CSCI 3110
Fun with Algorithms

Chris Whidden
whidden@cs.dal.ca

Faculty of Computer Science
Dalhousie University

Summer 2011

Modified from slides of Norbert Zeh
Given:

- $n$ women $w_1, w_2, \ldots, w_n$
- $n$ men $m_1, m_2, \ldots, m_n$
- $n$ marriages $(w_{i_1}, m_{j_1}), (w_{i_2}, m_{j_2}), \ldots, (w_{i_n}, m_{j_n})$
The marriages are \textit{stable} if there is no pair \((m, w)\) such that
\begin{itemize}
  \item \(m\) likes \(w\) better than his current partner and
  \item \(w\) likes \(m\) better than her current partner.
\end{itemize}
**Goal:** Find a set of marriages such that

- Every woman is married,
- Every man is married, and
- All marriages are stable.

```
<table>
<thead>
<tr>
<th>m2, m3, m1, m5, m4</th>
<th>w1</th>
<th>m1</th>
</tr>
</thead>
<tbody>
<tr>
<td>m5, m1, m2, m3, m4</td>
<td>w2</td>
<td>m2</td>
</tr>
<tr>
<td>m4, m2, m1, m3, m5</td>
<td>w3</td>
<td>m3</td>
</tr>
<tr>
<td>m5, m4, m3, m2, m1</td>
<td>w4</td>
<td>m4</td>
</tr>
<tr>
<td>m1, m3, m2, m4, m5</td>
<td>w5</td>
<td>m5</td>
</tr>
</tbody>
</table>
```

```w1, w4, w3, w5, w2```  ```w1, w4, w3, w5, w2```  ```w5, w2, w3, w4, w1```  ```w3, w4, w5, w2, w1```  ```w4, w3, w5, w1```  ```w4, w3, w5, w1, w2```
Stable Marriages: A Solution Inspired by Real Life

**PROPOSAL-ALGORITHM**($M$, $W$)

1. while there is an unmarried man $m$
2. do $m$ chooses his favourite woman $w$ he has not proposed to yet
3. $m$ proposes to $w$
4. if $w$ is not married or likes $m$ better than her current partner $m'$
5. then $w$ divorces $m'$
6. $w$ marries $m$
PROPOSAL-ALGORITHM($M$, $W$)

1. while there is an unmarried man $m$
2. do $m$ chooses his favourite woman $w$ he has not proposed to yet
3. $m$ proposes to $w$
4. if $w$ is not married or likes $m$ better than her current partner $m'$
5. then $w$ divorces $m'$
6. $w$ marries $m$

Is there always a set of $n$ stable marriages?
PROPOSAL-ALGORITHM($M$, $W$)

1. **while** there is an unmarried man $m$
2. **do** $m$ chooses his favourite woman $w$ he has not proposed to yet
3. $m$ proposes to $w$
4. **if** $w$ is not married or likes $m$ better than her current partner $m'$
5. **then** $w$ divorces $m'$
6. $w$ marries $m$

- Is there always a set of $n$ stable marriages?
- Does the algorithm ever terminate?
Stable Marriages: A Solution Inspired by Real Life

**PROPOSAL-ALGORITHM**\((M, W)\)

1. **while** there is an unmarried man \(m\)
2. **do** \(m\) chooses his favourite woman \(w\) he has not proposed to yet
3. \(m\) proposes to \(w\)
4. **if** \(w\) is not married or likes \(m\) better than her current partner \(m'\)
5. **then** \(w\) divorces \(m'\)
6. \(w\) marries \(m\)

- Is there always a set of \(n\) stable marriages?
- Does the algorithm ever terminate?
- Does the algorithm always produce a correct answer?
Stable Marriages: A Solution Inspired by Real Life

**Proposal-Algorithm** \((M, W)\)

1. **while** there is an unmarried man \(m\)
2. **do** \(m\) chooses his favourite woman \(w\) he has not proposed to yet
3. \(m\) proposes to \(w\)
4. **if** \(w\) is not married or likes \(m\) better than her current partner \(m'\)
5. **then** \(w\) divorces \(m'\)
6. \(w\) marries \(m\)

- Is there always a set of \(n\) stable marriages?
- Does the algorithm ever terminate?
- Does the algorithm always produce a correct answer?
- How efficient is the algorithm? Can we give an upper bound on its running time?
Course Outline

- Proof of correctness
- Analysis of resource consumption
- Design techniques
  - Graph exploration
  - Greedy algorithms
  - Divide-and-conquer
  - Dynamic programming
  - Data structuring
  - Randomization
- NP-completeness and intractability
General Information

Instructor: Chris Whidden
Office: 320 (Thursdays only)
Mona Campbell Building 4233

Office hours: R 1:30–3:30
Email: whidden@cs.dal.ca


Website: [http://www.cs.dal.ca/~whidden/CS3110](http://www.cs.dal.ca/~whidden/CS3110)

TA: TBA

Midterm: June 23, 2011
Grading

- 10 assignments ($A$)
  
  The best 8 assignments count; each has equal weight.
  
- Midterm ($M$)
  
- Final ($F$)

\[
\text{Final grade} = 40\% \cdot A + 20\% \cdot M + 40\% \cdot F
\]
Collaboration, Plagiarism, Late Assignments

*Collaboration*
- You are expected to complete and submit only your own work.
- You are allowed to discuss assignment problems and solutions with other students.
- All assignment problems can be completed based on the material covered in lectures and in the text.
- *Do not look to outside sources (such as the internet) and document any help you receive.*

*Plagiarism*
- *Plagiarism will not be tolerated.*

*Late assignments*
- Will not be accepted without a doctor’s note.

*Please see website for a detailed discussion of these rules.*
Things You Should Know

- Propositional logic
- Elementary combinatorics (counting permutations, combinations, ...)
- Elementary probability theory
- Elementary data structures (arrays, lists, stacks, queues, ...)
- Standard sorting algorithms (Insertion sort, Quicksort, Merge sort)
- Hash tables
- Binary-search trees (Red-black trees or (a,b) trees?)