CPSC 322
Introduction to Artificial Intelligence

October 29, 2004

Things...

Send me pairings for term project no later than Sunday, Oct. 31

Midterm exam 2 is Monday

Bring me your first midterms before the second midterm if you want to be retested on problem 3

Midterm 2

Some people have suggested that midterm 1 wasn’t difficult enough

Midterm 2

Be careful what you wish for...
you might get it

Midterm 2

All about search
(for example)

Intelligence = search?
Blind search vs. heuristic search
Different types of each
- Algorithms
- Behaviour
- Time/space requirements

What happens when we tweak algorithms in new ways?
Apply search strategies to problems you haven't seen before
Collecting paths
Generating states
Explain when one type of search is better than another
Game search
Read/write search-related CILOG code

Search is seductive

Search is an important part of AI, but don’t be seduced into believing it’s everything...intelligence requires knowledge

Without knowledge, search is pointless, so the real issue here is figuring out what knowledge goes into the AI system and how it’s represented

If the knowledge isn’t right, any search strategy will fail to find useful answers, no matter how fast the processors are or how many of them are being used

Tuning search heuristics won’t help either

Knowledge poses problems

It’s indispensable, but...

- it’s voluminous
- it’s hard to characterize accurately
- it’s constantly changing
- it’s organized in different ways depending on how it’s used

Those challenges make it really hard to do AI...
how can we be sure we’re doing it right?

How to do the knowledge thing right

AI approaches to problems are best when they exploit knowledge that's represented in such a way that...

- the knowledge captures generalizations
- the knowledge can be understood by people who must provide it, work with it, update it
- the knowledge can be modified easily for error correction or changes in the world
- the knowledge can be used in many situations even if it's not totally accurate or complete
- the knowledge itself can be used to help constrain the search for the knowledge most appropriate to the task at hand

Representing Knowledge

Here are some general software engineering questions you may want to ask before building AI systems:

- What exactly is the activity that you want from this system you're going to create to solve some complex problem?
- What does your system need to know in order to perform that activity?
- How are you going to encode or represent that knowledge inside your system? (e.g., What will the language of the symbols be? What will the symbols map to?)
- How will your system know which piece(s) of knowledge to use at a given time, and how will it get at that knowledge without looking at all the knowledge?
- Once the system finds the appropriate knowledge, how will it use the knowledge?

Representing Knowledge

Here are some general software engineering questions you may want to ask before building AI systems:

- What exactly is the activity that you want from this system you’re going to create to solve some complex problem?
- What does your system need to know in order to perform that activity?
- How are you going to encode or represent that knowledge inside your system? (e.g., What will the language of the symbols be? What will the symbols map to?)
- How will your system know which piece(s) of knowledge to use at a given time, and how will it get at that knowledge without looking at all the knowledge?
- Once the system finds the appropriate knowledge, how will it use the knowledge?

Defining the task: what the system does

Representing Knowledge

Here are some general software engineering questions you may want to ask before building AI systems:

- What exactly is the activity that you want from this system you're going to create to solve some complex problem?
- What does your system need to know in order to perform that activity?
- How are you going to encode or represent that knowledge inside your system? (e.g., What will the language of the symbols be? What will the symbols map to?)
- How will your system know which piece(s) of knowledge to use at a given time, and how will it get at that knowledge without looking at all the knowledge?
- Once the system finds the appropriate knowledge, how will it use the knowledge?

Defining the knowledge representation: what the system needs to know

Representing Knowledge

Here are some general software engineering questions you may want to ask before building AI systems:

- What exactly is the activity that you want from this system you’re going to create to solve some complex problem?
- What does your system need to know in order to perform that activity?
- How are you going to encode or represent that knowledge inside your system? (e.g., What will the language of the symbols be? What will the symbols map to?)
- How will your system know which piece(s) of knowledge to use at a given time, and how will it get at that knowledge without looking at all the knowledge?
- Once the system finds the appropriate knowledge, how will it use the knowledge?

Defining the process: how the system performs the task

Representing Knowledge

Here are some general software engineering questions you may want to ask before building AI systems:

- What exactly is the activity that you want from this system you're going to create to solve some complex problem?
- What does your system need to know in order to perform that activity?
- How are you going to encode or represent that knowledge inside your system? (e.g., What will the language of the symbols be? What will the symbols map to?)
- How will your system know which piece(s) of knowledge to use at a given time, and how will it get at that knowledge without looking at all the knowledge?
- Once the system finds the appropriate knowledge, how will it use the knowledge?

Most of the questions have something to do with knowledge

As an example...

Let’s say you’re asked to build a system to understand simple stories

John and Mary went to McDonald’s.
John ordered a Big Mac and fries.
Mary had a Quarter Pounder.
John put the trash in the wastebasket.
They went home.

- What exactly is the activity that you want from this system you’re going to create to solve some complex problem?

As an example...

Let’s say you’re asked to build a system to understand simple stories

John and Mary went to McDonald’s.
John ordered a Big Mac and fries.
Mary had a Quarter Pounder.
John put the trash in the wastebasket.
They went home.

- What does your system need to know in order to perform that activity?

As an example...

Let’s say you’re asked to build a system to understand simple stories

John and Mary went to McDonald’s.
John ordered a Big Mac and fries.
Mary had a Quarter Pounder.
John put the trash in the wastebasket.
They went home.

- How are you going to encode or represent that knowledge inside your system? (e.g., What will the language of the symbols be? What will the symbols map to?)

As an example...

Let’s say you’re asked to build a system to understand simple stories

John and Mary went to McDonald’s.
John ordered a Big Mac and fries.
Mary had a Quarter Pounder.
John put the trash in the wastebasket.
They went home.

- How will your system know which piece(s) of knowledge to use at a given time, and how will it get at that knowledge without looking at all the knowledge?

As an example...

Let’s say you’re asked to build a system to understand simple stories

John and Mary went to McDonald’s.
John ordered a Big Mac and fries.
Mary had a Quarter Pounder.
John put the trash in the wastebasket.
They went home.

- Once the system finds the appropriate knowledge, how will it use the knowledge?