Sun Aug 12, 1956

3 very Imp. Ideas tied together! (see 79), 67) 68)

1. sets of sets of sets .... ideas — that any set may be an element of an ntp.

2. ~~79~~ 79) idea that some abss. are in th. form of instructions on how to create their contents. These instructions are ∴ abss., and so (some abss.) themselves can be used directly to create new Abss.

I.e. th. abs. are th. instructions on how to create new abss.

Th. ~~instan~~ Abs. is stored in th. form "Operation$_i$ × Abs$_j$".

(Some other Abs. are of th. form: Perform operation$_k$ on th. abs. in question — if th. result is in ngmst α, then th. Abs. in question is a member of "this" set.

3. Th. scanning method, in which th. scanning mechanism that is used to find (p)ngms, is also used to find strs, and/or ntps.) 67), 68)

→ Essentially, "this" ~~s~~ set is defined by $\frac{\text{ngmst}\alpha}{\text{operation}_k}$.

or $\equiv (\text{operation}_k)^{-1} \times \text{ngmst}\alpha$. — So these 2 kinds of storage are both of th. form — "operation$_\ell$ × Abs$_j$"

ℓ or j or both may vary over a set of values.

from 88).40 — pugms like ⊕ ⊴ o wouldn't become probable until all relevant other $\begin{smallmatrix} | \\ | \end{smallmatrix}$ tetragms ~~have been~~ are incons.

It requires $\begin{smallmatrix} o \\ | \end{smallmatrix}$; | → $\left(\begin{smallmatrix} o \\ | \end{smallmatrix}, |\right)$

4 operations

$\boxed{\tfrac{1}{2}} \times \left(\begin{smallmatrix} o \\ | \end{smallmatrix}, |\right) = \left(\begin{smallmatrix} o \\ | \\ | \end{smallmatrix}\right)$

(also $\boxed{\tfrac{1}{2}} \times \left(o, \begin{smallmatrix} | \\ | \end{smallmatrix}\right) = \left(\begin{smallmatrix} o \\ | \\ | \end{smallmatrix}\right)$)

⊕ ; $\left(\begin{smallmatrix} o \\ | \\ | \end{smallmatrix}\right)$ → $\left(⊕, \begin{smallmatrix} o \\ | \\ | \end{smallmatrix}\right)$

$\boxed{1 \,|\, 2} \times \left(⊕, \begin{smallmatrix} o \\ | \\ | \end{smallmatrix}\right) =$ ⊕ ⊴ $\begin{smallmatrix} o \\ | \\ | \end{smallmatrix}$

.17 $\begin{smallmatrix} o & | \\ | & | \\ | & | \end{smallmatrix}$ requires 6 operations. $\begin{smallmatrix} | \\ | \end{smallmatrix} \xrightarrow{2\ ops} \begin{smallmatrix} o & | \\ | & | \end{smallmatrix} \xrightarrow{2\ ops} \begin{smallmatrix} o & | \\ | & | \\ & | \end{smallmatrix} \xrightarrow{2\ ops} \begin{smallmatrix} o & | \\ | & | \\ | & | \\ | & | \end{smallmatrix}$

woops!! see 91).15

SN Perhaps we should automatically reduce th. U aprip of operations that are very multivalued — like str × and addition of ☐, and certain inverses.

How to work with "sets of sets of . . ."

Just try them out in various combinations with other objects, and see what obtains, until you get experience with them, then try to make them do specific things.

.30 * These methods of getting new strs from old, should be gone into in greater detail. Th. method of getting U apri of th. new str. is O.K., but th. xfmn $\boxed{1}\boxed{2} \to \boxed{1 \,|\, 2}$ can be characterized in some way, and perhaps gn. its own empirical U. After all, one does have to describe th.3 xfmn in some way — mite as well associate an apri U with th.3 particular xfmn. type.

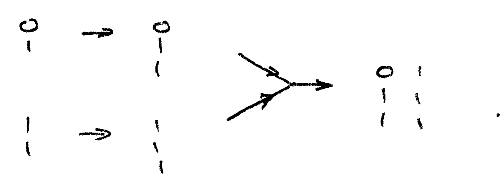
We have a heirarchy of 3 levels of U:

1) U of pagms.
2) U of ntps, ngms, strs.
3) U of xfmns to get U_{apri} of new ntps, ngms, strs, from $U_{empirical}$ of old " " "

We have, on 90).30 considered th. poss. of dividing up one of th. "methods" (of getting new abss. from old) into sub "methods" and assigning a ~~?~~ different ~~?~~ $U_{3)}$ to each.

Somehow, we want T.M. to do th. above sort of "division into sub methods" automatically.

.15 from 90).17 we can get ⚬ | | by an ~~?~~ apparently shorter method — i.e.



If we regard these 2 operations as being done at th. "same time". Actually, th. resultant U_{apri} will depend in a rather detailed way upon th. U_{apri}'s of th. individual ngms, as well as th. combination rules. It is hard to say anything exact about ~~these~~ rel. sizes of these U_{apri}'s.

I think we will want to let most U_{apris} be $\ll 1$, and let ~~?~~ th. case no. be much more imp. than U_{apri}. U_{apri}'s will be used mainly for distinguishing betw. various possible ~~?~~ ~~?~~ abss. to try in combination. Th. low U_{apri}'s will very probably result quite naturally, from most abss. being n.g. anyway.

It is possl. to get — which is cons.

~~?~~ This

~~?~~ has as hy U_{apri} as — perhaps even more, since | etc. has more empirical U than ...

actually

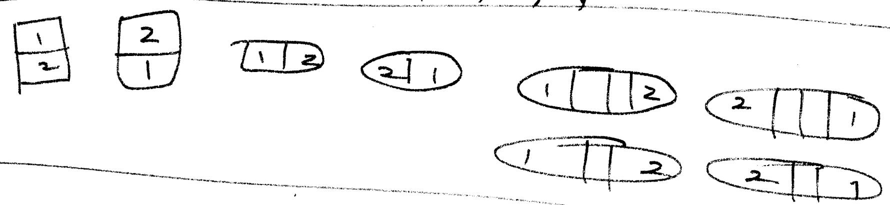
| 1 |
| 1 1 |
| 1 $\boxed{1}$ |

is a cons. pugm. and will continue to be so. Only when ⊗ is introduced, will it become incons.

At th. present time th. ngms that have had some empirical Verf.

O.K. — not ~~yet~~

0, 1, s, =, ~, ⊕, $\begin{matrix}0\\1\end{matrix}$, $\begin{matrix}1\\0\end{matrix}$, $\begin{matrix}1\\1\end{matrix}$, $\begin{matrix}0\\0\end{matrix}$, 10 ngms.



2 × 4 = 8 strs.

(1, 1) (1, 0) (0, 1) (0, 0)
(=, $\begin{matrix}1\\1\end{matrix}$) (=, $\begin{matrix}0\\0\end{matrix}$) (~, $\begin{matrix}1\\0\end{matrix}$) (~ $\begin{matrix}0\\1\end{matrix}$) 8 ntps.

→ Note also, that when ⊗ is introduced,

1 1
1 0
1 1

becomes incons.

<u>Woops!</u> even

1
0
1 $\boxed{1}$

or

1
0
0 $\boxed{1}$

is cons. and will be until ⊗ is introduced. These will have a reasonably large sample, also. There is no way to get around this — but to give problems like ⊕ s 1 s 0 0 / 0 1 0 / $\boxed{1}$ 1 0

There are essentially 2 kinds of imp. ~~[illegible]~~ situations brought up by a q. element or element.

.34 1) a pugm. can become incons. (by element or q. element)

.35 2) th. problem is ∋ there are no suitable pugms available, and a new one must be ~~[illegible]~~ constructed. (by q. element only).

.38 Course of action: take th. examples out to ⊕ with. th. use of th. pugm

1
0
1 $\boxed{1}$

, then write that after more ⊕ examples, ⊗ is introduced, and

~~the~~ pugms ~~the~~ of th. type (+) 5 1 ← must be ~~...~~ invented
0
$\boxed{1}$

(actually, they needn't be, in most cases & tho' ... needs it?

Note for +, th. same sort of thing will happend: T.M. will realize that "+" is being done, by noting that there are 4 lines rather than 3. This can be discouraged by some ad-hock problems — but such problems will give rise to a case of th. 2nd kind 92).35) and will not make th. old pugms incons.

write that "+" has been introduced also, and ~~th~~ ~~th~~ T.M.'s behavior investigated, — that a special form for + is used — with carry line. Then "−" introduced — some change of pugms used.

Write: Th. next interesting problem is th. ommission of th. "carry" line.

So: Rite now, th. problem is to replace (18) and (19) (page 87)) with ~~some~~ 3 or 4 examples that will ~~elim~~ ~~elim th. pm~~ render incons. th. 8 pugms.

(+) 0 1 0
1 0 0
1 $\boxed{1}$ 1 0

incons

1 1 / 0 $\boxed{0}$ 1 $\boxed{0}$ / 0 1

1 1 / $\boxed{1}$ 1 $\boxed{0}$ 1 / 0 1

10 / 10, 10 / 01, ect.

woops — there are 32 such pugms — 1 for each position of $\square$ but they can be rendered incons. in sets of 4.

(+) 0
0 0 0
1 1 1 0

to show all 32 incons., th. foll ngms must appear:

01 / 11, 00 / 10, 11 / 01, 10 / 00
10 / 11, 11 / 10, 00 / 01, 01 / 00

If A = 1 / 1, B = 0 / 0, a = 1 / 0, b = 0 / 1, then aAbBa and aBbAa elim. all 32

.38

so ~~(18)~~ (17) (+)
1 1 0 0 1
0 1 1 0 0
1 $\boxed{0}$ 1
1 1

~~(19)~~ (18)

(+) 1 0 0 1 1
0 0 1 1 0
1 0 1 1 1

Prediction is 1 viz 00 / 11

on th. other hand, altho th. 8 ngms $\begin{smallmatrix}1&0\\1&0\end{smallmatrix}$, $\begin{smallmatrix}1&0\\0&1\end{smallmatrix}$, ect.,
have not been used up to this point, ~~so there is no danger of T.M.~~ T.M. ~~[illegible]~~ would invent them if it could since they only take 2 steps to invent

e.g. $\begin{smallmatrix}1\\0\\0\\1\end{smallmatrix} \succ \begin{smallmatrix}1&0\\0&1\end{smallmatrix}$, which does have many cases. and was, untill recently, "consistant".

There is ~~an imp. prob~~ problem here of some import: we have th. pngms ⊕BBB $\begin{smallmatrix}1\\\boxed{1}\end{smallmatrix}$, $\begin{smallmatrix}1\\\boxed{1}\,1\end{smallmatrix}$, $\begin{smallmatrix}1\\1\,\boxed{1}\end{smallmatrix}$

$\begin{smallmatrix}1\\1\\0\mathrm{B}\,\boxed{1}\end{smallmatrix}$ $\begin{smallmatrix}1\\1\\0\mathrm{B}\,\boxed{0}\end{smallmatrix}$, etc. ⊕BBB $\begin{smallmatrix}1\\1\\\boxed{0}\end{smallmatrix}$

only $\begin{smallmatrix}1\\1\\\boxed{1}\,1\end{smallmatrix}$ and $\begin{smallmatrix}1\\1\\\boxed{1}\,1\end{smallmatrix}$ have ~~any~~ count >0, and none have any "cases"

The situation where one has many relevant pngms of reasonable Uaprip — that have no cases, I think one should chose the ones or ones that have ~~a coun~~ hyest count. My previous discn. ~~sugg~~ concluded that ~~cases of con~~ instances of conflict would be rare and the offending pngms would be soon eliminated. This may not be true. Their may be several pngms of zero count around that are relevant, and conflicting.

Consider th. foll: decision proccedure: Say one has just created a new bunch of pngms that are relevant to a □. (□ = interog. sq. = i.s.). If they conflict, take th. ~~predictedth~~ pngm. ~~of hyest~~ that has >0 cases. If there is > 1 with >0 cases, take th. one with most cases. If none have any cases, take th. one with greatest count. If none have any count, th. ~~situation is~~ prediction ~~it~~ can't be much good, and shouldn't be made. If ~~one~~ there is a conflict ~~most I decide~~ between ~~the pa conflicting~~ the predictions of 2 pngms with = no. of cases, or = count, then no prediction should be made.

.02 More simply: If the predictions of two relevant pngms conflict, take the prediction of the one with most cases. If both case numbers are equal, take the one with greatest count. If both counts are then equal, no prediction should be made. If both case nos. are equal and ~~greater than~~ greater than zero, the prediction accuracy may be expected ~~to be poor.~~ (Insert for α12.2.28)

≤ 0 : th. prediction is 1, via

1 / 1 / 1 $\boxed{1}$ and 1 / 1 / $\boxed{1}$ 1

Then try some random ~~⊕'s~~ ⊕'s:

~~(20)~~ (19) ⊕ 1011 / 0101 / 11$\boxed{1}$1

~~(21)~~ (20) ⊕ 10010 / 11001 / 1$\boxed{1}$0 1

in ~~(20)~~ (19) 1 / 0 / $\boxed{1}$ 1 and 1 / 0 / 1 $\boxed{1}$

both have count > 0.

⊕ 0 1 1 1 has o.k. 0, 0 / $\boxed{1}$ but zero count.

in (20) 0 / 1 / 1 $\boxed{1}$ and 0 / 1 / $\boxed{1}$ 0

Both have count > 0.

---

one could put > 1 □ on each q.element, (q.ele = q.ele.) (element = ele)

but this would make discussion more difficult.

See 92).38 for continuation. to (X)

A section should be written on how T.M. always tries to get simple, compact pngms. E.G. ~~we would like~~, In example 15 we would like T.M. to make the connection between the symbol "~" and the digm 1/0, but T.M. choses a simpler method of prediction. This method is again used in example 16.

Then discuss how T.M. learns ⊕, then how (X) ≢ seems good at first, but then T.M. finds a way to use local hexagms.

That in general, T.M. will not use a partic. method, unless it is th. <u>simplest</u> method, ~~to~~ its own idea of the word.

~~Eventually, T.M. will be~~

If, eventually T.M. is gn. problems, in which it becomes <u>necessary</u> to recognize ~~than~~ th. logical connection between, say "~" and ?, it will do so. But until that time, it will continue to use simpler prediction methods

---

Another section on NMTM. — How MTM ~~may~~ may be seriously disturbed forever by one ~~bad~~ ill-conceived example. NMTM. is probabistic and not <u>too</u> much disturbed by a single counter example. ~~Also has better possys.~~ ~~of [illegible]~~ Also is more directly adaptable to using its ~~own~~, normal prediction methods to improve itself.

Program for future work:

~~Ommission of carry line,~~

(1) Modification of definitions: Npst.
Ngmst
etc.

Major modification is change of def. of ntps so that a set may be a component of an ntp.

(2) Some additional comb. methods e.g. functions (give example)

(3) Modification of program from MTM to NMTM. because (a) less sensitive to small "errors" in trg. sequ. (b) Easyer to work out logic of modifying its own program, since efficacy of new ~~trial~~ methods of working problems is a probabilistic question.

(4) Specific problems:

subtraction
Ommission of "carry line"

Multiplication.
division
change to linear notation,

use of parenthesis

Alg. equation soln.
Theorem proving ← chess playing
Translation of English into logically tractable form.
Translation of logically tractable notation into English.
Th. Q.A. T.M.

interpolation.
extrapolation
Differentiation
Integration
Literal Solution of diff. equs.

Fri Aug 17, 1956

Talk gn. Fri Aug ~~18~~ 17, 1956 : Attending:

Solomonoff
Selfridge
Minsky
Mc Carthy

Bigelow
More
Rochester.

Things they seemed much interested in:

1) could machine learn "substitution?"

E.g. gn. $x+x=2x$, $y+y=2y$ could it learn to expect that $z+z=\boxed{} \rightarrow z+z=2z$?

or if $x=3$ and $y=2x+1$, $y=?$

A way T. M. learns this is by ngmsts.

$\boxed{1}\boxed{\dot{2}}\boxed{1}\boxed{3}\boxed{4}\boxed{1}(\beta, +, =, 2) = (x_i + x_i = 2x_i)$. Here β is th. ngmst $\beta \equiv x_1, x_2, x_3 \ldots\ldots$, so is a set of ngms.

$(\beta, +, =, 2)$ is created by a cart. product of th. ngmsts $\beta, +, =, 2$.

2) Th. extent to which strs, ntps, ngms which were once useful, are destroyed by a partic. gp. of ngms becoming inconsistant. This is catastrophic in th. present MTM, but not in NMTM.

3) A bit amused by my saying human memory need be $< .25 \times 10^9 = 2.5 \times 10^8$ bits.

3×10^7 sec/yr. say 10 bits/sec for 30 yrs.

$= 10^{10}$ bits in 30 yrs

we have 10^7 bits on 3" optical disc.

10 of these gives 10^8 bits — so we are not out of th. ball park, since 10 bits/sec for 30 yrs is a rather hi estimate.

Anyway, ~~though~~ I should have mentioned photodisk memory

4) McC suggested that $U_{ij} = U_{si} + U_{Ntj}$ is poor, unless th. U's are logarithmic — i.e. $U_{ij} = U_{si} \cdot U_{Ntj}$ looks a lot better.

Future work on T.M.

1. Possibly work out NMTM (probablistic **T.M.**) in a <u>little</u> detail — how th. ngms, ntps, strs do <u>not</u> ↓ in U very rapidly when an "inconsistancy" is found. "It is nec. to give more wt. to <u>near</u> past. Imp. ideas will, however, be frequently rejuvinated."

2. Work out + without "carry line".

3. Work out mechanics of ·pngmsts, ngmsts, strsts, ntpsts in more detail

4. Work out detail of "sets of sets" idea.

5. detail of sets defined by inversion or sequence of inversions. i.e. x is a memb. of set α if, when one performs operation T on x, one gets a member of set β.

6. Th. search problem — possible bearing of time-saving requirement on undesirablity of frequent use of (5).

7. Try to weed out some unnec. abs. comb. methods

8. Work out mechanics of pngm (or ngm?) memory scanner with strs, ntps, etc. See how mechanics are modified by notions of <u>ntpsts</u> (ntwl sets ↑?) <u>ngmsts</u> etc.

9. See if concepts of str, ntps, are really th. best to use. Perhaps <u>additional</u> entities would be expedient.

.38 10. Work out <u>actual</u> <u>U allocation</u> <u>routine</u> in some detail, using $U_{ij} = U_{si} \cdot U_{Nj}$, and hand program it, <u>or</u> make $U_{ij} \rightarrow \ln \frac{c_{ij}}{T}$ so $U_{ij} = U_{si} + U_{Nj}$ with U_{si} and U_{Nj} always < 0. .25 sugg. by McC

11. See if sets of sets idea makes it possl. for T.M. to realize that a particular method of generating pngms is good — with this to 103.01

"Method" being describable in many different ways:

e.g. a) if ~~a~~ $\boxed{1\,|\,2}$, $\boxed{1\,|\,1\,|\,2}$, $\boxed{1\,|\,1\,|\,1\,|\,2}$ are "good" would T.M. readily extrapolate to $\boxed{1\,|\,1\,|\cdots n \cdots|\,2}$?

b) ~~[illegible]~~ Can T.M. classify all of its own methods in ways that make it poss. for T.M. to extrapolate their efficacy? — This is the main problem.

Very imp !!

c) an example of b): T.M. performs operations α, β, γ and gets ~~a~~ pugm of hy U. Then T.M. performs α', β', γ' and gets another pugm of hy U. Can T.M. usually extrapolate to the conclusion that $\alpha'', \beta'', \gamma''$ will ~~be~~ probably result in a pugm of hy U? — The answ. is yes! — providing ~~α', β~~ (α, β, γ), $(\alpha', \beta', \gamma')$ and $(\alpha'', \beta'', \gamma'')$ are members of an ntpst that has been useful. In MTM (Math T.M.), we can even ask about the "consistency" of the set $(\alpha^{(r)}, \beta^{(r)}, \gamma^{(r)})$ $(r = 1, \ldots, k)$ — meaning the consistency of ~~the~~ the set of pugms it creates.

The problem seems to ~~to~~ be: how to ~~[illegible]~~ allow "operations" to become components of ntps. One must "factor" ~~[illegible]~~ sequences of operations.

Actually, in any T.M., ~~B~~ any procedure that is used for obtaining a pugm, must be described in some way. These sub-descriptions then can be ~~[illegible]~~ components of ~~[illegible]~~ ntps.

If this can be done, I think the entire T.M. problem is licked!

T.M.'s description of its own methods, or of its own entire operation, can be looked upon as an ngm. Better yet — since we want to give it ~~many~~ as many as possl. of such ngms, it would be better for these ~~ngm~~ to be descriptions of particular methods, or sub-methods of prediction.

I guess th. idea is to make T.M.'s methods of working problems ~~a~~ fairly good — open loop — without th. self-improvement idea. — Then when it has been running a while, give it th. problem of improving its own ~~structure~~ methods. Th. most imp. pt. here is to decide on a good ~~the~~ lang. for describing T.M.'s methods. This may be a fairly easy problem, ~~a~~ as it looks now.

Note that one can feed in a desire for speed into this closed loop — giving a ~~z~~ U to a "method" on th. basis of both speed and efficacy

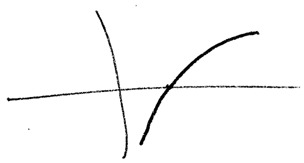
It is expedient to make th. descriptions of T.M.'s methods, ~ to th. problems that T.M. normally works — both in real content, and in "notation".

Write more on this "self-improvement" idea, since I usually arrive at these conclusions, but don't have as specific an idea in mind, as I have now. Th. idea depends much dept. of th. on just how T.M. operates in th. 1st place. Try it with th. simple **M.T.M.** First ~~a~~ invent a lang. that describes how it works — so this lang. is ~ to th. lang. of th. probs. that T.M. normally works. This latter shouldn't be hard, since M.T.M. "normally" works practically any type of problem.

12. larger List of specific probs : X957.23
  a) carry line ommission
  b) learning that − is inv. of +.
  c) $y = x^2$ ~~a~~ etc.
  d) ..

.01 from 99).38 For a computer, if the U_{si}'s, are small, it mite be useful to store them as positions of a 1 in a shift register. This mite make it easy to tell ~~whens~~ which of a set of prgms has a U > a certain amt. If $U_{2ij} = \ln \frac{c_{ij}}{T}$, it mite be possl. to use very approx. binary logs, by counting digit places of c_{ij} and T, with a shift register. Only c_{ij} digits need be counted, because T is th. same for all prgms.
I think that th. same approximations for U_{sj} and U_{Nj} may be used as on 22).15

Note that



no.	$\approx \log_2$		$\approx \log_2$ (better?)	$\log_2$ (better yet)
1.00	0.0	= 0	.0 wrong.	.00
1.01	.01	= $\frac{1}{4}$	.2	.32
1.10	.10	= $\frac{1}{2}$	.4	.59
1.11	.11	= $\frac{3}{4}$	.7	.81
10.00	1.	= 1	1.0	1.00

This makes a more accurate estimate of $\log_2 n$ rather easy.