BEINGS: KNOWLEDGE AS INTERACTING EXPERTS

Douglas B. Lenat
Artificial Intelligence laboratory
Stanford University, Stanford, California

Abstract

Knowledge may be organized as a community of intfiacting modules Each module is granted a complex stiucture, to simulate a particular expert in some small domain An extended analogy is drawn to a group of cooperating human specialists Based on this, an internal constraint is imposed on the modules Then structure must be standard over the entire community Some advantages of a uniform formalism are thereby preserved. An experimental community was implemented for the task domain of automatic programming It has managed to synthesize a few inductive inference LISP programs, nonformally. from specific restricted dialogues with a human user

1. Experts and Beings

Consider an interdisciplinary enterprise, attempted by a community of human expeits who are specialists in - and only in .. then own fields What modes of interactions will be productive? The dominant paradigm might well settle into *questioning and an sinning* each other Instead of a chairman, suppose the group adopts rules for gaining the floor, what a speaker may do, and how to resolve disputes When a topic is being considered, one or two experts might recognize it and speak up In the course of their exposition they might need to call on other specialists This might be by name, by specialty, or simply by posing a new sub-question and hoping someone rould recognize Ins own relevance and volunteer a suggestion Such tiansfers would be more common at the beginning, when the task is (by assumption) too geneial for any one member to compiehend As the questions form on more specific issues, single individuals should be able to supply complete solutions If the task is to construct something, (hen the activities of the experts should not be strictly verbal. Olten. one will recognize his relevance to the current situation and ask to *tto* something, clarify or modify or (raiely) create

What would it mean to *simulate* the above activity? Imagine seveial little programs, each one modelling a different expert What should each program, called a *Being,* be capable of It must possess a coipus of specific facts and strategies for its designated speciality It must interact via questioning and answering other Beings. Each Being should be able to recognize when it is lelevant It must set up and alter structures, just as the human specialists do.

Let us ietuin to our meeting of human expeits. To be more concrete, suppose their task is to design and code a large computer piogiam a concept formation system[2]. Experts who will be useful include scientific programmers, non programming psychologists, system hackers, and management personnel What happens in the ensuing session? When an expert participates, he will either be aiding a collegae in sume difficulty of else transferring a tiny, customized bit of his expertise (facts about his field) into a piogrammed function which can do something The final code reflects the member s' knowledge, m that sense One way the session might pioceed is for the specialists to actually *do* the concept formation task As they become familiar with what part of their own expertise is being called upon, and in what ways, they can begin to isolate it. When it is clear precisely what each is doing, they can take their extracted bits of knowledge, organize them, formalize them, and program them {A conscious erfoit along these lines was made in [8], where expeits gradually leplaced themselves by programs Instead of discussing how to write a speech ptogiam, they / speech recognition, until each one could introspect sufficiently into his own activities to formalize them For our task, one expects the psychologists to dominate the eaily discussions, later yielding to programmers The project sponsor might be passive, submitting a single specification order for the program, or active, paiticipating in the work as a (somewhat priveleged) member of the team. This individual is the one who wants the final product, hence will be called the *user*

How could Beings do this? Theie would be some little proyiim containing information about CONCEPT FORMATION (much mote than would be used in writing any single concept formation piogiam), anothei Being who knows how to manape a group to WRITE PROGRAMS, and many lower level specialist's, for example INFO-OBTAINFR, TEST, MODIFY-DATA STRUCTURE, UNTII -LOOP, VISUAL -PF.RCI PTION, AVOID-CONTRADICTION, PROPOSE-PL AUSIHt.E-NAME Like the human specialists, the Beings would contain far too much infoimation, far too inefficiently lepresented. to be able to say "we ourselves constitute the desired piogtam'" They would have to discuss, and perhaps cany out. the concept formation task They would write specialized versions of themselves, programs which could do exactly what the Beings did to carry out the task, no more not less (although they would hopefully take much less time, be more customized). This activity is relected to in the sequel as *automatic programming* Some Beings (eg, TEST) may have several distinct, stieamlined fractions of themselves in the final program. Beings (eg, PROPOSE-PLAUSIBLE-NAME) which only aided other Beings may not have *any* con elates in the final synthesized code

An experimental system, PUP6, was designed and partially implemented PUP6 synthesized a concept formation piogiam (similar to [7]). but the user. who is human, must come up with certain specific answers to some of the Beings' ctitical queues A grammatical infeience ptogram and a simple property list maintenance routine were also generated. A few new Beings had to be added to PUP6 to synthesize them

The next section illustrates how the experts might have cooperated on the task of writing the concept formation ptogram Section ? describes the program they produced. Next comes the Being hypothesis complex but standard anatomy Later sections explain this, both theoretically and by examining the behavior of the actual PUP6 pool of 100 Beings The appendix exhibits a typical Being.

2.ExerIsJnteractjng

The input/output behaviot of the desned concept formation piogiam is specified in this section, and we eavesdrop on a simulated group of specialists as they get to work on wnting it As the presentation of the experts* activities becomes moie specific, the te.tder's uiirenily vague conception of Beings will be made less amorphous (because Beings are constrained to cany on appioxtmately the same discussion as the expeits below do)

Externally, the concept formation task can be specified as followv pictures of structures (bui out of simple geometrical shapps) will he preser ne after anothei For each such scene, the concept foimauon piogiam, tall it CF. must guess its name The piesentrr will then tevral the conect name of the stMictuir. C.F must quickly learn to identify simple structures (ARCH. TOWER), and must never make the same mistake twice in a row Assume, as given, a process which extracts a clesciiption of a visual scene

Our group of expeits are given this specification for CF Assume that the usei (the financial sponsor) is available for lesolvmg important questions, via messenger, and he may in fact ask questions of the group Whenever an expert speaks, almost all the others in the room heai him Usually only a few can benefit from what he says, and fewer still care to react The conversation in the room might go something like the following (the suggestive names of the experts are of course comcidental)

GENL MANAGER: Can anybody here figure out what to do. what the usei's saying? (waves the input/output specifications in the air)

PGM MANAGER I *can* He *wanly* a computer program *to* be written If somebody will expl.un the task "con-cept-for ma tion" to me a little mote cleaily, I'll delegate It propeily

PSYCH Permit me to assist you I know all about concept toimatum In fact, my master's thesis.

PGM MANAGER Wait, the user must be told that we'll be able to handle the job for him

MESSENGER Here. I can take that message Go on with your work

PGM-MANAGER We need a name for (his program. Somebody get one. please

NAMER How'about "CONCEPT"? Maybe just "CF" Let's ask the user to decide

MESSENGER (panting) 1 just came back from there! Alright. I'm going User says to call it "CF"

PGM MANAGER Now then, I have four people in mind, one of whom must take over now in an important way Each of them always wants to do something different.

CHOOSER Give me then names and I'll choose the best one for you

PGM MANAGFR They are INFO OBTAINER. INFO-USER. PGMMER. and ANALYST

CHOOSER You four expeits each must justify to me why you should take contiol now

INFO OBTAINER We already have some unused information, so I guess 1 can wait

INFO USER Let me have the floor! I know what to do with the usei's specification

PGMMER Well, I suppose 1 could set up some "empty" top-level function, er.

ANALYST Marginal profits accruing from my working at this point in time do not warrant...

CHOOSER Okay, INFO-USER, you seem to be the only relevant expert

INFO-USER- First thing is for PSYCH to tell us how to actually *do* concept formation

PSYCH There are several decisions to be made, depending on what your independent variables and your experimental setup are For example, are we studying one individual, or must oui program adapt to simulate many diffetpnt subjects? Must stimulus items be just classified, or classified and ordeied, or classified and ordered and metnzed? Are the scenes left m view indefinitely, or just the anient scene, or is even that one just Hashed before the subject for a limited time? May the subject make written notes? Can he learn from positive instances of the concepts? fiom negative instances? Is there to be any positive transfer effect...?

DEFER: Hold on here, we can't swamp the user with lots of unnecessary questions Let's go through each one and see if it can wait. Consider the decision about the task being classificatoiy, tompai auvc, of metncal What is the first situation where that decision leally matters?

WHEN-NEXT: This involves a list of alternatives Who knows about that?

ALTERNATIVES I know a few tuck', here If all the alternative sequences of activities have the same initial subsequence, then do that common subsequence before making the decision

UTILIZE: In this case, all three begin by partitioning a domain of elements

DEFER: Temporarily, replace the task CF by the task of partitioning a domain *(The usei is informed of this.)* Now consider the decision about subject specific behavior being tequned

PSYCH This involves periodically inputting a description of the human subject

CODER That would mean adjusting the algorithms based on a vector of parameters

WHEN-NEXT This would probably affect the entire code

DEFER I can't defer this decision Someone resolve it

RESOLVE Ask the user about it

ASK USER Phrase this as a yes/no question Explain each alternative to the user

MESSENGER That's my job *<typcs on teletype>* User says no, don't simulate different people

DEFER The next decision

Eleven decisions ate ultimately proposed by PSYCH, and all but one are deferred

INFO USER I have no objections now if someone wants the floor

PGM-MANAGER Do any of the other three experts I mentioned earlier want to speak now?

PGMMER Yes I think that the top-level function CF can now be coded

CODER Give me the arguments and the body of the code.

PGMMER There are no known arguments The body is a call on *PARTITION DOMAIN*

CODER Okay I will precede that with a call to an *INITIALIZE* function, and follow it with a call to a *FINALIZE* function, which are both defined as NIL for now Is *PARTITION-DOMAIN* simple enough to be composed light now and filled in here?

MATHEMATICIAN Noway Any realuate.

CODER Uh, thanks There The function CF is defined as:

(LAMBDA () (INITIALIZE) (PARTITION-DOMAIN) (FINALIZE)).

ANALYST Remind me to examine the initialization and finalization functions at the end of our task. If either function is still null, it will be deleted

WARNER I have just put that note into the code for CF. as a comment of type "demon"

PGMMER Can someone advise me of what else to do to finish defining this function?

PGM MANAGER Each function should have a proper name Show the user the names you have picked, and let him choose other ones if he prefers

MESSENGER Okay The user agrees to all 3 names

INFO-USER Somebody, please tell the group how to *do* partitioning of a space of examples

A complete script, like the above, was constructed by hand In the sequel, this will br refeired to as the *protocol.* In all. 87 diffetent expeits were called for 17 speciftcly dealing with inductive inference tasks, and 70 dealing with progtamming, managing workers, and communicating with the user Near the end ol the protocol, the usei is asked which of the three types of concept foimation CF is supposed ro do He responds "*CL ASSIFI< ATORY only",* and the experts discovei that they are finished All the newly created code is dumped out onto a fresh hie After hundreds of pages, a concept formation piogtam meeting the user's specifications had been written The next section will desctibe that program detail

3.. The Program the Experts Wrote

One of the experts at the simulated meeting must have read P Winston's dissertation[7], because CF, the synthesized concept formation program, was remarkably similar to the one therein described. CF has a much simpler graph-matching algorithm, and relations on relations are stored in a different way than simple relations on objects Since CF was later synthesized by PUP6, the programmed pool of Beings, it is worth detailing here

CF repeatedly scans a scene and tries to name it. As a first step, the scene is broken into a set of objects and a set of features (relations on those objects) CF maintains a model for each differently named scene it has encountered. A model contains a description of the objects one expects in such a structure, a set of features which *must* be present in any scene having this name, a set of features which *must not* be present if the scene is to have this name, and a set of features which *may* be present or absent. Thus a model is an archetypical scene plus a name For example, part of a scene might be described as

OBJECTS a,b,c,d
RELATIONS (Green a) (Blue c) (Touches c d)
(Supports a c) (Supports *b* c)

CF's current model for an arch might be

NAME Arch
OBJECTS a,b,c
MUST (Supports a c) (Supports b c)
MUSTNOT (Touches a b)
MAY (Green a) (Wedge c) (Prism a) (Block b)
(Parallel a b) (Red a) (Red b)

Each time it is confronted by a new scene, CF must scan its models until it finds one which matches it A model is said to match a scene if all the MUST features associated with that model are observed in the scene, and all the MUSTNOT features are absent from the scene. CF informs the user of this guess, and accepts the proper name. If it guessed incorrectly, CF modifies its models. The wrong-guess model may have features added to its MUST or MUSTNOT sets This is sufficient to prevent CF from making the same wrong guess twice in succession. The correct-name model may have to be modified or (if it's a new name) created and inserted into the list of models, to ensure that CF will eventually learn that concept. A *concept* here simply means a model, i.e., all scenes having a given name.

Suppose that the target program reads in the above scene fragment and lues to match it to the above ARCH model The MUST relations should all be present Yes, the scene does contain (SUPPORTS a c) and (SUPPORTS b c). Next, the MUSTNOT relations must be absent from the scene Sure enough, (TOUCHES a b) isn't there So the model and scene are consistent, and the program announces that its guess is ARCH If the user venues this guess, then the MAY set of the ARCH model is augmented with the relations (BLUE c) and (TOUCHES c d), and the OBJECTS set is augmented with "d "

If the usri dentes that the scene is an arch, CF sees if there are any letations in the ARCH moclrTs MAY set which do not occur in the scene If so. one of them (e.g., (PARALLEL a b)) will be itamfenrd (mm the MAY to the MUST set If no such featuir eststed, the program would look for a feature prespni in the scene but not mentioned in any set of the ARCH model (eg. (TOUCHES c d)). and inseit if into the MUSTNOT set In either case, the user would be asked what (he tine name was, and that model would have its MAY set augmented by any new features in the scene and by any fe.ituies nu the tiue-name model's MUST or MUSTNOT sets which luntiadicted the scene.

4. Anatomy .of .Synergetic, Cooperation

Conwdei the buth uf one small ea necessary in the writing of C.F (eg. that of classifying a model's features into three categaties (MUST. MUSI'NUT, MAY)) No single specialist at the mrrting rould have had this idea by himself How do intellects mesh, effectively comniuiiKate, and unite then powersm' A tentative mechanism, which barely scratches the smfate of this mystery, will be hypothesized The Beings in PUP6 embody this concept, and are able to reproduce both the experts' discussion and the final CF program

Viewing the group of experts as a single entity, what makes it productive? The membeis must be very different in abilities, in order to handle such a complex task, yet similar in basic cognitive structure (in the anatomy of then minds) to peimit facilp communications to flow For example, each specialist knows how to direct a progiammer to do some of the things he can do. but the specific facts each expert has under this category must be quite unique Similarly, each member may have a set of strategies for ipcognmng Ins own relevance to a proposed question, but the c*entrnts* of that knowledge vaires fiom individual to individual The hypothesis is that all the expens can be said to consist of categorized information, where the set of categories is fairly standaid. and indicates the *tyfus* of questions any expert can be expected to answer An expeit is consideird *tyiuvalent* to his answeis to several standaid questions Each expert has the same mental "parts", it is only the values stored m these parts, then contents, which distinguish him as an individual

Aimed with tins dubious view nf intelligence, let us return to the design of Beings Each Being shall have many parts, each posseting a name (a question it deals with) and a value (a protedure capable of answering that question) Hencefoith. *"part"* will be used in tins technical sense When a Being asks a question, it is leally just one part who is asking tn fact, it must be that the *value* subpait of some part can't answer AM question without further assistance He may not know enough to call on specific other Beings (so he lers anyone respond who feets relevant), but he should *of ways* specify what Being *part* the question should be answered by By analogy with the experts, each Eeing will have the same "universal'' set of types of parts (will answer the same kinds of queries), and this uniformity should permit painless intercommunication Since the paradigm of the meeting is questioning and answering, the names of the parts should cover all the types of questions one expert wants to ask another Each part of each Being will have implicit access to this list it may ask only these types of questions Each Being should *not* have access to the list of all Beings in the system requests should be phrased in terms of what is wanted, rarely is the name of the answerer specified in advance. (By analogy; the human speaker is not aware of precisely who is in the room, when he feels inadequate, he asks for help and hopes someone responds) Another point is that Beings are not a recursive concept (like ACTORs(3] are) a part of a Being is a brief collection of knowledge (usually procedural), not another Being, a collection of Beings (also called a community, a pool, the system, or a group) is also not itself a Being There are no s*tuntuttd* clusters of Beings

Once again the concept of a pool of Beings is that many entities coexist, each having a complex stiuctuie. but that structure does not vary from Being to Being This idea has analogues in many fields transactional analysis in psychology, anatomy in medicine, modular design in archiierhtute

How can we test out this idea? We must build a pool of Beings, a modular program which will internet with a human user and genetate the CF program Recasting the idea into opeiational terms, we arrive at this procedure for writing a pool of Beings

(1) Study the task which the pool is to do. See what kinds of questions are asked by simulated experts

(2) Distill this into a core of simple questions, Q, in such a way that each inter-expert question or transfer of control can be rephrased in terms of Q. The size of Q is very important. If Q is too large, addition of new Beings will demand either great effort or great intelligence (an example of a system like this is ACTORS). If Q is too small, all the non-uniformity is simply pushed down into the values of one or two general catchall questions (all first-order logical languages do this).

(3) List all the Beings who will be present in the pool, and fill in their parts. The time to encode knowledge into many simple representation schemes is proportional to the square of (occasionally exponential in) the amount of interrelated knowledge (e.g., consider the frame problem). The filling in of a new Being is *independent* of the number of Beings already in the pool, because Beings can communicate via nondeterministic goal mechanisms, and not have to know the names of the Beings who will answer their queries. This filling in is *linear* in the number of Being parts listed in Q; all parts of all Beings must be (at least, should be) filled in.

(4) The human user interacts with the completed Being community, until the desired task is complete

Section 5 clarifies the effects of constraining that Q be constant (over all the Beings in the system). Theoretical aspects of Being systems follow, in section 6. Next comes an evaluation of PUP6's behavior. The uses and the problems with Beings are summed up in the final section

5. Internal Details of Beings

A set of 29 ubiquitous, questions were chosen, lepiesnitihg everything one expert might want to ask another At least. they natuially encompass those questions which *were* asked during the simulated meeting, hence should be sufficient for geneiatmg CF Q this universal set of Beinp, paits, is listed in Appendix I The reader is urged to glance at this now, and refer to it whenever a Being part is specifically mentioned

Each of the 100 Beings in PUP6 should have had a value for each pait (in reality, only 40%. of these 2900 slots weie filled in, only 30% were actually neceuaiy to generate (I) A value for a pait is simply a LISP pmgiam which can answer that question, otten by asking questions of the same Being. of other Beings, and of the user A pait may also assrit some fact, create or modify some structure (including demons., Beings, and parts of Beings) Appendix I shows the values stored under each part for the typical Being named "INFO-OBIAINEP"

The set of parts breaks into three rough categories (1) those parts which are useful in deciding which Being gets connoi, *(2)* those which are used once the Being gains contiol, and (3) those useful only to answer the user's questions and Veep him of lented The next section describes categoiies I and 2, the section after that explicates the third categoiy of Bemp pans

5 J..CqnJrnl in the PUP6 System

At the humans* meeting, only one expeit spoke at a time; in the Beings community, only one Being has contiol at any given moment He uses his parts to do things (ask, create, modify), and yields control either voluntarily or through intemiption

In slightly more procedural teims, the scenario is as follows One parl of a Being senses its televance (often the IDEN or EFFECTS parts, which are united with all such pails to form a large production system[5]) If more than one Being wants control at any time a special Bring. CHOOSER, seizes contiol momentauly He asks each competing Being to evaluate its WHEN part, to see how senously it needs to go immediately If some Beings are still tied for hrst place, he asks them to evaluate their COMPLEXITY parts, to see which is the simplest If any *stiff* ne for top. one is randomly chosen In any case, the winner is then passed contiul Onrr in coutrl, a Being arranges some of us parts in tome unet and evaluates them For example, the ARCS pajt n)i*:ht he hist, if it asks for some aiguments which no Bring has Mipplied. then the whole Being might decide to fail Some parts, when evaluated, might create a new Being, might ask questions which require this whole process to repeat iccuisively. etc This "asking" really means hioadcasting a request to one or two parts of every Being, for example "Is theie a known fast way of gronking tovess? would be asked as a search for a Being whose COMPLEXITY part indicated speed, and whose EFFECTS part contained a production with a template matching "gronking toves" A list of the tespondets would be returned (Incidentally, GERUND would recognize this, but latei give up when no one could tecognue "gronk loves") The questioner might pose some new questions directly to these Beings, might turn control over to them directly, etc One way or another, the Being eventually relinquishes control If it had no direct successor in mind, all the Beings are asked if they want to take over There will always be *some* Being who will take over; the general management types of Beings are always able - but reluctant -- to do so

How does each Being decide which parts to evaluate, and in which order, once it gains contiol[7] The answer might seem to be difficult or tedious for whoevet wntes Beings, since it might vary from Being to Being In fact, it doesn't! The commitment to a univeisal set of Being parts is inefficient in some ways (each Being *needed* only a thud of all the parts) bur allows for some simplifications tight here What paits should be evaluated, and in what order, when a Being gains control? This decision depends pnmanly on the *types* of parts present in the Being, not on then *values* But every Being has the same anatomy, so one single algorithm can assemble any Being's parts into an executable LISP function Moreover, this assemby *can* be done when the system is hist loaded (or when a new Being is first created), and need only be redone for a Being when the values of its parts change Such changes aie lare expeits are not often open minded The precise algorithm is sketched in the box below The parts useful here include ARCS, DEMONS, META CODE, COMMENTS, ARC-CHECK, and REOU1SITES

Assembling a Being into an executable function

When a Being *B* first gains control, its EXPLICIT-ARGS are bound. The IMPLICIT-ARGS are initialized, the name *B* is pushed onto the Being control stack, and any newly-activated DEMONS are so tagged. The Being who called *B* should have explained his reasons by assigning some phrase to the variable BECAUSE. This reason is now stored as a special sub-part of the WHY part of *B*. BECAUSE is rebound periodically in the META-CODE and COMMENTS parts, to keep current the explanation of each call that *B* makes. Each ARG-CHECK predicate is evaluated. If any returns NIL, the entire Being reports that it has failed; otherwise, the PRE-REQUISITES are examined. Effort is expended to make them true, if they are currently not satisfied. Each COMMENT is evaluated, then the *CO-REQUISITES, META-CODE*, and the current demons are executed in pseudo-parallel. Each POST-REQUISITE is then examined, and an effort made to satisfy it. The newly-activated demons are exorcized, *B* is

> popped from the Being control stack, and the value computed by the META-CODE is returned. Some heuristics were devised to take advantage of the fact that the Beings often didn't need many of the standard parts. For example, INFO-OBTAINER has no new demons or corequisites, so no parallel processing need be simulated.

5.2 Keeping the User Inforiitrcl

In the earlier conversation excetpts. the simulated human user had no trouble whatever understanding what the simulated experts asked him In the actual programmed PUP6 system, the human who was siting at the teletype quite *rarely* undeistood what was wanted by the Beings He frequently had to intenupt !hrn> and ask them questions about who was in tontrol, why, what he was trying to do, what had tecently tianspnrd. etc These ideally can be phiased as simple lernevals and FVALs of active Beings' part* The Benig parts most often called for by the user are the simple one ti*me* "m inn it ion" templates These include WHAT, MOW, WHV, and AFFECTS Foi theoretical reasons explained latei, the syuthesmd piogtam, CF. was writen as a pool of BES ns'lt (by PUP6, but not during the piotocol Actually, a foituitou* "bug" m PUP6 created this mtnguiiig situation) Although us question answering ability is infenoi to PUPfi. the fact that it has *any* such power was tuipnsing to the authoi In other words, one can inteirupt the taiget piogiam as it is lunnmg and ask questions Any Being un the tnntml stack will provide fully instantiated amwms to any of us 29 allowable queries (its parts), all othei r*einj;s will provide only hypothetical answers As an example, roimdei this actual excerpt of a human using the CF piogiarn synthesized by PUP6 (Some liberty has been taken with the English; e.g., the user really types *WHAT*,* not *What atr you* <frvw£*) *???" simply means "guess the name of the scene with these objects and relations" OF types in *ifoln* i. the usei in boldface.

> CF· *Ready to accept brand new scene*
> **User: (??? (A B) (Block A) (Wedge B) (Touches A B))**
> CF: *Not an Arch. Not a Tower. Might be a Row.*
> *My guess is Row. What is it* ***Really?***
> **User: Nothing**
> **CF:** ***Transfer from the May to the Must set of the***
> ***Row model, the feature: (Block B)***
> **CF:** ***Ready to accept brand new scene***

Here is the inwe excerpt, but with the user interrupting occasionally. "II:" signifies he has interrupted, and "OK." signifies "Co back to work now. CF."

II: What are you doing?
CF divide a domain tnto subdomains,
using specialised version of PARTITION-DOMAIN
How?
CF: By building up the partition gradually,
by adding one 'scene, name pan* OK.
CF: Ready to accept hand new scene
User: (>>> (A B) (Block A) (Wedge B) (Touches A B))
V: What are you doing?
CF Break OBJ CCTS puce off input scene to examine
What code i* Ihr current Reing goinp to execute?
CF The meta >cde is {(ADR argl) Loafs to (A B). OK.
CF Not an Arch U: Why?
C F The ARCH MUST adation (SUPPORTS A B)
is contradu ted in the SCENE OK.
CF Not a Town Might be a Row My guess is Row
What is it realty" U: Why are you asking?
CF: I must resolve it now. but can't, so I ask User
OK What you wanted is: Nothing
U: What are you doing now?
CF: Co though space oj concept names, and make user
aware of changes **What Beings might this affect?**
CF MESSAGE is possibly called, some version of TEST
is possibly called **OK.**
CF Transfer from the May to the Must set of the
Row model, the feature: (Block B) **U: Why?**
CF. Change model of ROW until it contradicts
the previous inputted Scene **OK.**
CF. Ready to accept brand new scene

6. Theory of Pure Beings Systems

We now discuss the constraints each Being, and each group of Beings, must conform to. Hopefully, ideas will be separated from implementation details, prejudices from plausible features

It would be aesthetically pleasing to restnet all entities in the system to be Beings However, this would cause an infinite regiess, as each part of each Being would have parts which had patts To stop this, one ran assert that at some finit" level, all construets are primitive ACTORs, for example, set this level to zero, Beinp,set i! to one ACTORs themselves a"* pnmitive. but only *fui)fi* of Beings can be Foi this leasnn, Beings can not he viewed as a convergent recuiwve defimiinn, as can ACTORs

Suppose ii weie decired that the only autniiomous entities possessing contiol abilities were Beings In particular, we forbid any plain *jumtions* to exist In the case of an automatic piugiammmg task, the Beings would have to write new Beinps, not new I.ISP timetmm The tai^et program would thus itself be a community of Brim;* In older to fill in all the parts, a vast amount of superfluous infoimation would be collected These supplementaly facts can be viewed as a standardized, oigani2ed body of *ttr*umrntali(m,* a formatted system of comments tacked onto each Bein£ piodured

Which Beings would write the new Beings[7] Looking back at our inteidisciphnary experts, we see that each expert is responsible for distilling his own essential contribution, which is then encoded by a programme) Perhaps each Being should be able to dnect constMiction of new, speftahied Being*, which lelate to it If no Being lelates to a task, then it can't be coded, if <rveial lespond, they should cooperate This ability is in teahty the SPECIALIZATIONS part of each Bring (s.re Appendix 1) The Being which actually does the creation (CODER) m the experimental system is almost ti ivial, getting very precise instiuctioiu from other Beings

Since the pool must communicate with the user, some Beings must translate quasi English phrases into calls on Brings Drawing again on out experts analogy, we icquire that each Being leco^nize his own televance So translation is merely the act of asking the whole pool "Who can recognize this ", collecting the respondei s, having *them* decide who should take contiol. and letting the winnei do the translation. Most communication is done as if it. too, were such a translation activity

One bias is the (ejection of debugging as a fundamental programming tool It is felt to be worth the extra effort to make the system'* internal model of the cuirrnt partial target proviam *cor ret* Debugging demands detective work, examing one's earliei effoits lot flaws, foi details which have been o vet looked Any tuelcss system should not ignote details, but tathei deter them, asseitmg a warning to this effect when it does so Piociastinatiuu is quite valuable, in PUPG, much effort is spent defeinng any unresolvable decision. Undefmable uniesolvable decisions must cause a backtrack point to be reluctantly set up Anothei prejudice is that most carelessness bugs can be eliminated by this deferral, feed-forward, and precise record-keeping Humans depend on then adaptability to compensate for limitations in then brain hardwaie. beitheir is no need for an *automatic* programming system it so. These busses ate not inherent in thr Beings formulation, but only in the design of the PUP6 system (and in the mind of the anitbor)

To clartfy what Beings are and are not, they are contrasted with some other ideas FKAMFS[4] are sufficiently amoiphous to subsume Brings In philosophy. FRAMES are meanl to model peiception, and intentionally tely on implicit detault values. Beings intentionally avoa linching decisions by default This is also the diheieiue between HACKER and PUP6 Since PUP6 wrnts stnutuird progiams. it should be distinguished from macro expansion Macro proceduies expand mechanically expamKsequence in_1 in_2) = (sequence expanti(in,) expanding))) Brings could use mfoimation gleaned duing expansion of in, to improve the Way m_2 was handled ACTORs[36], unlike Brings, have no fixed striutuir imposed, and do not bioadcast then messages (they specify who gets pach message, by name, to a buieauctaty)

Beings subsume (inefficiently) many popular AI features, the demolish ation will be briief A *demon* could be teplaced by a Bring whose ARC CHECK ptedicate was the tiiggerinp: piedicate. whose WHEN pan was high enough to epstitr firqurnt attention, and whose META CODE part was fhr body of the demon An *assertion m an* associative data netwotnuM be a Being with only an IDEN pait filled in, when it ir(ov.ni/.es its lelevanre. a fully instantiated assrition n. irtujnecl A *funition* is equivalent to a Being with only a MM A CODE. ARCS, and EVAL ARCS parts; one knows almost nothing about it beioie executing it The inefltciencies should be clear whenever a Being throws, a question open to the flooi. "Who tan ", it takes an amount of lime piopoi(tonal to the niimbfi of Beings in the system. One would intioriuce this huge time factoi by teplacmg any of the above mechanisms by Beings

The *numf'f)* ol Bring pans seems to indicate the balance between uniformity and stiuctlite in the community This was huulied mi as point (2) of Seition -I A small universal set of Being paits is netessai y to presei ve some of the advantages <>| tuutoimtty (easy addition of knowledge to the system, easy inter-Being communication) This demands that the numbei of pans o(each Being be. say, under 100 But it is the complex ttrtiuiire of a Being which makes complex behavmis leasable, including flexible < ommunitation as well as viable final products So each Being should have many parts, say at least ten Tins lange. 10 « ||Q|I *, 100, is wide foi the domain of automatic ptogiamming In other domains, it may be nairow or disappear altogether this would indicate that Brings could *not* br used eflrclively foi those tasks

7. Experimental Results

Two *nf* the most significant tfwestions for automatic programming systems are what progiams ate synthesized, and what kinds ol things the user must tell the system

7.1. Cliar.ii ter nf the DjaIngiicJfejuTt'n \'seiJH><LPJL)P6

By lumping all the parts of the PUP6 system together conceptually, the interaction is seen .is a *iltalogae* between a human user and an automatic programming system The level of detail maintained in the earliei polyloguc (discussions among the expeits) oInuued what the human actually saw and irspondrd Below is the same exertpt .\^ was piesented in Section *?,.* but as the *user* sitting at a teletype actually saw it These weir the initial few line* of *^n* actual 700-page long dialogue In section *?.,* this rxiript was presented the way expeits (the same way the PUP6 Beings) discussed it

```
PUP  1 want any task
USER  Write a progiam winch does concept formation
PUP: I am about to wntr a progiam foi the task
       specialized Kind of CONCrPT-f 0PMAT10N
PUP  Give me a proper name for this piogiam
          (I suggest CF or CONCEPT)
USER  CF
PUP  Must CF mimic diffeirnt human subjects[7]
USER  NO
PUP  I am temporarily replacing the task 'Vpeoahred
      kind ot CONCEPT-FORMATION" by the simpler
      task "specialized Kind of PARTITION-A-DOMAIN"
```

From the above, one gets the tiue impression that the dialogue was fairly tedious choosing, names and resolving design decision? *unit* the USPI'S mam iule There were, in sum, about WO such responses involved m genrj.mng CF l!< ciucial responses were aKo nrcrssjty, whrie the usei supplies needed information that was noi explicitly mpiested by the Beings A few of these key lines .UP *Write <7 program which docs concept foi motion. An "clement of the domain of stimuli" is a Matte scene. The guessing process is one of searching, not computing. Terminate the loop iff any model feature is incompatible with the scene's features*

Thr translation mechanism is simply the IDEN parts of all the Beings Each such part contains a few productions, which are united with those of the othei Beings into one laige production system The left side of a production is a template which is matched against the given phrase The right side is a small piogram to be executed in case the phrase is recognized Generally, that piogiam simply says to translate some subphrase of the phiase, and then (probably) pass contiol to the Being which this is in the IDEN part of While not up to the "state of the ait" in natuial language processing, this is both adequate and laithful to the spuit of distiibtiting problems over all the Beings |>y making them *parts,* so it is each Being's duty to know a Intlr about them

During the coinse of the dialogue, PUP6 originates the idea of a MUST/MUSTNOT/MAY stiuctuting of each model's features Here is how the user sees that portion of the synthesis Notice that only the frist user comment is other than a simple choice

```
PUP  How do we know when to teimmate this loop?
USER  Terminate iff any featute in model-features JS
    incompatible with scene.featuies
PUP  PUP wants USER to type in name for specialized
    version of CONTRADICTS
USER  Call it IMPOSS
PUP  Move cursor to IMPOSS type of CONTRADICTS I
    cati no longer defer this decision A Contradiction arises
    from featuies who have probability-1 of occurring B:
    Conn adiction arises from featuies who have
    probability*" of occurring C Contiadiction arises from
    features who have probability^-'-1 of occumng Please
    type some of A.B.C. or type Defei. ?. ot Details
USER  A. P., ^tui C are all relevant
PUP  I tecommend that each model's trituirs be structured
    (along I dimension) into these ". pieces PRESENT,
    which will hold features whose oicuneuce should have
    probability- I. ABSENT, whi'.h will hold featuies whose
    occuireiKe should have piubahilityxfv and EITHER,
    which will hold features whose ocnuirtuc should have
    any other probability Please iypr back YES. NO,
    UNSURE. DETAILS, and/oi ? new names
USER  YES Call them MUST BE PRESENT,
    MUSTNOT BE PRESENT. MAY-BE PRESENT
PUP: Too long Nicknamed Must, Mustmit. May
PUP  Located ?• references to model's (earnres in code
    generated so far In each case, the reference has been
    replaced by "(APPEND Must Musinut May)"
```

This is PUPG at its best inatimg just The "fight" interence .at the "nphi" nmr "I lie hwiidircls nt inter Being tianstns ate masked the system appears to type out just what is necessary The next few paiagiaphs dispell this false illusion, there is no mysreiy to PUPG's doing the tight things, and any *typrcal* dialogue excerpt would be botingly verbose

Because of its genesis frorn a single "experts meeting" piotocol. thp PUPG pool of Being was (1) easily able to repioduce that "piopei" dialogue, but () incapable of widely vanrd dialogues with the usei Two assumptions implicit in using the simulated expeits' discussion as a literal model turned out to be serious the abilities of any actual user of PUP6 must coincide with those *of* thp user who was simulated in the protocol, and the order of and wording of the actual user's comments must closely adhere to those of the user in the protocol Such difficulties must be overcome in any system designed for wide useiship, but were considered ignorable in the specific automatic code generation task at hand

Also as a result of this appioach to system specification, each Being had only those pans spouhed which it actually would *need* in the ensuing dialogue Pait of the difficulty with new dialogues stemmed fiom this minimal completion In the protocol, when a decision was made by experts, the knowledge necessaiy to folluw the *oth*i* alternative branch was not used, nor were such supeifluous facts supplied to the Beings in PUPG Thus the usei ul PUP6 must almost always resolve each choice the way the emulated (piotocol) user did It is felt that if all the pans of all the Beings had been faithfully filled in. this pioblem would have subsided Basically, (he difficulty is one of modelling all the possibly relevant knowledge an expert has. tathei than (as was done) just capturing enough of his knowledge to do a few given tasks

While all the Beings' intei actions were invisible to the user, the system still swamped hurt with data about what was going on F01 example, most of the entities he was asked to name wnr ncvei irfencd to again by name The converse problem existed as well it was nrcess.uy lo include a Being which simulated foigetfulness. to prevent, eg, anaphoia spanning minutes of ieal lime Oi inning ihe usei was not solved satisfactorily Pointeis into a giaph of generated code weir simulated, but often a usei wished to refei 10 a piece of code not by name or by pointing, but by some brief meaningful (to him only') phrase

7.2. The Range of Program* Synthesized by PUP6

The system, PUPG. did eventually synthesize CF. the target concept formation piogram PUPG was 200 pages of INTERLISPIO], CF was ?0 pages long (6 pages when coded by hand during the protocol) CF was generated in 60 cpu minutes (compiled, PDP-10 TENEX) The dialogue consisted of WOK characters typed by PUP6. and 4K by the user It occupied 300 pages, and five hours of ieal time.

Despite the lack of dialogue flexibility, it *was* felt that most of the Beings could he useful in generating other programs For this irasou, two additional target progiams were specified They were synthesized with little change to PIJPG, but only by someone familial with the system

The spcond target program, G1. is a ptammatical inferenie piogiam, which accepts strngs labelled LECAL, ILLEGAL, or ?? In the lattei case. GI must guess the liy.ably Internally, potential spts of mles are maintained. Of the original pool, 49 out of the 87 Beings were used in synthesizing both targets Four totally new Beings had to be added, related to fomal grammars and rules Unfortunately, the addition of *any* new Beings demands that the user be acquainted with the luimat conventions of PUP6 The GI piofiain geneiatpd was *20* pages long a hand coded version was one hllh that szie

PI was the final t. it get pi obtain attempted, a simple propertty hst mamipulatoi I' lepr.iirdly acrepts. inquests from the user to t, inspect, or delete some ieroid(s) Any unspecified helds ate tteated as don't LHJrs. so a simple pattern matchfi had to be synthesized Two Beings had to be *ai/dfil* to PUPG The iinpoitaut piece uf data is that about half rif the onginal PUPG pool of Beings were actually used in *a// three* taiget synthesizing dialogues

As piopo<-ed in Section G, the Beings generate othet Beings, uever plain functions This explains the huge incrases in tatget code lengths in the PUP6 versions compared to the verstiong pioduted by hand when simulating the expeits (who wiote the target piogiams as functions) CF was a pool of '>6 btand new Beings. GI ?7, and PL 24 As with PIJPG. one can inteirupt the target programs as they are tunning and ask questions Any Being on the control stack will provide fully instantiated answers to any of its 29 allowable queues (its pait<), all othet Beings will provide only hypothetical answeis Recall the exceipt fiom CF itself tunning, found m section 52

Some nf the difficulties stem horn the natilie of the task In any lony dialogue, the user often torgets, changes his mind, env. ill A vejy sophistn ated usn mcidel would be nere^aty to .H~cnmi>date this enoitul puness in a non debugging system Without such ihiluirs the system itself may be led into enor While most bugs *nn-* avoidable by raieful lecoid keeping, it ptoved unn,<listic to make no provision fot debugging a new thirty page program When a few eirots did nccut m CF, PUP6 itself had to be altered

8. Conclusions

sIL About PHP6

What have we learned fiom this expeumental study? The overall feasabilny of Beings was clemonstiated, but the difficulties of communicating with the usei made the system almost impossible to work with The set of questions the user was expected to want to ask is the same as the set that one Being can ask anothet the Being paits When the "nice" user intetiitpts. his questions are danslated trivially into a simple retrieval. Real users are seldom nice, the Beings generally misundei stood what useis asked

To modify PUPG to synthesize new ptogiams, it was necessary to add a few genei ;»l puipose ptogtamming and communication Beings, plus add several Beings specific to the new progiams domain, plus generalize a few existing Beings' paits The dialogue to produce the new progiam may be pootly suited to that domain, since most ol the tecogmzed phiasps stem ftom a single (CF-producing) piotocol

To impiove PUP6's performance, one could add some debugging specialist Beings, some dialogue specialists, some sophisticated usei psychology expeits (why is the user asking me that question, what needn't I tell him, how should I direct his attention), some Beings whose task is to aid the untrained usei m insetting new domain specific Beings, and perhaps a whole hbiaiy of varied specialist Beings

8.2 About Beings

The perfotmauce of the Beinys implementation itself in PUPG is mixed Two advantages weir hoped lor by using a unifoim set of Being pans Addition ol new Beings to the pool was not easy (tor unnamed useis) but communication among Beings *urns* easy (fast, naiuial) Two advantages were hoped for by keeping the Beings highly stiuctuied The *interactions (especially with the user) were brittle,* but the complex tasks put to the pool *were* successfully completed.

The crippling problems are seen to be with user-system communication, not with the Beings ideas themselves. Sophisticated, bug free programs *were generated,* after hours of fairly high level dialogue with an active user, after tens of thousands of messages passed among the Beings. Part of this success is attributed to distributing the responsibility for writing code and for recognizing relevance, to a hundred entities, rather than having a few central monitors worry about everything. The standardization of parts made filling in the Beings' contents fairly painless.

What *are* Beings good for? For which tasks won't the *problems encountered in* PUP6 *recur? The idea of* a fixed set of parts (which distinguishes them from ACTORs) is useful if the mass of knowledge is too huge for one individual to keep "on top" of. It then should be organized in a very uniform way (to simplify preparing it for storage), yet it must also *be highly structured (to speed up retrieval).*

For these reasons, the author is currently investigating, as a potential task domain, "research in elementary number theory". This has the added benefit of isolating the problems *in representation research from the staggering* complexities of natural language handling. Beings are big and slow, but valuable for organizing knowledge in ways meaningful to how it will be used. In the future Automated Mathematician system, Beings will be one -- but not the only -- internal *mechanism for representing* and manipulating knowledge.

References

[1] Green, Waldinger, Barstow, Elschlager, Lenat, McCune, Shaw, and Steinberg, *Progress Report on Program-Understanding Systems*, Memo AIM 240, Stanford Artificial Intelligence Laboratory, August, 1974.

[2] Hempel, Carl G., *Fundamentals of Concept Formation in Empirical Science*, Chicago: U. Chicago, 1952.

[3] Hewitt, Carl, *A Universal Modular ACTOR Formalism for Artificial Intelligence*, Third International Joint Conference on Artificial Intelligence, 1973, pp. 235-245.

[4] Minsky, Marvin, *Frames*, in (P. Winston, ed.) *Psychology of Computer Vision*, New York: McGraw Hill, 1975.

[5] Newell, A., *Production Systems: Models of Control Structures*, in (W. Chase, ed.), *Visual Information Processing*, New York: Academic Press, 1973.

[6] Teitelman, Warren, *INTERLISP Reference Manual*, XEROX PARC, Palo Alto, Ca., 1974.

[7] Winston, Patrick, *Learning Structural Descriptions from Examples*, MIT Project MAC, TR-231, September, 1970.

[8] Woods, W.A., and Makhoul, J., *Mechanical Inference Problems in Continuous Speech Understanding*, Third IJCAI, 1973, pp. 200-207

Many hours of creative discussions were quite important. In particular, the author acknowledges C. Green, R. Waldinger, D. Shaw, and E. Sacerdoti. Computer time was generously provided by the Artificial Intelligence Center of SRI.

Appendix 1: A Typical Being

We consider INFO-OBTAINER, a Being which is independent of task domain. Below is listed, for each part, its abbreviated name (in bold), an English question that it might (try to) answer, and the stored program which should try to answer it (often a simple template or a constant). The percentages given indicate how many of the (ultimately 100) Beings in PUP6 actually used that part during the synthesis of one of the three target programs.

WHAT What do you do? Summarize your basic idea 82%
(OBTAIN SOME INFORMATION WHICH CAN BE USED)

WHY Justification? Summarize your motivation 77%
(PUP HAS NO MORE INFO THAT IT CAN USE TO PROGRESS)

HOW Global strategy? Summarize your method. 72%
(OBTAIN NEW FACTS ABOUT OLD INFO, OR BRAND NEW INFO)

IDEN Do you recognize: "Find out more about frob gyrnation"? (if I see either phrase:
(INFO-OBTAINER any1) or (FIND OUT MORE ABOUT any1),
then I return: (INFO-OBTAINER (TRANSLATE any1))) 54%

EXPLICIT-ARGS What argument(s) do you take? (U) 63%

EVAL-ARGS Which are quoted, not evaluated? NIL 4%

IMPLICIT-ARGS What local variables are needed? NIL 11%

WHEN When should you take control (justify your answer)?
((if T then add in -10 because 19%
(I AM EXPONENTIALLY-GROWING, GENERALLY UNDESIRABLE))
(if New-Info-List then add in (Plus 100 (Length New-Info-List))
because (WE SHOULD WORK ON UNASSIMILATED NEW
INFORMATION IF THERE IS ANY)))

REQUISITES What must you ensure is true just before (pre) and after (post) you go? NIL 10%

DEMONS What demons should you activate while you're in control? NIL 7%

META-CODE What happens when you are in control? 70%

```
(DO (CHOOSE-FROM ((GET-NEW-INFORMATION U)
                  (TRANSLATE U)
                  (ANALYZE-IMPLICATIONS U)
                  (EXTRACT-RELEVANT-SUBSET U)))
   BECAUSE (WE CAN ONLY TRY TO OBTAIN USABLE
      INFO IN ONE WAY AT A TIME))
```

COMMENTS Do you have any special hints for filling in undefined subparts of this Being? NIL 16%

STRUCTURE Viewing this Being as a data structure, what can you do to it? NIL 4%

MAIN-EFFECTS Can you cause this goal to occur: "Usable information exists"? 27%
((to get (NEW INFORMATION any1) or to get
(USABLE INFORMATION any1), do (INFO-OBTAINER any1)))

AFFECTS *What other Beings might you call on directly?* 14%

```
((CHOOSE-FROM is called)
 (call on some Being who can satisfy the goal:
    (AWARE USER (ABOUT TO OBTAIN USABLE INFO))
 (GET-NEW-INFORMATION possibly is called)
 (TRANSLATE possibly is called)
 (ANALYZE-IMPLICATIONS possibly is called)
 (EXTRACT-RELEVANT-SUBSET possibly is called) )
```

GENERALIZATIONS What Beings are more general than you? (WRITE-PROGRAM SERVE-THE-USER) 27%

ALTERNATIVES What Beings are similar to you, to try in case you fail? 16% (USE-INFORMATION, OPTIMIZE, FIX-INCORRECT-PIECE, and FILL-IN-UNDEFINED-SECTION)

ENCODABLE Any special constraints on what order the parts should be evalled in? NIL 9%

COMPLEXITY-VECTOR How costly are you? 92%
(.5 .5 .9 .5 .1)

A vector of utility measures. The first component says that INFO-OBTAINER is of average difficulty to call. Next, there exists a *.5* chance that some descendant will call it again. Next, this activity almost always succeeds. The time/space used in allowing this Being to try is typical. Finally, there is no good reason for inhibiting it ever. In general, each component can be a program, not just a constant. These weights, like the contents of all the parts *of* all the Beings initially in the experimental PUP6 system, were decided upon and inserted by hand.