U^2}{2}$: then

22 on t. interval $0 \le U < 1$: ① $f(0) = f'(0) = 1$; $f'(U) = \frac{U(U-2)}{1-U}$ which is ≤ 0 on that interval from .19

Note: if $\boxed{\underset{\mu}{E} \sum U_i < -2\ln p_\mu}$; then $\underset{\mu}{E} \sum U_i^2 < -2\ln p_\mu$ since $U_i^2 < U_i$.

So this → is major result.

→ I __should__ check on this, hvr, t. juggling of inequalities is a bit __tricky__!

This is a neo derivation.

→ T. beginning at .00: $\frac{p_\mu}{~~\Sigma~~} \cdot \frac{1}{\prod(1-U_i)} < 1$ ← normaliz constant → $\prod(1-U_i) > p_\mu$: $\sum \ln(1-U_i) > \ln p_\mu$: α $-\sum \ln(1-U_i) < -\ln p_\mu$. ✓

30 Then via .15 ff. $\ln(1-U) < -U + \frac{U^2}{2}$; $-\ln(1-U) > U - \frac{U^2}{2}$: $U - \frac{U^2}{2} < -\ln(1-U)$; β $\sum U_i - \sum \frac{U_i^2}{2} < -\sum \ln(1-U_i)$

from α, β $\sum U_i - \sum \frac{U_i^2}{2} < -\sum \ln(1-U_i) < -\ln p_\mu$

So $\sum U_i - \sum \frac{U_i^2}{2} < -\ln p_\mu$.

∴ $\underset{\mu}{E} \sum U_i - \underset{\mu}{E} \frac{U_i^2}{2} < -\ln p_\mu$.

$E_\mu \frac{U_i^2}{2} < -\ln p_\mu$ i.e. __$E_\mu \sum U_i^2 < -2\ln p_\mu$__

so $\boxed{E_\mu \sum U_i < -2\ln p_\mu}$

3 main facts to give derivation:

① $\underset{\mu}{E} \sum_i \frac{U_i^2}{2} < -\ln p_\mu$

② $\prod_i \frac{1}{(1-U_i)} \cdot p_\mu < 1$

③ $U - \frac{U^2}{2} < -\ln(1-U)$ (if $0 < U < 1$)

Conclusion: $\underset{\mu}{E} \sum_i U_i < 2\ln p_\mu$.

→ (215.15

5/12/03

NIPS

202.20: Notes for Paper on Convergence Thms. for Chris Wallace "Festschrift."

4 Mon
2 aft lunch
5 comb!
late aft.

20 On t. NIPS talk: The idea is to Motivate people to read t. paper!

T. abstract should do much of this.

What I may do is start by showing that INV & OZ & predn. (QA) problems cover just about all well defined problems. Perhaps Give a few interesting/general examples.

Consider what we mean by "well defined problem".

1) Usual inversion problems

a) Inversion problems w. Badness threshold (how ~~far from exact~~ soln can one be?)

e.g. solving complicated alg. eqns.

2) Optzn: List several types. Maybe time varying, Noisy, Expensive evaluator,

10 Perhaps show how a cleverTM could help one "Well define" a problem (perhaps look at Letter to Ivan on this)

1 Book/day
= 1 h books in syrs
3k/decade
15k in 50yrs

So I want to show that this (CREATE a "t. Report") is a v.g. way to do A.I.

14 Argt: 1) ALP is a good, very General way to do induction

2) Just about all well defined Probs are INV or OZ (OZ covers a lot)

3) T. Main problem is "updating" & I think I have a good, gen'l. way to do it.

4) 2 phases of updating: Phase 1 simple Phase 2 Advanced.

18 5) Use of Lsrch; Then (Better) WON. → 212.00

20 SN In prediction of reals: We want to do search over continuous & discrete params at same time

21 (not "discretize, then optimize" — then new discretize, then optimize.)

Contrast speed of doing .21, with discrete param trials & associated random trials for contin params. In this coding, we will have a "Goodness of fit" criterion for how good t. code is, so we don't even have to hit t. "A" value exactly.

.25 There is some Q about whether this system will work, because

.26 it does not do Σ of pc's of codes; it just tries to find v.g. individual codes

There may be some special way I could devise to deal w. t. .25-.26 effect.

(On a previous look at something like this:) T. trials are not in pc order — so

30 → Not so Good for Lsearch for induction. I don't remember whether I found a good way to deal w. this diffy.

This is a more general problem (even for discrete predn.) that occurs when we do "lossy coding" w. "correction". We do not know, a priori, how big t. "correction" part of t. code will be. — So this is usually not such a good method to do "Induction"

Nips

AGAINST OCCAM

On t. Superiority of ALP: Reply to "Further Experimental Evidence ~~against~~ t. utility of Occam's Razor: Journ of AI rsrch 4 (1996) 397-417 | ~~Pr~~ in PS folder 6/13/02

≡ p(M) × p(corpus|M)

00 : Q: Suppose PEM₁ gets more wt. for a certain corpus, than PEM₂; But empirically,

.01 PEM₂ ~~gets~~ ~~better~~ predns? Well, that means that PEM₂ gets ~~better p~~ hyer

.02 p of corpus than PEM₁, but t. aprips are much inverted. It could be that

.03 aprip's ~~[illegible]~~ for PEM₁ & PEM₂ are way off (i.e., very bad computer lang.) or it could

be that PEM₁ is very a.H. & and actually will not continue to give predn.

.05 or, my system could be wrong!!

Most likely ~~that~~ that .02-.03 is true, that aprips were way off. If PEM₁ is really better a longer corpus will fix wts properly.

When .03-.05 (A Hoc) is true, one can usually ~~tell~~, tell by how t. 2 PEMs

0 were created.

So: If corpus is short enuf, main wt. is from apri, & can be way off, if apri info is poor!

Say t. "true M" has a very long dern. The only way one could justify "using it", would be to have a long corpus. With a short corpus, one would not even consider that "complex M".

One mite do it by using ~~many~~ very many continuous params: in which case A.H.ness could occur, & be apparent. — One would tend to select wrong M — A.H. correct for that corpus.

It mite be good to give examples of corpi w. continuous params, in

19 — which one used a M that was too large for t. corpus: Say t. "true M" was too

0 — large to be "(i.e. "too complex")" coded by t. corpus!

→ .19-.20 could easily be t. cause of .00-.01

T. forgoing is related to a paper that I have (in PS I guess) on some guys that tried using MDL on various corpi & found it worked well on some, but poorly on others.

If M has a long dern, than a "~~short~~ too short" corpus, will always give more wt. to PEMs w. shorter derns, that will not do as well as M. — But M is essentially "undiscoverable". If one does ~~[illegible]~~ Choose M, One must have reasons for choosing M, & if one does, its aprip is very hy.

20 Re: t. paper at top of page: If they had (wo. seeing new data) modified their Cafgrzn Algm. in ways that they thot would improve it — ~~[illegible]~~ or on t. basis of logical reasoning &/or their experience w. Cafgrzn in t. past, — then they were [illegible] & aprip of their model (≡ ↓ "complexity").

If they had no reason at all to make these modifns, then there are ~~many~~ very many other ~~[illegible]~~ very bad ones they should have tried.

If they feel that modifying a model increases its "complexity" they are thinking of "Complexity" in a purely syntactic sense. This is Bad. ~~The language has~~

Occam's razor is nice only, if t. lang. ~~[illegible]~~ has ~~been~~ modified so that occam's razor was correct in t. past. This is done by giving useful prediction rules, short codes.

T. lang. reflects t. regyrnts found in t. past, known corpus

If Occam's razor doesn't work, it's usually because t. aprips or language was not properly "updated"

Attempts to find upper bnd for $\sum_i p_c(u_i)$ — we know it converges to $<\infty$.

$\sum ~~$\bar{\ }$~~ \Gamma \equiv k'$ ~~ac~~ $\frac{\text{hd cost}}{\ln 2}$.

.00 : $\boxed{SN}$: $\prod \frac{1}{1-p_i(u)} = \prod \frac{1}{1-x_i} < \frac{1}{p_c(u)}$; $-\sum \ln(1-x_i) < -\ln z$

$1-p_i(u)$ ''' ~~$\equiv$~~ "x_i" $\quad \equiv z$

$z \ll 1;\ -\ln z > 0$

.03 $-\sum \ln(1-x_i) \overset{\le}{\underset{\le \sum x_i}{~~=~~}} k' \quad \left(\equiv -\ln \frac{1}{p_c(u)}\right)$

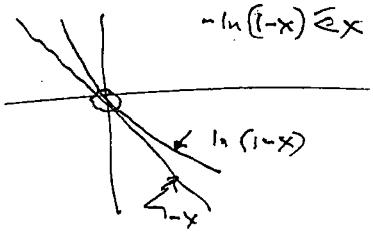
$\ln(1-x_i) = -x_i + \frac{x_i^2}{2} - \frac{x_i^3}{3} \ldots$

$> -x_i$

$\sum -\ln(~~1~~-x_i) \le ~~=~~ \le x_i \quad \sim \quad$ i.e. $\ln(1-x) \ge -x$; $-\ln(1-x) \le x$

So ~~[illegible]~~ while (.00) tells us $\sum x_i$ converges, it gives no upper bnd for $\sum x_i$..

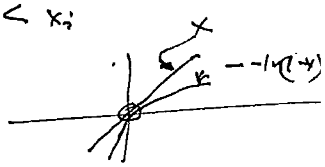
$-\ln(1-x) \le x$



10 $\underbrace{\sum x_i}_{} \ge \underbrace{-\sum ~~-~~\ln(1-x_i)}_{} > \underbrace{k'}_{}$

$-\ln(1-x_i) < x_i$

.00 $\prod \frac{1}{1-x_i} < \frac{1}{p_c(u)}$



~~$\sum \ln(1-x_i) \ldots k'$~~

$\sum -\ln(1-x_i) < -\ln p_c(u) = k' > 0$

16 $\ln(1-x) > -x_i \qquad -\ln(1-x) < x \qquad$ so $\boxed{\sum -\ln(1-x_i) < \sum x_i}$

18 hvr, we do have ~~a~~ an ~~expected~~ upper bnd for $\sum \frac{x_i^2}{2}$ which is k', from $p(U)^2$ conv. thm.

20 so $\sum x_i - \sum \frac{x_i^2}{2} + \sum \frac{x_i^3}{3} \ldots < K' \sim \frac{\text{hcost}}{\ln p} \sim 1.4 \times \text{hcost}$.

$\frac{\sum x_i^2}{2} < k'$ or $2k'$ or whatever $< ~~\sum~~$ for a seq. of n datapts. from $p(u)^2$ ... I don't know if this is k' or $2k^+$. ~~[illegible]~~

so $\sum x_i \overset{\sim}{\sim} < 2k'$

$\frac{1}{3} + \frac{1}{5} + \frac{1}{7} \ldots \frac{1}{2n+1}$

~~the~~ $\to \frac{1}{2} \ln n$.

If ~~$\sum$~~ $\{x_i\}$ is a positive, ~~[illegible]~~ non↑ seq. and $\sum x_i^2$ has known ~~[illegible]~~ upper bnd.

~~$\sum x_i < k'$, Can we say~~ $\quad$ ∴ $\sum x_i$ ~~[illegible]~~ is known $< \infty$.

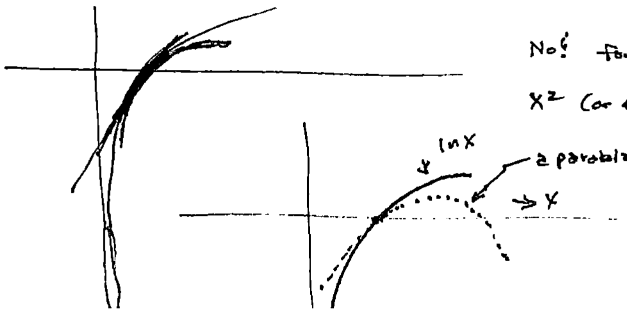
~~Can we say any thing at~~ If $\sum$

30 Upto .16 I hadn't ~~been~~ able to get ~~[illegible]~~ a bound on $\sum^+ p_i(u)$ .. only that I know it converges,

$-\ln(1-x) = x - \frac{x^2}{2} + \frac{x^3}{3} \cdot \quad < x$

~~[illegible]~~ ∟ is this $\gg x - \frac{x^2}{2}$? ~~$\exists\alpha$~~ $\ni -\ln(1-x) > x - \alpha x^2$?

No! for x betw. 0 & 1 the pole of $\ln(1-x)$ will dominate any x^2 (or finite power of x).



a parabola thru ~~a~~ 1,0, ~~having~~ matching first & second deriv. of ln x.

Drop this! It's an interesting puzzle, but not at all critical to TM!

→ 216.00

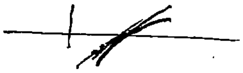
5.10.03
NIPS

original

Why $\frac{x}{x+y}, \frac{y}{x+y}$ is BEST Normzn possl. $\boxed{.10 - .19}$

Convergence of $\sum p_i(u)$: .22

$\sum_i \ln(1-\epsilon_i) = -\epsilon_i + \frac{\epsilon_i^2}{2}$



if $\sum \epsilon_i$ conv. then $\frac{1}{2}\sum \epsilon_i^2$ conv. Hur....

00: 206.40: whether include t. no. zero.

4) Monotonic ↑ funct of s.c. is s.c. But sin(s.c.) is s.c. only if (L_i, B) is all on an ↑ section of sin( ). $(s.c.)^2$ is always s.c.

5) sc − sc & $\frac{sc}{sc}$ are usually (almost always, always) not s.c.

E. T. result is that s.c. nos. are not very useful for arithmetic! — so sometimes one can use this property in proofs

→ 6) "enumerable" is not a good way to relate s.c. nos., because when one says a certain set of nos. is "enumerable" it's not clear whether one means cardinality or computability!

$\vec{f} = f_1, f_2 \quad = f_1(a,b), f_2(a,b) \equiv A, B$.

0 7) Say $\mu(x_i)$ is a semi measure, let $\vec{f}(a,b) \to A,B$ be a "normalizn", if $A+B=1$.

Then $F_1(\overbrace{\mu(X_i=0)}^{A_i})$, $F_2(\overbrace{\mu(X_i=1)}^{B_i})$ will be a seq. of real pairs.

$\boxed{D.16.04}$ I think this ↑ is backwards! μ is already normzd; we want to normalize M, t. univl. d.f. → Hur. this doesn't affect the Argument

12 for each i, there will be a maximum value of α_i ∋ $A_i \geq \alpha_i \cdot \mu(X_i=0)$ and $B_i \geq \alpha_i \cdot \mu(X_i=1)$

13 The normalization function that gives t. largest α_i will be t. usual normalization —

15 i.e.: $f_1(x,y) = \frac{x}{x+y}$; $f_2(x,y) = \frac{y}{x+y}$.

8) If t. norm of .15 is used, then t. $E\sum(err)^2$ betw t. normed and (t. normed measure that generated t. data) will be minimum

18 i.e. t. ratio $\frac{M_{sol}}{\mu}$ will be as large as possl. — using any other normzn will

19 ↓ This ratio. → See 4TM 313.10–.20 for more "proof".

20

9) Usually, for a long seq., t. proby of "u" will be very small; $\sum_i (p_i(u))^2$ ~~converges~~ is bounded.

EMPT CASE 23 → 10) If μ, t. generating d.f., is normal i.e. a "Measure", then t. normalization constant for M can't be > $\frac{1}{pc(\mu)}$, because this would make P_i too > 1.

(t. normzn constant for a single term is $\frac{p_i(u) + p_i(0)+p_i(1) = 1}{p_i(0)+p_i(1)} = \frac{1}{1-p_i(u)}$) ← $\boxed{\text{N.B. t. } p_i(\cdot) \text{ are all Conditional Probys}}$

26 T. product is $\prod_i \frac{1}{1-p_i(u)}$ which converges ⟷ $\boxed{\sum p_i(u) \text{ converges.}}$

i.e. is bounded ... so $\prod (1-p_i(u)) > 0$.

so, a fortiori, $\sum p_i^2(u)$ must converge.

GO OVER(10) carefully! .22–.23

we also have from that $\boxed{E \sum p_i(u)^2 \leq \ln \frac{1}{p(\mu)}}$

30 T. reason .26 must converge, is that if μ is a measure, t. normalization constant for it must be $\leq \frac{1}{pc(\mu)}$: $\boxed{\text{See .22–.23}}$

At first Glance, it would seem that t. convergence of $\sum p_i(u)$ could not be related to μ — But it is! μ generates t. sequence that decides under what conditions & u can should occur.!

More exactly $\prod_i (1-p_i(u)) \geq \frac{1}{pc(\mu)}$ = max size of Normzn. Constant.

NIPS

On Unimportance of semi-computability .35 ff
Significance of P(U)>0 (.00 – .30)

10 : $\boxed{SN}$ What is signif of P(U) > 0 ? : for SM or HR predn. it would be meaningless. Because there is no info (we know of!) in the data to imply "end of seq".

For daily weather data also, I suspect that "U" would be not acceptable.

There is a tendency to interpret "U" as "end of the Universe". I think it isn't a good interpretn.

A few pp ago I noted that (I looked back to 194 (4.24.03) ~~could~~ didn't find it):

If I had ~~[illegible]~~ 2 kinds of info about a seq. ① I knew last 100 bits ② I knew seq. had no "U" for last 10^6 bits. Then I should give very low pc for U being next bit.

The U symbol can be considered as being in a 3 ~~[illegible]~~ symbol alphabet for Bern. predn. Using Lap's rule, the U always has ≠0 proby of occuring. Modifying "Lap's rule," (because

0 we have aux info) we get very small pc for the 3rd symbol. ∴ Normally in statistics, if an event has never occured before, we usually ~~may~~ want to give it some possy of occurance. (particularly if it is an "alphabet" symbol (!) —

Normally ALP will give, for short seqs, a ~~[illegible]~~ "fairly large" pc for U, but this p(U) → 0 as maybe $\frac{1}{n}$ (or converges) ~~[illegible]~~ for many data seqs.

Laplace ~~[illegible]~~ (?) Example of $(365 \times 10^4)^{-1}$ as pc of sun not rising tomorrow, because he thot earth was ~ 10^4 yrs old — the sequence has only 10^4 bits. This corresponds to "aux info". In case of SM, "U" means the seq will stop at that pt, not that the Mkt will close for a days & open the next day. This ~~occurs~~ might occur during war, plague & conditions of Natl. emergency.

0 — The kinds of info that ALP is using ~~[illegible]~~ as a basis for the "U" prediction, is mainly, that the seq. is short. The fact that p(U) usually converges faster than $\frac{1}{n}$ means that aux info about past length of seq., ~~[illegible]~~ can be used to suggest, corroborate low expected pc for U.

While Mathematics is "The Handmaiden of the Sciences", I will admit that she certainly has a life outside that job (occupation). On the other hand, she really is employed ~~[illegible]~~ ~~[illegible]~~ by the Sciences. or "No all work and no plays she".

The way we do this — by renormzn — seems a bit AH., but it's a reasonable way to do it . . . ~~[illegible]~~ The suggestion that we consider p(U) >> 0 as a real possy. would have very poor corroboration in the past.

E.g. If we start w. 10 yo daily SM data: At first we get frequent occurance of large p(U)'s. They are always wrong. As the seq. continues, p(U)↓ rapidly to say ~ $\frac{1}{3600}$ ~ ~~[illegible]~~ .0003. ~(.03%). (its .003% if we consider 100 yrs of SM data)

0 So: All of the foreg. suggests that S. normzn should be used in usual prediction. If we have a meaningful ~~useful~~ interpretn. of "U" so that p(U) > 0 is a useful, reasonable pc, we should use G's normzn.

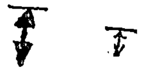
35 ~~[illegible]~~ On: {203.00 to here}: 1) First that S.C. v.s. non-S.C. (but "knowable limit") — S.C. isn't signifly much better, if one is looking for an approxn. of the number.

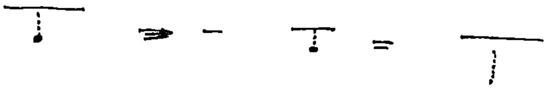
2) S.C. nos. are characterized by a) a seq. of monotonic (none) [α_i] reals. b) an upper bnd B c) the limit of a). L

~~[illegible]~~ the quantity B−L will be the smallest amt. of error one could know in L.

3) sum of S.C. nos are S.C. Products are S.C. ~~[illegible]~~ usually if the ranges (B_1, L_1) & (B_2, L_2)

NIPS



00: : If we take t. diffrence (or ratios) of 2 sc nos., t. seq has a limit, but it's not a monotone seq. ~~The~~ A "limit" is known, 

The resultant/non-monotone seq. has a limit: ~~The~~ It has a known upper bnd, but no known lower bound.

From a practical pt. of view, the s.c. value of ~~a~~ a semi measure is not knowable in a useful way. Say we want t. semi measure of t. seq. 01101110. 8 bits. There are 256 possl. seqs. of length 8. ~~By~~ finding as many codes ~~for~~ as possl. for all seqs. of length 8, we'll narrow t. upper bnd. on t. pc's of ~~the~~ all of t. seqs. If we include info on codes that stop before they print 8 bits or codes known to loop before 8 bits out, we have a ~~[illegible]~~ smaller upper bnd for pc of any 8 bit seq. If Δ is t. total pc of all codes that might be a code for an 8 bit seq. but don't ever converge & are of "unknowable" convergence, Then Δ is t. uncertainty in t. ~~[illegible]~~ ~~[illegible]~~ pc of any code of an 8 bit seq. If its "true" pc is p then we can't at best know its betw p & p+Δ.

My guess is that usually Δ ≫ p (& more likely Δ >> p) so Δ gives no useful bound on t. pc.

So t. moral is, that for actual practical approximations, t. known limit is too far from t. known pts to be of much use in getting accurate estimates. In most cases, & for either semi computable or more "totally incomputable" numbers, one will usually look at t. ~~[illegible]~~ apparent rate of convergence & extrapolate to ∞. — if one wants

19.— that info. More likely, for practical statistics, it's not necy to know how close one is to t. limit (= ALP). Even if one knew one had P_M exactly, we would still have to estimate how accurate it was, in t. "usual ways" (cross validation, with or w.o.

23 training set).

T. ~~[illegible]~~ subset of "U" cases that are "undecidable" gives a ~~[illegible]~~ lower bnd on t. precision of any estimate of a semi computable no. My impression is that normally ones' estimate of this ~~[illegible]~~ "undecidable"; & lower & upper bnds on what it could be — ~~are~~ too poor to ~~be~~ of use in ~~knowing~~ estimating the semi computable no. of interest.

In getting t. "non enumerable" nos. that are differences or ratios of s.c. nos., We really don't have much difference in our ability estimate accuracy of correct approxn. In this case, t. estimated sequence is not monotonic, & t. bounds on it are too far away to be of utility. — We estimate error by apparent rate of convergence ...

But anyway, t. remark on .19–.23 applys — i.e. this is usually not a ~~[illegible]~~ problem in ALP.

So: Motivation for Normized ALP: If μ is a cpm it will give more accuracy to t. estimate of P_0, P_1. Also true if μ is "approximatable" by a cpm. I'm not sure about cases in which μ is simply a sc. semimeasure.

5-7-03

Nips

Most of these don't have a "normzn constant"

10 8) Levin's analysis (4 ~~B~~, (203.2#)) (200.21) ← ? is relevant to ⓐ ~~[illegible]~~ weird normzns ⓑ Semicomputable m.
I think it is only of interest if the normz constant → ∞.

9) T. "natural" way of normzn gives max ↑ of "normzn factor" ∴ it gives least ~~ms~~ $\Sigma(\text{error})^2$.
Say p_0 & p_1 are pr's ~~and~~ $p_0 + p_1$ may be < 1. (p_0; $p_1 \geq 0$).
~~we~~ for normz, we want $p_0, p_1 \to x_0, x_1$; ∋ ~~①~~ $x_0 + x_1 = 1$, ②
Say α is the largest no. ∋ $x_0 \geq \alpha p_0$ & $x_1 \geq \alpha p_1$ and (α is a max mult factor of P(μ).) Then the x_0, x_1 mapping that gives (.00) max α is ~~[illegible]~~ $x_0 = \frac{p_0}{p_0+p_1}$; $x_1 = \frac{p_1}{p_0+p_1}$
0 This is ~~t.~~ Basic Normzn, indep of (8) (Levin's analysis).

·11 (10) A possl. interpretn of Levin: That even using my normzn, There is no normed UPD,
·12 That is ~~[illegible]~~ ~~[illegible]~~ ≥ all other normed UPD's (within a constant factor) (wc#).
If all normzn constants were bounded, ·11–·12 would be false. ∴ there must be seqs & univs for which the normzn "const" ~~→~~ → ∞: (This is a nec'y, but not sufficient, cond for
Def Lev. to be right about this. If we assume μ is a CPM, then Any Normzd UPD is $\boxed{\text{w.c.f.}}$ "better than any other" within constant factor

(11) If we know the seq. has been running for 10^6 bits, (but we only have last 100 bits for data)
then we should normz our predictions to $1 - 10^{-6}$ rather than to 1

20 $\boxed{SN}$ 3 dif's:
1) from −∞ to +∞ : ~~[illegible]~~ $= e^{\frac{(x-\mu)^2}{2\sigma^2}}$ Gaussian.
2) from 0 to ∞ $x^n e^{-mx}$ Gamma
3) from 0 to 1 $p^n(1-p)^m$ Beta (?)
These 3 cases would seem to cover most situations!

12) For discussions related to this stuff see $\left(L\text{-}V_{97}\right)$: pp 259 – 301.

(13) Re: "semicomputable nos:" a "s.c. no. ("enumerable") no. is defined by an ind. seq. of ~~i~~ increasing (non-↓) reals & an upper bnd. (or an upper bnd & an algm. to generate such a seq.)
30 ~~One can add 2 s.c. nos & get~~ s.c. is closed under, a) addition b) Mult by positive constants, or by another s.c. that is known to be > 0. c) any monotonic ~~(~~non↓) operation.
$(sc)^2 = sc$; $\sqrt{sc} = sc$ if sc is known to be > 0. Subtraction & division betw. 2 sc's are never poss.
Any way, t. point is, one can't do very much w. s.c. nos. (Usually one can't compute them — tho a regular real is a s.c. no. & often it can be computed & a s.c. nc.
If a is t. true value (limit) of a s.c. no. & b is its "~~[illegible]~~ upper bnd, then b−a is t. lower bound for uncertainty in t. value of t. no.

5.6.03

Nips

50 : On "Superiority" of Normed Univl pd. = Universal Proby Measure.

I had this idea that Levin proved that There was no Universal Measure that multiplicatively dominated all other Universal measures.

L.V. '97 (second edition!) (p 300 / p 301) 4.5.4: Possibly §4.7 pp 307-314 (History) is relevant.
Seems to contain no mention of Levin's proof.

The BONE of Contention: ~~Inform~~ The univl semimeasure & Measure both have ~~Ee~~ E Σ err² converge to < a constant — hvr, The constants differ by the normzn factor for the univl measure.

Th. Q is: How large can this factor be? — Can it be infinite? I think it can't be infinite —

Woops! The normzn "constant" is not ~~a~~ a single number for the entire Normed PD! —

It's usually diffrnt for every bit in its domain!

0

Say PM actually was ~~α~~ αM for the seq. of interest, x. Then the normzn constant would ~~approach~~
~~$\frac{1}{\alpha}$ as n → ∞, but the~~ be $\frac{1}{\alpha}$, for all ~~other~~ bits of x.

So $\frac{1}{\alpha}$ would be upper bnd on normzn constant — so ~~the~~ the Normed Univl pd. ~~would~~ could be $\frac{1}{\alpha}$ times as good as the semimeasure: (so ~~E~~ Σ ~~e~~ err² could be $\frac{1}{2}$ that of ~~the~~ semimeasure!) ← N.B. .18R The "Best" I'm considering, gives Normzn constant = 1, equivalent to pc = 1 for all! — This gives EΣ err² = φ! So while normzn ↓ err², we have no useful idea as to How Much.

.18 1) In general, this normzn constant has to be bounded by $\frac{1}{\alpha}$.

2) I am considering only prediction of cpm's (computable proby measures).

3) re: ① My impression is that usually the normzance will be << $\frac{1}{\alpha}$, since the normzn constant only fills out the "U" probabilities.

0

.2) N.B.* 4) What Levin probably showed was that if you considered all possl. normzn methods, there was no one that was "better" (within a constant factor) than any other. This may be false! It implies that some normzn constants must be unbounded! If so, the S7813 corollary would say they couldn't be useable probability distribns — i.e., S7813 wouldn't apply to them.

5) It may be that the arguments is mainly about predicting non cpm's.

6) Actually, only the usual method of normzn has a normzn "constant", i.e. the pc's ~~a~~ [illegible] for 0 & 1 [illegible] are mult by the same "constant" (tho the constant ↑ (usually) as the seq. length ↑.

Ackermann ~~funct~~ funct.

SN If a monotonic funct is prim recursive, then its inverse is prim. recursive. — maybe not!

30 Well, A^{-1} isn't defined for many of its args. Hvr, if, in "defining" the inverse of the Ack funct. one lists the args for which it is defined, then computing the inverse is trivially prim. rec.
To find $A^{-1}(k)$ ~~simply list the n'th args in order~~ Simply list the ~~#~~ legal args in order of size. The n^{th} one will be k; so $A^{-1}(k) = n$.

7) In view of .18R: Σerr² ↓ but we don't know how much; (φ is lower Bnd! ①).

If we computed the normzn "constants" we will know how big it is & how much Σ err² has ↓.

5.4.03
Nips

Paper on Convergence Thms .20

.00 : 3 interesting cases of things that seem ~.

1) ~~[illegible]~~ Mult & division needs to take much longer time than Addition Subtraction: This helps TM to discover "laws of Alg", since they can shorten time needed to do many Arith operations.

2) Discovery of concept of "option" ~~as of an~~ model for certain prob solns as being faster (shorter) than simply trying to copy ~~[illegible]~~ recreate directly examples of "option" that were given in the TSQ.

3) From examples of very simple "stories" in algebra, TM "invents" humans w. very simple properties. (A, B, C, : "Alice Bob, Charles, Dick and Jane ... etc) ~~At~~ Later TM sees stories about humans in R.W. ⇒ makes Models.

.0 A great breakthru in compression occurs when TM realizes "it" can use same/or analogous models for both Domains

A possl case in which things are "~": Restriction of TM (i.e. ~~[illegible]~~ HW or S.W. or info in TS.. (for clear: access to Macros, APLs, table of S's, ~~energy~~ concepts etc) can make ~~[illegible]~~ it possl. for certain concs to be useful that would not otherwise be useful. This fits examples (1 & 2): I'm not sure about whether ~~E~~xample 3 fits.

4) TSQ's: carefully designed at first, then less care as time goes on.

Esp: where valuable!.

Also probly that: there is particularly (not 100 bytes of P — coupons known to be long, that its likely that P(U) is very small.. (i.e. Normed.

20. Paper on "Convergence Thms": Some things I can include: GOOD!

1) Gens of UPD each have their Conv. Thm: So Give Gens & Great. Conv. thms. Usually the proofs will be ~~about the same~~. Include Multidimens, missing data, multidirectional, {Bags, some bag elements have finite length, others semiinfinite, others are or (hyper) ~~semiinfinite~~. → also $E\Sigma P(u) < 2\ln$ [illegible]

Includes: Conv. of ~~EΣ~~$(P(u))^2$. Also all corrolarys ~~of~~ corresponding to Sol 78 Th3 corrolary, ← Mention Tony Fine's discovery, suggesting this Corollary.

Mention C. Wallace correspondence as inspiring ~~a~~ genesis of bags & Gauss. to Multidim, ~~both~~ multi infinite corps.

Give each Conv. Thm in detail: Give proof (briefly: if not obvious or if not given elsewhere ~~[illegible]~~, — tell where proof is) Explain what kinds of problems it applies to: perhaps give examples.

30 Perhaps discuss signif & lack of signific. of ~~complete~~ incompletability of UPD('s).

6/8/03 A possl. way to write paper: Write prelimy introduction: But mainly, write up a minimal paper to start. Then add in more material from .20 – .30. Then rewrite introduction;

finally: write Abstract.

I seemed to have promised to get paper to ~~[illegible]~~ Douw in final form by "Jun 2005"

Not clear when in June: try June 30. 22 days ≈ 3 wks! I better get started in more detail!

In intro, Give Motivation, importance of Conv. Thms.

NEPS

Revision of Kol Talk paper; section on Incomputability of UPD

To follow: "Not a Bug but a feature" p3 of paper:

Let me explain: Many years ago in ancient Greece, the Pythagoreans sp? discovered that √2 could not be expressed as the ratio of two integers. It took the mathematical community many centuries to get a good understanding of this problem, but well before that time, approximations were made and used. None of the approximations were actually √2, but they ~~were~~ got arbitrarily close.

In the case of the UPD, we can make a sequence of approximations and just as for √2, the approximations will eventually get arbitrarily close to ~~[illegible]~~ the ~~[illegible]~~ Platonic Ideal ... the true UPD. The difference in the two situations is that for √2, ~~we~~ for each approximation, we have a good upper bound on how large the error is. ~~[illegible]~~ On the other hand, for our approximations to UPD, we cannot ever know a useful upper bound on how much we deviate from the ~~[illegible]~~. ideal UPD.

Fortunately, for almost all applications, we don't need this information. What we usually want, ~~need~~ to know, is not how close our approximation is to the ideal, but ~~how~~ rather, how accurate is our approximation for prediction. If we have a reasonable sample size ~~then~~ we can estimate this accuracy by cross validation, but often we can do better. ~~If~~ If the approximation we use is entirely a priori (devised before the data was known), we can use all of the data for testing, since none is needed for training the model.

Though the incomputability of the UPD is usually not relevant to problems in practical prediction, it is of much interest in the Philosophy of Science.

~~Many~~ Many 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. However, the incomputability of the UPD assures us that this cannot ever happen. With any amount of data and finite computing resources, we can never be certain that we've found the ~~[illegible]~~ last, the Final Theory.

Some of us are not at all disturbed by this state of affairs, but find it instead to be a neverending source of joy in discovery. Newton ~~said that his discoveries were built~~ progressed by standing on the shoulders of giants. It is often necessary to ~~progress~~ move along by climbing over the ashes of dead heros.

The charred remains of the vanquished heros of the past

Newton attributed his success to his standing on the shoulders of Giants