Stable matchings

Alexander Shen

LIRMM / CNRS, University of Montpellier. France

Outline

University Entrance in Russia

Old system:

University Entrance in Russia

Old system:

- each university had entrance exams

University Entrance in Russia

Old system:

- each university had entrance exams
- all at the same time

University Entrance in Russia

Old system:

- each university had entrance exams
- all at the same time
- only one application

University Entrance in Russia

Old system:

- each university had entrance exams
- all at the same time
- only one application
- failure: no university this year

University Entrance in Russia

Old system:

- each university had entrance exams
- all at the same time
- only one application
- failure: no university this year

New system:

University Entrance in Russia

Old system:

- each university had entrance exams
- all at the same time
- only one application
- failure: no university this year

New system:

- country-wide tests

University Entrance in Russia

Old system:

- each university had entrance exams
- all at the same time
- only one application
- failure: no university this year

New system:

- country-wide tests
- many applications

University Entrance in Russia

Old system:

- each university had entrance exams
- all at the same time
- only one application
- failure: no university this year

New system:

- country-wide tests
- many applications
- universities make offers

University Entrance in Russia

Old system:

- each university had entrance exams
- all at the same time
- only one application
- failure: no university this year

New system:

- country-wide tests
- many applications
- universities make offers
- applicants choose

Simplified Model

Simplified Model

- *n* candidates

Simplified Model

- *n* candidates
- *n* positions

Simplified Model

- *n* candidates
- *n* positions
- full information: lists of preferences

Simplified Model

- *n* candidates
- *n* positions
- full information: lists of preferences
- matching: *n* pairs (candidate, position)

Simplified Model

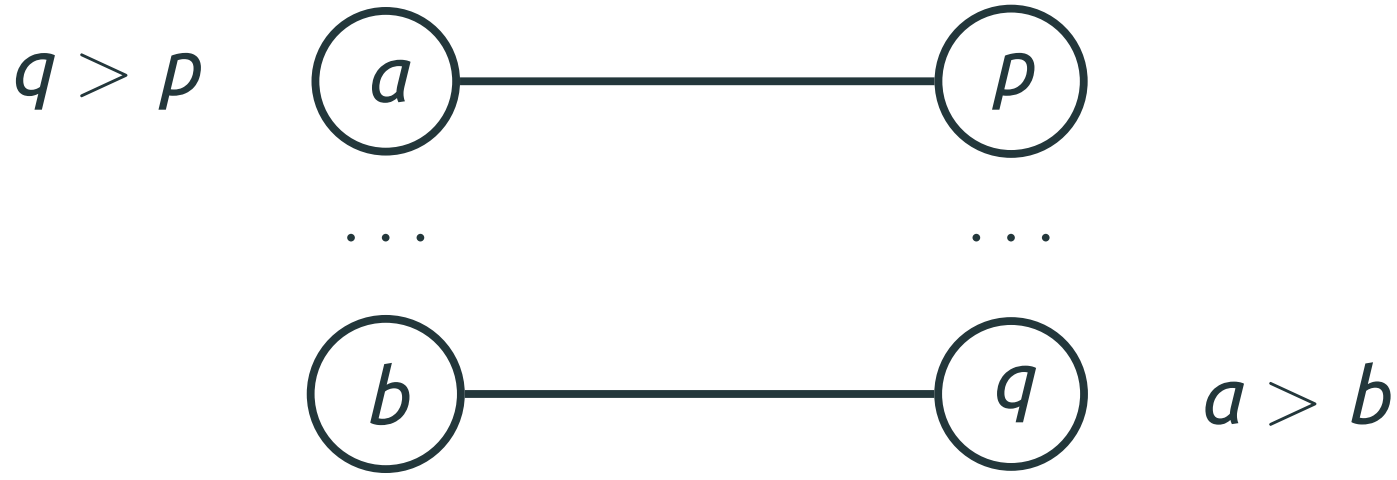
- *n* candidates
- *n* positions
- full information: lists of preferences
- matching: *n* pairs (candidate, position)
- graph theory: “perfect matching”

Simplified Model

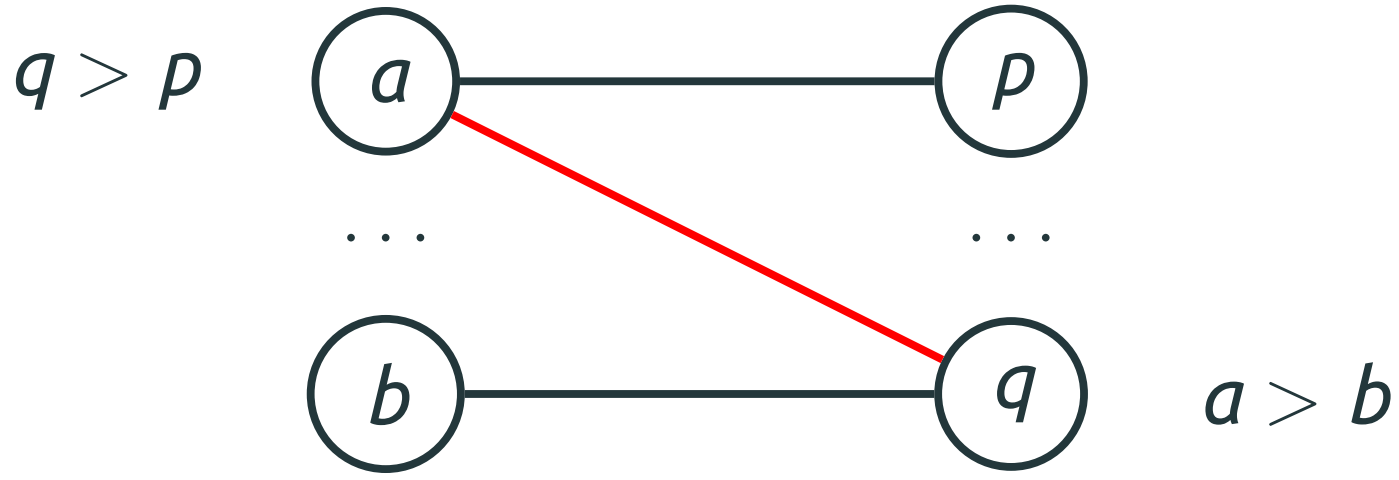
- *n* candidates
- *n* positions
- full information: lists of preferences
- matching: *n* pairs (candidate, position)
- graph theory: “perfect matching”
- minimal requirement: stability

Unstable Matching

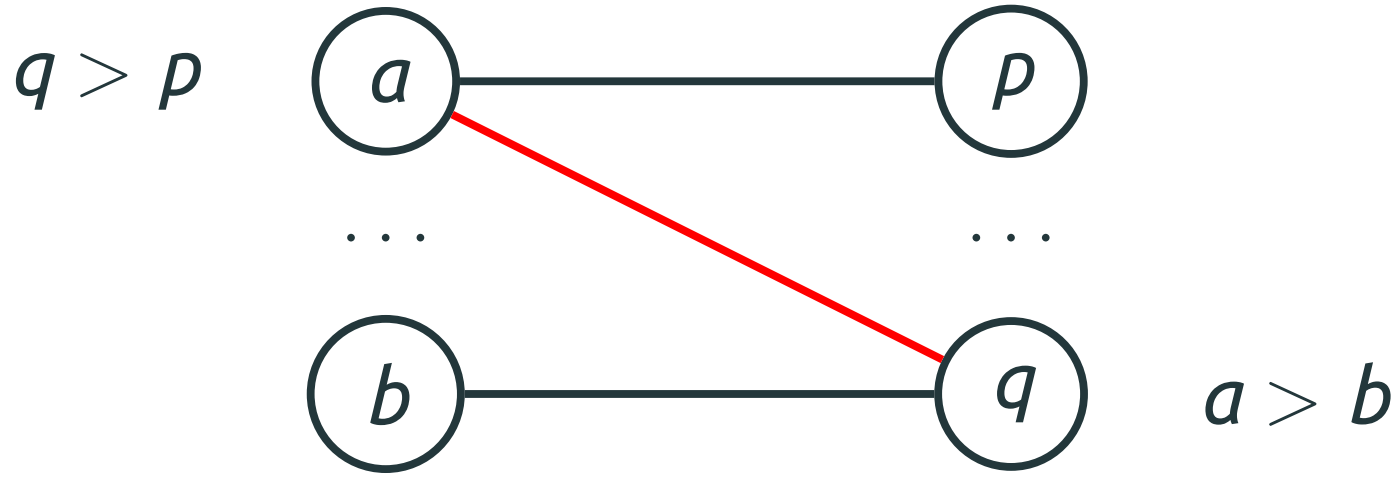
Unstable Matching



Unstable Matching



Unstable Matching



Is there a stable matching?

Outline

How to Get a Nobel Prize…

How to Get a Nobel Prize…

- Nobel memorial prize in economics (2012): Alvin Roth, Lloyd Shapley

How to Get a Nobel Prize…

- Nobel memorial prize in economics (2012): Alvin Roth, Lloyd Shapley
- “Roth recognized that Shapley’s theoretical results could clarify the functioning of important markets in practice… helped redesign existing institutions for matching new doctors with hospitals, students with schools, and organ donors with patients.”

…by Writing to a Wide Audience

…by Writing to a Wide Audience

- David Gale (1921–2008), Lloyd Shapley: College Admissions and the Stability of Marriage, 1962

…by Writing to a Wide Audience

- David Gale (1921–2008), Lloyd Shapley: College Admissions and the Stability of Marriage, 1962
- American Mathematical Monthly
  `https://www.jstor.org/stable/2312726`

Warning: The Terminology

Warning: The Terminology

- men and women instead of applicants and jobs

Warning: The Terminology

- men and women instead of applicants and jobs
- matchings = marriages

Warning: The Terminology

- men and women instead of applicants and jobs
- matchings = marriages
- no claims about real life!

Warning: The Terminology

- men and women instead of applicants and jobs
- matchings = marriages
- no claims about real life!
- “her husband” is shorter than “applicant that fills this position”

Warning: The Terminology

- men and women instead of applicants and jobs
- matchings = marriages
- no claims about real life!
- “her husband” is shorter than “applicant that fills this position”
- more symmetric

Outline

Unique Stable Marriage

Unique Stable Marriage

- *n* men and *n* women

Unique Stable Marriage

- *n* men and *n* women
- each participant: ordered list

Unique Stable Marriage

- *n* men and *n* women
- each participant: ordered list
- perfect matchings (marriages)

Unique Stable Marriage

- *n* men and *n* women
- each participant: ordered list
- perfect matchings (marriages)
- stability: no pair prefers each other to current partner

Unique Stable Marriage

- n men and n women
- each participant: ordered list
- perfect matchings (marriages)
- stability: no pair prefers each other to current partner
- $n = 1$: no problems

Unique Stable Marriage

- n men and n women
- each participant: ordered list
- perfect matchings (marriages)
- stability: no pair prefers each other to current partner
- $n = 1$: no problems
- another case: all men have the same preferences

Unique Stable Marriage

- n men and n women
- each participant: ordered list
- perfect matchings (marriages)
- stability: no pair prefers each other to current partner
- $n = 1$: no problems
- another case: all men have the same preferences
- unique stable marriage: why?

The Same Ordering

The Same Ordering

- all men: $w_1 > w_2 > \ldots > w_n$

The Same Ordering

- all men: $w_1 > w_2 > \ldots > w_n$
- w_1 prefers some m_i

The Same Ordering

- all men: $w_1 > w_2 > \ldots > w_n$
- w_1 prefers some m_i
- m_i prefers w_1

The Same Ordering

- all men: $w_1 > w_2 > \ldots > w_n$
- w_1 prefers some m_i
- m_i prefers w_1
- m_i–w_1 in any stable marriage

The Same Ordering

- all men: $w_1 > w_2 > \ldots > w_n$
- w_1 prefers some m_i
- m_i prefers w_1
- m_i–w_1 in any stable marriage
- m_i, w_1 cannot create instability for the rest

The Same Ordering

- all men: $w_1 > w_2 > \ldots > w_n$
- w_1 prefers some m_i
- m_i prefers w_1
- m_i–w_1 in any stable marriage
- m_i, w_1 cannot create instability for the rest
- induction

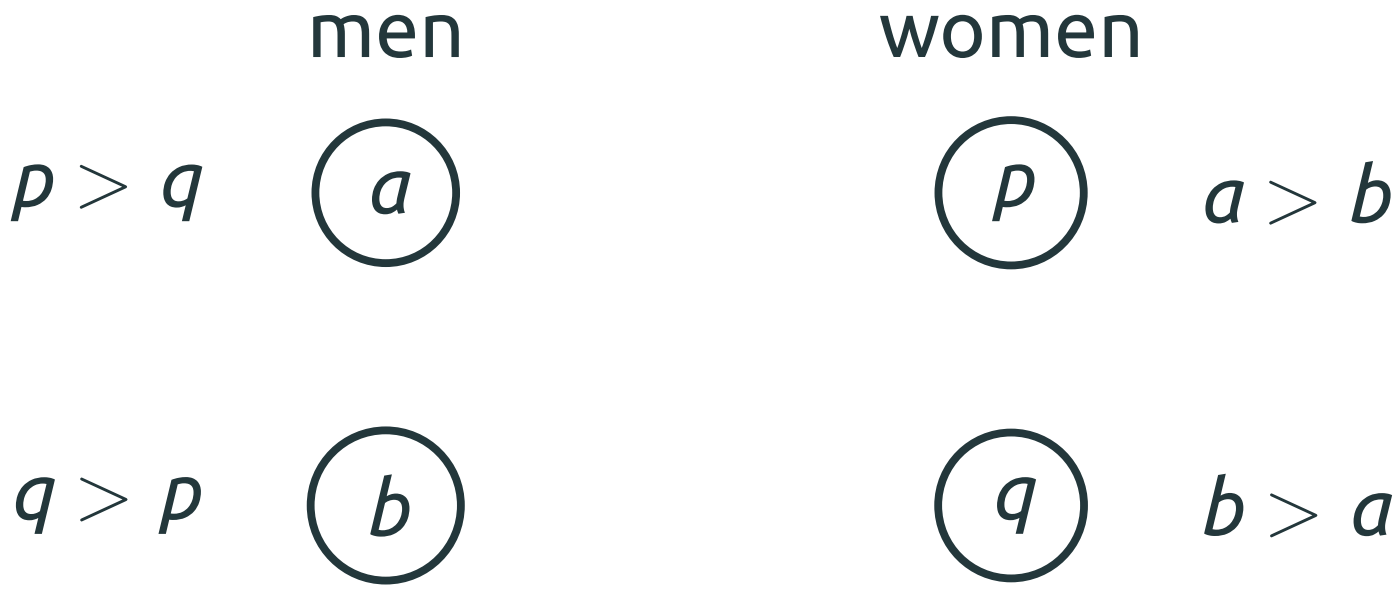
The Same Ordering

- all men: $w_1 > w_2 > \ldots > w_n$
- w_1 prefers some m_i
- m_i prefers w_1
- m_i–w_1 in any stable marriage
- m_i, w_1 cannot create instability for the rest
- induction
- recursive algorithm

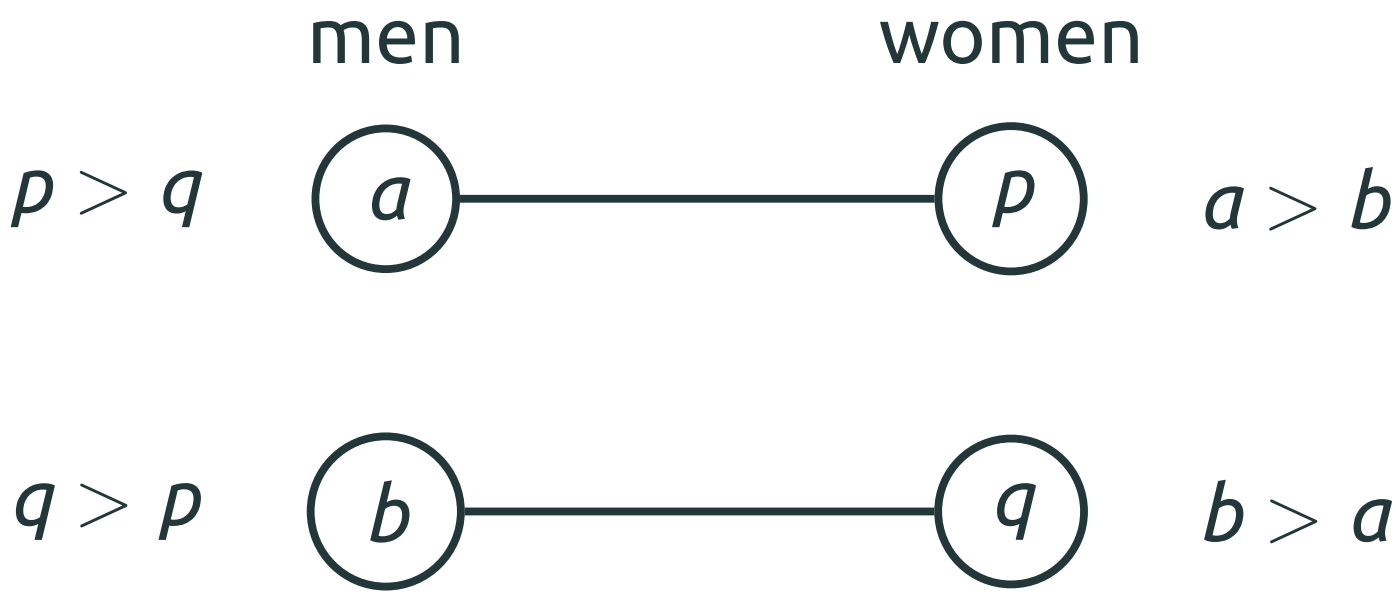
The Same Ordering

- all men: $w_1 > w_2 > \ldots > w_n$
- w_1 prefers some m_i
- m_i prefers w_1
- m_i–w_1 in any stable marriage
- m_i, w_1 cannot create instability for the rest
- induction
- recursive algorithm
- symmetry

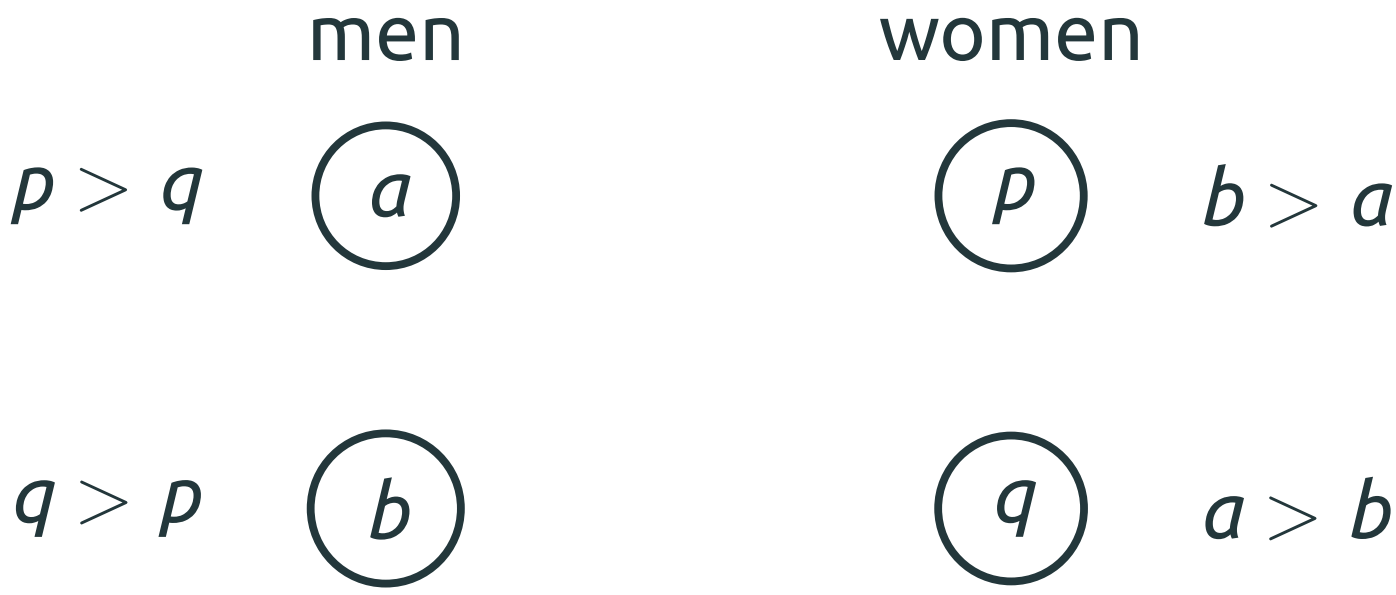
$n = 2$: **Everyone Is Happy**



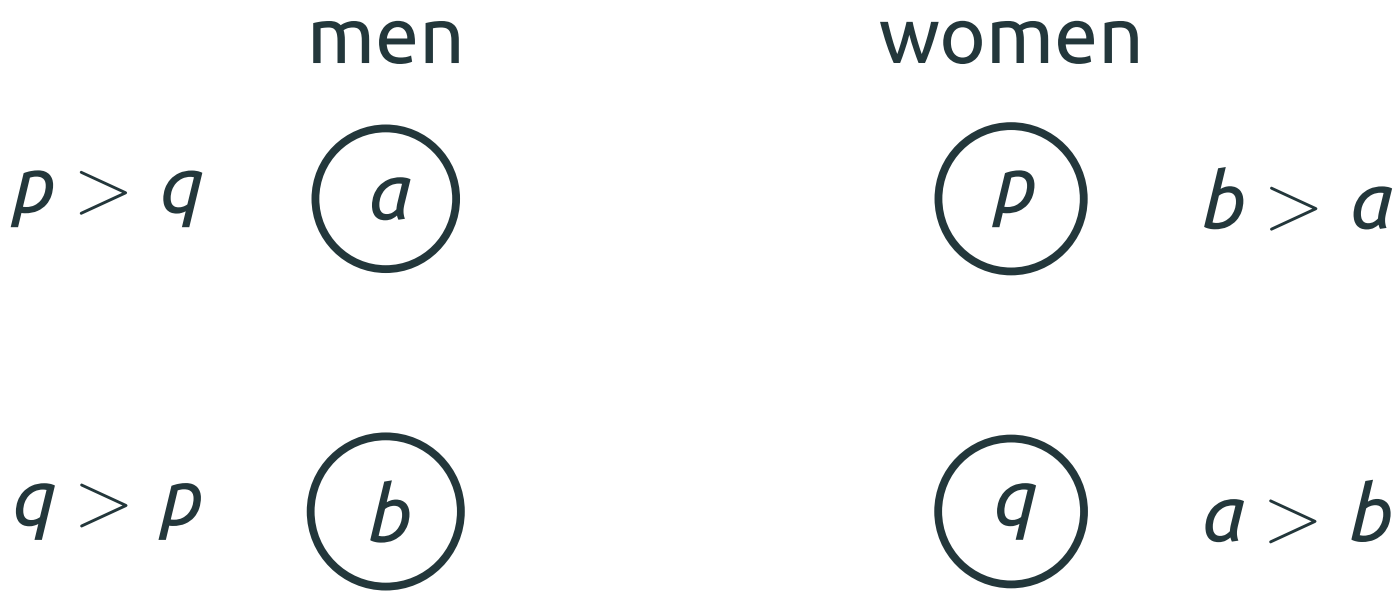
$n = 2$: Everyone Is Happy



$n = 2$: Two Stable Matchings

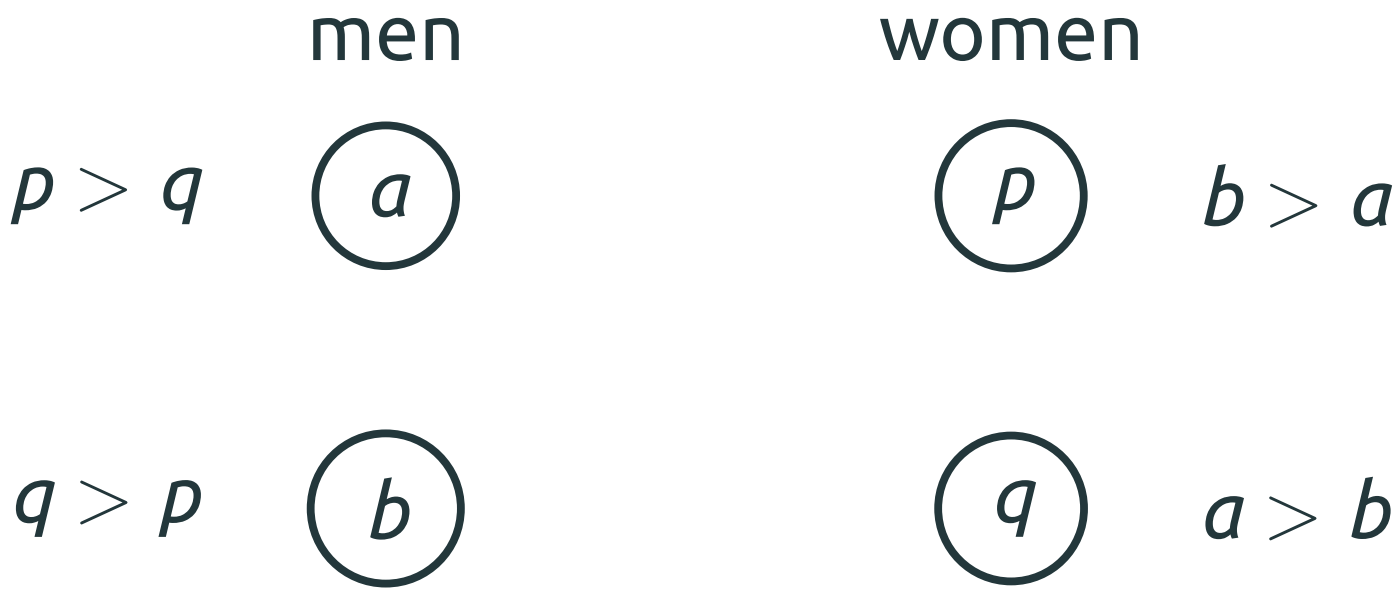


$n = 2$: Two Stable Matchings



two bad but stable matchings

$n = 2$: Two Stable Matchings



two bad but stable matchings

H. Heine / R. Schumann (“Dichterliebe”)

`https://www.youtube.com/watch?v=UzfWyCLmHLc`

Outline

Laissez-Faire…

Laissez-Faire…

- “laissez-faire” approach:

Laissez-Faire…

- “laissez-faire” approach:
- start with some matching

Laissez-Faire…

- “laissez-faire” approach:
- start with some matching
- if there is an pair that wants to marry, let them do it

Laissez-Faire…

- “laissez-faire” approach:
- start with some matching
- if there is an pair that wants to marry, let them do it
- former partners make another pair

Laissez-Faire…

- “laissez-faire” approach:
- start with some matching
- if there is an pair that wants to marry, let them do it
- former partners make another pair
- repeat until a stable matching is obtained

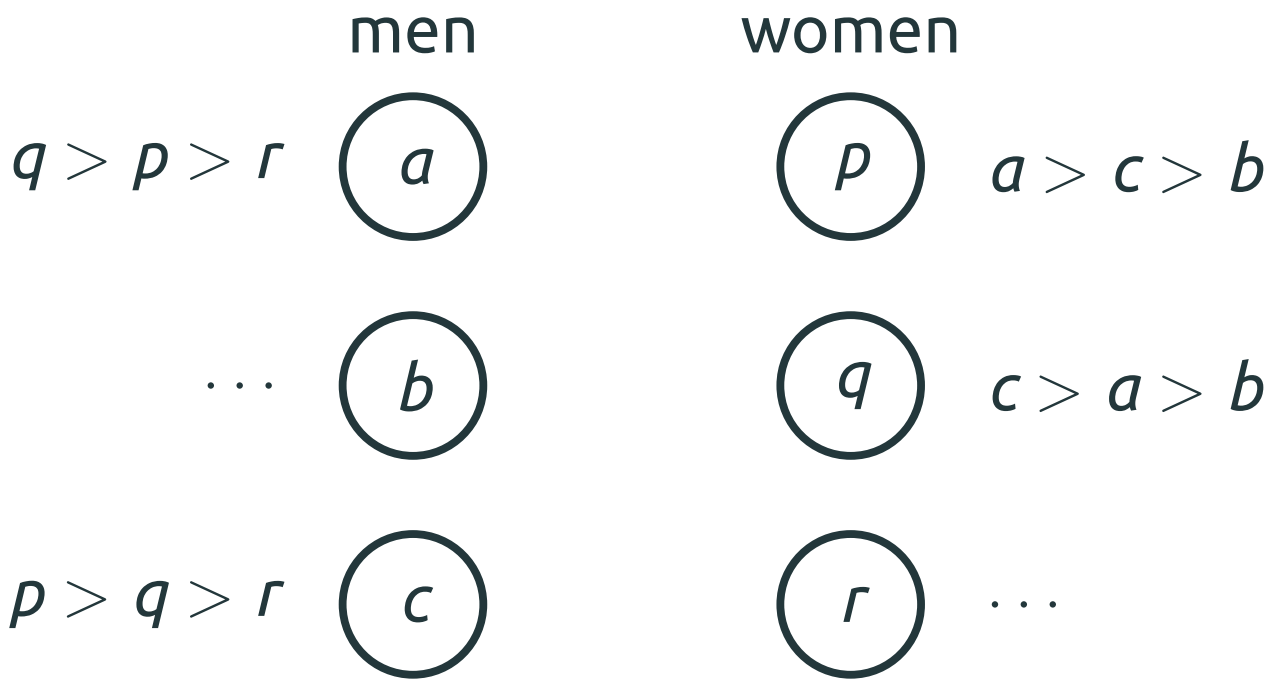
Stabilization?

Stabilization?

…not guaranteed

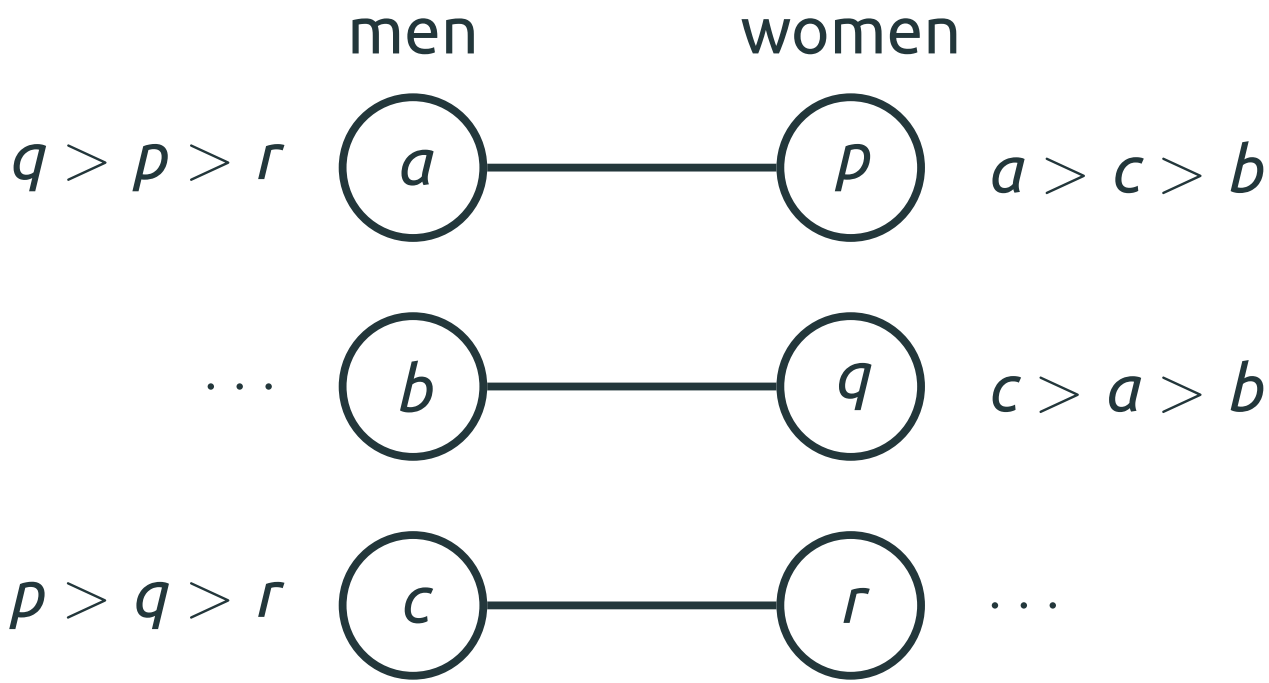
Stabilization?

…not guaranteed



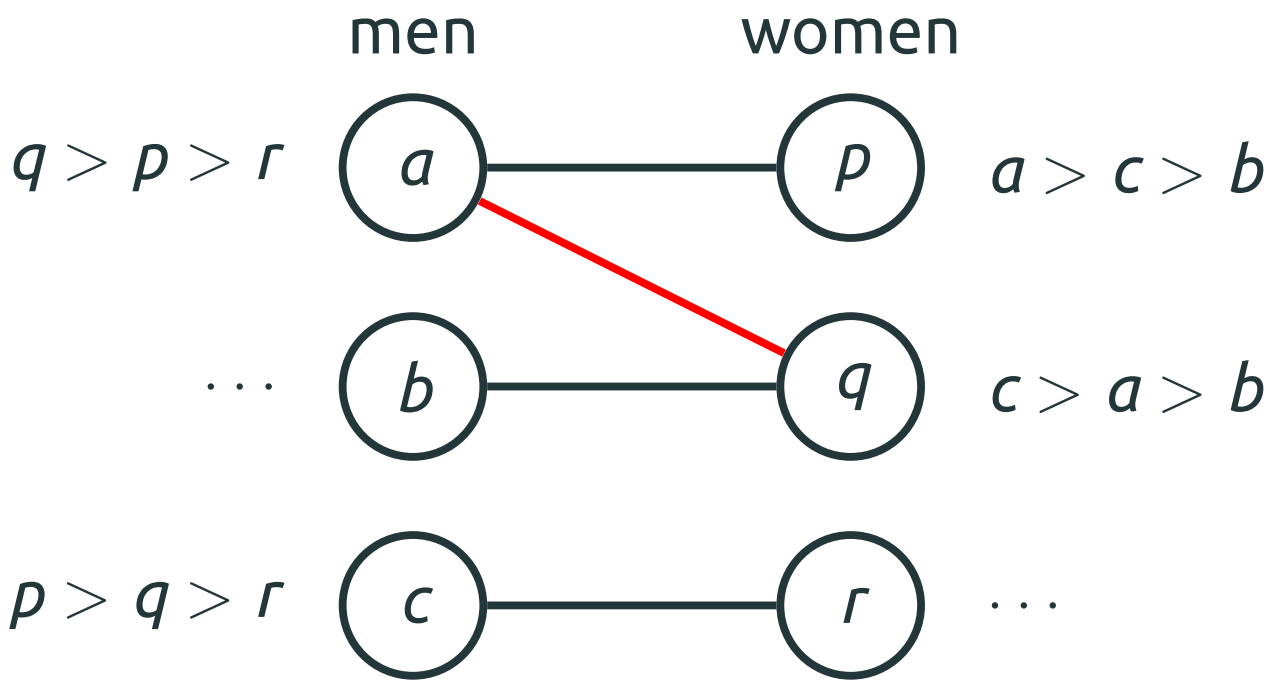
Stabilization?

…not guaranteed



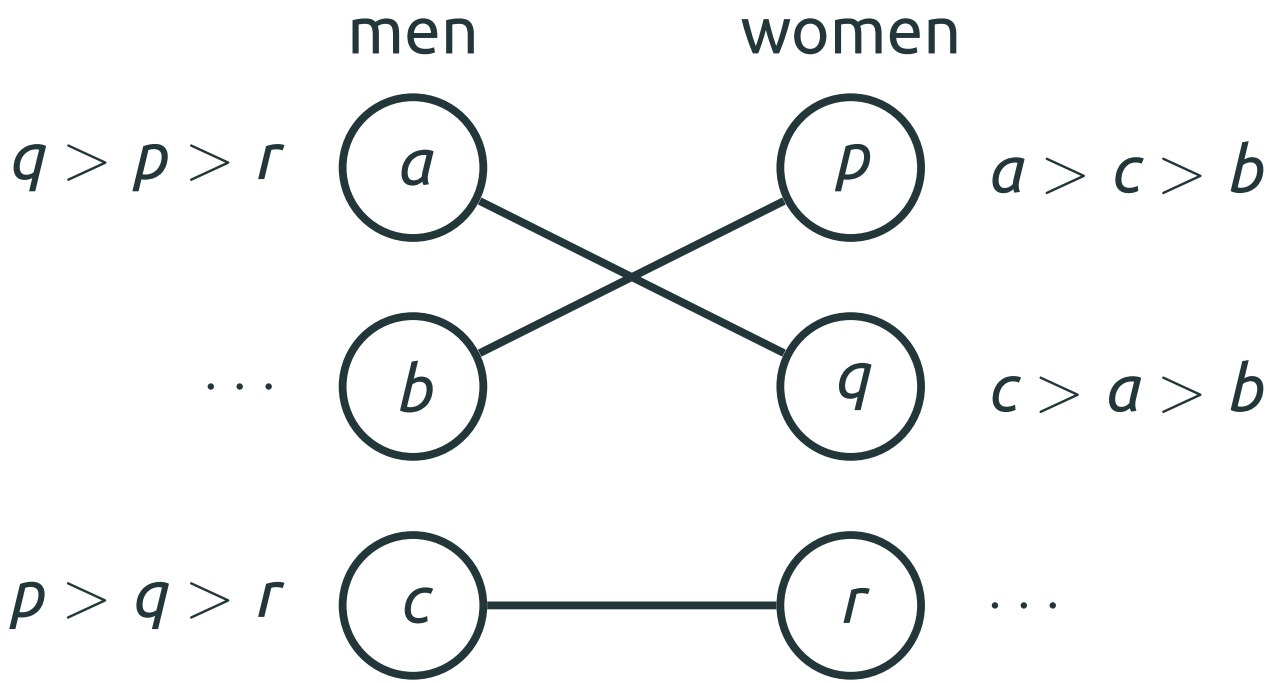
Stabilization?

…not guaranteed



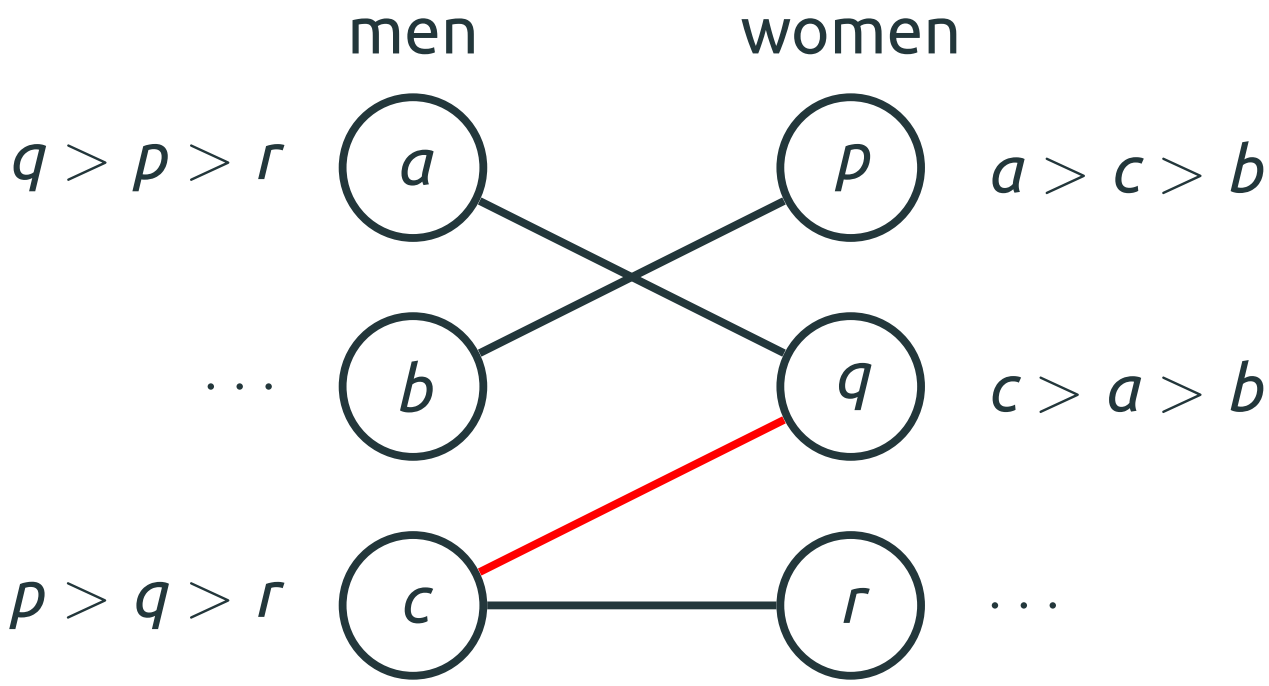
Stabilization?

…not guaranteed



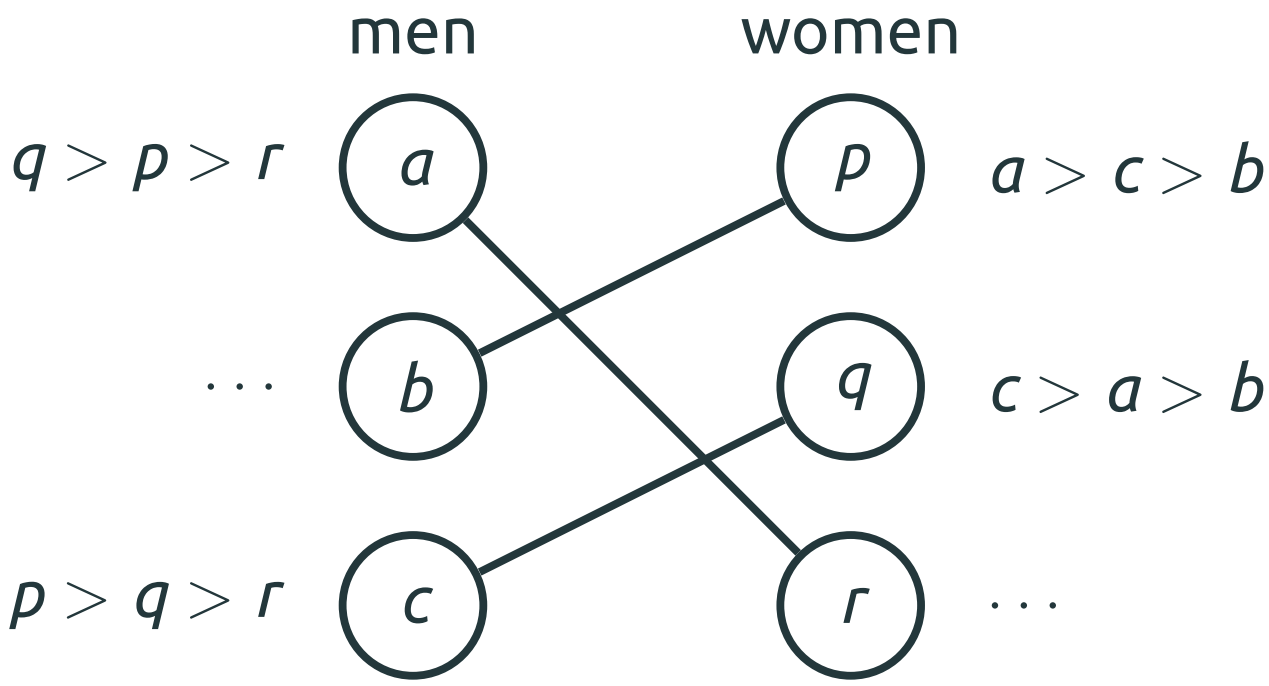
Stabilization?

…not guaranteed



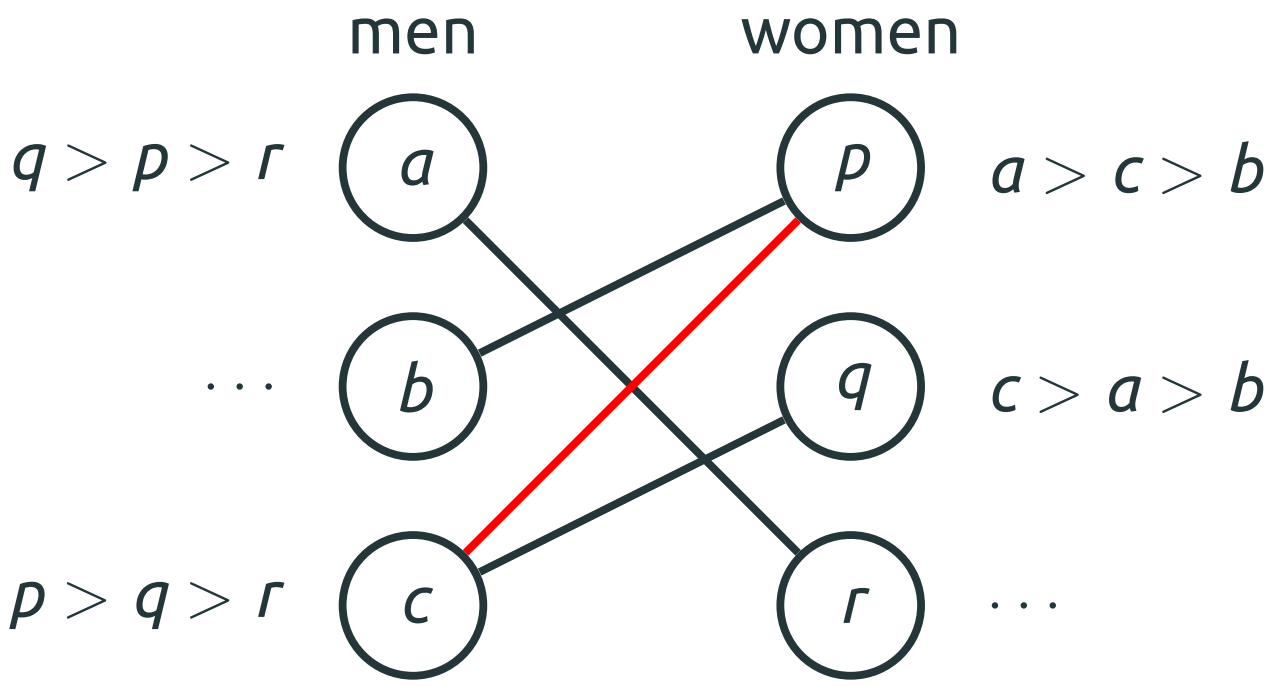
Stabilization?

…not guaranteed



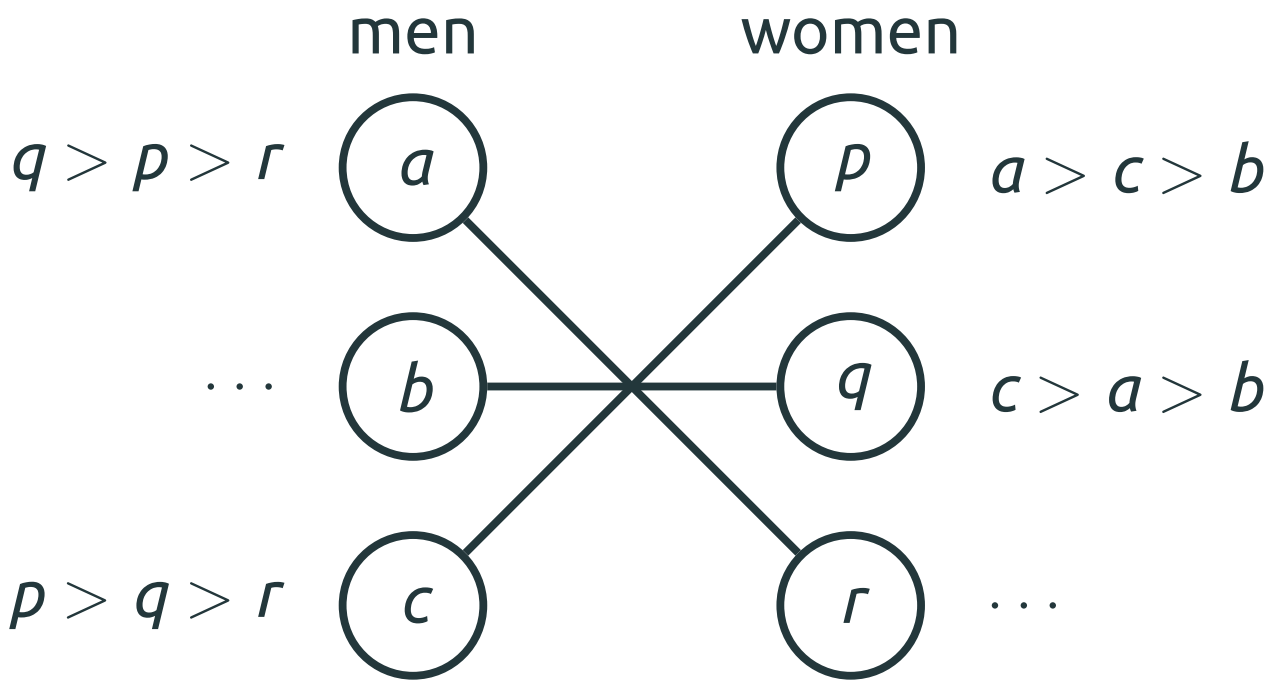
Stabilization?

…not guaranteed



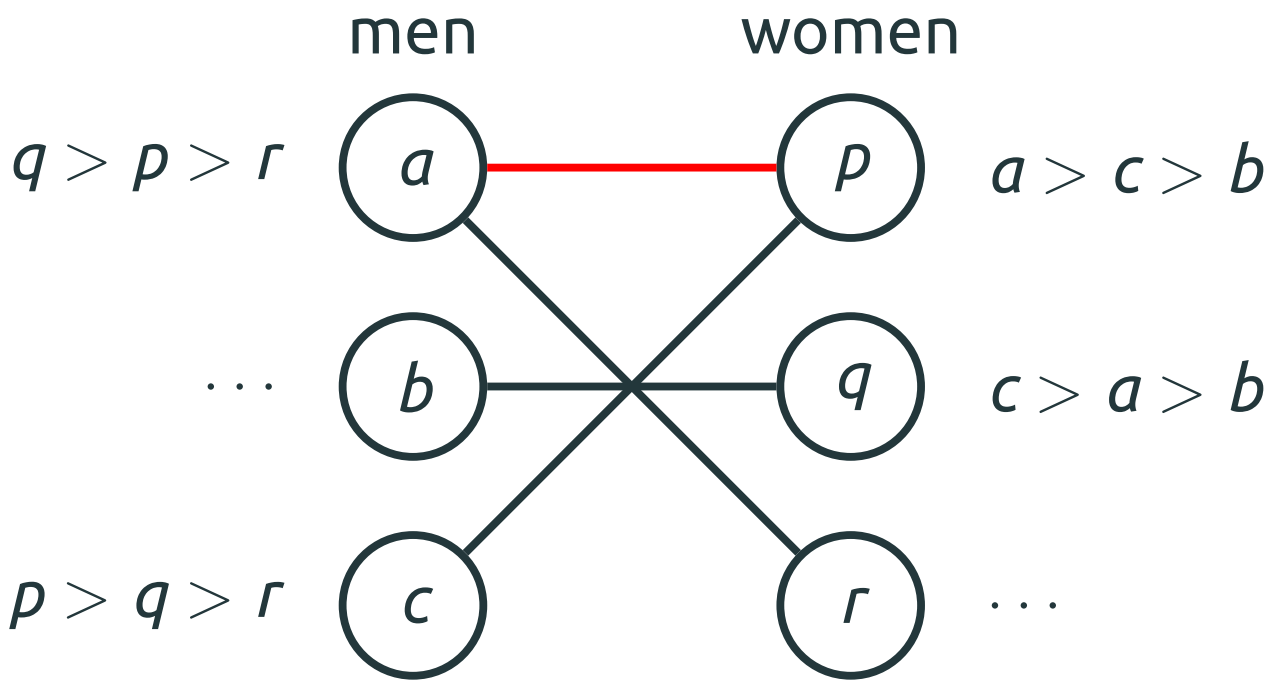
Stabilization?

…not guaranteed



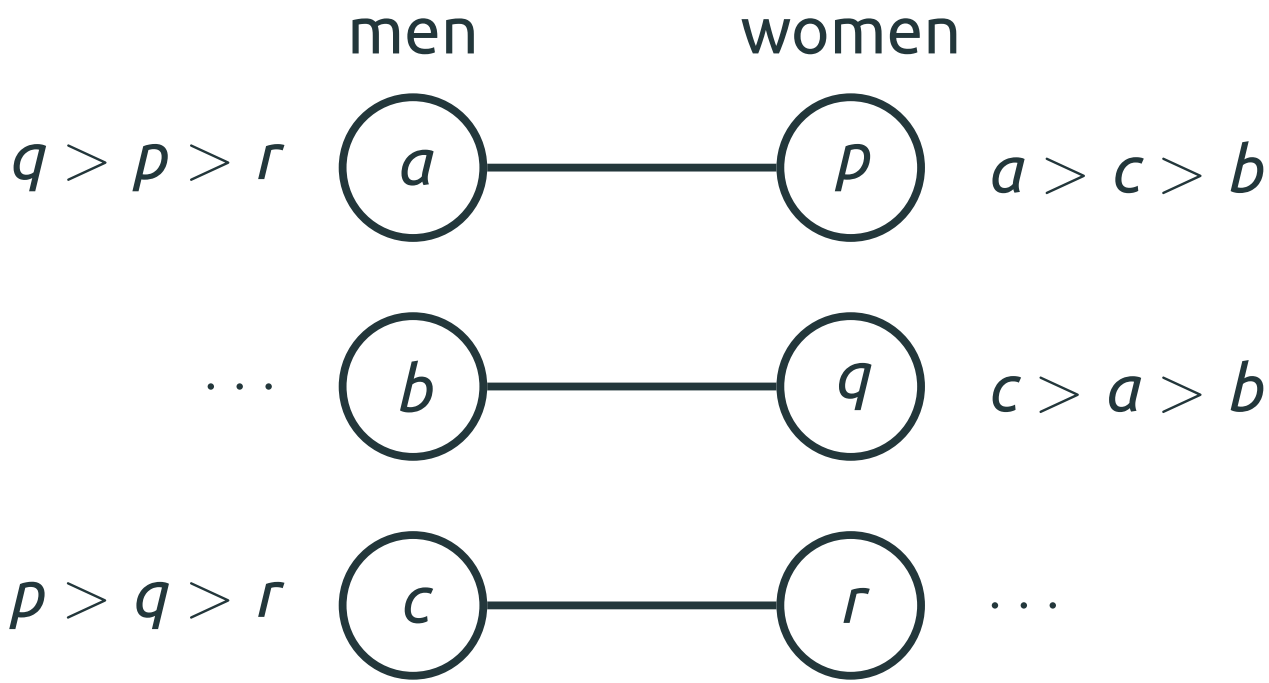
Stabilization?

…not guaranteed



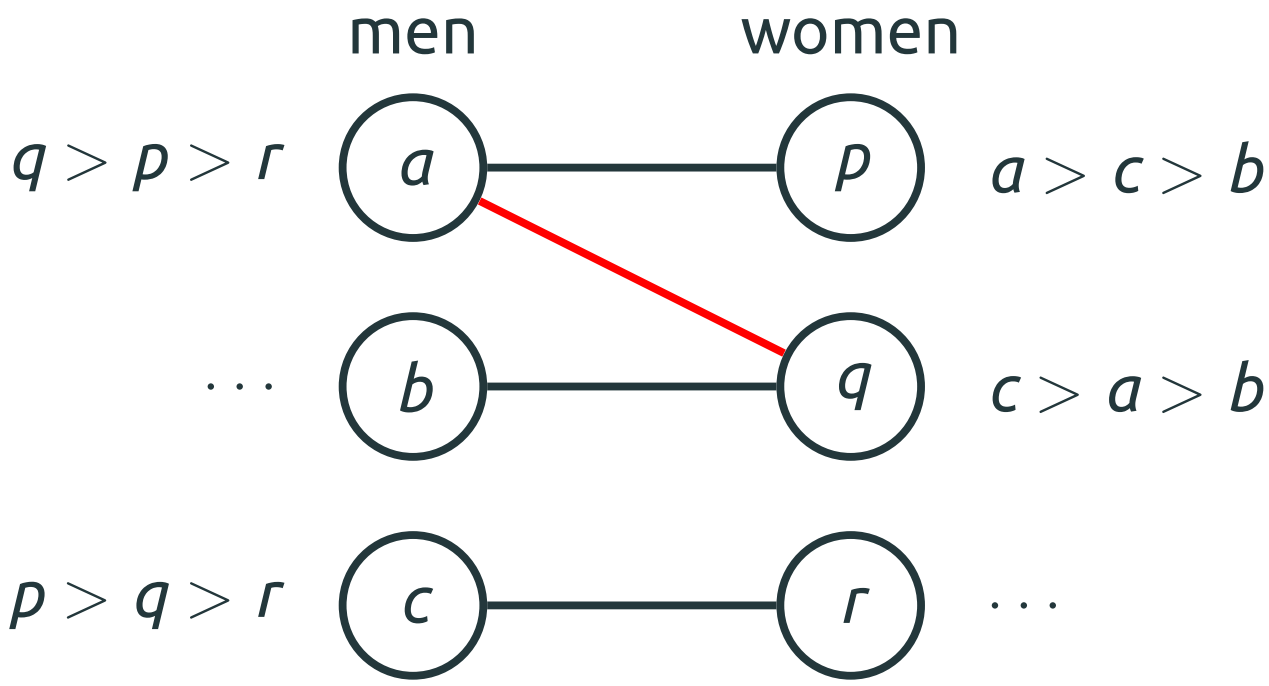
Stabilization?

…not guaranteed



Stabilization?

…not guaranteed



Outline

The Algorithm

The Algorithm

- at each stage: partial matching

The Algorithm

- at each stage: partial matching
- man without a partner makes a proposal…

The Algorithm

- at each stage: partial matching
- man without a partner makes a proposal…
- …that is accepted or rejected

The Algorithm

- at each stage: partial matching
- man without a partner makes a proposal…
- …that is accepted or rejected
- acceptance may destroy an existing pair

The Algorithm

- at each stage: partial matching
- man without a partner makes a proposal…
- …that is accepted or rejected
- acceptance may destroy an existing pair
- repeat until everybody is married

The Algorithm

- at each stage: partial matching
- man without a partner makes a proposal…
- …that is accepted or rejected
- acceptance may destroy an existing pair
- repeat until everybody is married
- two things to specify:

The Algorithm

- at each stage: partial matching
- man without a partner makes a proposal…
- …that is accepted or rejected
- acceptance may destroy an existing pair
- repeat until everybody is married
- two things to specify:
- whom to propose?

The Algorithm

- at each stage: partial matching
- man without a partner makes a proposal…
- …that is accepted or rejected
- acceptance may destroy an existing pair
- repeat until everybody is married
- two things to specify:
- whom to propose?
- accept or reject?

The Algorithm

- at each stage: partial matching
- man without a partner makes a proposal…
- …that is accepted or rejected
- acceptance may destroy an existing pair
- repeat until everybody is married
- two things to specify:
- whom to propose?
- accept or reject?
- non-deterministic

Men’s behavior

Men’s behavior

- make proposals according to the ordering (ignoring the existing pairs)

Men’s behavior

- make proposals according to the ordering (ignoring the existing pairs)
- never leave the current partner on their own

Men’s behavior

- make proposals according to the ordering (ignoring the existing pairs)
- never leave the current partner on their own
- never repeat a rejected proposal

Men’s behavior

- make proposals according to the ordering (ignoring the existing pairs)
- never leave the current partner on their own
- never repeat a rejected proposal
- memory: position in the list / current partner

Men’s behavior

- make proposals according to the ordering (ignoring the existing pairs)
- never leave the current partner on their own
- never repeat a rejected proposal
- memory: position in the list / current partner
- things are getting worse with time

Women’s behavior

Women’s behavior

- accept a proposal if no partner…

Women’s behavior

- accept a proposal if no partner…
- …or an improvement

Women’s behavior

- accept a proposal if no partner…
- …or an improvement
- memory: current partner

Women’s behavior

- accept a proposal if no partner…
- …or an improvement
- memory: current partner
- things are getting better with time

Outline

Three Things to Prove

Three Things to Prove

- termination

Three Things to Prove

- termination
- perfect matching

Three Things to Prove

- termination
- perfect matching
- stability

Termination

Termination

- at each step a new proposal is made

Termination

- at each step a new proposal is made
- proposals are never repeated

Termination

- at each step a new proposal is made
- proposals are never repeated
- at most n^2 proposals

Termination

- at each step a new proposal is made
- proposals are never repeated
- at most n^2 proposals
- is rather fast

Termination

- at each step a new proposal is made
- proposals are never repeated
- at most n^2 proposals
- is rather fast
- (brute-force search: $n!$)

Perfect Matching

Perfect Matching

- if a man remains without a partner…

Perfect Matching

- if a man remains without a partner…
- all women rejected him…

Perfect Matching

- if a man remains without a partner…
- all women rejected him…
- …when he proposed or later

Perfect Matching

- if a man remains without a partner…
- all women rejected him…
- …when he proposed or later
- so all women were married at some point

Perfect Matching

- if a man remains without a partner…
- all women rejected him…
- …when he proposed or later
- so all women were married at some point
- and remain married

Perfect Matching

- if a man remains without a partner…
- all women rejected him…
- …when he proposed or later
- so all women were married at some point
- and remain married
- contradiction (#women = #men)

Stability

Stability

- assume there is a dangerous pair (m, w) at the end

Stability

- assume there is a dangerous pair (m, w) at the end
- m: w is better than his current partner

Stability

- assume there is a dangerous pair (m, w) at the end
- m: w is better than his current partner
- w is earlier in m's list than his current partner

Stability

- assume there is a dangerous pair (m, w) at the end
- m: w is better than his current partner
- w is earlier in m's list than his current partner
- w rejected m at some stage

Stability

- assume there is a dangerous pair (m, w) at the end
- m: w is better than his current partner
- w is earlier in m's list than his current partner
- w rejected m at some stage
- at some time her partner was better for her than m

Stability

- assume there is a dangerous pair (m, w) at the end
- m: w is better than his current partner
- w is earlier in m's list than his current partner
- w rejected m at some stage
- at some time her partner was better for her than m
- but things could only improve for her since then

Stability

- assume there is a dangerous pair (m, w) at the end
- m: w is better than his current partner
- w is earlier in m's list than his current partner
- w rejected m at some stage
- at some time her partner was better for her than m
- but things could only improve for her since then: a contradiction

Gale–Shapley Theorem

Gale–Shapley Theorem

Theorem: a stable matching always exists

Gale–Shapley Theorem

Theorem: a stable matching always exists

Proof: use the algorithm

Gale–Shapley Theorem

Theorem: a stable matching always exists
Proof: use the algorithm

remarks:

Gale–Shapley Theorem

Theorem: a stable matching always exists
Proof: use the algorithm

remarks:

- stable matching is not unique

Gale–Shapley Theorem

Theorem: a stable matching always exists
Proof: use the algorithm

remarks:

- stable matching is not unique
- may depend on the order?

Gale–Shapley Theorem

Theorem: a stable matching always exists
Proof: use the algorithm

remarks:

- stable matching is not unique
- may depend on the order?
- in fact not

Gale–Shapley Theorem

Theorem: a stable matching always exists
Proof: use the algorithm

remarks:

- stable matching is not unique
- may depend on the order?
- in fact not
- unfair: favors men

Gale–Shapley Theorem

Theorem: a stable matching always exists
Proof: use the algorithm

remarks:

- stable matching is not unique
- may depend on the order?
- in fact not
- unfair: favors men
- a symmetric algorithm

Outline

Best for Men

Best for Men

- a man may have different partners in different stable matchings

Best for Men

- a man may have different partners in different stable matchings
- **claim**: algorithm gives the best one

Best for Men

- a man may have different partners in different stable matchings
- **claim**: algorithm gives the best one
- corollary: non-determinism inessential

Best for Men

- a man may have different partners in different stable matchings
- **claim**: algorithm gives the best one
- corollary: non-determinism inessential
- corollary: no collisions if for every man the best possible partner (in stable matchings) is chosen

Best for Men

- a man may have different partners in different stable matchings
- **claim**: algorithm gives the best one
- corollary: non-determinism inessential
- corollary: no collisions if for every man the best possible partner (in stable matchings) is chosen
- reformulation: if m was rejected (in any way) by w during the algorithm, (m, w) cannot appear in a stable matching

Proof

Proof

- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching

Proof

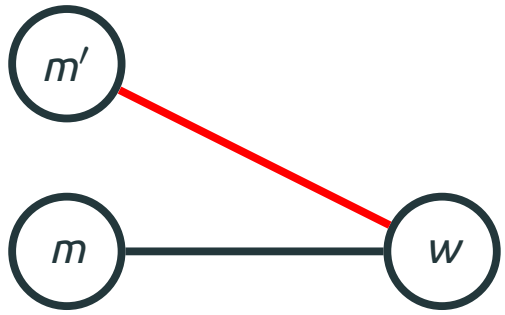
- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching
- induction by the rejection moment

Proof

- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching
- induction by the rejection moment
- (m, w) in a stable matching

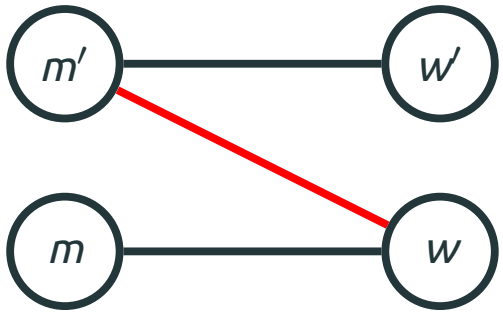
Proof

- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching
- induction by the rejection moment
- (m, w) in a stable matching
- but w rejected/left m because $w : m' > m$



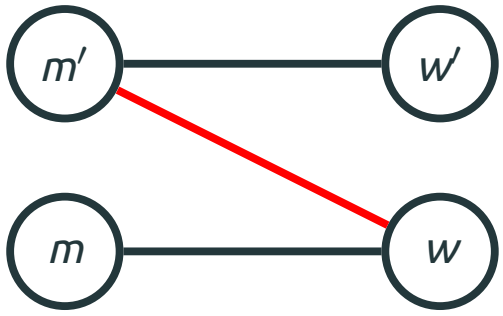
Proof

- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching
- induction by the rejection moment
- (m, w) in a stable matching
- but w rejected/left m because $w : m' > m$
- (m', w') in (the same) stable matching



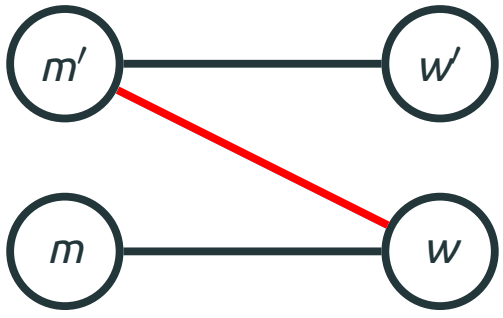
Proof

- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching
- induction by the rejection moment
- (m, w) in a stable matching
- but w rejected/left m because $w : m' > m$
- (m', w') in (the same) stable matching
- $m' : w' > w$ (otherwise not stable)



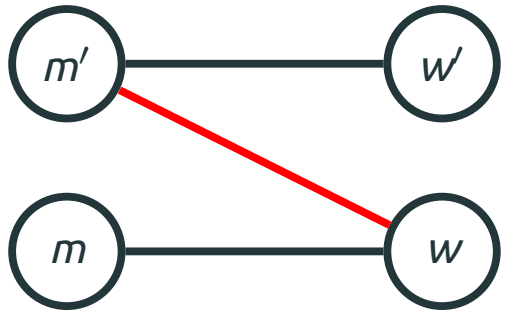
Proof

- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching
- induction by the rejection moment
- (m, w) in a stable matching
- but w rejected/left m because $w : m' > m$
- (m', w') in (the same) stable matching
- $m' : w' > w$ (otherwise not stable)
- m' was paired with w during the algorithm



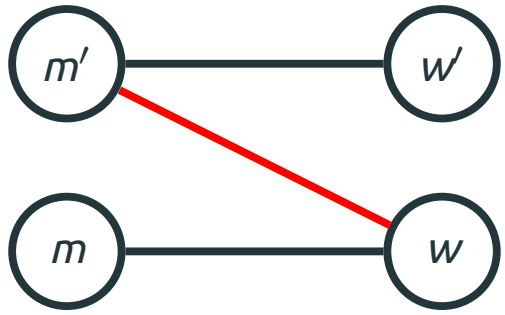
Proof

- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching
- induction by the rejection moment
- (m, w) in a stable matching
- but w rejected/left m because $w : m' > m$
- (m', w') in (the same) stable matching
- $m' : w' > w$ (otherwise not stable)
- m' was paired with w during the algorithm
- so m' was *earlier* rejected by w'



Proof

- **claim**: if m was rejected by w during the algorithm, (m, w) cannot appear in a stable matching
- induction by the rejection moment
- (m, w) in a stable matching
- but w rejected/left m because $w : m' > m$
- (m', w') in (the same) stable matching
- $m' : w' > w$ (otherwise not stable)
- m' was paired with w during the algorithm
- so m' was *earlier* rejected by w'
- so (m', w') cannot appear in a stable matching



Outline

Worst for Women

Worst for Women

- a woman may have different partners in different stable matchings

Worst for Women

- a woman may have different partners in different stable matchings
- **claim:** the algorithm gives the worst one

Worst for Women

- a woman may have different partners in different stable matchings
- **claim:** the algorithm gives the worst one
- reformulation: if (m, w) is in some stable matching, the algorithm cannot pair w with some m' that is better (for w)

Proof

Proof

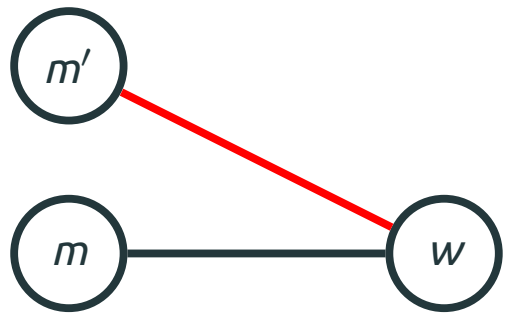
- **claim**: if (m, w) appears in some stable matching, the algorithm cannot pair w with some m' that is better (for w)

Proof

- **claim**: if (m, w) appears in some stable matching, the algorithm cannot pair w with some m' that is better (for w)
- (m, w) in a stable matching

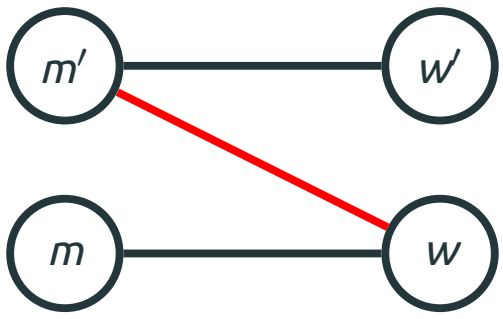
Proof

- **claim**: if (m, w) appears in some stable matching, the algorithm cannot pair w with some m' that is better (for w)
- (m, w) in a stable matching
- the algorithm pairs w with some m', and $w : m' > m$



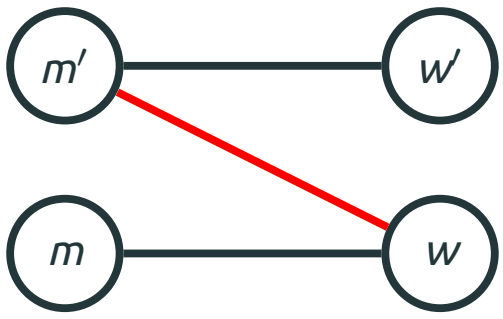
Proof

- **claim**: if (m, w) appears in some stable matching, the algorithm cannot pair w with some m' that is better (for w)
- (m, w) in a stable matching
- the algorithm pairs w with some m', and $w : m' > m$
- (m', w') in (the same) stable matching



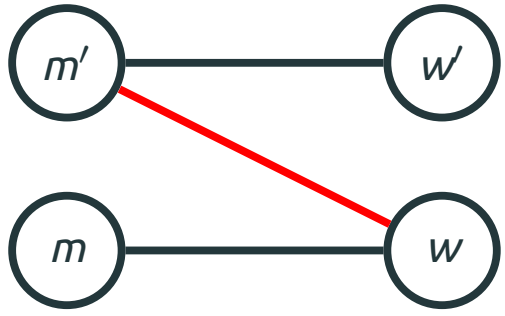
Proof

- **claim**: if (m, w) appears in some stable matching, the algorithm cannot pair w with some m' that is better (for w)
- (m, w) in a stable matching
- the algorithm pairs w with some m', and $w : m' > m$
- (m', w') in (the same) stable matching
- $m' : w' > w$ (otherwise not stable)



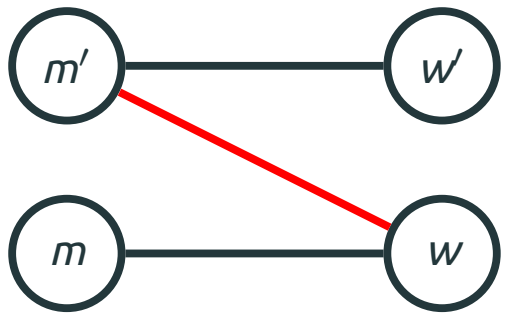
Proof

- **claim**: if (m, w) appears in some stable matching, the algorithm cannot pair w with some m' that is better (for w)
- (m, w) in a stable matching
- the algorithm pairs w with some m', and $w : m' > m$
- (m', w') in (the same) stable matching
- $m' : w' > w$ (otherwise not stable)
- the algorithm gives not the best result for m' (w instead of w' in a stable matching)



Proof

- **claim**: if (m, w) appears in some stable matching, the algorithm cannot pair w with some m' that is better (for w)
- (m, w) in a stable matching
- the algorithm pairs w with some m', and $w : m' > m$
- (m', w') in (the same) stable matching
- $m' : w' > w$ (otherwise not stable)
- the algorithm gives not the best result for m' (w instead of w' in a stable matching)
- contradiction



Closing remarks

Closing remarks

- Counterintuitive: men never leave their partners

Closing remarks

- Counterintuitive: men never leave their partners
- …while the women are allowed to

Closing remarks

- Counterintuitive: men never leave their partners
- …while the women are allowed to
- still the algorithm favors “men”

Closing remarks

- Counterintuitive: men never leave their partners
- …while the women are allowed to
- still the algorithm favors “men”
- practice: not 1-1 correspondence

Closing remarks

- Counterintuitive: men never leave their partners
- …while the women are allowed to
- still the algorithm favors “men”
- practice: not 1-1 correspondence
- partial preferences (equality)

Closing remarks

- Counterintuitive: men never leave their partners
- …while the women are allowed to
- still the algorithm favors “men”
- practice: not 1-1 correspondence
- partial preferences (equality)
- some candidates not acceptable at all

Closing remarks

- Counterintuitive: men never leave their partners
- …while the women are allowed to
- still the algorithm favors “men”
- practice: not 1-1 correspondence
- partial preferences (equality)
- some candidates not acceptable at all
- cheating?

Closing remarks

- Counterintuitive: men never leave their partners
- …while the women are allowed to
- still the algorithm favors “men”
- practice: not 1-1 correspondence
- partial preferences (equality)
- some candidates not acceptable at all
- cheating?
- economics!