CogSysIII Lecture 8: Cognitive Architectures

Human Computer Interaction

Ute Schmid

Applied Computer Science, Bamberg University

last change June 14, 2007

Computational Mind?

Is the mind a computational phenomenon? No one knows. It may be; or it may depend on operations that cannot be captured by any sort of computer. ...
Theories of the mind, however, should not be confused with the mind itself, any more than theories about the weather should be confused with rain or sunshine. And what is clear is that computability provides an appropriate conceptual apparatus for theories of the mind.
This apparatus takes nothing for granted that is not obvious. ... any clear and explicit account of, say, how people recognize faces, reason deductively, create new ideas or control skilled actions can always be modelled by a computer program. (Johnson-Laird, The Computer and the Mind, 1988)

Computer Models of Cognition

- Cognition is about mental activities
- Not directly observable via input-output behavior
- Information processing paradigm, symbol systems
- Empirical approach: hypotheses about processes which lead to assumption about how task differences result in differences in reaction/solution times, errors
- Modeling approach: tool for formulating complete and consistent runnable models of cognitive processes
- Weak AI position (contrast: strong AI)
- Cognitive AI should not be “sloppy AI”

Characteristics of Good Theories

- Consistent (internally and externally)
- Parsimonious (sparing in proposed entities or explanations)
- Useful (describes and explains observed phenomena)
- Empirically Testable & Falsifiable
- Based upon Controlled, Repeated Experiments
- Correctable & Dynamic (changes are made as new data is discovered)
- Progressive (achieves all that previous theories have and more)
- Tentative (admits that it might not be correct rather than asserting certainty)

Intrinsic Problem

- One and the same I/O behavior can be realized by infinitly many computer programs!
- Restrictions for cognitive models necessary!

Information Processing

- Physical symbol system hypothesis (Newell and Simon)
- Church-Turing hypothesis applies to computational models of cognition
- see

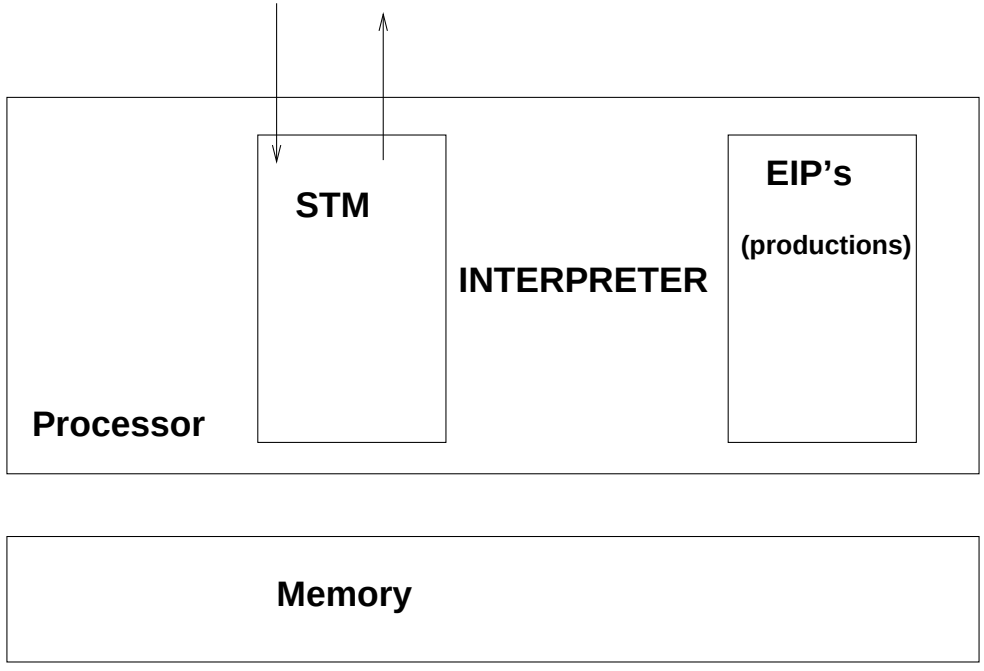
  `http://www.rci.rutgers.edu/~cfs/472_html/CogArch/IPS.html`

IPS

1. There is a set of elements, called symbols.
2. A symbol structure consists of a set of tokens (equivalently, instances or occurrences) of symbols connected by a set of relations.
3. A memory is a component of an IPS capable of storing and retaining symbol structures.
4. An information process is a process that has symbol structures for (some of) its inputs or outputs.
5. A processor is a component of an IPS consisting of:
   - a (fixed) set of elementary information processes (eip's);
   - a short-term memory (STM) that holds the input and output symbol structures of the eip's;
   - an interpreter that determines the sequence of eip's to be executed by the IPS as a function of the symbol structures in STM.
6. A symbol structure designates (Equivalently, references or points to) an object if there exist information processes that admit the symbol structure as input and either:
   - affect the object; or
   - produce as output, symbol structures that depend on the object.

IPS cont

7. A symbol structure is a program if (a) the object it designates is an information process and (b) the interpreter, if given the program, can execute the designated process. (literally this should read, “if given an input that designates the program.”)
8. A symbol is primitive if its designation (or its creation) is fixed by the elementary information processes or by the external environment of the IPS.



Objects of IPSs

The term object is used to encompass at least three sorts of things:

- symbol structures stored in one or another of the IPS's memories, which are often usefully classified into (a) data structures, and (b) programs.
- processes that the IPS is capable of executing;
- an external environment of sensible (readable) stimuli. Reading consists in creating in memory internal symbol structures that designate external stimuli; writing is the inverse operation of creating responses in the external environment that are designated by internal symbol structures.

Note: the relation between a designating symbol and the object it points to can have any degree of directness or indirectness.

Symbols and Symbol Structures

- Symbol tokens are patterns that can be compared by the IPS and judged equal or different.
- A class of all tokens that are judged to be identical is called a symbol type.
- Thus, tokens of the same symbol type differ from each other only in being distinct occurrences or instances.
- Symbol structures are built up from symbol tokens and relations.

Five General Propositions

1. Humans when engaged in problem solving in tasks such as solving cryptarithmetic problems, proving theorems, playing chess, ..., are representable as information processing systems.
2. This representation can be carried to great detail with fidelity in any specific instance of person and task.
3. Substantial subject differences exist among programs, which are not simply parametric variations but involve differences of structure and content.
4. Substantial task differences exist among programs, which are also not simply parametric variations but involve differences of structure and content.
5. The task environment (plus the intelligence of the problem solver) determines to a large extent the behavior of the problem solver, independently of the detailed internal structure of his information processing system.

Theories of Human IP

Four propositions that shape the theory of human information processing:

1. A few and only a few, gross characteristics of the human IPS are invariant over task and problem solver.
2. These characteristics are sufficient to determine that a task environment is represented (in the IPS) as a problem space, and that problem solving takes place in a problem space.
3. The structure of the task environment determines the possible structures of the problem space.
4. The structure of the problem space determines the possible programs that can be used for problem solving.

Invariants

Additional Characteristics of the IPS that seem to be invariant over problem solver and task:

- Size, access characteristics, and read and write times for the various memories of the Human IPS.
- Serial character of the information processing and the rate at which elementary information processes can be performed.
- Program organization is production-like and goal-like in character.

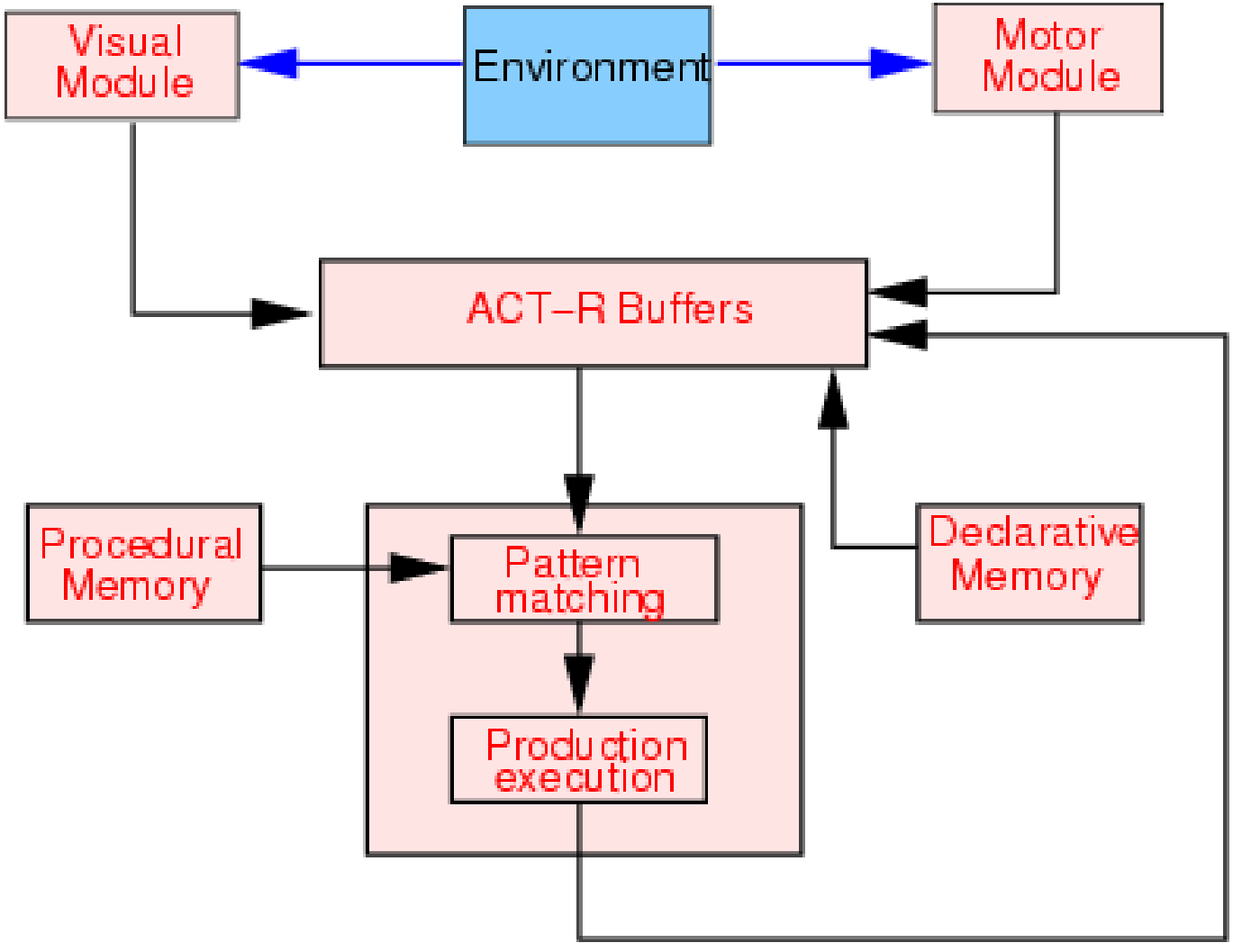
Cognitive Architectures

- vs. special purpose models
- “unified theory of cognition”
  - explicit definition of basic mechanisms of information processing
  - assumption that these mechanisms are constant over all domains (problem solving, language understanding, pattern recognition etc.)
  - basic mechanisms: control of interaction with environment, representation of information in memory, strategy to select rules
  - Advantage: different models realized in the same architecture get comparable
  - Disadvantages: Modeling language often over-constrains what is modelled, modelling can easily get “unnatural” and “clumsy”
- Examples: GPS, EPIC, SOAR, ACT

History of ACT-R

- Anderson and Bower (Standford): 70ies HAM (Human Associative Memory), declarative memory with information spreading
- Carnegie Mellon University: ACT* (Adaptive Control of Thought), book 1983
- Covering: Knowledge Representation, Spread of Activation, Control, Fact Memory (not very satisfying: procedural learning)
- Quite succesful applications in intelligent tutoring (Lisp Tutor, Geometry Tutor, Algebra Tutor)
- ACT-R (rational), book 1990; human memory and categorization
- Current: applications, e.g. in the domain of modeling air traffic controlers
- ACT is a production system (see CogSysI) implemented in Common Lisp
- see http://act-r.psy.cmu.edu/

Basic Structure of ACT



ACT Components

- Memory modules:
  - declarative memory: consisting of facts such as Washington, D.C. is the capital of United States, France is a country in Europe, or $2 + 3 = 5$
    know what, accesible, verbalizable
  - procedural memory: made of productions. Productions represent knowledge about how we do things: for instance, knowledge about how to type the letter “Q” on a keyboard, about how to drive, or about how to perform addition
    know how, skills, control of cognition
- Perceptual-motor modules: interface with the real world; most well-developed perceptual-motor modules in ACT-R are the visual and the manual modules

ACT Components cont.

- Buffers: contents of the buffers at a given moment in time represents the state of ACT-R at that moment
  Special buffer: goal buffer, holding current goals
- Pattern Matcher: searches for a production that matches the current state
- Execution: Match-Select-Apply; heuristics for selection

Declarative Units

- Organized in chunks with slots:

```
Action023:
    isa chase
    agent dog
    object cat

Fact3+4:
    isa addition-fact
    addend1 three
    addend2 four
    sum seven
```

Spread of Activation (1)

- Classical experiment and model (already in HAM): Fan-Effect (Anderson, 1974)
- Study sentences, where people and locations occur in one, two, or three of the sentences (fan)

```
 1. A hippie is in the park.
 2. A hippie is in the church.
 3. A hippie is in the bank.
 4. A captain is in the park.
 5. A captain is in the cave.
 6. A debutante is in the bank.
 7. A fireman is in the park.
 8. A giant is in the beach.
 9. A giant is in the dungeon.
10. A giant is in the castle.
11. A earl is in the castle.
12. A earl is in the forest.
13. A lawyer is in the store.
```

Spread of Activation (2)

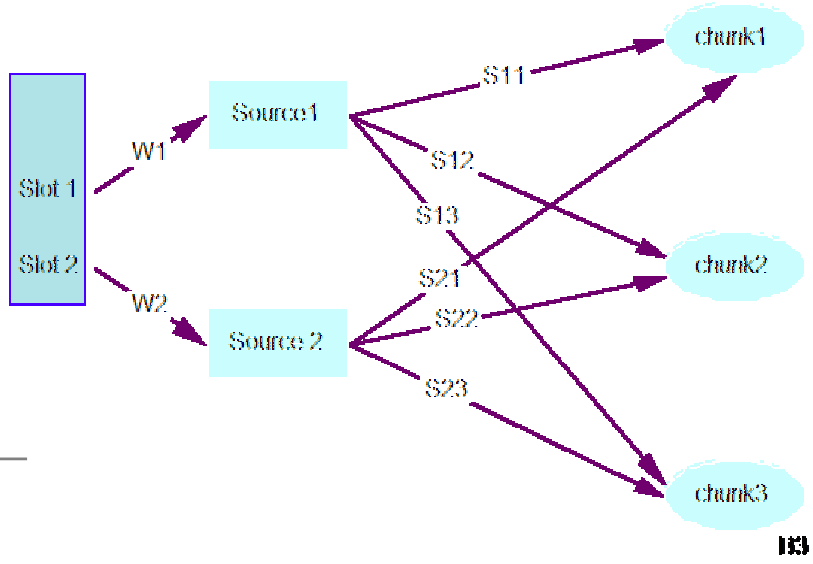
- Judge whether the facts were seen before, mixed with new sentences (foils)
- Main Result:

	Targets				Foils			
Location Fan	Person Fan 1	2	3	Mean	Person Fan 1	2	3	Mean
1	1.111	1.174	1.222	1.169	1.197	1.221	1.264	1.227
2	1.167	1.198	1.222	1.196	1.250	1.356	1.291	1.299
3	1.153	1.233	1.357	1.248	1.262	1.471	1.465	1.399
Mean	1.144	1.202	1.357	1.20	1.236	1.349	1.340	1.308

When fan increases, the time to respond increases. Foil sentences take longer to respond to than the targets

Spread of Activation (3)

- Goal buffer provides a context in which to perform a retrieval.
- Chunks spread activation to the other chunks in declarative memory based on their relations to the other chunks which we call their strengths of association. This essentially results in increasing the activation of those chunks which are related to the current context.



Spread of Activation (4)

Activation A_i of a chunk i

$$A_i = B_i + \sum_j W_j S_{ji} + \epsilon$$

with

- B_i: The base-level activation reflects the recency and frequency of practice of the chunk (effect of prior knowledge), can be set as parameter
- $S_{ji} S - ln(fan_j) =$: strength of association between source j and chunk i
  S is a parameter to be estimated
- W_i: amount of activation from source j
- ϵ: noise value

Productions

- condition-action pair
- condition (left-hand side) specifies a pattern of chunks that must be present in the buffers for the production rule to apply
- action (right-hand side) specifies some actions to take; basically modifications of chunks or requests for other chunks

Production Example

```
(p add-ones
   =goal>
      isa add-pair
      one1 =num1
      one2 =num2
      one-ans waiting
   =retrieval>
      isa addition-fact
      addend1 =num1
      addend2 =num2
      sum =sum
==>
  =goal>
      one-ans =sum
      carry waiting
  +retrieval>
      isa addition-fact
      addend1 ten
      sum =sum
)
```

English Description

```
If   the goal
is to add a pair of numbers
and =num1 is the ones digit of the firs
and =num2 is the ones digit of the secc
and the ones digit of the answer is wai
and the retrieved chunk
is of type addition-fact stating
    =num1 plus
    =num2 equals
    =sum
Then
change the goal so that
the ones answer is =sum
and note that a carry is being processe
and request a retrieval
of a chunk of type addition-fact
with addend1 being ten
and the sum being =sum
```

Tower of Hanoi in ACT (1)

```
(chunk-type disk size peg state)
(chunk-type peg name)
(chunk-type tower-task largest current goal)
(chunk-type move-disk disk to from other test at)
(chunk-type countfact first then)
(chunk-type encode-configuration size state)
(add-dm
   (disk1c isa disk size 1 peg c state current)
   (disk2c isa disk size 2 peg a state current)
   (disk3c isa disk size 3 peg b state current)
   (disk4c isa disk size 4 peg b state current)
   (disk1g isa disk size 1 peg b state goal)
   (disk2g isa disk size 2 peg a state goal)
   (disk3g isa disk size 3 peg c state goal)
   (disk4g isa disk size 4 peg c state goal)
   (fact01 isa countfact first 0 then 1)
   (fact12 isa countfact first 1 then 2)
   (fact23 isa countfact first 2 then 3)
   (fact34 isa countfact first 3 then 4)
   (fact45 isa countfact first 4 then 5)
   (a isa peg name a)
   (b isa peg name b)
```

Tower of Hanoi in ACT (2)

```
(p start-tower
"IF goal is to solve tower task of size =size and =size greater than 1
 THEN set a subgoal to move disk =size checking disk =new and
      change the goal to solve a tower task of size =new"
   =goal> isa tower-task
          largest =size
          - largest 1
          current t
          goal t
   =fact> isa countfact
          first =new
          then =size
==>
   =goal>    goal nil
             largest =new
   =newgoal> isa move-disk
             disk =size
             test =new
   !push! =newgoal
)
```

Tower of Hanoi in ACT (3)

```
(p move
"IF the goal is move disk of size n to peg x
    and all smaller disks have been checked
 THEN move disk n to peg x and pop the goal"
   =goal>
      isa move-disk
      test 0
      from =from
      to =to
      disk =size
   =disk>
      isa disk
      size =size
==>
   =disk>
      peg =to
   !pop!
)
```

Tower of Hanoi in ACT (4)

- Chunks given completly
- Only two productions (of many) and only partially in ACT-R language
- Specific models for: memory for subgoals, learning with practice

User Modeling

- No running model online
- Lebiere, C, Anderson, J. R., & Bothell, D. (2001). Multi-tasking and cognitive workload in an ACT-R model of a simplified air traffic control task.