ARTIFICIAL INTELLIGENCE WORKING GROUP REPORT

Chairman, William Dress
Secretary, Ray Adams

Attendees:

J. Waldron, Y. Racine, A. Cotterman, S. Gulick, D. Mateen, S. Lewis, M. Worrel, G. Haydon, R. Miller, J. Basile, S. Rose, D. Ruffer, C. Moore, P. Reynolds, T. Rayburn, J. Garst, C. Rogers, M. Perry, R. Dixon, M. Glidewell, M. Kristo, L. Atkinson, P. Lambrix, J. Bender, P. Moreton, D. Jagerman, J. Lundin, B. Davis, J. Shifrin

Chairman's Comments

- Use Forth to describe AI problems - as opposed to writing an AI language in Forth.
- Fast version of an expert system, Expert II.
- Prolog done in Forth (by Joel Springs?); Expert II has been of some use.
- Natural Language Processing - use Forth to vector into/out of various states.
- Savvy program and savvy user's group.
- Mike Perry interested in knowing what tools are of use in the various AI fields - ex. dynamic allocation.
- Henry Harris' talk - generation of storage, but de-allocation by hand - nevertheless he had programmed some very sizeable applications.
- Garbage collection equated to floating point (by Chuck Moore) in terms of desirability of avoiding.
- Distinctions between editor complexity and operator tools vs. AI construct usefulness.
- Person new to AI interested in knowing which structures that AI investigators need in their work, and to see if those could be provided via Forth constructs.

AI Interest Areas

- Expert Systems
- Language Understanding
- Theorem Proving Machine
- Learning
- Auto Instruction/Teachings
- Vision/Speech
- Games

Techniques

- Pattern Matching
- Object Oriented Messages Program Languages
- Logic Programming
- Productions, Fames
- Search
- List manipulations (Data Structure)
- Probabilities
- Recursion
- Parsing

- Because of fuzziness of our natural language descriptions, we have a problem in understanding our own discussions.

- Dick Miller mentioned that Chuck Moore sometime ago suggested that Forth would be the most natural way to get machine language (spoken) capability once we can get the machine to listen to us.

- Data Structures suggested as one of the easiest things to do in Forth.

- 75% of Group used or interested in Expert Systems.

- Dick Miller interested in who is using Expert II.

  - one person has used and written a paper on its results.
  - Miller thought the easy access (back into) Forth would provide relief for some experienced problems.

- Why are Lisp and Prolog used to such a great extent in Expert Systems, AI research? What is it about these languages that is so attractive - could Forth fulfill these needs?

- The heart of Expert II was published (somewhere) written in Forth.

- Expert II suggested as most useful in figuring out (learning) what an Expert System is as a starting point, then strike out on your own for further study.

- Because Forth can handle data and programs the same (as does Lisp), why can't Forth be used as well as Forth?

- The (one of the) interest difference between Lisp and Forth is the greater ease with which Lisp can treat data structures as potential program instructions.

- More to the point than the details of the language is the need to figure out what elements of (human) intelligence can be represented in an effective way (as an operator) by a computer program.

- Comments on recursion as a concept, as a procedure (relative to data structures) vs. iteration as an algorithm.

- Lisp is thought of (by Mike Perry) as trading off things that appear more natural to the programmer for (at the expense of) things that are hard for the machine.

- Chuck Moore pointed out that the NOVIX chip will easily implement the processing of tree structures.

- Words used as data seem to be an important attribute of Lisp; Forth really can act (naturally) the same way (e.g. case statement - vectored case). A Forth programmer need only vector 5% of an application.

- Disadvantage of the large size of Lisp that prevents use in a small computer, as opposed to Forth.

- There is a rich source of useful problems to be solved by a small computer that could be done in Forth. These are being ignored by the Expert Systems community.

- What are some of the AI techniques that we need to focus on in Forth (given that Forth could easily treat data as programs)?

    - Dictionary and Vocabulary searches (each vocabulary should know how it is organized-tree structure, linear, etc.)

    - Lisp does dynamic allocation/deallocation of memory well (Bill Dress' paper suggests one way - but more work is needed).

    - What about relocating dictionary words (forgetting one word at a time).

    - Vocabulary for compiled overlays.

    - Re-locatable jumps cost twice as much as absolute jumps (C. Moore), therefore.

- Mike Perry argues that "Objects are Crap" and lead to inefficient, clumsy handling at run time that are better done at compile time.

    - Counter argument (that I did not comprehend) which was recountered by Perry, who offered the use of hidden vocabularies.

    - Others suggested that Perry's objection does not change the desireability of handling objects from the standpoint of the user rather than the programmer.

- Observation that Chuck Moore's comments on difficult problems and how they may be more easily solved is related to how fast compilation can be achieved -- save the data and recompile.

- Comment on Lisp and the desireability of being able to easily handle lists. By comparison, how easily are these handled by Forth? Suggestions on how several Forth constructs do easily handle lists.

- What kinds of data structures?

- Suggested that Forth Dimensions should have another competition (like for case statements); this one on list handling.

    - Or AI programming contest in Forth.

    - Or not restrict to Forth.

    - Could someone suggest a general AI problem that could be a competition?

        - Optimal search problem
        - Rubics cube
        - Maze problems (not algorithmic)

    The solution published for the sieve problem was so bad it stimulated people to improve on it.

Another suggestion for a competition.

- Devise an optimal strategy for solving an adventure game.

- Challenge Forth Researchers to do some useful work in AI coded competitivly in Lisp.

- Solution limited to two blocks of code.

- Initiates the challenge by asking for problem submittals

- Does speed - as from new Forth engines-make some AI (sub)problems trivial, or much easier, so that previous stumbling blocks disappear?

- Proof of program correctness is a research area that could be profitable - but this concept was offensive to C. Moore, because of his experience (and the modularity) and observations of the fundamental nature of Forth programming style and practice.

- C. Moore observation from two conferences (Asilomar and Rochester)

  - We are genuinely puzzled by what AI is all about, whereas the AI community is openly hostile to Forth.

  - Some suggestions about the psyches of AI and Forth practitioners and computer science practitioners.