Lecture 4

CSE 331

Sep 7, 2016

Read the syllabus CAREFULLY!

CSE 331 Introduction to Algorithm Analysis and Design Fall 2016

University at Buffalo

Department of Computer Science & Engineering

CSE 331 — Introduction to Algorithm Analysis and Design

I'll need confirmation in writing. No graded material will be handed back till I get this signed form from you!

I, ______________________________ (PRINT name), acknowledge that I have read and understood the syllabus (and the homework policy document) for this course, CSE 331 *Introduction to Algorithm Analysis and Design*.

I also acknowledge that I understand the definition of academic integrity as outlined in the syllabus, and that I will minimally receive a grade of F in the course if I am found to have breached academic integrity, *even if it occurs for the first time*. In particular, I understand that I cannot claim that I did not understand the rules if I am found to have breached academic integrity.

Signature: ______________________________ Date: ______________

Sign-up for mini projects

Deadline: Monday, Sep 26, 11:59pm

Email me your group (=3) composition

Separate Proof idea/proof details

</> Note

Notice how the solution below is divided into proof idea and proof details part. **THIS IS IMPORTANT: IF YOU DO NOT PRESENT A PROOF IDEA, YOU WILL NOT GET ANY CREDIT EVEN IF YOUR PROOF DETAILS ARE CORRECT.**

Proof Idea

As the hint suggests there are two ways of solving this problem. (I'm presenting both the solutions but of course you only need to present one.)

We begin with the approach of reducing the given problem to a problem you have seen earlier. ⇒ Build the following complete binary tree: every internal node in the tree represents a "parent" RapidGrower while its two children are the two RapidGrowers it divides itself into. After s seconds this tree will have height s and the number of RapidGrowers in the container after s seconds is the number of leaf nodes these complete binary tree has, which we know is 2^s. Hence, the claim is correct.

The proof by induction might be somewhat simpler for this problem if you are not comfortable with reduction. In this case let $R(s)$ be the number of RapidGrowers after s seconds. Then we use induction to prove that $R(s) = 2^s$ while using the fact that $2 \cdot 2^s = 2^{s+1}$.

Proof Details

We first present the reduction based proof. Consider the complete binary tree with height s and call it $T(s)$. Further, note that one can construct $T(s+1)$ from $T(s)$ by attaching two children nodes to all the leaves in $T(s)$. Notice that the newly added children are the leaves of $T(s+1)$. Now assign the root of $T(0)$ as the original RapidGrower in the container. Further, for any internal node in $T(s)$ ($s \geq 0$), assign its two children to the two RapidGrowers it divides itself into. Then note that there is a one to one correspondence between the RapidGrowers after s seconds and the leaves of $T(s)$. ⇒ Then we use the well-known fact (cite your 191/250 book here with the exact place where one can find this fact): $T(s)$ has 2^s leaves, which means that the number of RapidGrowers in the container after s seconds is 2^s, which means that the claim is correct.

On matchings

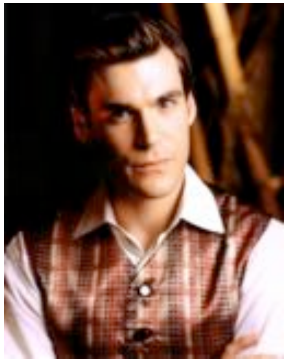
Mal

Inara

Wash

Zoe

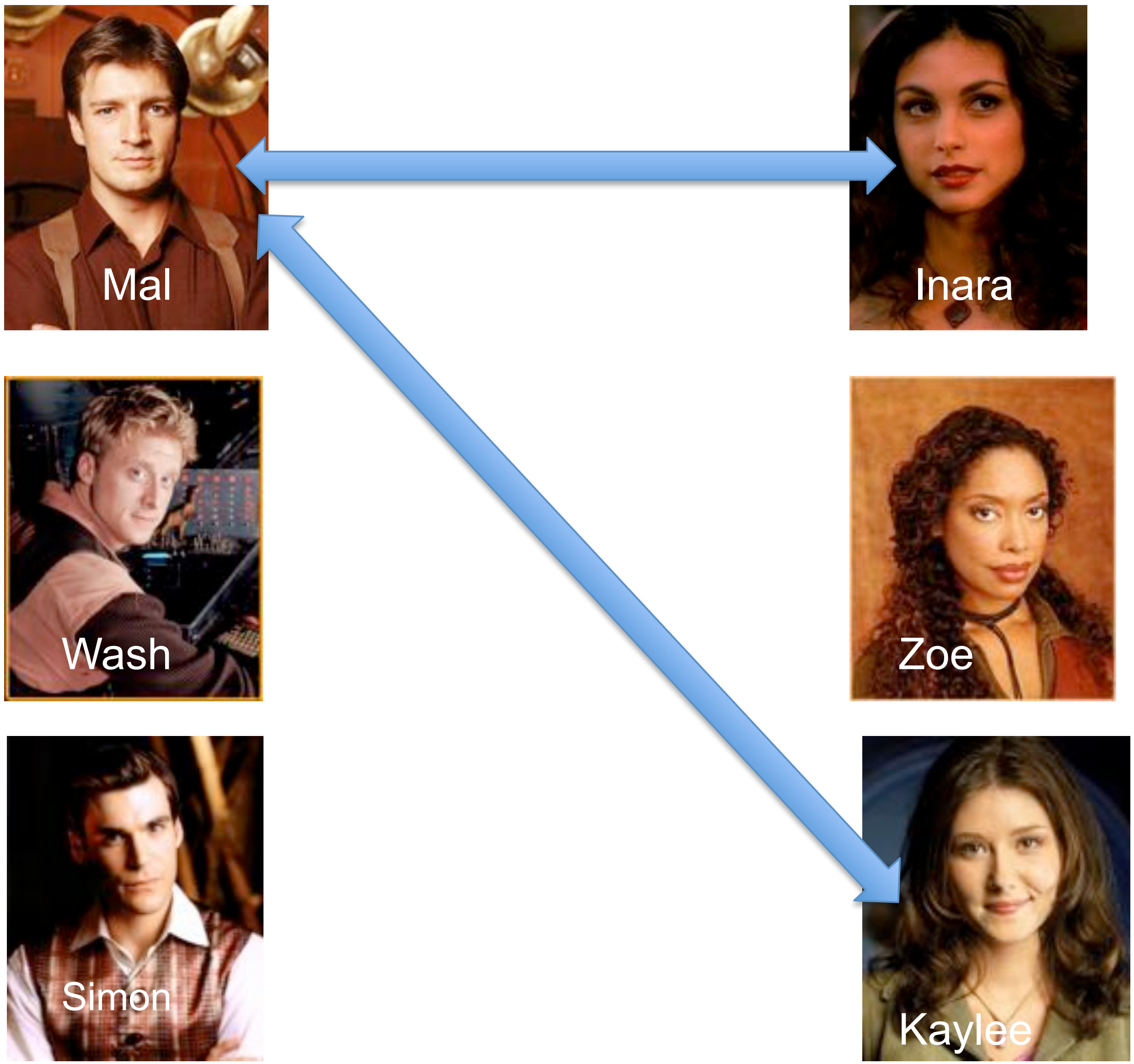
Simon



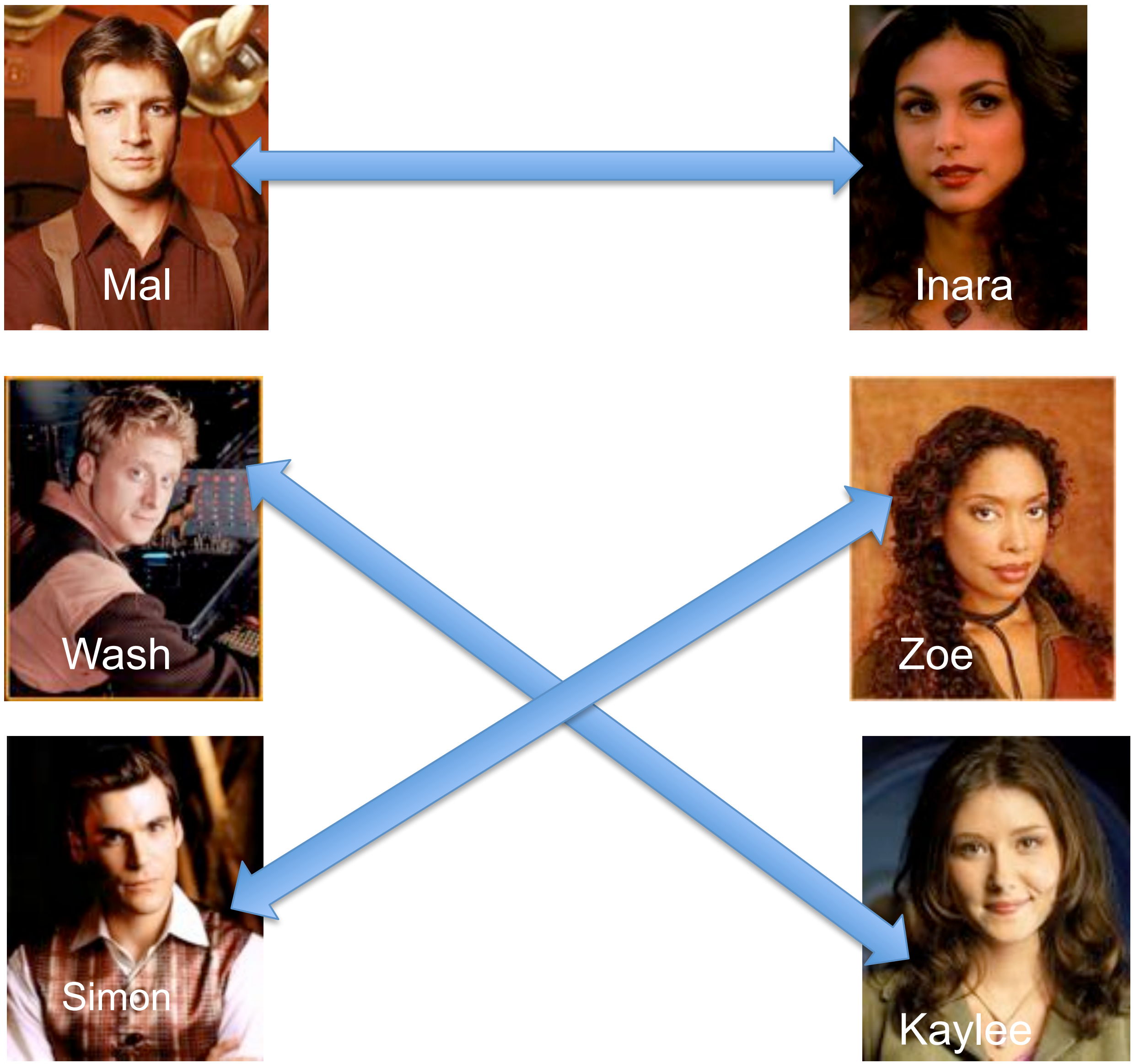
Kaylee

A valid matching

Not a matching

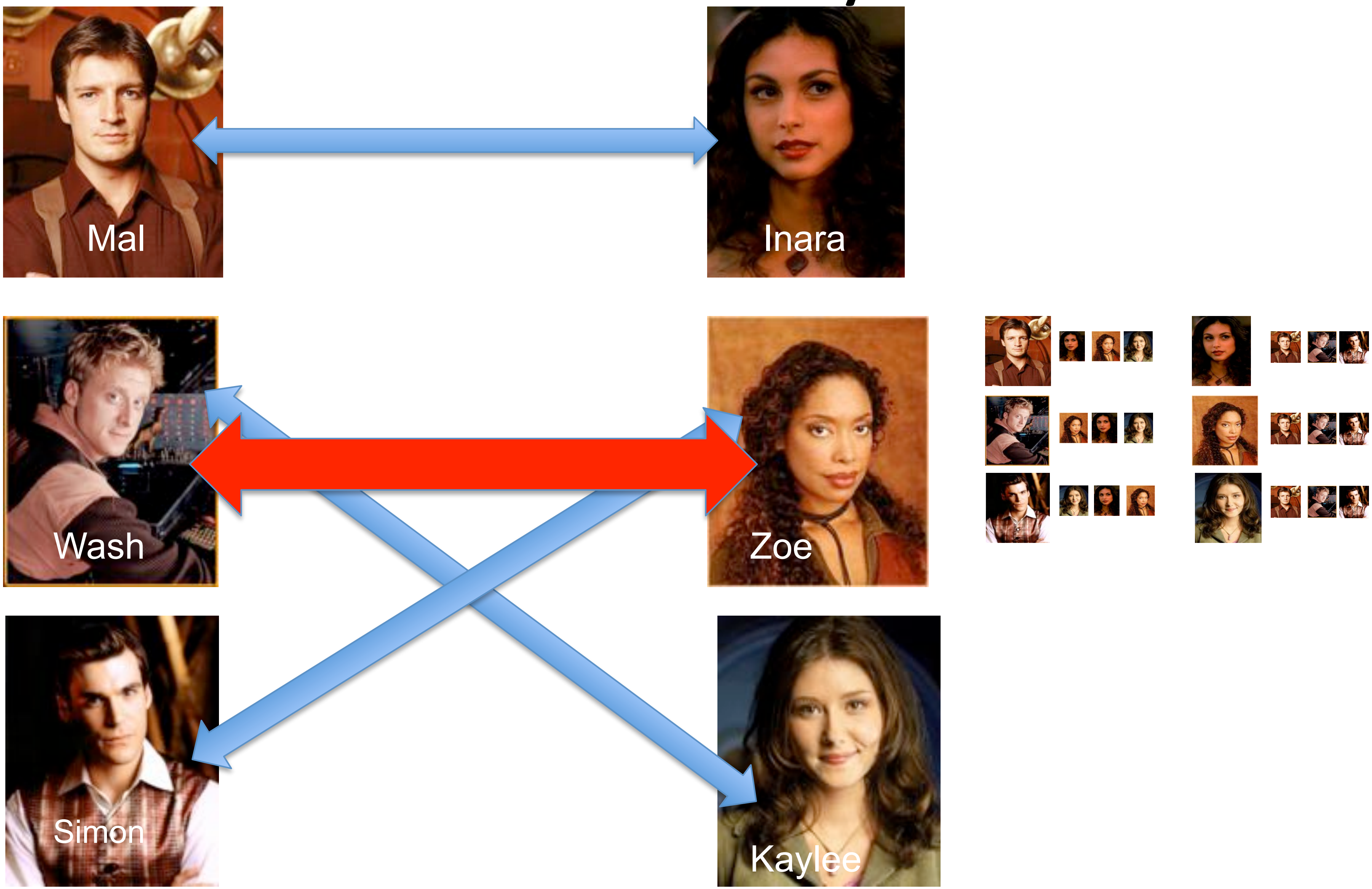


Perfect Matching



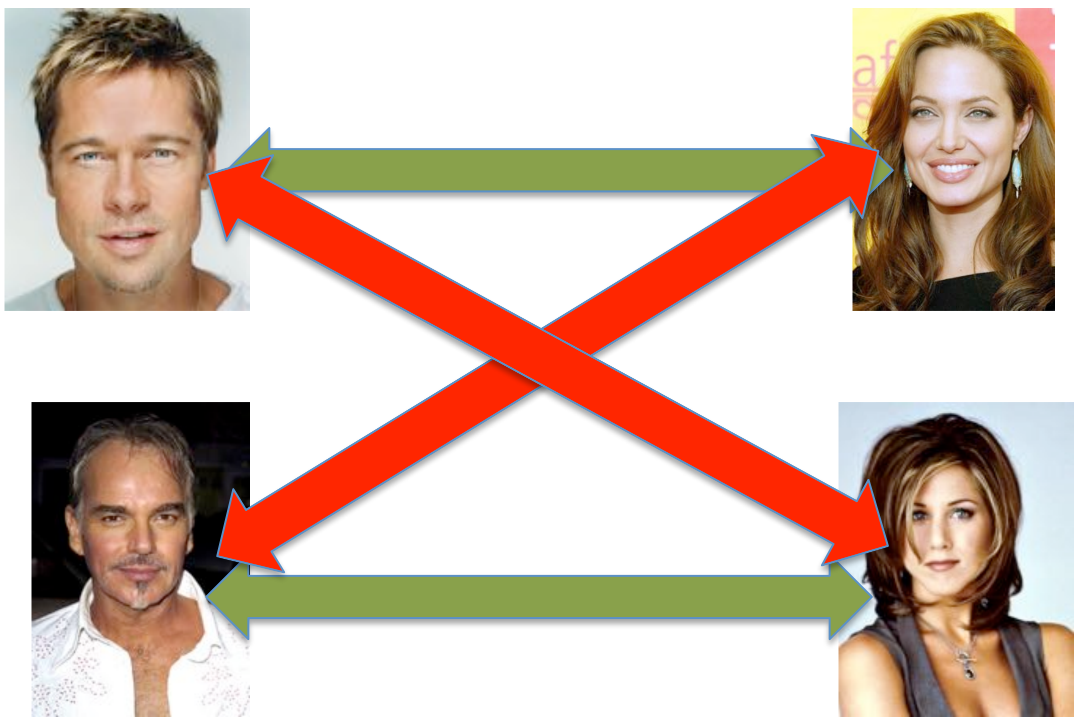
Preferences

Instability

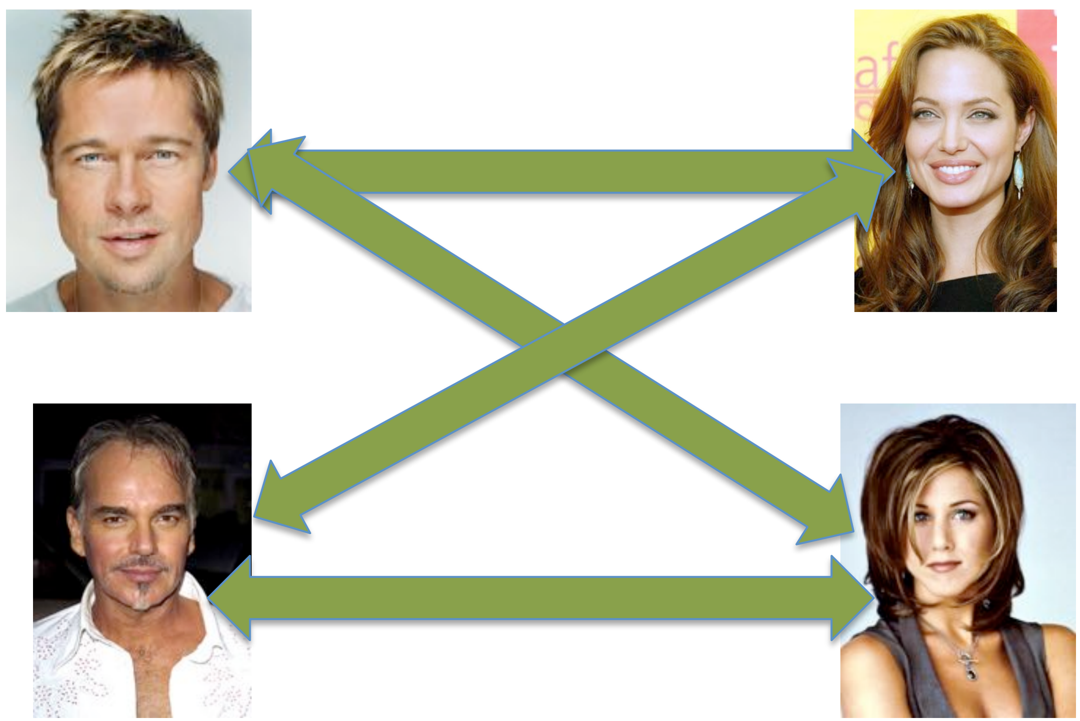


A stable marriage

Even though BBT and JA are not very happy



Two stable marriages



Stable Marriage problem

Set of men M and women W

Preferences (ranking of potential spouses)

Matching (no polyandry/gamy in M X W)

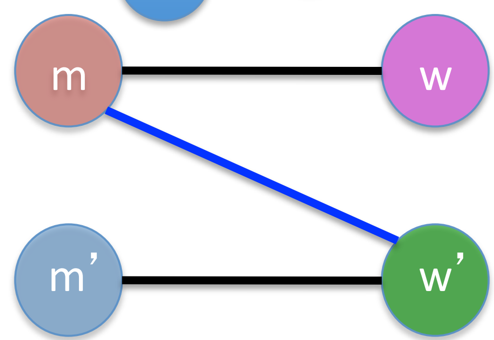
Perfect Matching (everyone gets married)

Instablity

Stable matching = perfect matching+ no instablity

Input: M and W with preferences

Output: Stable Matching



Questions/Comments?

Two Questions

Does a stable marriage always exist?

If one exists, how quickly can we compute one?

Today’s lecture

Naïve algorithm

Gale-Shapley algorithm for Stable Marriage problem

Discuss: Naïve algorithm!

The naïve algorithm

Incremental algorithm to produce all n! prefect matchings?

Go through all possible perfect matchings S

If S is a stable matching

then Stop

Else move to the next perfect matching

Gale-Shapley Algorithm

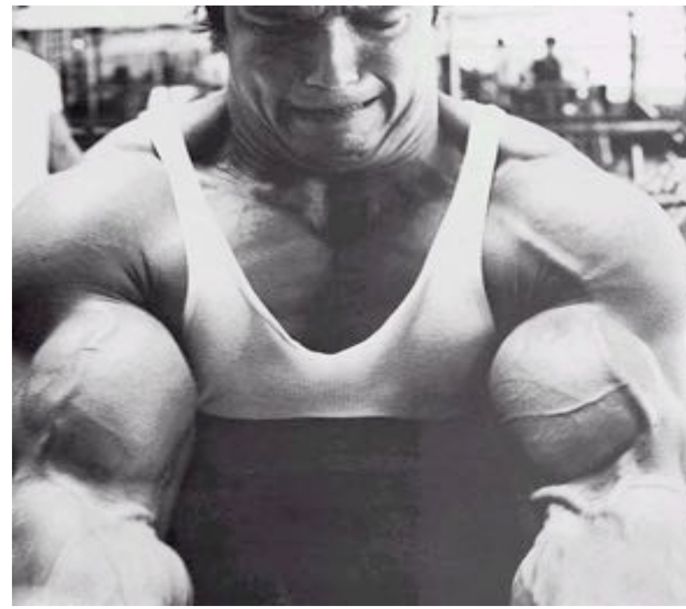
David Gale

Lloyd Shapley

O(n^3) algorithm

Moral of the story…


>



Questions/Comments?

Gale-Shapley Algorithm

Intially all men and women are free

While there exists a free woman who can propose

Let w be such a woman and m be the best man she has not proposed to

w proposes to m

If m is free

(m,w) get engaged

Else (m,w') are engaged

If m prefers w' to w

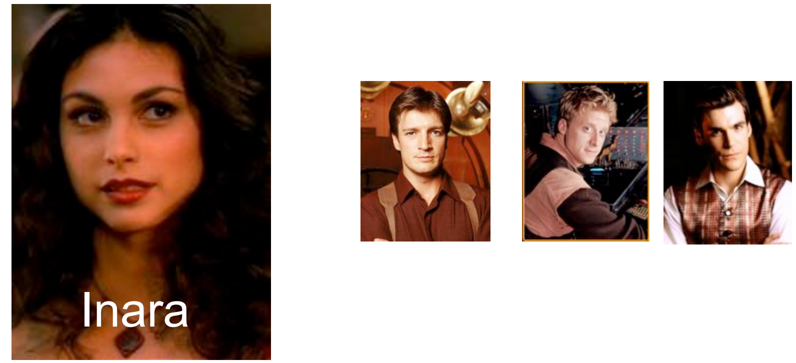
w remains free

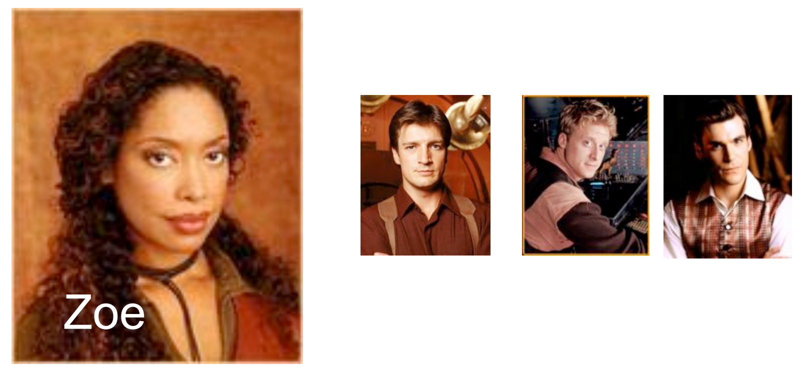
Else

(m,w) get engaged and w' is free

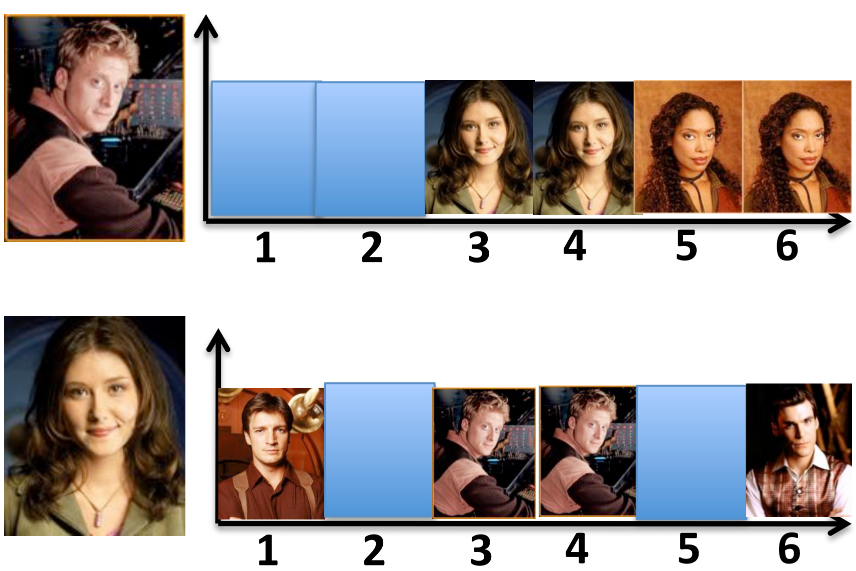
Output the engaged pairs as the final output

Preferences





GS algorithm: Firefly Edition



Observation 1

Intially all men and women are free

While there exists a free woman who can propose

Let w be such a woman and m be the best man she has not proposed to

w proposes to m

If m is free

(m,w) get engaged

Else (m,w') are engaged

If m prefers w' to w

w remains free

Else

(m,w) get engaged and w' is free

Once a man gets engaged, he remains engaged (to "better" women)

Output the engaged pairs as the final output

Observation 2

Intially all men and women are free

While there exists a free woman who can propose

Let w be such a woman and m be the best man she has not proposed to

w proposes to m

If m is free

(m,w) get engaged

Else (m,w') are engaged

If m prefers w' to w

w remains free

Else

(m,w) get engaged and w' is free

If w proposes to m after m', then she prefers m' to m

Output the engaged pairs as the final output

Questions/Comments?

Why bother proving correctness?

Consider a variant where any free man **or** free woman can propose

Is this variant any different? Can you prove it?

GS' does not output a stable marriage