SFTM 387-388

N·15·73 SFTM Thave no many previous refs: I don't know t. most Recent. TM 387

Some ideas:

1) A βntm (Boolian Net TM) is ~~some~~ sort of Boolian Net w., at any particular time, a particular I, O relationship. This relationship can be modified ~~to~~ to be various "near by" relationships by suitable / mutations. ~~TB~~

To operate it: we trie various mutations Run Test each according to some criterion. If it is "good enuf" we retain t. mutated — if not, we try a new mutation from t. old pt. as center.

If "good enuf" (or even just better than before) we do new mutations from this new pt. as "center".

2) Q! If we have a net that is "Universal" in some sense, is it always possl. to apply it at some pt. → its "closeness function" is any particular function that one wants? (The usual ones for induction are those involving some mixture of ccost & prost).

3) T. system of 1) isn't really so good. We would perhaps like to try a mutation, obtain its Gorc. We do this for many mutations. We remember t. Mutation, Gorc pairs, hvr. T. set of best mutations are then tested further, using a Gorc of greater ccost, & t. best mutation is then retained & used as a base for new mutations.

I think a system that could do 3) was the one with a very large, memory of relatively slow access time, used by many arith units in ||, each having its own reasonably sized cache memy. [I wrote much about such a device in t. first ½ of 1973]

It can do backtracking if needed.

So 2) is a Very Impt Q.

In t. system of •30, since each arith unit is a sort of unc., we should be able to implement any "closeness" functs we like. : Perhaps just use prost, & make ccost part of t. Gorc! — Tho it would be nice if we could reject by ccost trials before even trying them!

388.01

SFTM: .01: 387.40: In this 387.30 system, we can save all of t. cache unit memys in "archival" (unchangable) memy, so we can backtrack arbitrarily. Also, we may want to go back & actually have "offspring" machines constructed, having starts at certain "Branch pts." of t. evolu. of t. system.

.05 4) Note, hvr, that 387.30 is not at all SFTM. SFTM was a very large Bnet w. very fast transistors in it as components. It was very fast, assynchronous, perhaps partly analog, & very inexpensive

5) Humans may implement 387.20 by using short term & long term memys

6) T. System of 387.30 may be adequate for most any TM, that I will initially consider. After I get one running, I can try to get it more like into SFTM form (like .05).

7) A (almost purely) sequencial machine usually gets more computing done per cost. Probably a sequencial with unit (1 bit at a time) is best — particularly inasmuch as one need only use as many bits accuracy as is needed. Another big waste is "look ahead".

Planning 944 - 951
73TM

TM 444

A scenario of Planning & debugging in Electronic Ckt design

MD 2073

Planning

Discn w. G. Sussman: Outpourings of some ideas.

[See also his

Given a problem: to express the problem in a form → A "solu plan" is suggested.

A "solu plan" is a very large set of possl. solns.

It takes the form of a rather imperically derbd. Soln.

Solving the problem then consists of narrowing down the possl. solns.

One common method is to take a possl. soln. that conforms in a central way to the plan. Then, after trying out the soln., it is clear that certain sections of the soln. do not conform to the plan. Also the way they don't conform suggest modifns. to make them conform.

In the prog. ~~XXXX~~, design, the plan was divided into sections (sort of "subgoals"), so that one could try to get each section of the proposed soln. to conform to the corresp. section of the plan.

If certain subgoals appear, after a while, to be impractical, or if, all subgoals are achieved & the soln. is still n.g., we go back & try to devise a new plan.

I think the prog. discn. is too much tied up w. the idea of subgoals. The idea of a plan can be more gen'l. than that, & I should write a more gen'l. discn.

.01 D2473 : On Planning in General: A Plan by definition,
.02 corresps to a (usually large) set of possl. solns. to a problem.
1. Language of Plans is quite impt. — in at least 2 aspects of import

(1) Easy to induce plan from problem. This is based upon good laws for plan & prob. Also relevant experiences in the "area" of the problem. So — from a descr. of the Problem in Problem lang → a descr. of one or more plans in plan lang.

(2) Easy to solve the prob. once one has a plan in the plan lang. This would be because the plan lang. facilitates natural ways to break prob into parts (i.e. sub-goals is one way to look at this —

e.g. Problem → plan → plan → sub goals; then each goal → plan → subgoals or goal → soln. or goal → plan → soln.
(some goals or plans are very close to solns. for a properly experienced TM).

A prob can be broken into subgoals in a non structure (I don't know if this is most gen.) as in SP.

A sub class of this is for it to be broken into a set of nearly independent problems, that may or may not be sequentially constrained (i.e. the exact nature of Bs can't be determined until A is solved)

T. defn. of a "plan" .01-.02 seems fairly general. The idea of its facilitating the breakdown of the problem into a tasknet (i.e. SP) seems rather non general, but has perhaps received all or almost all the attention of AI workers (perhaps following GPS) up to now.

.38 A "non-nearly tasknet" concept of working via a plan: A working of the problem consists of narrowing down the specificity of the plan, which

D2973 (Planning) ~~SP~~ ← see separate copy make copy TM117 copy 2 (3)

.01 ≡ initially a large set of poss. solns). ~~[crossed out]~~

In the task net approach, the plan is a sort of "factorization" of the set of solns. One solves each factor by narrowing it down. More generally, factorization of this form is not poss., & one must somehow narrow down the soln. set "as a whole" (or perhaps not as a whole, but not in factored form).

This working out the plan in not factored form is often done on the specific (subgoals) (factors) of a larger plan. It can be done if e.g. the goal is recognized as being of a particular type & being amenable to certain (non-factoring) approaches.

.18

(SN) On SP: I suspect that normally one does not solve large task nets using probty curves for each? known sub task in the net. Instead, there is a certain set of net forms that are readily solvable (perhaps by table, analog or MC means) & the form of any actual SP net is "reduced" to one of those forms as an approxn. Some methods of "reduction", are to express a set of ‖ or sequential tasks as a single task, or any single input, output object sub net as a single task with suitable probty curve.

Viewed in this way, if a large task net is decomposable into series, parallel subnets, one might be able to solve it, by obtaining a G curve (i.e. probty of soln, failure v.s. time) for each ‖ or serial set of tasks — like an ohm's law soln. of a Resistance network.

Furthermore, if one could do the analog of Δ to Y & Y to Δ xfms, one could perhaps solve all nets!

How to construct plans

In the task net a design to cover all poss. solus.

— see May 66 CACM paper

→ 6.0

D2473: Planning

01: 3.18: **To obtain** a more goal. approach to Planning is: soln. finding: Consider 1 goal. idea of 2.37—3.18. Look at various planning methods. Also various soln. methods. See how they fit into the goal. defn. of a plan of 2.37—3.18. I think they ought to all fit.

In particular look at Miller, Galanter & Pribram — "Plans & the Str. of Behvr." for ideas

(2) Polya for Prob. solving methods to fit into the goal. Schema.

Remember N & S method of Getting Plans: Simplify the problem in some way; Solve the resultant problem. Use the soln. as a Plan to solve the original problem.

The Planning lang. will be closely linked to the methods one uses to solve problems.

Another closely related idea — to which prob. solving & planning methods & langs are related — is the "form in which info is perceived & stored. Minsky's "frames" idea is relevant here... — for perception, storage, memory, thru low order abss.

In a good Planning language, it should be possl. to work out various further constraints on the soln. — in some v. g. plan langs, it might be possl. to work out the complete soln.!

One very good kind of plan is a very non-detailed descn. of a soln. — e.g. "To go from home to airport, I will walk along streets A, B & C." Another kind of plan would give a set of goals — with each

Fig. 01

D2373 Planning.

01 ! 450.40 goal specified either exactly, or with much inexactness.
While staying within the planlang, one might sharpen up the
goal(s) (Make them more specific), or decide on the order
in which they are to be done. One might find
that ~~one~~ goal X must be achieved before
one can specify the next goal w. any exactness.

This characterization of Planning seems v.g. What I can
do, is to look at various workings of problems (& protocols, but
perhaps just introspective), & try to characterize them in
a common way — of seeing just how the set of possl. solns.,
as gen. by the initial plan, is reduced in each (successful)
step of the soln. of the problem.

Devising/using subgoals, 'working backwards', use of diagrams to suggest & discourage
subgoals, Thms. (An analog of diagrams (in Geom) is making up lots of
random-ish examples for logic ~~proof~~ proof problem), can all
be discussed in terms of narrowing down the "plan" to a more specific
(= narrow) set of possl. solns.

Certain "plans" can be "complete" in the sense of
one being able to prove that ~~the~~ all possl. solns. must
be included in it. Once then might be able to narrow down
the plan & still have the resultant plan be "complete".

His (Gaul) | 379–382

History of my TM work.

His ; Gen
tory

.01: 378.40 : Now it seems to me, that many times in the past, I was at the pt. of which of 378.06 - .40. Just why didn't I continue? In fact in ~~the~~ 1962 "Tug sags for Mach Ind" was pretty much the forgg. idea. I wasn't as confidentiation understanding of CMI — also I may not have had PMTM — but why didn't I continue?

I did take off on Stochastic Part, also some shortcut ideas on Superfast TM in more recently (this year) on Sag. property Machines. But still, what a clear cide tracks were passed??

For an analysis of ~~the~~ why I didn't continue on ZTB 143 (Tug sags etc). See 70TM (K6) 220.01 - 221.03
For more analysis of what I spent my time on 1965 - 1972 See MI folder.

In 1964, PMTM and GIM (inductive Gen'l Machine) were the main track. I had devised some Tug sags at that time — & I may have devised some earlier as well — perhaps that's what "Dart TM" Sequence was.

After browsing a bit thru the work ~~done~~ of 1965 (before fire): Let us compare 3 situations:

A) At time of ZTB 143 (Early 1962)) 3½ yrs
B) Just before Fire. (late 1965))
C) Now. (late 1973)) 8 yrs!

(1962) A) MI had 1 idea of ~~=~~ TM1 ≡ TM2. (≡ BSTM = JS)
2) Also that Gray Search was very close heuristicly to BW search
3) Ideas on a good Tug sag. of Arith, alg, etc.
4) Basic ideas of CMI.

(1965) B) In addition to A) I also had
1) A more exact idea on how to xfm a/Gen'l soln. of BW srch into a gen'l. soln. for Gray search.
2) Some gen'l. ideas on how to do searches (Stochastic Part ≡ SP) 3) Also how to xfm most, if not all AI probs into the form of a Gray or BW search.
4)

His : Gen

.01: 379,90 ; 1973 C) All of B) available plus via Willis.

Gen

1) Assurance that CMI is rigorously correct & that my method of doing Grammatical induction w. CFG's is correct. (This is probly impt. for ruff extrapoln. of hours).

2) Work on ik Gilborn GS Grimsdale Ummer method & other very recent - i.e. 1973 shortcuts to TM, involving & sequencial property (This is was done after Willis' paper).

SQP m's, of this sort, will not work, hwr. & I know just why they wouldn't.

3) Understanding of SVH soln. to induction problem - also estimates of expected error. — The little rigorous work, & no idea of speed of convergence.

4) Some ideas on IPC : Digital logic, Memory hierarchy utilzn & IPC of Human Brain. These are not immediately of value, but will be probly impt. if anyone TM is to be constructed that will take optimum advantage of the hardware available.

$((1+x/3)x/2+1)x+$ 113

Actually, from Mid 1965 (~ 8 1/2) untill Mid 1969 (~ 4 yrs) I did relatively little new work on TM.
Mid 1965 to the end of 1965 was a slowing down.
Jan 1969 to Mid 1969 was a re beginning.
Actually Oct 13, 68, I started reading W's first report.] serious work — willis
On Jan 9, 69 I had perhaps finished reading much of it.
On July 11, 69, hwr. I had only done 30 pp on TM proper.
What happend from Jan 9 to July 11? [cornell folk fabs, constructed & oper Apr 25 - Jun 11
Angy back Mar 69
J.

Genl.

440-442

D1973 Gen | Previous Gen Ref: 407.40 | Also R439.25-.40
On t. SQPM idea!
01: R 439.40 :
See Blum 374.10 -.40 for most recent summary of objections
" " 129.02 for previous " " "

TM440

Anyway, when looked at in t. foregg. way, in terms of those "Batches", this SQPM method is pretty much like just mutating old Pgms to find/code trial new ones.

As such, it lends itself well to parallel processing, à t. discus. of ll proce. affity, use of heir memys, IPC, etc.

There appears to be a big difrnce, hwr, betw. t. present concept, à t. old SQPMs: In t. present thing, t. machine looks at t. Macrosymbol (≡ Batch of corpus), before trying to devise codes for it. In t. old SQPM's t. trial code was devised for t. corpus before t. Machn could look at t. corpus, à then t. out code was rejected if t. output didn't conform to t. corpus.

Hwr., in t. "new" Mach., it seems that there is something that has t. old sql. propy. We have a seq. of corpus coding methods (≡ sort of Pgms). These methods are tried, then mutated, recombined, tect, in a "random-ish" way to form new t. next trial coding method.

So our "object to be extrapolated" is this batch of Pgms (≡ coding methods) à parts of Pgms. As t. corpus grows, this object also grows. Also t. wts. assigned to various parts of it changes as it à t. corpus grow.

Re! Backtracking: As t. set of Pgms à parts of Pgms à comutation rules à their wts (≡ basis of naming them) changes, TM, tries to recode t. past corpus in terms of t. modified bunch of Predn. objects. If a signifly new code is found, t. wts. & à objects in t. ("prediction body") are suitably modified. This / can take a lot of ccost in a machine, à it certainly can do t. same in a human!

reforming of t. Predn. Body

My mind isn't clear on just what t. pt. of this "New View" is:

Perhaps that new trial Pgms (corresp to new trial codes) are formed by mutations etc. of old codes à sub codes — but that this is done in a sequential manner — so that we have a fixed pt.

441.01

~~Gen~~ Gen I

D2073

TM441

01:440.40 a dcrn of t. "prodn. body," that works up to that pf. We normally modify this prodn. body (mutation, recombinn. of parts) by adding symbols to t. "prdn body" dcrn. I'm not sure if we devise new definitions in this normal "sequencial coding". We may, hvr, do Backtracking — which is to apply newly devisd successful (or almost successful) ideas to t. old corpus to recode part of it.

One of t. problems in any random mutation scheme a/o recombinn. scheme, is to keep it sequencial, so one doesn't have to try recoding t. entire old corpus, whenever a new mutation a/o recombinn. is tried.

Perhaps it is possl. to show that usual sequencirality, with occasional backtracking, results in a Universal ind. device — ie, one that can dcr any CPM.

Note that t. "prodn. body" plus some error rutn, constitutes a complete dcrn. of t. corpus, so, one should be able to apply ideas of CMI to it wrt t. problem of whether or not it is a univl. ind. device.

Well, my impression is that in order to be univl., t. system ~~could~~ need scan over only a finite (tho very large) set of mutations & recombinations — but somehow, as time went on, t. CB would have to ∧. This corresponds to trying "only" (finite now) 2^{ℓ} possl. codes for t. corpus, but ↑ t. C.B. to $\to \infty$.

From a practical standpt., hvr, t. no. of tryable codes is $\ll 2^{\ell}$, & so one will have to try $< 2^{\ell}$ & also let CB be $< \infty$. 2^{ℓ} is sort of like ∞ from a practical pt. of view — so one will take some path in CB, code space! ie. CB will be some funct. of t. no. of codes tried — also a funct. of t. precst of t. code being tried.

For most, perhaps almost all, progs we might consider, CB is of approximately known size — definitely bounded.

Some actually useful functions (like t. inverses of some easily defined functs), have short dcrns but take an enormous time to compute.

[e.g. $F(x)$ = The middle n bits of x^x — where x has n bits. F^{-1} is lots of trouble to compute — also I don't know if it's single valued — (The one could specify t. smallest value. — also we don't know if it exists for all values of its argt.!]

44201

D2073 Gen

01.441.40: [redacted] Hvr., it would be well to look at one's mutation, recombn., backtrack schema, to estimate to what extent it is "univl." Perhaps a system might start out not univl., but after it had a certain typ. seq., it would have certain new mutations & abss. — so that then it would be "univl."

88 → Actually, I think that is the Genl. approach I expect to restart on soon:

cf ZTB 141, (376.18 – 378.40 Genl), 387–383 sfTM (Also see (391.28 Bm).]

Typ.

Universality is Not a problem: I will have this seq. of problems: If TM is unable (or even unlikely) to solve any problem, I will know it, & add something or modify the system so it will be able to do this. Since Universality is not a diff't thing to get, it is likely that I'll have it early in the typ. seq.

As for this Business of Backtracking: It would seem to be desirable for optimum predn. w. agn. SSZ. (or corpus). Hvr., strictly speaking, it is not absolutely necy, providing for universality, t. TM remains univl. I.e. at any stage of TM's "tracking t. corpus" — say t. rule of t. underlying Rdss is suddenly completely changed — TM should be able to discr this rule, no matter what this new rule is. Hvr., t. CB necy may be very large if t. new rule is very distant from t. old one — or more specifically, t. new rule is very distant from t. most likely "near" pgms that would be tried by t. [redacted] st prediction body" that was attained in following t. old typ seq.

I think t. idea is to keep t. "predn body" good in t. sense of being likely to track near by" (to t. previous pgm tracked) pgms, but still not so narrowly apriorid — so that more distant pgms have reasonable chance w.o. excessive CB.

PLAN : T.S. 248 - 406

Plan

VHM = Van Heerden Method = Shortest cod. 248

On t. Q. of what to work on ~~re.~~ SVHM:

~~Prob~~ (1) Should I spend more time on t. SVHM Thm itself à t. 2 possl. forms of SVHM (max prost of single code for PEM. v.s. max Σprost of PEM)?

(2) Working on max ~~MMMMM~~ v.s. Direct v.s. Standard methods.

Whether or not SVHM is "true", (2) is of much value!

a) In ironing out just how to apply SVHM — how to evaluate prosts of various types of PEM (Tho much of this, I have done befor — I should try to find it à index it).

b) T. basic idea of (2) is to see if it is a counterexample of SVHM — i.e. that t. direct method may give a better Gorc that t. ~~Σ~~ max Method.

There is some Q about t. relevance of my comparison of t. max & direct methods: Shall t. comparison be for larges S (= ss≥) or for S ≃ 2N? — or should the N value in t. max method be automatically chosen for each predn. so as to max t. past Gorc ??? — so that in t. max method, even N need not be part of t. descr of t. Pam.

On second hot, hwr, I know that in a certain specifiable sense, max method will always give better predns than t. direct method — so under all rounds (any values of N & S) max ought to give a better Gorc.

(no. of params ↑, ssz ↑)

So perhaps consider S = ~~MMMMM~~ N+2. I think in cases where S is just a bit larger than N, max meth may win, because direct has those extra N params to specify.

July 25, 73 Plan

250

.01: 248.40: It may be that I can get some good comparisons between Direct, Standard & Max methods w.o. a lot of work. One of the main problems is punctuation — the goal problem of coding scopes of nos., etc.

.09

.10 Re: T. most immediate goal of the "Review Paper" on the validity & sufficiency of CMI": In order to get others to work on this, I'll have to do more than just present those results. One thing I can do, is to show how CMI can be used to solve various problems: e.g. Gramm. induction, — How to use it for Abduction (discovery of new sci hypotheses to try), show how it unifies & quantifys a lot of other work on Patt. discovery, induction, etc.

Sept 18, 73: T. work of 148.01–.40, 250.01–.09 has been more or less sloppily completed. No proofs, but strong conjectures. 300.01 – 302.40 is a review of the results. 250.00 remains to be done. I will now take a bit of time off to Referee's paper on Gramm. Induction.

398.01

Tng. sags. — on Tng. Sags.

There are some recent (this y.) refs. Also, I would do well to try to make an index of older references. There are various direct references to tng. sags. in the set of Indexes in MF.O. 200. Also, see notes on my indexing of the Burnt papers after y. '65 fire.

Re: The [redacted] Tng Sag. of 2143 — ie. arithmetic learning.

For the "with learning" part, the problem was learning a fn from Algebraic to Polish notation. As we usually think of it, T. Alg notation involve partial ordering of rules wrt precidence.

If the symbols used in Alg & polish notation, are mainly the same, e.g. + ⇔ + ; × ⇔ × , etc., the problem is much simpler than if other notations are used.

I.e. say the input is 4+3=7 ; i.e. we want the machine to be able to write a pgm. → if its input is 4+3= , the output will be 7. The permissable symbols in the pgm are 0, 1, 2 . . . all integers, +, −, ×, ÷, (,), = . | +, −, ×, ÷ are polish operators so +(3,4) means "adds to 4". So we want "3+4=" to be translated as "+(3,4)".

We are given the i.o. pair 3+4=, 7

One way to solve this general type of problem, is to "work backwards". I.e. we have many examples, = occurs in all of them & is the not relevant to this set. If the examples are all of the form $A(\pm)B = C$, then we factor them into the sets $\alpha \equiv \{A_i + B_i = C_i\}$ & $\beta \equiv \{D_j - E_j = f_j\}$.

We then try to solve each of these 2 sets individually, by trying to combine A_i & B_i in the machine instructions, to get C_i. A natural thing to try is

$$\begin{pmatrix} + \\ - \\ \times \\ \div \end{pmatrix} \begin{pmatrix} A, B \\ B, A \end{pmatrix}$$

— only 8 trials needed to get the both right answers. — An average of 4 trials

Tng says

01: 380.40: If it is suspected that the "+" in t. input à t. "+" in the machine lang. mite be similar, we try

+ (A, B) than + (B, A) — so we get t. rite answer rite off!
Same order Reverse order.

Other impl. ideas: ① t. idea of "an object" i.e. if (3+4) occurs in an expression, we can rewrite t. expression as "7" à t. expressn. will be equiv.

② T. idea that expressns w. fewer symbols are easier to evaluate. — so whenever we can do ①, this reduces t. no. of symbols in an expressn à is ∴ usually desirable.

Ideas ① à ② should make t. idea of "precidence" in alg expressions, come easier.

So, in this Tng sep. TM, system, I have a sequence of "hours", along with a sep. of "solns". Hours are for TM_2, "solns" are for TM_1. At any particular pt. in time, a "soln" is an (I, O) operator. — So here we have a very nice example of a TM_1, TM_2 combination, but TM_1 cannot = TM_2. I am TM_3, supplying hours for TM_2.

Eventually, TM_1 can begin getting into problistic problems, — problems of improving programs, problems of devising hours — So we can train TM_1 = TM_2.

Perhaps at a not very advanced pt. (say when TM_1 is able to extrapolate using CF Grammars) we can present TM_1 w. TM_2's entire seq. of hours, à ask TM_1 to suggest good trial hours from this seq. — Hvr., for serious hour guessing, TM_1 should be at t. pt. where it should be able to "reason out" why certain hours are good, à construct new trial hours based on principals

383.20 spec
400.01 ff

N.20.73 PLAN

TM398

01: 250.40: See 341.28(BM) for Gen'l plan: A More detaild descr. of t. plan of work on TM propar: Gen 376.18 – 378.40 is fairly Good. Other discussions of General Modes of Constructing TM's should be looked into → See INDEX 281.28 RT for some refs. Also stuff on "Tng Seqs" is good. Gen 375.01 – 376.17 is also good.

.08 As for t. Mechanization: see SFTM 387.01 – 388.40. This is pretty much t. old (Early 1973) idea of having a large, slowish common memy, used by many arith units in ll, that

.10 have each their own cache memy & cannot write into t. common memy.

This latter assumes that t. main problem is more or less search w/o. any interaction betw. trials. To t. extent that this is true, .08 – .10 is a v.g. method. If, on t. other hand, trials are very much determined by t. results of previous trials a/o comptns., then it is not clear that that system is at all good.

On t. other hand, for essentially sequencial problems, serial hardware that is very fast, is much better than lots of slow parallel hardware — so modern computers could be far ahead of t. hum. brain in this respect!

Marvin suggests that in most present AI pgms, very few trials are used (say < 10), & it may well be that I work is essentially sequencial.

Anyway If one has a lot of TM-type problems to do, at t. same time, it is probably possl. to multiplex various sub-operators betw. t. various problems — so All in all, it is t "IPC" needed by each problem, that is important.

My present impression is that I should first spend a lot of time on t. Tng Seq. T. seq. & t. soln. should be written out in English in as much detail as possl.. Note that it may be necy. to introduce "side" tng seqs. so that certain concepts, theors, ect., are available at t. rite times,

I now, presumably, have available to me, more tools for doing tng seqs. than I had in 1962. Hwr, it would be well for me to have some of t. problems in solving this tng seq. in Mind as I review t. 1962–1973 yrs. — i.e. I may find suggestions for Solng.

Perhaps Parallel w. t. direct work on T.M., write "Study Paper" or "Position Papar", etc. for CIAP. Consider Meeting, Informal "College", visting people, writing lots of letters, & tc.

M26.73 PLAN Tue Seq

TM400

.01 : 398.40 : 378.40 Gen Reviewing this old material is very stimulating !.

Some recent ideas on what the "Main line" should be ;

Again 376.18 – 378.40 Gen — but some new ideas!

In my working on the problem, it is not necessary to start at the beginning of a tug seq. & work up to complex probs. Instead, work on probs. that are interesting, that I have ideas about. Later I can try to fit the probs I've worked on into a tug. seq. & make sub tug. seq. "bridges" betw. them.

Some problems: (1) My newest view of "artificial learning": That if it's properly presented it will take few trials (say 2 or 8 only); (see recent "Tug Seqs" for discn. (2) That the machine should use concepts that a human could use. — e.g. the idea of a "quantity" in arith learning is the soln. of equations.

(3) In solving equations or systems of eqns: There are many ways to solve eqns. We may want TM to be able to do several of them. Tg. devry of each involves several impt. concepts that would make that devry likely.

e.g. One way of solving systems of eqns involves successive approxn. For a computer, this can be very fast, because computation of pts. is very fast. Some ideas needed (a) T. idea of "closeness" in numbers, so TM can know that an approxt. soln. is almost adequate. (b) T. idea of continuity – that small changes in params will produce small changes in the system. (c) Some topological ideas — That if the $f_{scalar}(\text{system}) = a$ & $f(\vec{s}_-) = b$, then there are usually one or many paths of values of $\vec{s}$ that pass thru all pts. from a to b.

Using ideas (a), (b), (c), it is possl. to devise successive approxn. methods, by "trapping" the soln. within a manifold & cutting down the size of the manifold with ideas (d) The idea of Taylor expansion, w. gradually. higher order coefs probly ↓, one might be able to devr. Newton's method for more clever methods

SN For economy on cost & machine should be as sequential as possl. e.g. Sequential arithmetic means that only as many bits accuracy as are needed, need be processed. Also "Look Ahead" in arith units is wasteful of cost.

401.01

N26·73 PLAN — Tng Seqs. TM401

01: 400.01 : Another set of impt. ideas ~~are~~ necy in solving equs. by t. usual methods of algebraic manipulation until we get x = 3, say. Also, t. idea of elimination of variables ~~[illegible]~~ (by substitution or (subtraction, etc.)) in solving systems of equs. i.e. that if one can solve 1 eq. in 1 unknown, then one should ~~[illegible]~~ try to get 2 equs in 2 unks. ~~[illegible]~~ "reduced" to that state.

By writing up schema (in English) for t. soln. of more difft probs., one will know what concepts have to be developed in earlier parts of t. tng. seq.

Some other very Big problems: Inventions: neg nos, ("why" $-1 \times -1 = +1$.) fractions, irrational nos, imaginary, complex nos., A^x where x is ~~complex~~ non integral, ~~[illegible]~~ complex, vector, matrix. Get motivation for these inventions. What Gore (?) would make them good inventions? Defining new functions like e, e^x, $\ln x$, $\sin x$, $x!$. $x!$ for non integral x; T. idea of analytic continn.

Calculus. Try dif cal. first, then int cal. } see if either
" int " " " diff " } is superior.

Learning PSGd: More specifically — learning to translate commands written in some simple command lngg — into machine instructions).

Learning to "Read" — in t. sense of get info from text & put t. info into a rapidly usable form.

.30 So: For very difft problems — like invention of Relativity, Q.M., etc., an historical analysis would be useful, to try to find t. ideas that were "around" that made those discvys "reasonable". Very probably most of t. ~~[illegible]~~ analysis that has been written on t. history of Sci discy is a lot of foolishness — but it may be that ∃ certain good writers.

Hopefully, I may be able to recruit some useful interest in Phil of Sci & His. of Sci departements, using this pitch — Since what I have is something like a real objective criterion for t. (rel) goodness of an "explanation". —→ Also see Pol 373.33 402.01

01: 401.40: Also, it mite be possl. to use work of other A.I. workers. Any work on learning pgms mite be usd — e.g. Sussman's & Goldstein's pgms. Or, perhaps more commonly — Given S's or G's pgm as a working device that can solve certain kinds of problems — what is an adequate set of abstractions that a machine would have to have, & what kind of typ sep. would it have to have to learn/perform those particular tasks in these ways?

T. reason I mite want to chose heur pgmg. tasks & t. pgms that do them as "things to learn" is that these kinds of tasks & t. way they are solvd, were meant to be pottend after t. way a person mite do them or try to do them.

It would seem like a good idea to get some experience w. simple problems (like arith (learning), befor getting involvd w. t. more complex kinds. Hwr., in t. case of arith learning, I'm not sure I can define t. problem in a way that will make its solu. contain abss. that will be useful in more diflt probs. T. problem of a child learning arith. with learning seems very complex.

Hwr., I could define t. problem as I have in TS 380 — 381. This mite be enuf to get me to more diflt probs. — if not, then add to t. typ sep. or change to a difunt initial problem.

Anyway, if I do have input in alg. notation, it would be necy for t. machn to learn precidence rules; a/o that certain things always or almost always can be treated as objects — t. meaning of parentheses, etc. After t. machine learns alg. notation, try teaching it Polish - both forwards & backwards; then perhaps a part of APL, then some ordinary machine pgmg. lang that expresses t. prime things;

One approach would be to start w. Sussman's a/o Goldstein's thesis ideas. Another would be Minsky's Frame Theory, which is meant to be a model of child learning up to a certain pt. of sophistication. In t. case of S & G's work, I'd have to outline t. main heur pts.; 406.01

N·28·73 PLAN T.S.

01: 402.40 then try to devise the sequs - leading from " " a simpler set of hours to those pts.

In the case of Mary's work, perhaps the outline of the entire thg seq. is more or less implied — along w. the seq. of hours used. — but I haven't looked at the "Frames" report yet.

One idea from "Frames"! That in chosing a problem field, we must be careful to try to make it a manageably small universe. — so that we can get essentially all necy hours from that small Universe.

Anyway, in chosing Problems, at first, try to stay away from (1) Human Lang & Grammar (2) Vision! Because probably these have very many special-purpose tricks that don't generalize well to other fields. Also, one should be able to find very seriously diffl creative discovery (say in Math &/o Science) in which these 2 things are not impt. (tho vision & bss. may be impt. in Physics, say).

P. W.

Penn Wtng. Problem 384–399

PW ≡ Pam Wtng Problem : see 397.01 for REV (TPM

01: This is essentially a continuation of the old SVH - since it got mainly into this problem.

Previous refs 350, 349, 345, 329, 325, 27, 324
317.10 318-320? on TPM's

The idea is: Say I have 3 poms, $P_{1,2,3}$. I have their b costs or k cost or prosts of dcrn. I also have f. prosts of t. corpus wrt. each. Using f. full CMI, can I compute f. best (linear or other wts to use for these 3 poms)

SN: Even tho there is overlap of component Dcrns f. all poms method may be correct. It may have to be renormalized, hvr. T. normzn factor may → 0 ? if so, will have to be inserted w. care. Idea is that "t. all poms method" is probably "unbiased". It has extra codes for t. corpus, i.e. many are duplicated while other unbiased 'all poms methods' use less than all possl. codes — i.e. they leave out many, but in an unbiased way. I think t. all poms method is both unbiased & has a normzn factor of 1.

Possl. Criteria for a good combination rule:

(1) t. resultant pom has max expected value of future prosr of t. corpus. (If this is zero for all combinations considered, take some sort of Geom mean prosr/symbol)

(2) smallest expected mean sq. error in future or smallest total sq. error in future (if this is zero for many possl. combins).

Consider t. next 100 symbols of t. corpus. There are 2^{100} possl. seqs. Via CME, t. i^{th} one has probby q_i ($\sum q_i = 1/2^{100}$).

Each of t. 3 poms has prosr $r_{j,i}$ ($j = 1, 2, 3$) for this corpus.

So of t. 3 corpi, t one w. best expected future corpus prosr is that for which $\sum_i r_{j,i} q_i$ is max.

If we give P_j wt. a_j then what we want to select a_j's is $\sum_{i,j} a_j r_{j,i} q_i$ is max, subject to t. constraint that $\sum_j a_j = 1$.

PW

.01: 384.40: Consider 384.37 —— Lets expand q_i in t. form

Gu. by some version of t. "all pams method":

let $R_{l,i}$ be t. pcost of t. corpus wrt. t. l^{th} pam.

" α_l " " wt. of t. l^{th} pam in this particular "all pams Method".

So $q_i = \sum_l \alpha_l R_{l,i}$

384.37 gives ~~$\sum$ ...~~

.10 $\sum_{l,i} \alpha_l R_{l,i} r_{j,i}$

384.34 gives

.13 $\sum_{l,j,i} \alpha_l R_{l,i}$ [illegible] $a_j r_{j,i}$

If, in .13, ~~the 3 pams considered are included in those considered in t. all pams method,~~
we had not 3, but M pams (where $l = 1/M$ as well as $j = 1/M$)

.20 & those M pams were t. same as those considerd in t. "all pams method" — then I guess $a_l = \alpha_l$ for all l —

but can I prove this? We then have .13 →

$$\sum_{l,j} \alpha_l \underbrace{\left(\sum_i R_{l,i} R_{j,i} \right)}_{\text{symmetric}} a_j$$

t. symm matrix has positive real eivalues — so rotate t. system until t. matrix is diagzd. we get $\vec{\alpha} \cdot \overset{\leftrightarrow}{S} \cdot \vec{a}$.

Th. soln. for $\vec{a}$ is normalized $\vec{\alpha} \cdot \overset{\leftrightarrow}{S}$, not $= \vec{\alpha}$, unless $\overset{\leftrightarrow}{S}$ has all unity eivalues — which maybe it does!

Hvr, even if constraint .20 is not true, & we still have only 3 /apica pams to work with, $\vec{a} = (\text{normz})\, \vec{\alpha} \cdot \overset{\leftrightarrow}{S}$ is t. solution.

False! it would be true if we had t. constraint $|\vec{a}|^2 = 1$

Lets go back to 384.37: say

$\sum_i r_{j,i} q_i = b_j$

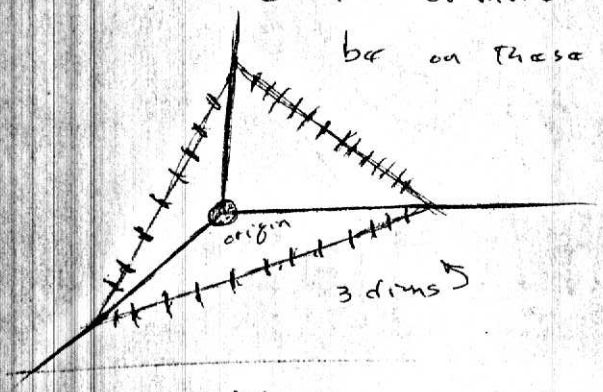
01:385:40 : Now, in general, to solve $\vec{a} \cdot \vec{b} = \max$, w. r. constraint $\vec{a} \cdot \vec{I} = 1$ ~~is a problem~~ with the further condition $a_i \geq 0$ for all i.

In 2 or 3 dimensions, the solution is always

$a_j = 1$ if $b_j = \max$ of all b_i. Otherwise $a_j = 0$,

.05 Another solu. : If n of the b_j's are all equally max.,
.06 then the corresp. n a_j's can be any set of value $\Rightarrow$ their sum is 1.

In 3 dims, $\vec{a} \cdot \vec{I} = 1$ means $\vec{a}$ has its nose in a certain plane. On the other hand, the locus of all vectors $\Rightarrow$ $\vec{V} \cdot \vec{b} = k$ is a plane $\perp b$. If we consider all planes $\perp \vec{b}$, they will all intersect the plane $\perp \vec{I}$ in st. lines. In general, the prize corresp. to a plane of max k, will pass thru 1 or more corners, so the intersection must be on these lines.



This result can be obtained directly in n dims: Consider $\sum a_i b_i$. We know the b_i's. Clearly, if b_k is the largest b_i, then $a_i = 1$ for $i = 1$, 0 otherwise. If several b_i have max value, a linear comb. is OK. (.05-.06)

This is an interesting result! It says that one can always chose just one poem, that is at least as good as any combination of poems.

Note that this does almost contradict the results I got w. linear regression, because the max method was almost one of the 6 linear poems I considered. It didn't have a gaussn. error distribn., but it would if there was either a lot of data or a large no. of coefs (I don't remember which).

Well, perhaps Max is "Best" if averaged over the future rather than the past corpus!

In 384.37 note that r_{ji} and q_i both include both the known corpus & the unknown future 100 symbols.

N.14.73 PW

01: 386.40 ! Well, not so surprizing — that one Pgm should be "Best"! But not that this is "Best" wrt t. "correctly known" ~~future~~ probty distribn. of t. future corpus. TM3.8

→ Suppose that all one knows is ≈ knowls # of Y.3 Pgms à prosts of a sub corpus w.r.t. these pgms.

One could consider t. average (over all future corpi of length 100) of all Pgms w. ~~the~~ cost-bnost of subcorpus past sub corpus = α_1. Compare it w. similare average of α_2 à α_3. Then It would seem that probly Y. one w. min α_i would be best, but t. cross correlns. betw. Y. 3 pgms are not "consulted".

Perhaps a better way to look at it: we regard Y. 3 pgms as each 3 distnt estimators of Y. future. If they were "indip" - it mite be easy to estimate their relative wts. for a better estimate than any of them. However, if they are not "indip" Y. prob. is much harder.

Perhaps we can discover Y. amt. of lack of indip. by sampling probty values from Y. 3 in Y. past corpus. Say use random samples or use entire past corpus.

O.k.! Say each pgm P_i ~~has~~ is a sequence of values ($i = 1/3$)

.25 q^i_j ~~j = t. symbol of t.~~ j = Y. order of symbol in t. known corpus. ($j = 1/50$) say

Say q°_j is t. true probty. (which we don't know).

So we have Y. 3 vectors $\vec{q}^i$. We might be able to estimate their relative wts. for optimum estimate of t. next probty pattern of t. 51st symbol by knowing only t. cross correlns. betw. Y 3 vectors, $\vec{q}^i$.

A counter-example: Say P_1 gave $q^1_j = 1$ for all j,

P_2 " $q^2_j = .5$ " " "

P_3 " $q^3_j = .2$ " " "

It would seem (unless P_1 has by knost), that P_1 is v.g. à that P_3 is very poor. — I don't see how "cross correln/students" would give P_1 any more wt. than, say, P_3.

58 → One approach: Assume that Y. corpus is a determinstic seq. — so probys are all 0 or 1. In this way t. (error)^2 for each component of $\vec{q}^i$ will be $(1 - q^i_j)^2$. 330.01

N·15·73 PW TM390

01:389.40: In this way, we can give each pam wt ∝ its (variance)$^{-1}$.

2 troubles w. this!

1) We haven't taken the cost of pams into account. We may be able to do this by computing expected future variance of each pam.

2) Cross correlations betw. the pams were not considered — but this can be done.

.08 Getting the rite coifs is (as I'm modeling it now) identical to the linear regression problem. The linear regn. soln. is exactly correct for a n-dim. Gaussn. distribn, à / suitable a priori of coiffs. (uniform) — we get the Max Likelihood Soln.

The correlation matrix of interest is the 3×3 matrix

$$M_{\ell m} = \sum_{j=1}^{50 \leftarrow 389.25} q_j^\ell q_j^m .$$

The constant vector (components are) $K_\ell = \sum_{j=1}^{50} q_j^\ell \cdot 1 = \sum_j q_j^\ell$ ← the "y" values always $q_j^0 \equiv 1$.

.18 The wts. are then $\vec{X} = \vec{M}^{-1} \cdot \vec{K}$.

SN A "criticism"! We could just as well have used $q_j^i - 1$ instead of q_j^i $(i = 0|3)$, & we'd have no eqs. — but then, $q_j^0 - 1 = 0$ always so $\vec{K} = 0$ & we get nothing! Woops! But then we would have to allow a constant to be one of the pams — so it may be OK. (This is equiv. to a constant term in the regression)

While .08 – .18 takes cross correln. into account, cost of pams is not considered. Perhaps we should do this by modifying the $\vec{K}$ coifs. In my proof that Willis' information/error convergence implies Σ sq. error convergence, I got a relationship of proportionality betw. the 2. I can use SVH ideas to estimate ↓ of mean(-ln probty) error. — how can I relate this to ↓ of $1 - q_j^i$ error?

In general, we may always want to introduce a constant term. — But if we use q_j^i as the variables, the constant term will correlate w. q_j^0 ·100%! thus destroying the whole thing!

SN Another criticism of linear regression is that the sum of the coifs. $\sum_{\ell=1}^{3} K_\ell$ may not = 1. (i.e. $\sum_\ell X_\ell$) ·18 for def of $\vec{X}$ 391.41

pl: 390.90 → Another Guess at a Soln: Just do 390.08 − .18, & obtain $\vec{X}$. Then mult t. ~~[illegible]~~ coifs of $\vec{X}$ by t. proper ~~[illegible]~~ prosts of ~~[illegible]~~ dctn of corresp^g Pawns ! [illegible]

In t. case of linear regressn., we chose t. wts. so that t. resultant Pawn has least ms ~~error~~ in t. known corpus. A corresp^g ⇒ SN | T. coifs may occasionally be <0 — & give probly values that are ⇒ method would ~~[illegible]~~ chose coifs so that t. prost of t. known corpus w.r.t. t. linear Σ corpus, would be max. — Hvr. again, we did not consider prost of dctn. — so a Pawn that got $q_j^i = 1$ for all j ~~[illegible]~~ would get wt = 1, even tho it ~~[illegible]~~ was completely a.h. (i.e. large & cost). → 392.20

In t. linear regression approach to PW: what I would like is that in ~~[illegible]~~ 390.18 t. coifs of $\vec{X}$ & $\vec{M}$ are debiased w.r.t. infinite future corpus. The ad hocknass due to non-zero dctn costs perhaps makes t. coifs of $\vec{M}$ as derived from t. known past corpus, about rite — i.e. no reason for systematic error. But t. coifs of $\vec{X}$ are always excessively large due to ad-hocknass. → No!! → see .38

One way to estimate how much $\vec{X}$ components should ↓:

Say S is t. length of t. known corpus, & A_i = $-\log_2$ prost of Pawn dctn. Then

$$X_\ell \to X_\ell \times 2^{-\frac{A_i}{S}}$$

→ for all S symbols of t. corpus

$2^{-\frac{A_i}{S}}$ is about how much each of P_i's probly estimate is "hy" due to ad-hocknass.

This is $\left(2^{-\frac{A_i}{S}}\right)^S = 2^{-A_i}$ — or a cost of A_i'.

T. components of $\vec{M}$ can be similarly corrected, but $M_{\ell m} \to M_{\ell m} 2^{-\frac{A_\ell}{S} - \frac{A_m}{S}}$

01: 391.40: $2^{-\frac{A_i}{s}} \approx 1 - \frac{A_i}{s} \ln 2$ if $\frac{A_i}{s} \ll 1$.

similar approxns for $2^{-\frac{A_e}{s} - \frac{A_m}{s}} \approx 1 - \frac{A_e + A_m}{s} \ln 2$

There may be a simple approxy. effect on $\vec{\vec{M}}^{-1}\vec{x}$ that is gn. by these corrections.

06 → Another impt idea: In my recent work on SVH, I got several diffnt. corrections to obtain expected future post/symbol. These more exact corrections should be used instead of $2^{-\frac{A_i}{s}}$. The difference is, I believe, a sizable factor, under most conds. → see 399.18 for some Q's

10

09 Also note: T. cross correlns in $\vec{\vec{M}}$'s components can be obtained by sampling — i.e. it is not necy to do all of t. cross products.

20: 391.15: It is possl. to write $\prod_{j=1}^{50} \left(\sum (a_i q_j^i) \right)$ = prod

& then try to chose t. a_i's so this prod is max. —

→ one can "correct" t. q_j^i's using t. methods of 391.30–.40 (or more exactly 392.06–.10).

There may be a good way to do t. expressn. of .20 — but at first glance it looks like a real diffl. thing. If we only have 3 or 4 comps, then we could do a lot of runs w. diffnt. values of $\vec{a}$ — a H.C. problem. If t. hill is suffly smooth & non-bumpy a soln. may not be hard to get — one can use various tricks to speed up t. work.

32 → One rapid way to evaluate t. product of .20: add $\ln q_j^i$, 33 but chose i's randomly w. probly a_i. This method can perhaps find a ruff max, & then more exact evalns can be made in t. "good" region. → Wrong! see 393.23

Another/MC way: when always chose j at random ($j = 1/50$) — uniform distribn. If one has tables of $\ln q_j^i$, one can really do this very fast. (Flt pt. add may take too much time, so use fixed pt values of $\ln q_j^i$. So we just → 393.01

N·16·73 PW

TM393

.01: 392.40: Keep chosing these random values and adding them on, using an exponential update (x window) to slowly change the $\vec{a}$ values.

→ There remains a basic Q. of whether the method of 392.32–.33 actually gets the correct expected values of the products.

While the Forge method does give a lot of noise, remember that even if one does it "exactly", there is still a lot of noise due to finite SSZ — so one can perhaps chose the "x wind" value to be the scanning parameters so one gets a maximum expected rate of hill climbing per cost of computation

Note: To do the Forge analysis, one need only have a set of sample pts for P_1, P_2 & P_3 — they need not, however, have the same j values for the same i values. However, the mode of selection of samples must not have a systematic bias.

Actually, the Forge — MC method can be applied to any type of Gore — to not only max pcost, but best ms error — or any other

---

.23 → Well: the method of 392.32–.33 is wrong!

.24 We want $E(\ln(a_1 q_j^1 + a_2 q_j^2 + a_3 q_j^3))$ — that the chequ would get

Vs $E(a_1 \ln q_j^1 + a_2 \ln q_j^2 + a_3 \ln q_j^3)$.

To get .24; $\ln(x) = (x-1) - \frac{1}{2}(x-1)^2 + \frac{1}{3}(x-1)^3 \ldots$

.28 $\approx E\left(\sum a_i q_j^i - 1 - \frac{1}{2}\left(\sum a_i q_j^i - 1\right)^2\right)$ [illegible].

The three partial divives wrt a_i's give 3 linear eq's in 3 unks.

.32 The coefs [illegible] of the a_i's are probly $E_j(q_j^2 q_j^m)$, or some similar thing - woops!

.33 in .24, $\frac{\partial}{\partial a_1} = E\left(\frac{q_j^1}{\sum a_i q_j^i}\right) = 0$

However, since the q_j^i & a_i are always ≥ 0, I don't see how this is possl.

As a result, it must be that we are at an edge — i.e. either a_1, a_2 or a_3 must = 0 or 1. Perhaps we can use this argt. to show that all but 1 of the a_i's must be 0!

Woops! I forgot the constraint $\sum a_i = 1$

→ 394.25

394.01

N.17.73 PW

TM394

.01; 393.40: The expansion of 393.24 -.32 can be just carried to 2nd degree — in which case the eqns are solvable, if not too many parms are involved. If one does it to > 2nd degree, the eqns can be solved by h.c. methods. We need the hyper order cross correlns of the various parms (e.g. $E(q_j'(q_j^2)^2)$ etc. These can, as before, be obtained by sampling.

.10 Since we also solve 3 linear eqns in 3 unks in the "linear regression" method of 391.01 ff, it may well turn out that the soln. of it as well as the quadratic case of 393.24 -.32 are identical! This would be easy to check.

.17

.18 The corrections of the forgg. solns for the cost of parms ($\equiv A$) > 0, can be done to make corrections to the correlns & cross correlns. as in 391.30 -.38.

Alternatively, it may be possible to insert these corrections after one has the solns. for $\vec{A} = 0$. In such a case, it will be a lot clearer as to just how the A_i's effect the results. However, eq. .26, if true, suggests that this (latter) may not be very easy.

.24

.25 : 393.40! (Perhaps the 393.33 eqns and upwd $\rightarrow$ $E_j\left(\frac{q_i'}{q_j}\right) = \lambda$ · const for all i. (i.e. $\frac{\partial}{\partial a_i}\left(\ln q_j + \lambda(\sum a_i - 1)\right) = 0$ YES! $q_j \equiv \sum_{i=1}^{3} a_i q_j'$ $\rightarrow$ 396.23

.26 $\rightarrow$ In the SUM discn. of A_i (the cost of discn of P_i), we get an upper bnd on an estimate of the cost/symbol of the future corpus. We can, then, expect a fair amt. of 1 sided error in such estimates. An impt Q. is — what is the effect on such discns as 391.30 -.38 in general, on the optimality of the recent results using corrections like 391.39 - 38? An analysis like .18 -.24 might make this a lot clearer! — but eq. .26 (if true) suggs that this might not be poss!.

395.01

01: 394.40: Well, O.k. say we do have means for combining Poms linearly or any other way, so that a Gorc (MS error or prost) is extremized. If we do take t. k cost of Poms into account via ≈ 391.30–33 — of what signific. is this? — Clearly we will end up w. an approach b cost/symbol ≤ any of those expected from any of t. component Poms. Will this "combination Pom" really on t. average, be any better than t. components?

Remember that since, if we have n Poms combined, we have just adding in an ad-hocness assoc w. that n! If t. combin rule doesn't gain us more than we lose via that n — then we will do better by either using t. "best" Pom or a combination of < n Poms.

.20 I think t. exact value of t. ad-hocness added by this combining of n poms can be obtain via t. method that I used in analysing linear regression (q.v.). To do this, it would be necy to know t. $\frac{\partial^2}{\partial a_i^2}$ (pcost Gorc) — or corresp thing for any other Gorc. (Again, note that w. t. Max Method, I didn't use this simple Gorc, yet t. calcns were not diffly. Ingeneral, this $\frac{\partial^2}{\partial a_i^2}$ can be obtaind by obtaining 2 2n+1 empirical pts. — n being t. no. of dims. of $\vec{a}$

Anyway, it is clear that while I have derbd but 2 combination methods for Poms (actually, 1 method (linear) w. 2 diffnt Gorcs), there are an ∞ of such combn. methods — the Max expected pcost being t. "ultimate" Gorc. Which are "best" or even "reasonable," I don't know.

An impt thing in the Pom combination is getting the "A" value (of pcost) of t. combination rule. Perhaps we could devise a combination rule like t. Max Method, so this A value would be zero. Note that t. "A" value is not "zero because it is optimum" — but "zero because t. rule is fixed befor 396.01

N.18.73 PW

01: 395.40 t. corpus is seen — i.e. it is a truly "a priori" method

or → {If I have a fixd no. of pams that was obtaind in no particular way / Then I suspect that no $A=0$ method exists. Certainly t. Max / method (a $A=0$ method) is not this sort — i.e. t. pams are chosen apri, & wts are gn. to them apri. [e.g. w. 3 pams 2 may be almost identical]

So, it would appear that I would do well to try to devise special combn. rules. Linear is, of course 1 type — using

.10 t. E (post corpus) Gorc. Another trys to combine several pams by devising obs to detect when each of them is expected to be unusually good. I can devise an upper bnd for t. "A" of such a combn. method — it is t. prost of t. combn rule plus t. prosts of all t. component pams — but I suspect that there is a better way to look at this (/than regarding t./combn. rule plus t. derns of all t. component pams, as a new pam w. a large dern). [i.e. better dern off.]

→ Another Approach (not necly difint from t. forgg) is to try to see how Humans combine pams.

My method of dealing w. t. shness's due to A_i's (i.e. $q \to q \times 2^{-\frac{A}{5}}$) will have to be analysed more rigorously.

.25: 394.26 To solve $\sum_j \left(\frac{q_j^i}{q_j}\right) = \text{const}$. $q_j \equiv \sum_i a_i q_j^i$

① Let $b_i(\vec{a})$ for some particular $\vec{a}$

To get t. next approxn to $\vec{a}$, use t. deviation of t. $\vec{b}$ components from constancy, & use this to correct $\vec{a}$.

② Compute t. first & second dirivatives of $\vec{b}$ wrt. $\vec{a}$ exactly. Solve t. resultant n linear eqs in n unks, for t. $\Delta\vec{a}$ components for t. next approxn.

$$\frac{\partial b_i}{\partial a_k} = \frac{\partial}{\partial a_k} \sum_j \frac{q_j^i}{\sum_\ell a_\ell q_j^\ell} = -\sum_j q_j^i \frac{q_j^k}{\left(\sum_\ell a_\ell q_j^\ell\right)^2}$$

so t. 2nd partial is simply t. cross correln. w. wts $\left(\frac{1}{q_j}\right)^2$

t. first " " " " "mean" " " " $\frac{1}{q_j}$

∴ not diflt to compute — certainly about as easy as

.39 doing t. linear approxn. 399.01 Spec 397.01

.40 T. set of linear eqs to be solvd. is of t. form $\vec{a}\vec{M}\vec{a} + \vec{B}\cdot\vec{a} = 0$ this has → 399.25 Spec ← No!; see 399.26

01 : 396.40 Review of 384.01 – 396.40

T. Basic idea of P-w W'tng is : Given a bench of Pems, how should we best combine them into a new pem, at least as good as any of t. components?

One method is to use linear wtng of t. n component pem probys. The wts, a_i are chosen ⇒ $\sum_{i=1}^{n} a_i = 1$ & t. prost of t. known corpus is max.

If q_j^i is t. probty of t. j^{th} corpus symbol wrt t. i^{th} pem, we

.10 chose $\vec{a}$ ⇒ $\prod_{j=1}^{S} \left(\sum a_i q_j^i \right) = \max$. [S is t. corpus length]

Because the q_j^i's have been obtaind w. adhocness ($A_i \neq 0$), the q_j^i values must be reduced (391.30–.38). The correction $q_j^{i\prime} = q_j^i \cdot 2^{-\frac{A_i}{S}}$ is not unreasonable, and these $q_j^{i\prime}$ values, rather than q_j^i should be used in .10. Better correction factors under various conds (392.06) are discussd in t. recent work on SVH. T. difference in correction factors can be appreciable.

The n–1 degrees of freedom of $\vec{a}$ give an additional ah-ness to t. result (395.20); this can be computed in a manner similar to that which I used in my most recent analysis of linear regression, & t. Max method

Some Mathematical techniques of approxn, involving taylor expansion of $\ln(1-x)$ are gn. on 393.24 – 394.26. Also, because approxns are used, statistical sampling can be used to obtain various corrs & cross corrs. (292.09).

An alternative Govt using MS error, is discussd in 389.38 ~~391.38~~ This may give t. same results as t. more exact max prost Govt if only upto quadratic cross-corrs are used in t. approxn (394.10–.17)

A Genl. discussn of t. Signifc. of these Results (396.02–.40): That t. forg. combn. methods will often be important is that these methods of dealing w. ahness may be good approxns. Hvr. other methods of combining Pems should be considerd. 396.10 is an impt type. watching humans is another.

A more rigorous analysis of the $q \to q \times 2^{-\frac{A_i}{S}}$ correction is needed.

N·22·73 PW (see 397.01 for REV) NO!

396.39 spec

01: 397.40 : Note a serious/kitty in Y. maxm of $\prod_j q_j^{N_j}$

If one of Y. pams gives $q_j^i = 0$ for an event that occurd, that pam will get zero wt — even tho its other q values are v.g. (even if they are all 1 !!) This argt. is also true to some extent if q_j^i is very small for some i, j

This may be some sort of inherent weakness of Y. use of $q_j = \sum_i x_i q_j^i$ (i.e. linear combn of pams).

— Or could it be a fault of Y. Gore ?

On second Rot .01 ff is all wrong. If $q_j^i = 0$ this does not cause any trouble w. Y. i^{th} pam.

8 On Y. other hand, if $q_j^i = 0$ for some i, j, then Y. post of Y. corpus for that pam is, indeed ∞, so we would get very small expected future post/symbol for that pam. However, using Y. corrections 391.30–.38 (like $q_\ell^i \to q_\ell^i \times z^{-\frac{N_i}{s}}$), it would be OK.

We might run into trouble if we tried to use Y. more sophisticated corrections of 392.06

25: 396.90 2 solns: $\vec{a} = 0$ and $\vec{a} = \vec{M}^{-1}\vec{B}$ ← No!

26 This is wrong! The we want to maxz $\vec{a}\vec{M}\vec{a} + \vec{B}\cdot\vec{a}$ wrt $\vec{a}$.

We do end up w. something like $\vec{a} = \vec{M}^{-1}\vec{B}$ or $\vec{a} = \frac{1}{2}\vec{M}^{-1}\vec{B}$ so its OK. !!

SP 447 - 449

Stochastic Part.

D2473 SP ~~Planning~~ ← see help into copy [make copy] copy 1 TM441 (3)

.01 i.e. initially a large set of poss. solns). ~~[crossed out]~~
In the task net approach, the plan is a sort of "factorization" of the set of solns. One solves each factor by narrowing it down. More generally, factorization of this form is not poss., and one must somehow narrow down the soln set "as a whole" (or perhaps not as a whole, but not in factored form).
This working on the plan in not factored form is often done on the specific (subgoals / factors) of a larger plan. It can be done e.g. if the goal is recognized as being of a particular type and being amenable to certain (non-factoring) approaches. → 6.01 .18

How to construct plans

Is the task net adequate to cover all poss. solns?

May 66 CHM paper

(SN) On SP: I suspect that normally one does not solve large task nets using probty curves for each known subtask in the net. Instead, there is a certain set of net forms that are readily solvable (perhaps by table, analog or MC means) and the form of any actual SP net is "reduced" to one of those patterns as an approxn. Some methods of "reduction", are to express a set of ∥ or sequential tasks as a single task, or any single input, output object sub net as a single task with suitable probty curve.
Viewed in this way, if a large task net is decomposable into series, parallel subnets, one might be able to solve it, by obtaining a G curve (i.e. probty of soln, failure v.s. time) for each ∥ or serial set of tasks — like an Ohm's law soln. of a Resistance network.
Furthermore, if one could do the analog of Δ to Y and Y to Δ xfms, one could perhaps solve all nets!

D 2473 SP

It would seem to me that the  ;  x fmns are always possl. — The f. resultant G curves are not ordinarily in the same functional forms as those of the component tasks,

In particular these x fmns can be very worth while if there is a nearest or sibling dependence in the G functions of the subnets ← so one would try to design subnets so as to get a net w.o. such dependencies! If the dependencies are outside the set, then one is in trouble!

The f. sov-pov. idea still needs some thought,

Consider the $Y \to \Delta$, $\Delta \to Y$ Q. Just what would it mean if it were true? — say $Y \to \Delta$:

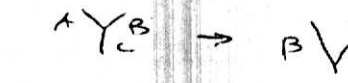
Say we had the sub net : 

This would mean that dW unit of work on A, say, is equiv to $f_\alpha dw$, on α, $f_\beta dw$ on β ; $f_\gamma dw$ on γ.

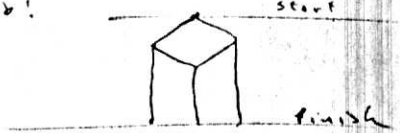
Any motion up or down on the G curves of α, β, γ is mapped into corresp. motion on the G curves of A, B, C & vice versa.

But: while doing A & C is equiv to doing β
& " B & C " " " " α

I don't know what α is equiv. to — if anything!

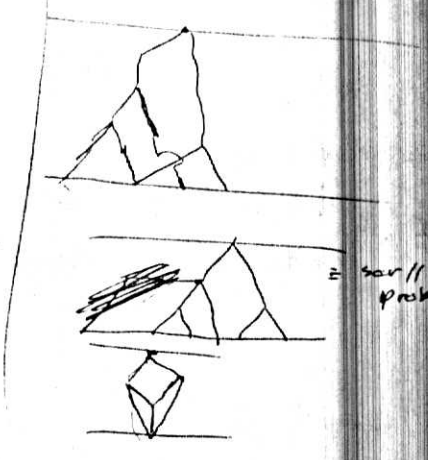
Perhaps the xfmn is  .

Also 

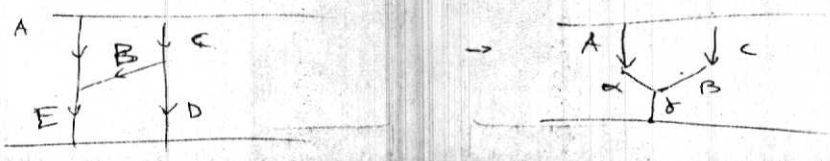
A possl. prob: 

or 

Note diodes, ABE is a soln., but not ABD.



D2473 SP



soln. of **E** ≡ soln. of α & γ

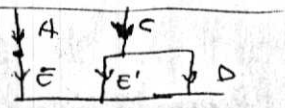
soln of **D** ≡ soln. of α & β

" " **B** ≡ partial soln of α & β.

This is f. same as

with E & E' being 100% correlated

(i.e. identical).

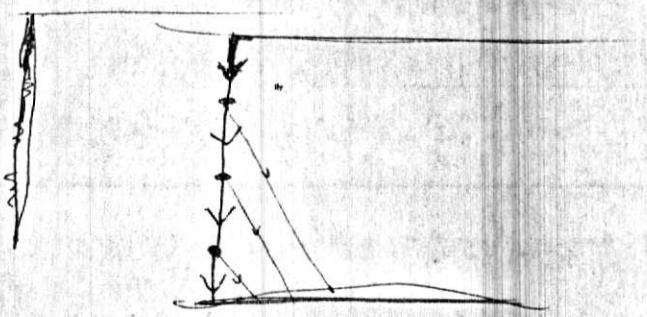


Perhaps: A single task w. a gn. $G_{1,2}(f)$ curve

(E ≡ probty of success, failure, other effort f) can be approxtd

by a sequencial set of tasks w. (a) uniform completion times,

(b) probty of success or failure ‖ not so clear!



VHM (old) 345 – 17

(23) (24)

Sept 24, '73

VHM = SVH (old)

01: 294.40: Re: T. "Thrm" of 2.99.03: If we can just show that t. error in using $2^{-k\,\text{krost}}$ of a corpus, is within some known fact bounded of t. true aprip of that corpus, we can use $2^{-k\,\text{krost}}$ as an approxn. to aprip — (say for a Poem or an operator or grammar) — & with this bounded error in krost, we can be assured that for a long enuf extra (over complete CMI) ss =, we will get O.K. convergence.

↑ is 299.40 meant?

This would legitimatize t. use of krost for aprip of "patterns" & give us idea as to upper bnd. on error (≡ non-optimality).

D2173: Discuss w. Willis: He said he had proved something like krost is within ±1 of krost, & that he would try to find t. proof & send it to me. He gave me t. idea that this relation was true of a certain particular Machine. It certainly is true of t. machine generated in his thrm. 12 (Poem → FOR Thrm).

Re!
Anyway, At. 299 thrm: If we make t. thrm say t. no. of 0's = 1000 x t. no. of 1's, then our units are 1000 0's & 1, 1 — which makes t. grain size quite large. Another way to do it, here — t. no of 0's = (n x no. of 1's) + k, with $-500 \le k \le +500$.

Here, we end up with a set of codes that seems to be "unbiased" toward any particular P (set of Poem(s)), yet it is a very probly univ. code.

[Hrr., codes w. ≈ 1000 times as many 0's as 1's are probly good particularly good for something! — Just what is this something — just what Poems?]

→ Anyway, Poem(Thrm 12), Say we have a UMC, M_U, & a Poem P_0. M_U can derb. P_0 in k bits, in t. sense that $P_0 \leftrightarrow R_0$ (R₀ is an FOR) ≡ & $R(\le k)\, M_U$.

Consider t. set of all CPM's; P_i. Say we want to know t. krost of a gn. corpus / C, wrt a gn. UMC, M_U. We know krost of C wrt M_U. What is (long) bnd on (log₂) krost of C wrt M_U? We can use M_U to derb. ≥ 1 R_i's. ($P_i \equiv P^{(R_i)}$)

(The R_i's have t. cond. that their krost ≤ log₂ krost + 1.)

→ 12.01

1574 (VHM) Previous big Reference exists.

345.90 : On relation of Kcost to Pcost : Letter from Willis : 1.2.74 : TM12 1,2,9

Say we have a umcuom : $0^\frown\alpha$ gives a universal machine for α input.

$M_U(1^\frown N^\frown 0 1^\frown\alpha)$ gives as output ~~|~~ $M_U(0^\frown\alpha) = M'_U(\alpha)$ & w. M'_U being a UMC.

→ The output of a C.B. on M'_U with α as input.

This CB on M'_U is t. τLU with $\tau = F(N)$. N is a binary no. coded by doubling digits so 0110 binary → 00111100, with $F(00111100) = 0110$. 01 is a sort of punctuation, so one knows when "N" ends.

N_U is t. same as M_U, except that $N_U(1^\frown N^\frown 01^\frown\alpha)$ gives the output of t. FOR $R^{p(\text{C.B. of } F(N) \text{ on } M'_U)}$, This latter being constructed via W's Thrm 12.

M_U is (1) a umc
(2) has at least 2 kinds of codes for a corpus ——

(a) t. normal codes via $0^\frown\alpha$.
(b) codes via $1^\frown N^\frown 01^\frown\alpha$.

The (a) type codes have pcost, P, say.
The (b) " " " shortest dern of length $3 + \log F(N) + \lceil \lg P' \rceil$ "ceiling" (P' being t. pcost of t. corpus w.r.t. t. C.B.

If $F(N)$ is large enuf, ~~so that~~ ~~M'_U's shortest code for t. corpus of~~ P & P' are very close.

If instead of $\tau = F(N)$ we let τ be a very rapidly ↑ funct of N, we have a pcost of P for t. corpus, & a "shortest dern" of length $\approx 3 + \lceil \lg P \rceil + G(N)$ where G is a very slowly ↑ funct of N. It's like lg of t. inverse of this function.

30 Q.: Say R_T are successive T limited CB's on M_U (a umc). Say we construct t. sequence $R^{p(R_T)}$. Could we do so in a way that would yield a "limit umc"? This Limit machine would have t. property that for every corpus, $A^{(Q)}$, t. shortest code is of length

$$-\left\lceil \log P^{(\text{limit Mach})}(A^{(Q)}) \right\rceil.$$

Whether or not this is possl., t. limit would certainly be a MSM, & probably provably convergent. Whether t. MSM would be umc is unclear.

13.01

≡ SVH (old)
1774 (VHM) [Relation of kcost to prost]

TM 13

.01: 12.40: Perhaps it would be best to try to prove t. opposite!
Some possl. Thrms:
.03 1) $\forall$ k > 0, $\exists$ a umc M_u $\ni$ for all $A^{(\ell)}$ (or for at least 1 $A^{(\ell)}$)
$-\log(p^{(M_u)}(A^{(\ell)})) <$ (kcost of $A^{(\ell)}$ wrt M_u) $- k$.
i.e. $-\log$ pcost (Brost) & kcost difer by an arbly large amt.

.06 2) for every umc., there is a corpus $A^{(\ell)}$ $\ni$ $-\log$ pcost & kcost differ by an arbly large amt.

Say 1) is true i.e. $\exists$ one umc. w. this property, — then I think 2) must be implied. Yes!
Because: Say 2) is false, so $\exists$ one umc. $\ni$ kcost + log Pcost is bnd for all $A^{(\ell)}$

Def: Bcost ≡ $-\log$ Pcost: There is no possy. now of confusing Bcost w. "t. length of t. shortest code" — since this is defined to be kcost. I think I'll henceforth let Bcost ≡ bcost.

So if kcost − bcost is bnd for all $A^{(\ell)}$ for a gn. umc, it is bndd for all umcs. Because — say its bndd for M_1:
so: kcost − bcost < k_1. Consider a umc, M_2. Kcost of xltn. from M_1 to M_2 is α_{12}; xltn. t. other way kcosts α_{21}.
If $kcost_1$ is t. kcost wrt. M_1 of a gn. $A^{(\ell)}$, then its $kcost_2$ (wrt M_2) must be between $kcost_1 + \alpha_{12}$ & $kcost - \alpha_{21}$.
Similar bnds for $bcost_{1,2}$ make it clear that
$\therefore$ $kcost_2 - bcost_2 < k_1 + \alpha_{12} + \alpha_{21}$ — which is ∴ bndd.

O.K. so just try to prove 1) (.03), since it implys 2) (.06).
Also t. falsity of 1) ⊃ t. falsity of 2)
So: try to find a umc. for which either Bcost − kcost is Bndd
or " " unBndd.
See R 38.18 for "Proof" of bndness of kcost − bcost for UMSM.

11274 VHM
13.40 put in contents en 345.40

.01: 345.40: 299.90 space The idea of extremely biasing the probys of 0 and 1 so that t. "mesh
.02 grain size" is very small; Say we have t. no of 0's =
.03 (k × no. of 1's) ± ½ l (−½ k < l < ± ½ k). So we use these codes,
.04 & select t. code of least b cost — i.e. t. shortest code.

A method close to this: Use t. "Model of t. World" model of CMI:

Here we send a biased stoch. seq. into a UMC — w. probys
ε, 1−ε; — w. very small ε. T. output still has t. same distribn. as w. probys = ½, ½.

This ↑ biasing gives, for normal CMI, a very sharp wt. —
particularly for large l, of codes where
no. of 0's = (no. of 1's × 1/ε) ± 1/ε × √no. of 0's. If we selected
not ± rather best code from this → sum we would get .02, .04.

(5) VH 349 – 45

(73) (74)

[

Skočajske VHM } (1986)

Oct6,'15

SVH

321.12 (321.12) In t. decn. string of "A" bits, clearly some bits will have more effect on prodns. than others. E.G. m.s.b's in the coifs in linear regressn — so one would want to give less penalty to t. S^2 on the account of these bits.

Hvr., Normally, one will use only enuf bits in t. coifs, so that t. total no. of bits in t. decn. (coifs bits + σ^2 bits) is minimal. If one does not minimize here, then A will, indeed, be spuriously large.

.12 "t. General SVH Thm:" If A = 0, then t. "expected value" of t. ncost for t. next symbol will be $\frac{F(s)}{s}\Big|_{s=s}$ or, better, $\frac{dF(s)}{ds}\Big|_{s=s}$. This exp. val. is wrt. all possl. corpi w. aprips assigned to t. corpi via CMI, — T. meaning of F(s) is unclear — See 321.01 .12

Another (perhaps) possl. interpretation of "exp'd val" would be wrt various pams w. similarly assigned aprips CMI

A=0 simply means that one knew, apri, nothing about t. corpus that was coded. This ought to be easy to show, or modify t. thm so that it is true. 27 → 350.01

For A≠0 I will perhaps have better insight after doing .12

SN This SVH Thm is sort of nice (if true in any way). "Completes" t. CMI Theory in a neat way. Via CMI, one can, by 1 CB, approach "true" probys as a limit — Hvr., ~~one never knows how close one~~ while one can ordinarily make limits of error at any pts. These limits do not → 0.

Using SVH Thm, hvr, a CMI w. 1 CB., we have A=0 & so we can estimate, not tot. error in probys, but $\frac{1}{S}$, t. expected value of ncost from t. next symbol. an upper bnd

Actually, this can in some circumstances be of more value than just knowing an upper bnd on t. error in probys

350.01

0.6.73 SVH

TOM 350

.01: 349.40 : (spac 349.27) → Note ↑ form of SVH from on 321.01 -.12

It may be much more or less diffl to prove it in this form!

D3.73 : first note that SVH may be restricted to Pam's for which a mean Entropy exists.

Second. Can I find ~~mean~~ ~~prob~~ "Expected value" of ~~prob~~ prost of next symbol (or of bcost of next symbol) (a) for Pam$_0$ (b) any other pam.

(c) Pam that has a best value of [corpus bcost + Pamprost] or some other Gorc.

Of all Pams, that have a mean Entropy: t. one w. t. largest σ for its Σ(bcost), is Bern seq., $p = q = \frac{1}{2}$.

It's it a max or min!

T. distribn of prost is $2^{-N} \frac{N!}{(N-n)!\,n!}$ - which is a gauss distribn. of (σ) width $\propto \sqrt{N}$.

All seqs are equally likely. (= prost of any (seq).

Hvr., if ∠. $p \neq \frac{1}{2}$, then I get $p^n (1-p)^{N-n} \frac{N!}{(N-n)!\,n!}$

$\left(\frac{p}{1-p}\right)^n = e^{[\ln p - \ln(1-p)]\,n}$ — which moves t. center of t. gauss, but leaves its width invariant! i.e. $e^{-n^2} \cdot e^{kn} = e^{-n^2+kn}$ will have a width that is indip of k. (Seems that I should have known this anyway — it's t. same as t. result on 'errors in estimation of mean of a gauss variable.)

Now — this seems like a fine result — but is it relevant?

Well! just what do I want?: say I have Pam$_5$: Its most likely seq. of Length N, has a prost of α. How many of t. 2^N seqs of length N have a prost betw α & kα? (k = e^4, say)?

Perhaps a no. (indip of N) will have a cost betw. $\ln\alpha$ & $\ln\alpha + \epsilon\sqrt{N}$.

→ Wrong! If $p = .9$ (probly for 0), all 0's is t. most likely seq. I.e. most likely no. of 0's is .9N.

D.373 SVH

TM408

01: 350.40 : Hexvens! What looks like a fantastically silly mistake! Not so silly!

On 289.40, I got the idea of essentially $\xi = \frac{F(s)+A}{s}$

$\Rightarrow$ ξ = expected future ncost/corpus symbol

s = No. of symbols in corpus thus far

A = ncost of pam.

$F(s)$ = ncost of corpus wrt pam for 1st s symbols of it.

Then on 293.07 I got $\xi = \frac{f(s)}{s} - A$

Thus Avgt. of 292.25 — 293.07 seems quite clear — hvn., in the examples given on 292.25, $\frac{F(104)}{104}$ is "correct" since $F(104)$ is probly not known, but $F(103)$ is known

$\begin{pmatrix} 100 \\ 101 \\ 102 \end{pmatrix}$

& $\frac{F(103)}{103-3}$ is about right, but a bit hy. $\frac{F(103)}{103} \times \frac{103}{103-3}$

.20 Perhaps 292.25 should be modified. Saying that if next 3 symbols are 000, is not a v.g. way

SN Note: HM is not a sequential coding method

to code them. If we are getting k corpus symbols/bit of code, at that pt (i.e. $F'(s) = k$ at that pt. for a no. load code NLC) then we want to multiply $F'_{observed}(s)$ by $\frac{s}{s - \frac{3}{k}}$

i.e. $F'_{adj}(s) = F'(s) \cdot \frac{s - \frac{3}{F'(s)}}{s} = F'(s) - \frac{3}{s}$

3 bits of code would give us $\frac{3}{F'(s)}$ symbols of corpus for corpus length s.

i.e. factor $\frac{s}{s - \frac{3}{k}} = \frac{ks}{ks-3} \approx \frac{ks+3}{ks} \approx \frac{F(s)+3}{F(s)}$.

$\frac{df}{ds}$

So $\frac{F(s)}{s} \to \frac{F(s)}{s} \cdot \frac{F(s)+3}{F(s)} = \frac{F(s)+3}{s}$ ← i.e. f.

results of 279.40

While .20 ff is not all so rigorous, still, it suggests that 293.07 is too lenient in penalizing A's & that 279.40 write by just right! i.e. $\frac{F(s)+A}{s}$; or perhaps $F'(s) \cdot \frac{F(s)+A}{F(s)}$ 409.8

01; 408.40: $\frac{F(s)+A}{s}$ seems o.k. I'd have to go into f. argt. more for f. $F'(s)$ correction. $\frac{F(s)+A}{F(s)}$ does seem like a reasonable dilation factor for $F'(A)$.

The arguments on when $F'(A)$ can be used, & f. various refinements on f. old $\frac{F}{s-A}$ formula (which is wrong), are probably all applicable to these formulæ.

15 ¾ (12674): Since W's Method (used as a limit) is SVH method, both obtain limits for probty, for fixed ℓ, perhaps one can compare them easily, for fixed ℓ.

W's method for fixed ℓ: Actually, $A^{(\ell)}\to 3$: $A^{(\ell)}\to 2$ are also possl. consider all codes for f. 2 corpi $A^{(\ell)}\to 0$: $A^{(\ell)}\to 1$ get ratio of Σ prodcts, using a C.B., T. Let $C_B\to\infty$. But forget about varying C.B. for awhile.

In SVH, Consider all nonprinting seqs. of length ℓ or less, ~~(illegible)~~. Perhaps renormalization of obs probty values may screw up inclusion theorems!

~~Certain codes for the~~ Actually, I'm not sure that every non-printing seq. defines a pgm ($\equiv$ CPM).

22 (SIU) Why confine f. Σ defn. to n.p. seqs! What we want to do is get f. machine into a certain state. If it has to do printing to get there — O.K. — we can simply ~~(illegible)~~ neglect f. printing, & concentrate on $\Sigma,0$ that occurs after we get f. f. desird state. T. input seq. that got it into that state is a defn. of that state & ∴ of f. CPM that that state realizes.

24

25 (That old stuff about pgms & ipgms may be irrelevant — but I should look it over again) — Anyway — Say any non printing n.p. seq. defines a pgm.

Def Then f. KMM (Kolmogorov version) KSVH method says, "pick f. pgm defn string, S, $\Rightarrow$ (N_S + cost of f. corpus wrt S) is min." Anyr, I think that in this case S is always f. null string! I.e. f. cost pgm that defined by UMC! To see this — put it in f. form of pcost rather than bcost. — so we want S $\Rightarrow$ [2^{N_S} × (prod of corpus wrt S)] is max. ($N_S \equiv$ probty of f. string, S.)

Now α is simply f. Σ prod of all codes for f. corpus that start w. S! So clearly, since the null defn pgm (i.e. that defined by f. UMC) has some other codes in addition to these, it will have a greater α.

If we take the (with this version of SVH) $\equiv$ WSVH, we chose f.

74TM 55.01

74TM 55.01

12674 SVH

01; 409.40 Pen → [Σ (prob of that path) × (prob of corpus w.r.t that path)] → = α_W is Min

= Σ prob of all strings that can get it into that state or an "equivalent" state that gives identical probs to all corpi

Hvr, again α_W is the Σ prob of a large bunch of codes for t. corpus, and again, the null string (→ ∃ one) has more codes than those for α_W.

i.e. 0 value for α_W.

Yet the SVH method does seem to have some meaning & validity!

Perhaps not! — T. main meaning in SVH being the use of

$$\frac{-\log_2 \alpha_W}{(\exists N_{corpus} = \text{length of corpus})}$$

as an upper bnd estimate for $\mathcal{G}$ —

10 t. expected mean bcost/symbol of t. future corpus.

i.e. it is True but trivially identical to Willis' method.

→ So 409.25 – 45.10 looks like a demolishment of the old SVH method! There are still, perhaps, some Q's about!

1) ~~whether~~ Just how a pen should be defined — whether by t. longest possl. n.p. string, or by any n.p. string, or whether primitive strings can also be legal (see 409.22–.24 for a discn of this latter pt.).

---

20 on C 88.30: For large corpus of RW data, it is likely that $\mathcal{G}$ will not exist for t. corpus as a whole — i.e. difnt part will have difnt $\mathcal{G}$ values. Just how one should deal w. this is unclear. Hvr, it seems clear that one can't simply use $\frac{F(S)}{S}$ [in this RW case, A ≠ 0 since amt of data ∃ (≡ S) is very large & we can chose our own "Apriori".], F'(S) may be more reasonable — but it is not clearly defnd.

Punct, Coding 99-100

1974

3.21.77 Punctuation, coding

This is
:/ Pretty much in t. spirit of t. previous writing on this stuff. TM 99

Some new ideas, hvr.

1) Use t. method of Willis' thms to code a) Bern seq. b) Bern seq. w. changing probys as a funct of previous corpus (≃ Markof chain) — which is an improvement on Huffman.
W's method mite be ≃ Hufmann's, for t. most likely code — but w. ∞ "Block length" →
c) Bern seq. w. varying alphabet at each symbol pt. (≃ Markof chain).

d) Coding w. defns. e) PSG coding.

In general, i think th. methods usd in I & II were wrong for these problems. (But I. e. T. method for coding t. Bern seq was wrong at t. begining) — But t. ideas are correct enuf to be useful in further work.

---

2) Suppose we are coding a time series of nos. by linear regressn.

T. code consists of t. following seq. of nos.

a) The first N values of t. seq.

b) " N coifs

c) the σ*

d) Th. seq. of corrections involving σ* as scale factor

(— This latter is not exactly rite but is O.K. for illustration)

Initially, N is not known exactly, but a probty distribn. is known. — Similarly for t. no. of signift bits in t. values of t. seq., t. coifs & σ. If any of these are partly computable & say, probilistialy, from previous data in t. code, then this info should be usd to narrow t. probty distribn. & ultimately reduce coding cost.

T. code itself; start with a trinary alphabet: 0, 1, Δ

Δ is punctuation. Its signifc. will vary at various pts. in t. coding. T. relative probys of 0, 1, Δ will vary, often at each symbol pt.

At first, we must write out t. first value in t. time series, T. first bit must be 1, so we don't write it. We start w. t. 2nd bit. This can be 0 or 1, or there can be no 2nd bit — in which case Δ is an "end of no." symbol. The relative probys of 0, 1, Δ will depend on how many bits we expect to be in t. no.

100.01

01: 99.40: Anyway, say we write $\overline{01101}\Delta$ (meaning 10101 is t. final value)

[There is some Q about using floating pt., in which case, each no. would be expressed as a pair of integers.]

If there is only 1 coif, Δ is t. next symbol. If not, then 5 binary bits follow — it is impossl. for Δ to come before t. final binary bit. Anyway, so we continue coding, using Δ, $\Delta\Delta$, etc, to designate various punctn. types. Hvr. I. supt thing is that t. probty distrbn over 0, 1, Δ should use all info available — both apri (external to t. seq) & all previous info in t. seq.

Instead of using Δ, $\Delta\Delta$, $\Delta\Delta\Delta$ as distnt. punctn. types, we could use Δ_1, Δ_2, Δ_3; but t. end result in code cost would be t. same.

In t. case of t. coding of t. errors (wrt α), this can be done directly: knowing α makes it possl. to compute t. probty of each possl. error that is written out.

~~There will be problems involving coding costs.~~

T. above decrn. of coding, gives us costs in a very direct way, for a "known" coding method. If new defns are devisd, hvr., it's not so clear as to what t. coding costs are.

E.g. say in t. linear regress. situation, we want to define a new variable = t. product of t. ~~2nd~~ 5th (say, $\ast$). We can write a punctn. symbol, indicating t. fact 2 integers (2 & 5), tell which logs are to be multd. to get t. new variable. Just what is t. probty of this punct. symbol, hvr.? T. probtys of 2 & 5, resp. can be $\frac{1}{N}$ each, (if there are N coifs thus far), or it can be other things if there is any previous info on this.

Hvr., t. Forge coding method (viz. t. one for linear regress.) is not universal, so I would, certainly, need a larger vocaby of instructions (or a distnt set of instructions).