Telling a computer how to behave
(via pseudocode -- a workaround for Computing’s Tower of Babel.)

COS 116, Spring 2010
Adam Finkelstein
Pin All Your Romantic Hopes on Google

When you think about it, love is just another search problem. And we’ve thought about it. A lot. Google Romance™ is our solution.

Google Romance is a place where you can post all types of romantic information and, using our Soulmate Search™, get back search results that could, in theory, include the love of your life. Then we’ll send you both on a Contextual Date™, which we’ll pay for while delivering to you relevant ads that we and our advertising partners think will help produce the dating results you’re looking for.

With Google Romance, you can:

- **Upload your profile** – tell the world who you are, or, more to the point, who you’d like to think you are, or, even more to the point, who you want others to think you are.
- **Search for love** in all (or at least a statistically significant majority of) the right places with Soulmate Search, our eerily effective psychographic matchmaking software.
- **Endure**, via our Contextual Dating option, thematically appropriate multimedia advertising throughout the entirety of your free date.

Learn more: Take the Tour, Press Release, FAQ
Scribbler

Inputs
- Stall sensor
- Light sensors
- Obstacle sensor emitter
- Obstacle sensor detector
- Line sensor (underneath)

Outputs
- Speaker
- Motor/wheels
- Light outputs
Scribbler’s “Language”

- Several types of simple instructions
  - E.g. “Move forward for 1 s”
- Two types of compound instructions

**Conditional**

```
If condition Then
{
    List of instructions
}
Else
{
    List of instructions
}
```

**Loop**

```
Do 5 times
{
    List of instructions
}
```
Scribbler language illustrates essential features of all computer languages

- Fundamental features of human languages: nouns/verbs/adjectives, subjects/objects, pronouns, etc.
- Computer languages also share fundamental features: conditional/loop statements, variables, ability to perform arithmetic, etc.
For a computer, everything’s a number

Audio waveform

Sequence of Numbers representing frequency, amplitude, etc.

Image

Sequence of Numbers representing color value of each pixel.
A simple problem

- Our robot is getting ready for a big date…

- How would it identify the cheapest bottle? (Say it can scan prices)
Solution

- Pick up first bottle, check price

- Walk down aisle. For each bottle, do this:
  - If price on bottle is less than price in hand, exchange it with the one in hand.
Similar question in different setting

- Robot has \( n \) prices stored in memory
- Want to find minimum price
Memory: a simplified view

- A scratchpad that can be perfectly erased and re-written any number of times

- A variable: a piece of memory with a name; stores a “value”

\[ i = 22.99 \]
Examples

\( i \leftarrow 5 \) \hspace{1cm} \text{Sets } i \text{ to value } 5

\( j \leftarrow i \) \hspace{1cm} \text{Sets } j \text{ to whatever value is in } i. \hspace{0.5cm} \text{Leaves } i \text{ unchanged}

\( i \leftarrow j + 1 \) \hspace{1cm} \text{Sets } i \text{ to } j + 1. \hspace{0.5cm} \text{Leaves } j \text{ unchanged}

\( i \leftarrow i + 1 \) \hspace{1cm} \text{Sets } i \text{ to 1 more than it was.}
Arrays

- $A$ is an array of $n$ values
  - $A[i]$ is the $i^{th}$ value

Solution

- Pick up first bottle, check price

- Walk down aisle. For each bottle, do this:
  - If price on bottle is less than price in hand, exchange it with the one in hand.
**Procedure findmin**

- $n$ items, stored in array $A$
- Variables are $i$, $best$

```
best ← 1
Do for $i = 2$ to $n$
    {
            best ← $i$
    }
```
Another way to do the same

\[\begin{align*}
    &best \leftarrow 1; \\
    &i \leftarrow 1 \\
    &\text{Do while } (i < n) \\
    &\quad \{ \\
    &\quad \quad i \leftarrow i + 1; \\
    &\quad \quad \text{if } (A[i] < A[best]) \text{ then} \\
    &\quad \quad \quad best \leftarrow i \\
    &\quad \}
\end{align*}\]
```c
#include <stdio.h>
int main(void)
{
    int count;
    for (count = 1; count <= 500; count++)
        printf("I will not throw paper airplanes in class.\n");
    return 0;
}
```
New problem for robot: sorting

Arrange them so prices **increase** from left to right.
Solution

Do for $i=1$ to $n-1$
{
    Find cheapest bottle among those numbered $i$ to $n$
    Swap that bottle and the $i$’th bottle.
}
Swapping

- Suppose $x$ and $y$ are variables. How do you swap their values?

- Need extra variable!

```plaintext
tmp ← x
x ← y
y ← tmp
```
Algorithm

- A precise unambiguous procedure for accomplishing a task

- Named for Abu Abdullah Muhammad bin Musa al-Khwarizmi

- Examples:
  - recipe, long division, selection sort.
Love, Marriage, and Lying

Standard disclaimer.
Stable Matching Problem

Problem:
Given $N$ men & $N$ women, find “suitable” matching
- Everyone lists their preferences from best to worst.

<table>
<thead>
<tr>
<th>Man</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Victor</td>
<td>Bertha</td>
<td>Amy</td>
<td>Diane</td>
<td>Erika</td>
<td>Clare</td>
</tr>
<tr>
<td>Wyatt</td>
<td>Diane</td>
<td>Bertha</td>
<td>Amy</td>
<td>Clare</td>
<td>Erika</td>
</tr>
<tr>
<td>Xavier</td>
<td>Bertha</td>
<td>Erika</td>
<td>Clare</td>
<td>Diane</td>
<td>Amy</td>
</tr>
<tr>
<td>Yancey</td>
<td>Amy</td>
<td>Diane</td>
<td>Clare</td>
<td>Bertha</td>
<td>Erika</td>
</tr>
<tr>
<td>Zeus</td>
<td>Bertha</td>
<td>Diane</td>
<td>Amy</td>
<td>Erika</td>
<td>Clare</td>
</tr>
</tbody>
</table>
Stable Matching Problem

Problem:
Given $N$ men & $N$ women, find “suitable” matching
- Everyone lists their preferences from best to worst.

<table>
<thead>
<tr>
<th>Woman</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
<td>Zeus</td>
<td>Victor</td>
<td>Wyatt</td>
<td>Yancey</td>
<td>Xavier</td>
</tr>
<tr>
<td>Bertha</td>
<td>Xavier</td>
<td>Wyatt</td>
<td>Yancey</td>
<td>Victor</td>
<td>Zeus</td>
</tr>
<tr>
<td>Clare</td>
<td>Wyatt</td>
<td>Xavier</td>
<td>Yancey</td>
<td>Zeus</td>
<td>Victor</td>
</tr>
<tr>
<td>Diane</td>
<td>Victor</td>
<td>Zeus</td>
<td>Yancey</td>
<td>Xavier</td>
<td>Wyatt</td>
</tr>
<tr>
<td>Erika</td>
<td>Yancey</td>
<td>Wyatt</td>
<td>Zeus</td>
<td>Xavier</td>
<td>Victor</td>
</tr>
</tbody>
</table>
Stable Matching Problem

- What do we mean by “suitable”?
  - PERFECT: everyone matched monogamously.
  - STABILITY: no incentive for some pair to elope.
    - a pair that is not matched with each other is UNSTABLE if they prefer each other to current partners
    - unstable pair: improve by dumping spouses and eloping

- STABLE MATCHING (Gale and Shapley, 1962)
  = perfect matching with no unstable pairs.
Lavender assignment is a perfect matching. Are there any unstable pairs?

Yes. Bertha and Xavier form an unstable pair. They would prefer each other to current partners.
Green assignment is a stable matching.
Example

Gray assignment is also a stable matching.

<table>
<thead>
<tr>
<th>Men’s Preference List</th>
<th>Women’s Preference List</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Man</strong></td>
<td><strong>Woman</strong></td>
</tr>
<tr>
<td><strong>1st</strong></td>
<td><strong>1st</strong></td>
</tr>
<tr>
<td>Xavier</td>
<td>Amy</td>
</tr>
<tr>
<td>A</td>
<td>Y</td>
</tr>
<tr>
<td>B</td>
<td>X</td>
</tr>
<tr>
<td>C</td>
<td>Z</td>
</tr>
<tr>
<td><strong>2nd</strong></td>
<td><strong>2nd</strong></td>
</tr>
<tr>
<td>Yancey</td>
<td>Bertha</td>
</tr>
<tr>
<td>B</td>
<td>X</td>
</tr>
<tr>
<td>A</td>
<td>Y</td>
</tr>
<tr>
<td>C</td>
<td>Z</td>
</tr>
<tr>
<td><strong>3rd</strong></td>
<td><strong>3rd</strong></td>
</tr>
<tr>
<td>Zeus</td>
<td>Clare</td>
</tr>
<tr>
<td>A</td>
<td>X</td>
</tr>
<tr>
<td>B</td>
<td>Y</td>
</tr>
<tr>
<td>C</td>
<td>Z</td>
</tr>
</tbody>
</table>
Propose-And-Reject Algorithm

- Guarantees a stable matching.

**Gale-Shapley Algorithm (men propose)**

Initialize each person to be free.

while (some man $m$ is free and hasn't proposed to every woman)
{
    $w =$ first woman on $m$'s list to whom he has not yet proposed
    if ($w$ is free)
        assign $m$ and $w$ to be engaged
    else if ($w$ prefers $m$ to her fiancé $f$)
        assign $m$ and $w$ to be engaged, and $f$ to be free
    else
        $w$ rejects $m$
}
Extensions

- Unacceptable partners
  - Every woman is not willing to marry every man, and vice versa.
  - Some participants declare others as “unacceptable.”

- Sets of unequal size
  - Unequal numbers of men and women, e.g. 100 men & 90 women

- Limited Polygamy
  - e.g., Bill wants to be matched with 3 women.
Matching Residents to Hospitals

- Hospitals ~ Men (limited polygamy allowed).
- Residents ~ Women (more than hospitals)
- Started just after WWII (before computer usage).
- Ides of March, 13,000+ residents are matched.
- Rural hospital dilemma.
  - Certain hospitals (mainly in rural areas) were unpopular and declared unacceptable by many residents.
  - How to find stable matching that benefits rural hospitals?
Homework for Thursday
(email your answers to pu.cos116@gmail.com by 2/11 noon)

- Write out pseudocode for selection sort.
- Try Gale-Shapley algorithm for previouslyShown Amy-Erica / Victor-Zeuss preference lists, but vary the order of choosing man $m$. Does this affect the outcome?
- Try the version where women propose. Does this affect the outcome?
- Bonus question: Try to justify this statement: \textit{When the Gale-Shapley algorithm finishes, there are no unstable pairs.}
Lessons Learned

- Powerful ideas learned in computer science.
- Sometimes deep social ramifications.

- Hospitals and residents…
- Historically, men propose to women. Why not vice versa?
- Computer scientists get the best partners!!!