Talking to the Machine

Amr Sabry

Department of Computer

Indiana University

9 February 2007

Talking to children

Talking to children

Did you notice how we talk to children?

We don't say:

> Assuming the standard three-dimensional physical space, please identify the coordinates of the spherical object in question.

Instead we say: Where . . . is . . . ball?
and then we might say: Here is ball!

Why?

Why do we talk to children this way?

- Children can only understand a simple vocabulary
- Children only understand simple ideas
- Children can only understand slowly and clearly-enunciated sounds
- Children understand the emotion in the sentence more than the actual words

Instructing a child

We talk to children to express love, to make them feel safe, etc., and sometimes to make them do things for us.

We talk to machines to make them do things for us.

Instructing a 3-year old to get you a drink

- Walk to the fridge
- Open the fridge
- Get the milk. Be careful, it's heavy.
- Put the milk on the counter.
- Close the fridge
- Get a glass from that drawer.
- Pour some milk. Not too much. It might spill.
- Bring me the glass. Use two hands.
- Put the milk back in the fridge. Don't forget to close the door.

Instructing a 3-year old to get you a drink

Our instructions are tailored to what we understand about the child's abilities.

- We assume that the child can interpret basic English sentences, that the child can walk, that the child can exercise the right muscles to carry a heavy container, etc. We don't give any instructions about how to perform such tasks.
- We are careful to use instructions that the child can understand and has a chance at performing. We don't ask the child to go to the store to buy some milk or ask the child to read the labels on the containers.

Instructing a 3-year old to get you a drink

- We assume the child can carry a full glass but we know this is challenging. We give detailed instructions about such tasks.

- We assume the child can only remember a few things at once. We break the task into small steps and wait for feedback before giving the next step.

Instructing a 10-year old to get you a drink

I am thirsty.

The child understands that when you are thirsty, the nice thing to do is to offer you a drink. We can rely on the child to have a memory of our preferences and to get a glass of milk; if there is no milk we expect the child to make a decision to get water instead, etc.

Lessons on how to talk to children (or machines!)

We must tailor our instructions to the machine.

- We must understand what the machine is capable of doing! Can't ask your cell phone to heat your pizza
- We must understand the language that the machine can interpret. Can't ask the microwave to heat your leftovers by singing to it. Must push the right buttons on the panel.
- We must monitor the progress of the machine and give appropriate instructions. Turn the microwave off when the popping sounds are more than a second apart.

Microwaves, cell phones, etc. What about computers?

Well, microwaves and cell phones are very special-purpose computers and we "talk" to them all the time.

Talking to simple machines

Talking to a microwave

- Sometimes it is very easy to say what we want to say. If we want to say "heat this item for one minute" all we have to do is push one button.
- Some things are very hard to say. Have you ever tried to thaw a chicken using a microwave? It involves a long complicated process with many buttons and it usually does not work!
- Some things are easy to say but only once you convert them to the right language. When a recipe says to cook the food for 3 minutes, it generally assumes a standard 700-watt. If your oven is a 1400-watt oven, for how long should you cook the food?

Talking to a cell phone

- Dialing a phone number is trivial.
- Maintaining a "Contacts List" is almost trivial.
- Turning your favorite song into a ringtone is more complicated.

Programming

Talking to a machine is generally called programming the machine. With microwaves and cell phones we can usually express two kinds of programs:

- The simplest programs just tell the machine to do something it was designed to do automatically — with the push of one button. Heat food for one minute.

- More complicated programs involve a sequence of commands. Set the power to 30%, then cook for 10 minutes, then increase the power to 50%, and cook for 10 more minutes.

Programming

With more advanced machines (computers!), we can express more sophisticated programs:

- Repeat the following commands until a certain condition holds. Keep downloading pictures until the user clicks "Done".

- Try to do the following commands. If something goes wrong, abort the commands and execute these other commands instead. Try to copy this folder and the files it contains to the data stick; if the data sticks becomes full in the middle of copying one file, erase this one file, and tell the user that the operation failed.

Talking to computers

Programming a "bare" computer

- When you buy a new laptop, it says "Intel inside"
- What if we only had that "Intel" thing and nothing else?
- To print `Hello` on the screen in x86 assembly language:

```
Message db "Hello"
.code

mov dx,OFFSET Message
mov ax,SEG Message
mov dx,ax
mov ah,9
int 21h
mov ax,4c00h
int 21h
END
```

System software

- For convenience, modern computers come with layers and layers and layers of software who job is to translate from some human-like language to the low-level assembly language.

- Even better it is very easy to teach a computer to speak almost any language of our choice, provided somebody knows how to translate this language to the low-level assembly language — or to some other language that has a translation to assembly language.

What is programming again?

A program is a set of directions telling a computer exactly what to do.

A programming language is a precise notation for specifying sequences of directions to a computer. Unlike English there are no ambiguities and no nuances.

An algorithm is the "general idea" with which we try and solve the problem.

Algorithms

Sorting

Let's do something simple: sorting a deck of cards

Algorithm I:

- Find the ace of spades and put it upside down
- Find the 2 of spades and put it upside down on the ace of spades
- Find the 3 of spades ...
- ...

Sorting

Algorithm I

- Find the ace of spades and put it upside down: might have to go through 52 cards
- Find the 2 of spades and put it upside down on the ace of spades: might have to go through 51 cards
- Find the 3 of spades ...
- ...

Might have to go through $52 + 51 + 50 + \ldots$ cards.

Sorting

Algorithm II

- Separate the four suits: Goes through 52 cards
- Sort each of the suits using Algorithm I: might have to go through 13+12+11+. . . cards four times

Best algorithm: goes through at most 205 cards

The Seven Bridges of Königsberg

Can one walk around the city in a way that would involve crossing each bridge exactly once ?

Computer Languages

Choosing a language

If I want to say something to this computer, the first decision is to choose the language in which I want to say it.

There are millions of computer languages; some are general-purpose, and some very specific to a class of applications.

Even better, we can make up our own language!

Invent a language

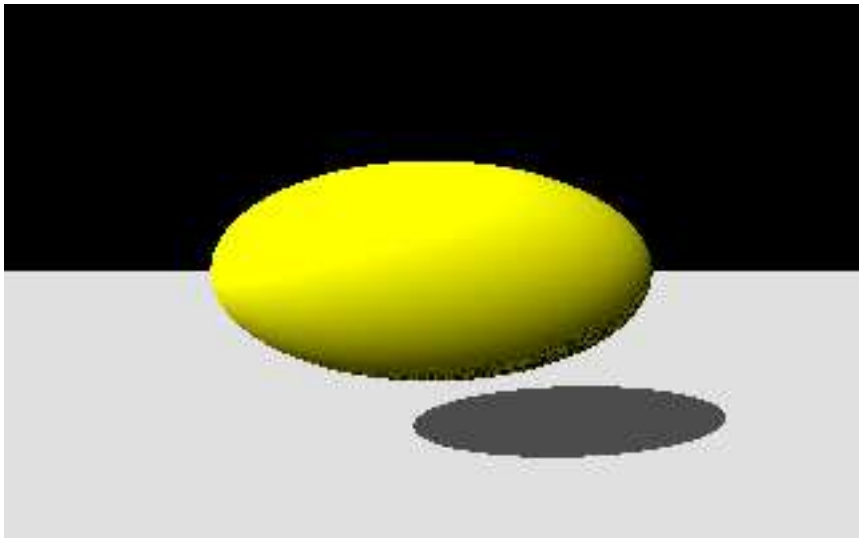
- A honors project for H211 students.
- Take a description written in this made-up language:

```
yellow matte apply sphere 2.0 1.0 1.0 scale
                          0.0 0.0 3.0 translate
white matte apply plane 0.0 -2.0 0.0 translate
union /scene

1.0 -1.0 1.0 point
1.0 1.0 1.0 point light /l
0.3 0.3 0.3 point
[l]

scene 1 90.0 320 200 "ellipsoid.ppm" render
```

And the output is. . .



How?

Write an interpreter from our made-up language to some language that the computer already understands.

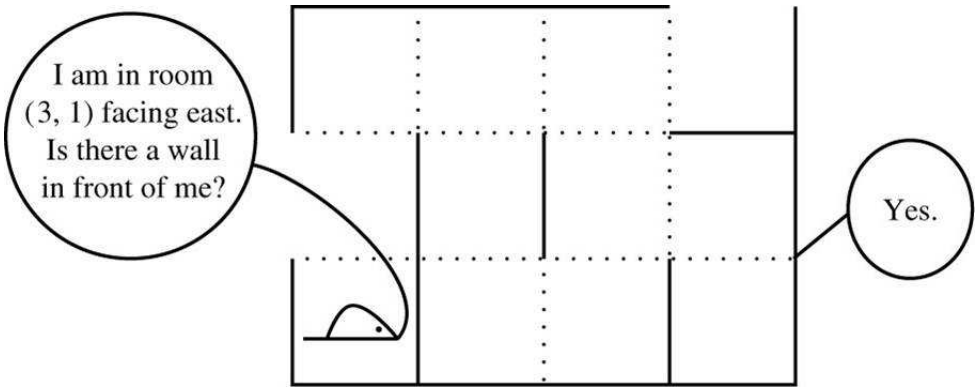
Really once you master interpreters you can say anything you want to the computer and in the way you want it!

This is the idea behind one of our most distinguished courses here at IU C311.

A Complete Example

Mouse in a maze

Let's try to tell the computer how to find its way in a maze.



Commands

Let's invent a language with the following commands:

- Step forward
- Turn left
- Turn right
- In maze?
- Facing wall?

Problem

- Construct a program that gets the mouse from entry to exit!
- Use the commands on the previous slide.
- Don't go through walls!

Ideas?

Ideas?

Hug the wall!

- Have the mouse walk hugging the wall to its right.

Algorithm

```
step forward;
while (inside the maze?) {
  turn right;
  while (facing a wall?) {
    turn left;
  }
  step forward;
}
```

Conclusion

Conclusions

- Machines are supposed to work for us.
- To tell them what we need done, we must learn their language.
- Machines are very precise: they do what you tell them verbatim! We must be very careful and accurate when talking to machines.
- Sometimes it feels like we work for them. But when we make the machine do something complicated for us — something that we couldn't have done without the machine — it is all worth it!