CS 696i, Spring 2005, part III

Computer Vision

(Making Machines See)

and other miscellaneous stuff

Kobus Barnard

Computer Science, University of Arizona

Administrivia

Second and third assignments should now be available:
http://kobus.ca/teaching/cs696i/spring05/

Off campus use will require:
login: me
pw: vision4fun

Extra office hours after class and tomorrow from 3:30 to 5.

TOC---main points from last time

We consider a computationally intelligent system to map integers to integers (discrete)

If we are to accomplish this using regular digital computer (modeled by a Turing machine), then we can appeal to what is known about computability and tractability

Computer systems are general purpose and equivalent in they can emulate each other and Turing machines.

If a system can be emulated on one of these machines, then it can to no more than them.

TOC---main points from last time

The number of mappings from integers to integers is uncountable (like the real numbers)

The number of computers is countable

===> Most input / output relations can not be produced by a computer

Theory of Computation

Does this matter?

Possibly not:

1. The integers that any machine will see are limited. The behavior on all integers is beyond the scope of what an intelligent system has to deal with.

2. One could further conjecture that the excluded mappings are not interesting---that the useful mappings by nature have (perhaps approximately) some structure making them ones that are computable.

Theory of Computation

Why study this then?

The main point is to provide intuition as to whether what we call intelligence can be computational and by what. The arguments revolve around these two issues:

1. Mappings from integers to integers are sufficient to (and appropriate) to characterize what needs to be done.
2. That it is feasible to do those mappings with a “sensible” computational strategy.

Theory of Computation

An example of a non-sensible method of providing the mapping is to memorize it (why?).

One characterization of "sensible" is that there is some clear competitive advantage to doing it that way.

(I do not simply mean "tractable" by current computers).

Theory of Computation

The notion of competitive advantage suggests that there must be an efficiency gain due to being able to construct the needed mapping with a system which is simpler than the map.

Put differently, we are looking for something which looks like a clever algorithm.

Theory of Computation

The notion of competitive advantage suggests that there must be an efficiency gain due to being able to construct the needed mapping with a system which is simpler than the map.

Put differently, we are looking for something which looks like a clever algorithm.

Discussion point: If it is not an algorithm, then what would it be like? Can we imagine it? Does this say something about the problem, or just something about our way of thinking.

Tractability

Tractability refers to how the computational resources required to solve a problem scale with problem size.

Example problems:

1) Searching
2) Sorting
3) Clustering

Tractability

Problem tractability is determined by looking at how computational requirements grow with problem size, n.

Sorting, which has known polynomial time running time algorithm (specifically n*log(n)) is tractable.

Many interesting problems have no known polynomial time algorithm. Typically they seem to require exponential time (e.g. 2^n).

NP completeness

It is assumed (but not known) that a large class of problems which *can* be solved in exponential time, *require* exponential time.

NP problems: Proposed solutions can be processed in polynomial time and the problem can be solved by trying all possibilities. The technical device for producing the "guesses" is the non-deterministic Turing machine

NP completeness

If an NP problem can be used to solve another (with polynomial overhead), then the first problem is at least as hard as the second.

“NP complete” means that the problem is as hard as any other NP problem (it can be used to solve the lot).

Cook’s result: There exist an NP complete problem (SAT).

It has yet to be proved that these problems are NOT solvable in polynomial time.

However, “intractable” generally means NP complete.

Tractability

Tractability is a function of of your computational system

If quantum computers become an everyday reality, then that will change our notion of tractability.

It is not clear (at least to me) what the right model for the brain is to ask about tractability!

However, parallel computers do not change the notion of tractability much as you generally pay linearly for power, but the problems get harder exponentially.

Tractability

Tractability discussion point:

Would the problems in computational intelligence be largely solved if we could run exponential (or worse) algorithms in polynomial time?