The way to count, ~~...~~ in this case, is to note that it take 3 xfms to go from =1 to =BBB1. While these 3 xfmns are being made, a ~~...~~ triple can be constructed, to be assoc. with this process.

The idea is how to partly characterize

=BBB1 by the ~~...~~ integer 3, ~~...~~ using the way that T.M. stores the meaning of "integer".

We want =B1 to be described as (=, 1, <1>)

=BB1 " " " " (=, 1, <2>)

=BBB1 " " " " (=, 1, <3>)

Where the <n>'s refer to T.M.'s concept of "integer"

If we can ~~...~~ express this ideas as (=, 1, <3>), ~~...~~ then we will know by experience that many things that work for a characterization of 3 will also work for 4 ~~...~~ — i.e. that the "One greater than" relation is a useful one. ~~...~~ Also use the concept of "function" [as used with binary relations] to change 3 into 4.

SIV The idea of "function" as a binary relation.

We want $ntpst(\alpha_i, \beta_i, \gamma_i) \xrightarrow{ntpst} (\alpha_i, \delta_k, \gamma_i)$

for all β_i, δ_k → (β_i, δ_k) is a certain known Binary relation (function). = pair set.

To do this, $(\alpha_i\ \beta_i\ \gamma_i) \to (\alpha_i\ \gamma_i\ \beta_i)$

combine with the pair set (β_i, δ_k) to get $(\alpha_i, \gamma_i, \beta_i, \delta_k)$

permute and omit by suitable xfmns to get $(\alpha_i, \gamma_i, \delta_k)$

Mon July 30, 1956

For more Final EXPOSITION, give many rather interesting cases, in which the various abs. methods are used.

Reduce no. of special definitions to a minimum: using written out defs. if nec.

After First ~~intro~~ introductory opus, "publish" expansion of ideas mentioned in "future work" —

i.e. 1) ntpsts, ngmsts, etc. explain how "counting" is simply sum of counts of individual ntps, or ngms, or etc.

2) Some new comb. methods appropriate to them

3) How ~~[illegible]~~ hyer order sets are introduced.

4) Many examples of how these new objects work

5) More detail on th. search process in th. ~~hyper~~ 2nd order T.M.

There may be some kind of backward working process, by which this 2nd order T.M. is able to work from th. ◪, to the pngmsts that fit th. case.

A structure may be expressed as a relation — i.e. an ntpst.

e.g. th. str $\boxed{1|2}$ ↔ th. set of all triple (α, β, γ) ∋ $\gamma = \boxed{\alpha|\beta}$. Now th. q. is — how to "score" this str. — or how $\gamma = \alpha\beta$ ever got into this ntpst.

Cart. division of an ntpst by an ntpst may give a kind of "function" generation.

It is possl. to make $(\alpha, \beta, \gamma) \rightarrow (\alpha, \beta, \delta, \gamma)$

Thru th. 2 steps $(\alpha, \beta, \gamma) \rightarrow (\alpha, \beta, \gamma, \delta) \rightarrow (\alpha, \beta, \delta, \gamma)$.

Now, if, in general, this sort of operation is found to be very useful, will T.M., using hyer order classes, be able to ~~to~~ realize this?

It may be best to use "count no." rather than "case no." to determine th. "count utility" of th. pngm. ~~Reg~~ There may be a hyer correlation betw. th. ~~coutn~~ apri count utility and observed ~~cout~~ count utility than betw apri U and " U.

Th. decisions as ~~to~~ what to keep in th. memory, will be based on hy U, however.

On th. other hand, it is hy U_{apri} that we are mostly interested in, in constructing new pngms.

Th. importance of "count Utility", is that it is used to decide between competing pngms.

There is some Q. as to whether the examples I give from ~~[illegible]~~ arithmetic learning, will give much of a suggestion that problems of any difficulty can be worked.

more specifically — that any seq. of operations can be described and generalized.

A more imp. example!

$(\alpha, \beta, A_1, \gamma, B_1, \delta)$ is useful.

also (A_i, B_i) is a useful ~~[illegible]~~ ntpst.

Now how does ~~[illegible]~~ T.M. realize that

$(\alpha\ \beta\ A_i, \gamma, B_i, \delta)$ is of some reasonable U_{apri} for all i?

[this utpst is of some U_{apri}, and then, since it is useful, becomes hyly reinforced.

In general, we simply have th. problem of how can T.M. describe any abstraction process ~~[illegible]~~ using ~~[illegible]~~ abss. ~ to those of English, but using its own vocabulary — so that T.M. can easily generalize in th. way we want it to.

Well, consider th. problem of 38).37:
= 1 , = 81 .... etc. Perhaps T.M. should ~~first~~ develope th. concept of "distance",
Th. sequence of strs 1 2 , 1 8 2 , 1 8 8 2 etc. is a better case to work with. We can describe these strs. by coords:

1 : (0,0)	1 : (0,0)	1 : (0,0)	1 : 0,0 etc.
2 ; (1,0)	2 ; (2,0)	2 ; (3,0)	2 ; (4,0)

If we have th. ngmst, 1, 2, 3, 4, ... ∞, we can easily place it at this position. This is a bit ad-hock, however, and I would like T.M. to develope th. concept of "integer" if possible, from a suitable trng. seq.

Also, I feel that T.M. should be able to "see" this and extrapolate without anything as sophisticated as th. concept of "integer".

A possl. method: Try to generate th. sequence of pairs: $(\boxed{1|2}, \langle 1 \rangle)$; $(\boxed{1|8|2}, \langle 2 \rangle)$ $(\boxed{1|8|8|2}, \langle 3 \rangle)$ . . .
where $\langle i \rangle$ ≡ th. set of all i tuples.

Perhaps th. more general problem, of remembering and generalizing any seq. of useful operations, is easier to approach

Some Definitions:

1) Ngmst, ntpst, pngmst, strst.

2) an ntpst is a set of ntuples (of th. same n)

3) an n tuple is an ordered set of n objects.

each "object" may be a strst, an ntpst, ngmst or pngmst.

defs 2) and 3) are circular. They do not define anything very specific, since an "obect" may be anything, and still be compatible with this definition. This may be O.K., however.

There may be some confusion as to just what an ntpst. is. i.e. just how would one recognize one if one saw one? This may be, perhaps fixed up by saying that an "object" is ~~the contents~~ an address of a register. An ntp. is, then, an ~~set~~ ordered set of n addresses, or a set of instructions that will determine whether a gn. set of n addresses satisfies that ntp. / there is, however, only 1 such set of n addresses An ntpst is a ~~set of instructions to tell whether~~ ~~test~~ test — a black box into which n ordered addresses are plugged and a "yes" or "no" comes out. Usually there are > 1 such sets of n addresses.

Underline all and only all words that are ordinary English words, and ~~may~~ have ~~a~~ special meanings in th. text.

We may use U, ngm, pngs str, ntp. without underlining.

Write in Numbered sections: as 1.0 introduction, etc. This will eliminate paging problem.

Indicate sections to be read at a first lite reading, by stars. *

This business about using U as a criterion to decide whether we want to keep a pngm. in th. memory seems O.K., since we do want to keep pngms of hy U. On th. other hand, in deciding a conflict of 2 pngms, it seems that pngm "count" (rather than "case no.") is imp. ~~But~~, Essentially, ~~[illegible]~~ we are giving pngms of very low ~~[illegible]~~ U, ~~[illegible]~~ zero wt. in combining conflicting pngms. (B.G. problem) — rather than making th. combination on th. basis of ~~[illegible]~~ "count." ~~[illegible]~~ This may, perhaps, be justified, on th. basis ~~that~~ pngms of low U, ~~[illegible]~~ tend to be "inconsistant." "Count" is an index of th. wt. to be assigned to ~~a~~ pngm., only as long as those pngm has been consistant. Th. wt. is meant to be a measure of th. probability that a gn. pngm will remain consistant forever.

We have, then, Th. foll. Q.: What is th. best criterion of storage of a pngm? We want a criterion of rejection that gives us hyest prediction accuracy, using th. stored pngms.

I think that th. reason that U is a good criterion, is that "probability of consistancy" would be a criterion that would waste too much of th. memory. There are probably very many pngms that are consistant — yet they do not have enuf cases to make their storage worth while. — This fact, along with th. fact that hy U tends to ↑ probty of consistancy, make U seem like a reasonable approximate criterion of retention of a pngm. in th. memory.

In "Definitions" write:

Pngm (abreviation of Prediction ngram)

Ngm (" " ngram)

In getting variance of Uapri — say for pngm = str x ntp., assign partial variances to each str. and each ntp., so that th. variance of $S_i \times N_j$ = "variance$_{S_i}$" + "variance$_{N_j}$".

This may be a better way, than assigning a single variance to all pngms U'_{aprips} created by str x ntp.

——— It may be useable only after a large enuf sample is available.

There is some confusion about U_{apri}'s.

Th. "same" U_{apri} can be obtained in ≥ 1 way. e.g. on α 97 is a way to get them for strs and Ntps in str x Ntp = pngm, by optimizing their values so that $U_{S_i} + U_{N_j}$ approx's $U_{pngm\, ij}$ as well as possl. Also one gets U_{S_i} by attenuating the U_{S_j} of a neighboring str.

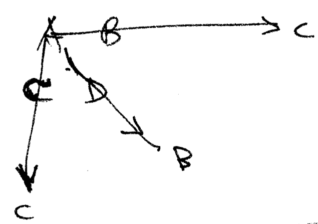
A way to ~~account~~ explain This: The "neighboring str." idea is a way to get strs. of hy Uaprip, with practically no computation. This Uaprip is used untill one gets more empirical info. Getting the U from neighboring Strs. may be looked upon as the Uapri of th. str.

— The U gotten by optimization is an empirical U.

Riteo

Serious Confusion in problem of assigning several U's to various objects, in accord with their ~~use~~ ~~effe~~ effectiveness in each of ~~several~~ operations. Say we have ~~4~~ kinds of objects A, B, C, D (Ntp, ngms, etc).

~~of~~ type A has N_A U's.
" B " N_B " etc.



(Actually we only need ntps, and strs)

$$\begin{tikzcd} S_{tr} \arrow[d] \arrow[r, dash] & N_{tp} \arrow[r] \arrow[d, dash] & N_{tp} \\ S_{tr} \arrow[d] & N_{tp} \arrow[d] & \\ S_{tr} & N_{tp} & \end{tikzcd}$$

~~An~~ Str has 2 uses.

1) to combine with strs. to produce strs.
2) " " " ntp " " ntps

An Ntp has 3 uses

1) to combine with strs to produce Ntps
2) " " " Ntps " " "
3) to predict.

Trouble is, if a str has 2 U's, then, if we want to combine 2 strs, these 2 U's must be functions of th. 4 U's of th. 2 strs.

T.M. makes various abstractions, and uses these abss. in some sort of optimum way, for prediction. It appears that T.M. can, with th. present apparatus, or but slight modification of its make any abs. class that ~~exists~~ is describable in th. Eng. lang.

Th. main q.'s are: 1) Will T.M. find ~~them~~ with reasonably hy probty, some of th. abs. classes known to be very useful in prediction?
2) I th. method of prediction, using these abs. classes, ~~soft~~ good enuf?
3) Is th. search for abss that apply to a partic. question too time consuming?

---

1) can be solved by demonstrating some interesting, important abss. that can be formulated by T.M. thru this method. 2) Is fairly likely 3) Has not been sufficiently investigated by me.

Let us clear up th. q. of just which U's are apri, ~~[illegible]~~ and how empirical U's of each type are obtained.

1) Th. empirical U's of pugms are clear.

2) Th. empirical " of strs. with resp. to their mult. by ntps to produce pugms is clear.

3) Th. empirical U's of ntps. in 2) is clear.

4)

I think, that to start off, we will have only one U for each object. Then as T.M. has more data, more U's will be added.

An imp. new way to get ntpsts:

let s_1 be a str. let N_i be th. elements of an ntpst.

then th. newly created ntpst will have elements

N_i, $S \times N_i$; or alternatively, $(N_i, S, S \times N_i)$.

It is clear that one can have an ∞ of U's for a ntp., say. Consider N_1 and N_2 are ntps that combine (by cart. mult, or division) to produce ~~a~~ new ntp N_3.

$N_1, N_2 \rightarrow N_3$. say N_1, N_2, N_3 have Us, of U_1, U_2, U_3 for some application.

then N_1 acquires ~~a~~ new U; U_1' that gives its utility in producing U, when combined with another N_i.

So now we have 2 U's per N_i; U_i^* and U_i'.

We then also must have U_i'', which is th. utility of N_i, when combined with another N_j, to yeild anew N_k.

In a similar way we can have U_i''', U_i'''', etc.

(?) → It would seem that th. sample size for these hy order U's would be small, so one could only introduce them after T.M. had been operation

not very sure of this.

Anyway, I think in humans, there can't be very many U's per object.

This point could be discussed at greater **length**, but for th. present, I will just use 1 U per object, for illustration.

.01 Another direction of complexity with these U's: each object has a U. For each ~~a~~ method of xmn or combination, th. U_{apri} ~~of~~ the resultant, is a funct. of th. U's of th. component entities. At first this function is th. mean of th. components, but as soon as more ~~info~~ data is available, more general first and second degree term coifficients are inserted.

If only linear coiffs are used in all applications, I think that th. simultanious optimization problem of all coiffs, and ~~[illegible]~~ σ^2's, may not be too difficult.

~~[illegible]~~ Th. T. M. equations: Consider a simplified T.M. It has pngms, ~~[illegible]~~ ntps, strs, ngms.

only binary ntps
" " strs.

.24)	σ_1^2	strs ~~[illegible]~~, ntps → pngms
.25)	σ_2^2	ngms, ngms → ntps
.23)	σ_3^2	strs, ~~[illegible]~~ ngms → ~~[illegible]~~ ntps

U_{S_i}, U_{Nt_i}, U_{Ng_i}, U_{P_i} are U's of pngms, ntps, strs, ngms.

Note ~~[illegible]~~ C is th. case no. of ~~[illegible]~~ consistent prgm., but C=0 for any inconsistent pngm.

We want

$$\sigma_1^2 = \sum\sum\left(\frac{W_{ij}\,(AU_{S_i}+BU_{N_j}) + C_{ij}}{W_{ij} \quad + \quad \cancel{[illegible]}\ 1} - AU_{S_i}+BU_{N_j}\right)^2$$

$$W_{ij} = \frac{(AU_{S_i}+BU_{N_j})}{\sigma^2}$$

Fix A, B, U_{S_i}'s, U_{N_j}'s ∋ σ_1^2 is min.

say 'ngm₁, ngm₂ → ntp' is a comb. method in which order of ngm₁ and ngm₂ is imp.

We want to chose th. U_{S_i}'s, U_{Nt_i}, ect. also th. combination coeffs → in all cases, the mean difference between th. apri U's and ultimate U's are minimal. [how does this agree with th. optimizing ultimate goal of "good prediction"?] — I am assuming that this is a good sub-goal.

In all 3 cases of combination, we must ~~find~~ express a U_{apri}, a wt. and a "true value of U".

~~[illegible]~~ in 48).24,

$$U = \frac{W\,U_{apri} + c}{W + \tau}$$

c = case no.
τ = no. of interrog. squs.

$$U_{apri} = A_1 U_{S_i} + B_1 S_{Nt_j}$$

$$W \approx \frac{U_{apri}}{\sigma^2}, \quad \sigma^2 = \overline{(U_{apri} - U)^2}$$

in 48).25

$$U_{apri} = A_2\, U_{N_{gj}} + B_2\, U_{N_{gk}}$$

$$U = U_{Nt\,jk}$$

~~th. relative wt~~ $U_{Nt\,jk}$ is determined "emirically" from

Just ~~what~~ how much variance is expected in $U_{Nt\,jk}$?

If this is known, th. mean sq. error of U_{apri} can be found

However, I think that it may be O.K. to assume th. expected error in this $U_{Nt\,jk}$ to be zero.

Then the weight to be assigned is unimp., and we need only optimize A_2, B_2, $U_{N_{gj}}$'s and $U_{N_{gk}}$'s.