Stable Marriages: An Introductory Example

**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 *stable* if there is no pair \((m, w)\) such that

- \(m\) likes \(w\) better than his current partner and
- \(w\) likes \(m\) better than her current partner.
Goal: Find a set of marriages such that
- Every woman is married,
- Every man is married, and
- All marriages are stable.
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 \(m\)
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?
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?
Stable Marriages: A Solution Inspired by Real Life

**PROPPOSAL-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: Norbert Zeh

Office: 314

Office hours: F 0900–1015

Email: nzeh@cs.dal.ca


Website: http://www.cs.dal.ca/~nzeh/Teaching/3110

TA: Chris Whidden whidden@cs.dal.ca

Midterm: June 26, 2008
Grading

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

Final grade = $\max(F, 40\% \cdot A + 20\% \cdot M + 40\% \cdot F)$
Collaboration, Plagiarism, Late Assignments

Collaboration
- Groups of up to three people are allowed to collaborate on assignments.
- Every group hands in one set of solutions; every group member gets the same marks.
- Collaboration between groups is not allowed.

Plagiarism
- Plagiarism will not be tolerated.
- Collaboration between groups is a form of plagiarism.

Late assignments
- Will not be accepted without 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
- Red-black trees