Post-Midterm Review

Shout-out: Tomoyuki Miura, with the top score of 96%

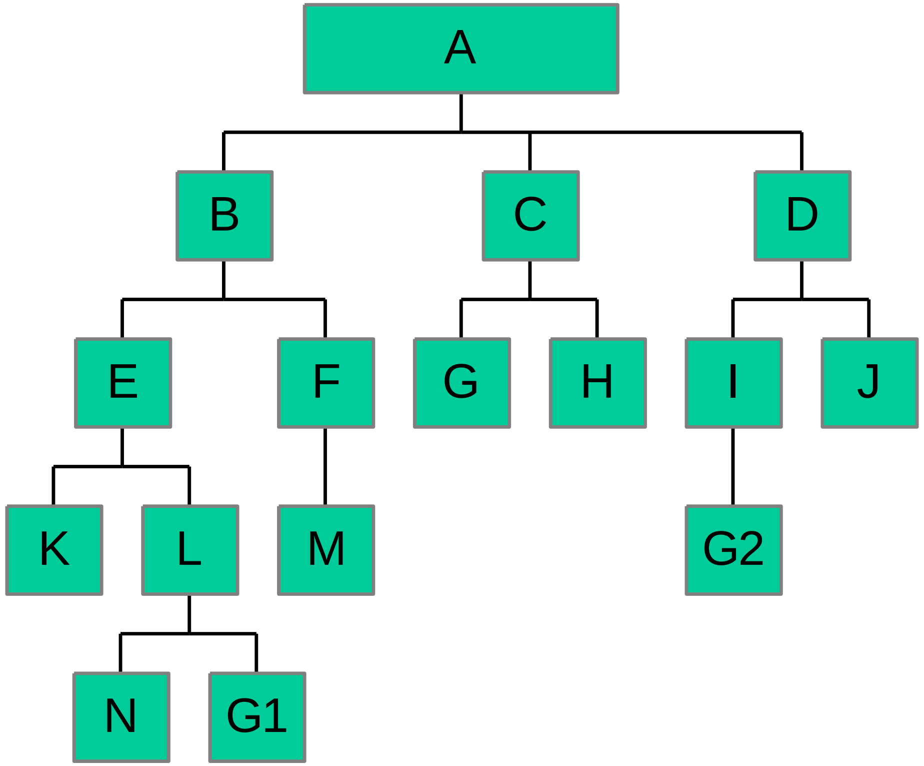
Turing Test

- A test of intelligence
- One judge, trying to decide which of two conversationalists is the human and which is a computer
- If computer is judged to be human 50+% of the time, must concede that computer is intelligent
- Unlimited, in both length and scope
- Never passed
- Criticisms: Lots of possibilities, e.g. computers don’t have a soul, computers can’t feel emotions, test is too strong, Chinese room.

Searle's Chinese Room (CR)

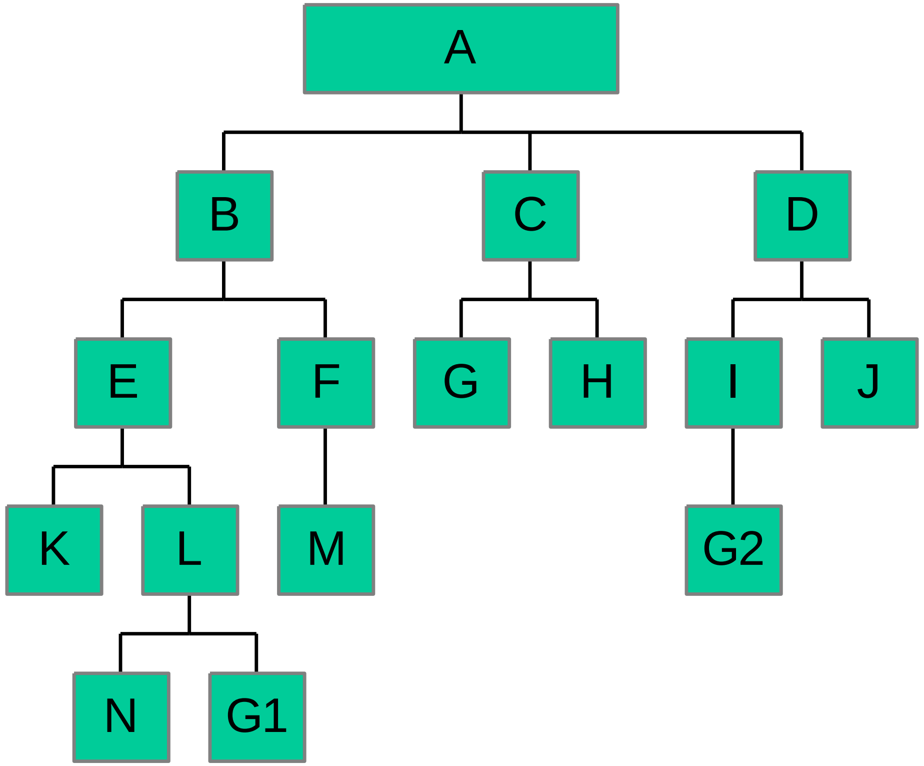
- A program that *passes the Turing Test in Chinese* is reimplemented as a library, with the code in the books and a *non-Chinese speaking* person acting as CPU.
- The CR should still pass the TT, albeit very slowly!
- Searle's point: Where is the understanding? It's not in the books, not in the human (he doesn't understand the inputs, the outputs, or anything in between!)
- Several possible responses, e.g. the intelligence is in the *whole system.*

Depth-first search



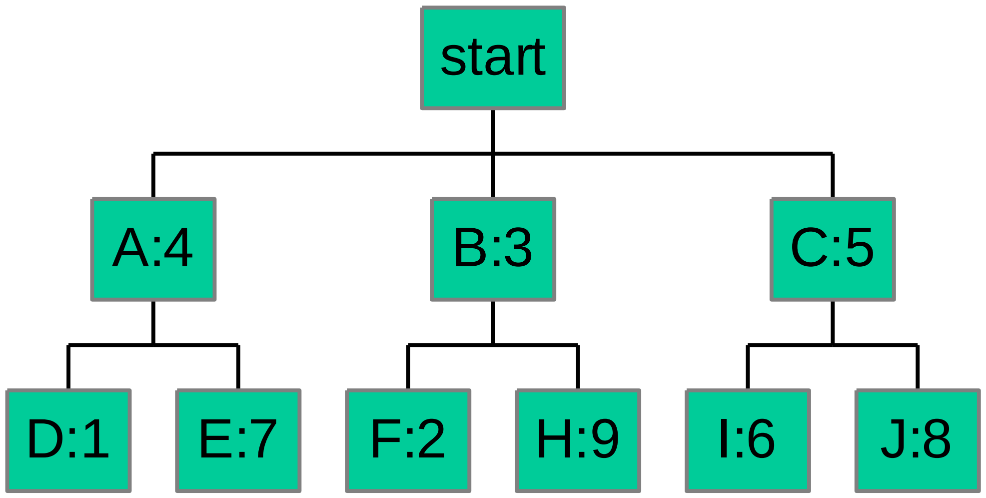
OPEN	CLOSED
A	
B C D	A
E F C D	A B
K L F C D	A B E
L F C D	A B E K
N G1 F C D	A B E K L
G1 F C D	A B E K L N

Breadth-first search



OPEN	CLOSED
A	
B C D	A
C D E F	A B
D E F G H	A B C
E F G H I J	A B C D
F G H I J K L	A B C D E
G H I J K L M	A B C D E F
H I J K L M	A B C D E F G
I J K L M	A B C D E F G H
J K L M G2	A B C D E F G H I
K L M G2	A B C D E F G H I J
L M G2	A B C D E F G H I J K
M G2 N G1	A B C D E F G H I J K L
G2 N G1	A B C D E F G H I J K L M

A* search



OPEN(f(x)=g(x)+h(x))	CLOSED
B(4) A(5) C(6)	
F(4) A(5) C(6) H(11)	B
A(5) C(6) H(11)	B F
D(3) C(6) E(9) H(11)	B F A
C(6) E(9) H(11)	B F A D
I(8) E(9) J(10) H(11)	B F A D C
E(9) J(10) H(11)	B F A D C I
J(10) H(11)	B F A D C I E
H(11)	B F A D C I E J
	B F A D C I E J H

Horizon effect

- In non-exhaustive competitive search (informed or uninformed), there is some maximum number of nodes that can be evaluated from the current node.
- Informed search uses heuristics to make sure we're evaluating those nodes in a promising order, but heuristics are *fallible.*
- So, there's always the possibility that there will be a very positive or negative node just outside of search range.
- In competitive search, Alpha-Beta Pruning prevents the search from wasting time/memory on irrelevant branches, allowing the search to go deeper on the branches that are important.

FOPL

- Loki is a cat and he is fat.
  - cat(loki) ^ fat(loki).
- All fat cats love food.
  - $\forall X\ \forall Y$: cat(X) ^ fat(X) ^ food(Y) → loves(X, Y).
- Some cats do not love Loki.
  - $\exists X$: cat(X) ^ ¬loves(X, loki).

Weighted sum membership function for "bird"

Should look something like this:

- A reasonably sensible set of features, e.g. "has wings" (HW), "lays eggs" (LE), "has feathers" (HF), "can fly" (CF)
- A weighted and normalized (i.e. so that it returns 0-1) sum function, e.g.: $\mu(x) = (3 * HW + 1 * LE + 4 * HF + 2 * CF)/10$
- The calculated membership function for each picture:
  - Roadrunner: (3+1+4+0)/10 = 0.8
  - Platypus: (0+1+0+0)/10 = 0.1
  - Mynah: (3+1+4+2)/10 = 1
  - Pteradactyl: (3+1+0+2)=0.6

Artificial Neural Networks (ANNs)

- ANNs represent knowledge as weights between nodes and thresholds at nodes.
- The Back-Propagation Algorithm trains nodes in a network with one or more hidden layers.
  - Initialize network. [no points deducted for leaving this out]
  - Present one piece of training data at input, and propagate values forward.
  - Compare actual output to desired output and calculate the error
  - Propagate adjustments backward through the network, depending on how much each connection contributed to the error.
  - Repeat, until error on training data is below some threshold.

World Knowledge Problem

- In order to solve even very ‘simple’ real world problems, a LOT of knowledge is required. For example, understanding:
  - a natural language sentence, e.g. “I dropped the carton, so I guess we’re having an omelet”
  - the preconditions/outcomes/constraints on an event, e.g. “I walked from Waikiki to Kaneohe starting at 8am this morning.” What has this changed in the world? E.g.
    - Where am I now?
    - What time is it?
    - How old am I?
    - How many legs do I have? 2? 1? 6?
    - How many hearts?
    - The capital of Canada was Ottawa when I started – what is it now?

Imprecise, incomplete or uncertain knowledge representation

Looking for:

- Bayesian (classical) probability. Pro: mathematically solid. Con: hard to get conditional probabilities, especially from a human
- Certainty factors: Pro: more intuitive than Bayesian, easy to calculate. Con: Not so solid, garbage-in/garbage-out.

Also accepted: Neural networks (robust under noisy/incomplete inputs), fuzzy logic/ES (allows for reasoning over qualitative, non-crisp sets)

LISP

- Q1: Let’s run it!
- Q2: Make a new definition for “spiffy” so that it returns NIL if its argument is an atom, T if its argument is a list, and its argument in all other cases. See file.

Quick-fire round

- *You are building a diagnostic expert system rule base. The goal is to determine whether a patient has malaria or not. It should use: Backward-chaining.* Why? You have a hypothesis (malaria), and backward chaining doesn't waste time on rules/data that have nothing to do with the hypothesis.
- *You're building a system to recognize insect species that takes images of bugs as input. You should use: An artificial neural network.* Why? Of the options given, only ANNs are good with images.
- *Your boss has told you to build a user interface that will understand absolutely anything the user says. Your biggest problem is: the World Knowledge Problem.* Why? See earlier question on the World Knowledge Problem.

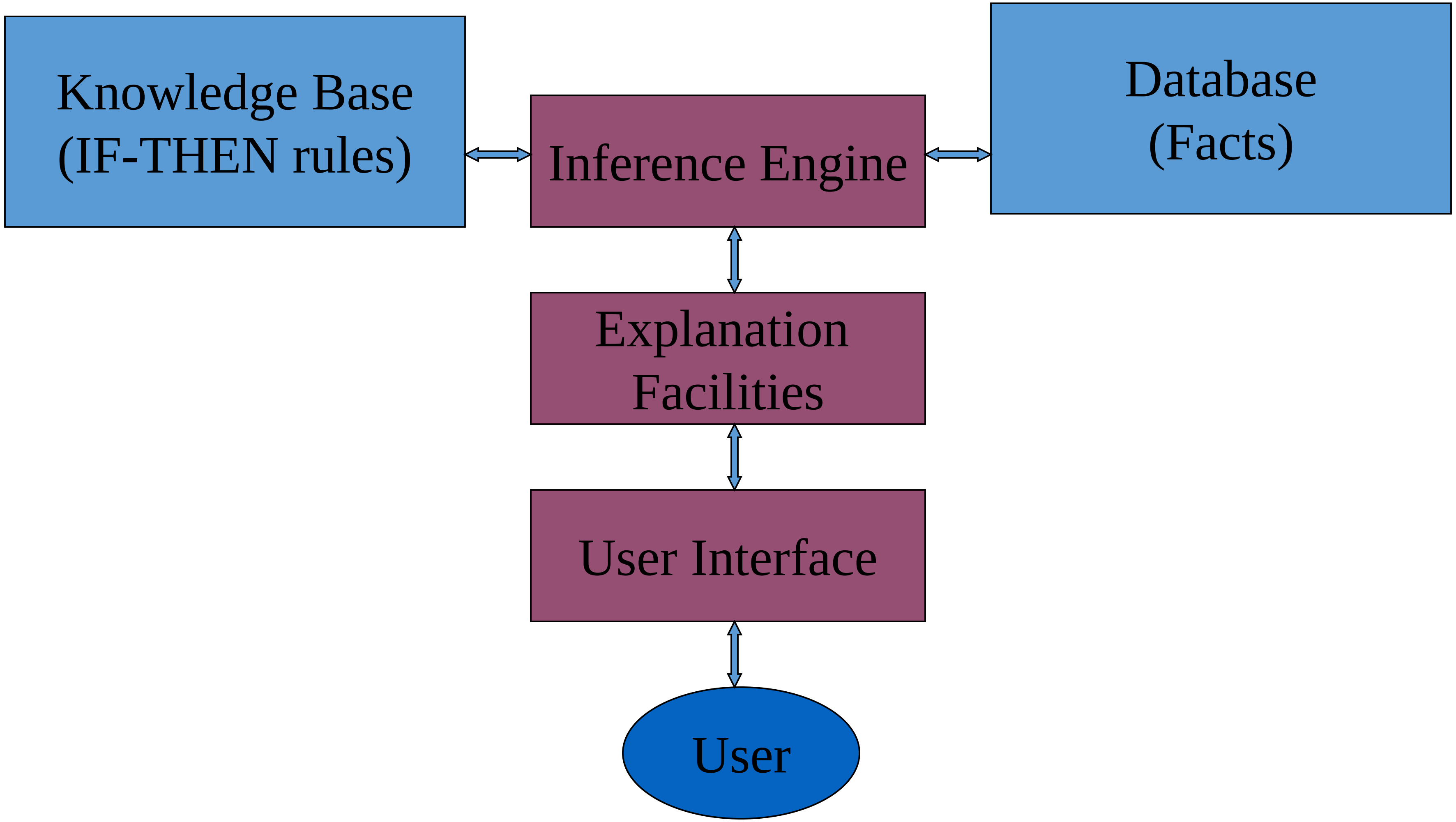
Quick-fire Round 2

- *You’ve trained an artificial neural network to recognize images of the employees of your company. Good job! However, whenever someone wears a new shirt or has a new haircut, the system rejects them. This is an example of: Overfitting.* Why? If you overtraining a neural network, it has a tendency to memorize the training data, losing the ability to generalize to new cases.
- *What is one potential solution to the problem above? Cross-validation.* Why? If you reserve some training data, and *don’t* train the network on it, you can test for overfitting. If the error continues to go down on the training data, but goes up on the cross-validation data, the network is probably overfitting.

Quick-fire Round 3

- *For the 8-tile puzzle, which of these heuristics is the most informed? Manhattan distance.* Why? Manhattan Distance ≥ Straight-Line Distance ≥ for all states.
- *For the 8-tile puzzle, which of these heuristics is/are* ***NOT*** *admissible. H(n)=2.* Why? For an admissible heuristic, h(n) ≥ h*(n) [the *actual* distance to the goal] for all n. Let’s choose an n that is one step away from a goal (h*(n) = 1). Here h(n) overestimates, so it is not admissible.

Structure of a rule-based expert system



Calculating Certainty Factors (formula given!)

IF A AND B THEN C

cf(A)=0.5

cf(B)=0.8

cf(A^B)=min(0.5,0.8)=0.5

cf(C)=cf(A^B)*cf(rule)=0.5*0.6=0.3

Combining certainty factors if two rules fire

- cf(rule1)=0.3 [given by previous question]
- cf(rule2)=0.7 [asserted by this question]
- Both positive, so:
  cf(combined) = 0.3 + 0.7*(1-0.3) = 0.3 + .49 = 0.79
- Note that order is irrelevant:
  cf(combined) = 0.7 + 0.3*(1-0.7)=0.7+0.9=0.79

$$cf(cf_1, cf_2) = \begin{cases} cf_1 + cf_2 * (1 - cf_1) & \text{If } cf_1 \text{ and } cf_2 > 0 \\ \dfrac{cf_1 + cf_2}{1 - \min(|cf_1|, |cf_2|)} & \text{If } cf_1 \text{ xor } cf_2 > 0 \\ cf_1 + cf_2 * (1 + cf_1) & \text{If } cf_1 \text{ and } cf_2 < 0 \end{cases}$$

Fuzzy calculations

μ_{windy}(today)=0.7

$$\mu_{windy}^{very}(x) = [\mu_{windy}(x)]^2 = 0.7^2 = 0.49$$

$$\mu_{\neg verywindy}(x) = 1 - \mu_{verywindy}(x) = 1 - 0.49 = 0.51$$

Who is Cosmo?

More Prolog

Remember:

Prolog tries very hard to be declarative. So:

- There are no functions: nothing is *returned.*
- There are no procedures: you can’t explicitly say “do this, then this, then this.”

• All Prolog can *do* is depth-first search (so, the order of the clauses DOES matter) and unification.

Equality in Prolog

See http://www.swi-prolog.org/pldoc/man?section=arith for exact definitions, but roughly:

- A = B. means: Can they be unified, symbolically?
- A == B. means: Are they exactly the same (barring operator syntax shifts), before evaluation?
- A =:= B. means: Do they evaluate to the same thing? (i.e. arithmetic equal)
- A is B. means: As close as you are going to come in Prolog to assignment of a value to a variable. A is not evaluated, B must evaluate, and they must be unifiable.

In-class exercise: Test cases

Write down what you think will happen in each of these cases. Then, test them in Prolog.

```
a(b,c) = a(C,B).
A(b,c) = a(b,c).
a(b(c),C) = a(C,B).
a(b(C),C) = a(C,B).
a(b,c) == a(b,c).
a(B,c) == a(b,c).
a(b,c) =:= a(b,c).
4+5 =:= 10-1.
4+5 is 10-1.
X is 10-1.
a(X,c) is a(10,c).
```