University of Glasgow

Profile-based optimal matchings in Stable Marriage

Frances Cooper

Joint work with: Prof David Manlove

Outline

Outline

- Matching problems

Outline

- Matching problems
- Fairness

Outline

- Matching problems
- Fairness
- Finding fair stable matchings

Outline

- Matching problems
- Fairness
- Finding fair stable matchings
- Experiments

Frances Cooper

Outline

- Matching problems
- Fairness
- Finding fair stable matchings
- Experiments
- Future work

Matching Problems

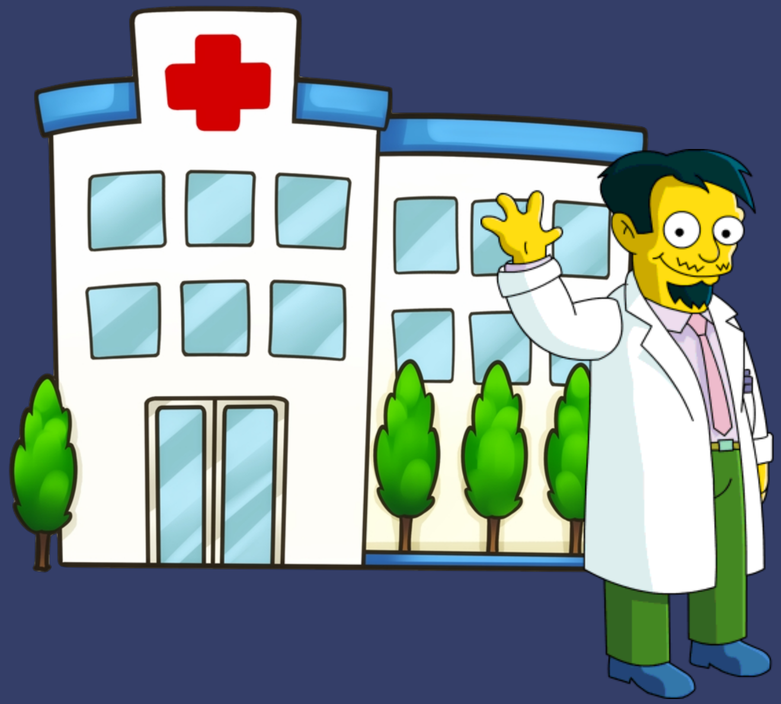
Matching Problems

Matching Problems

Matching Problems

- Assign one set of entities to another set of entities

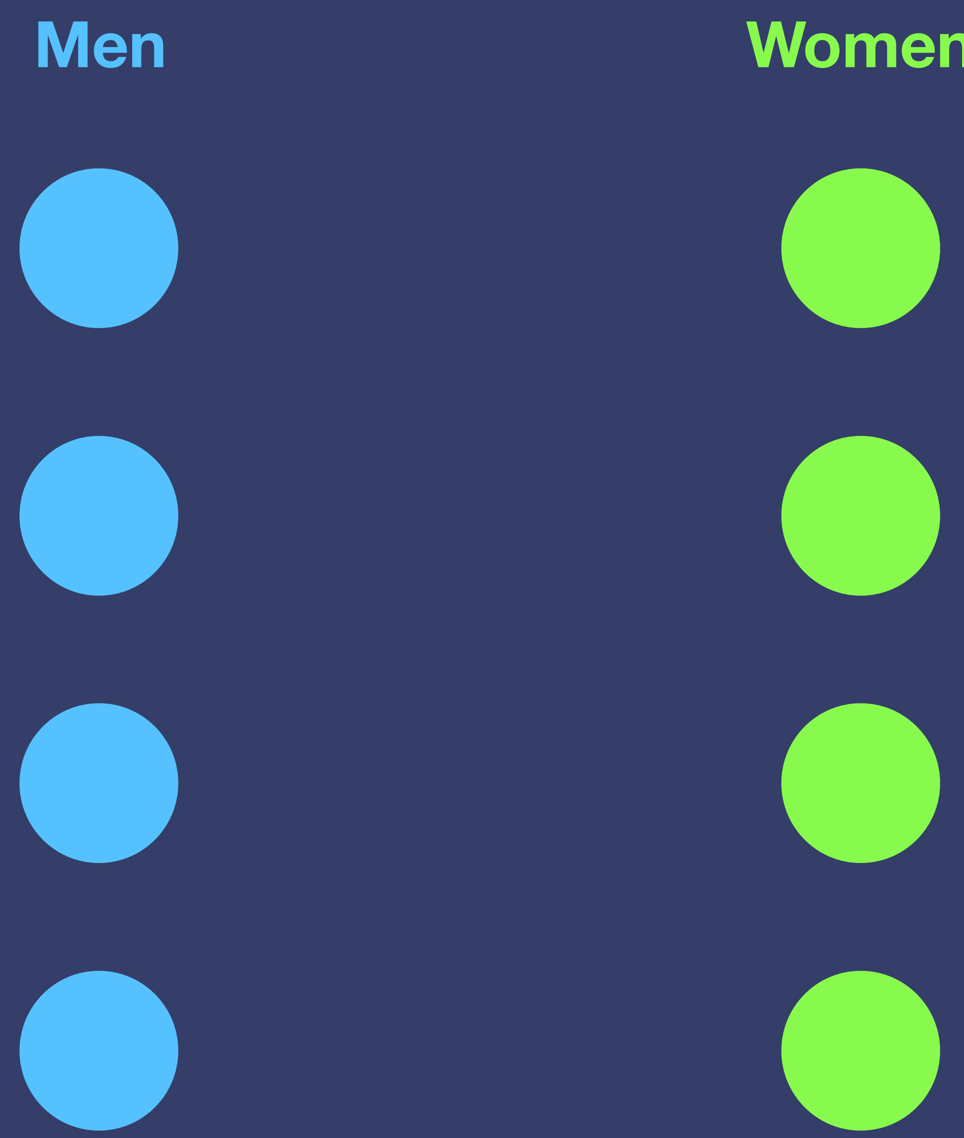
Matching Problems



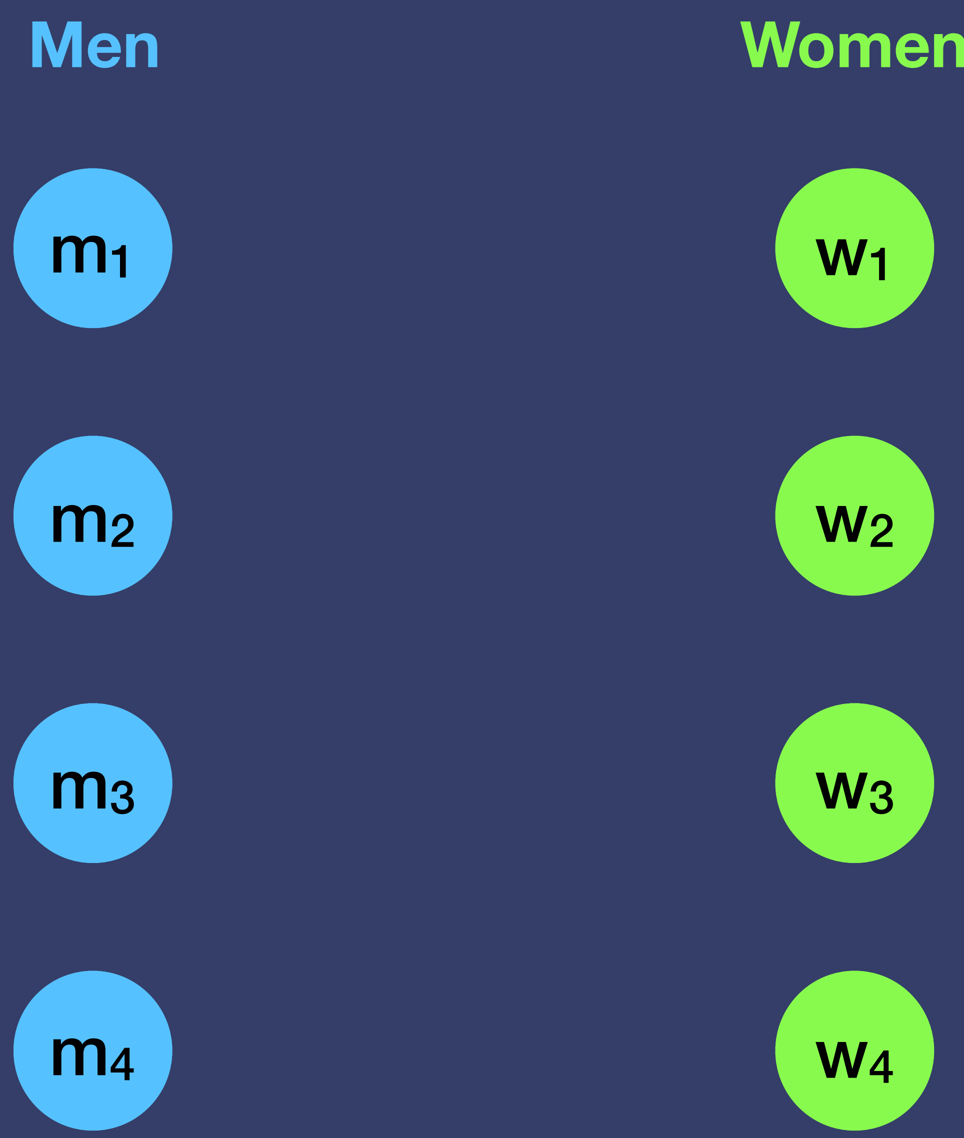
- Assign one set of entities to another set of entities
- Based on preferences and capacities

Stable Marriage

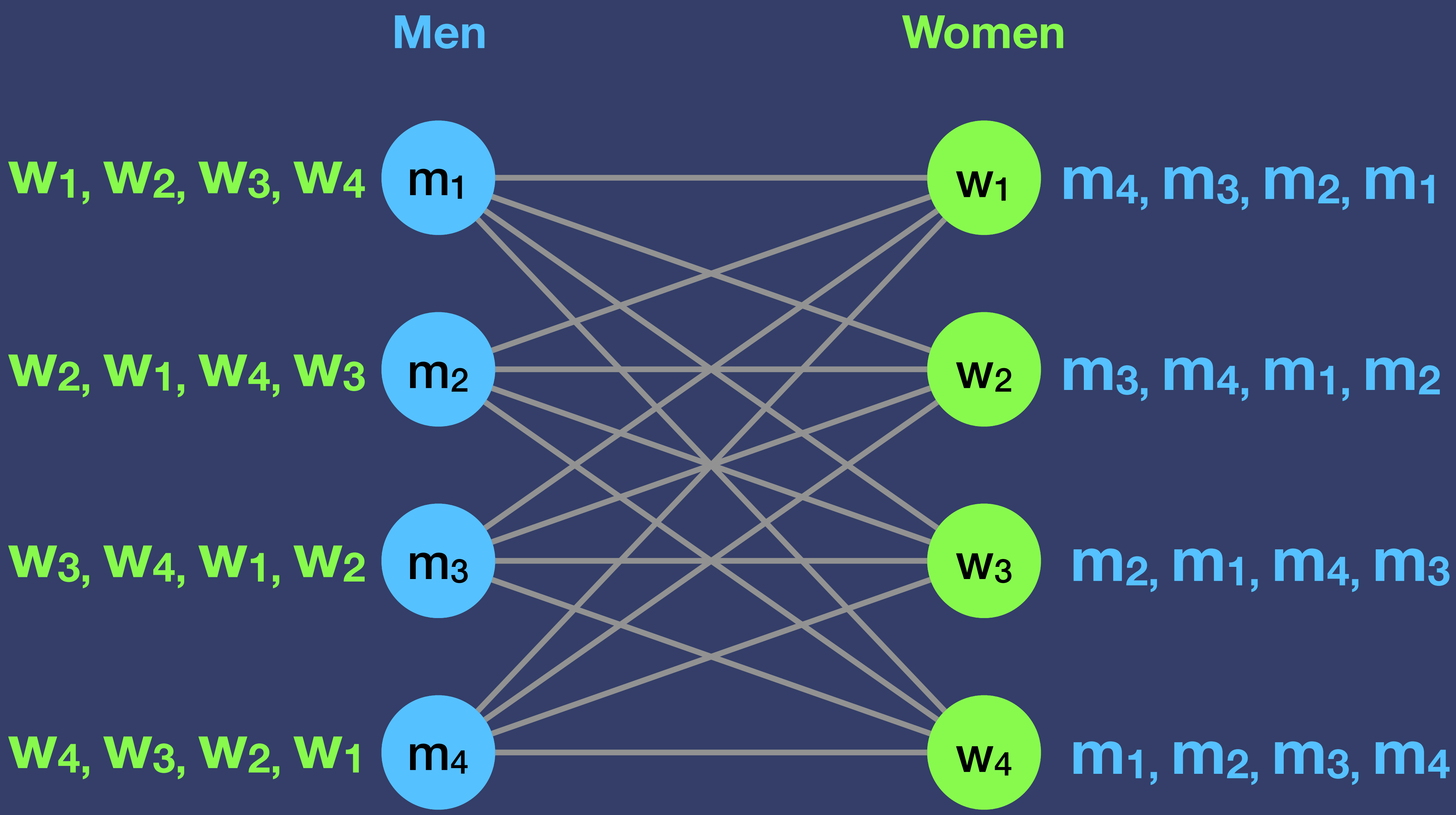
Stable Marriage



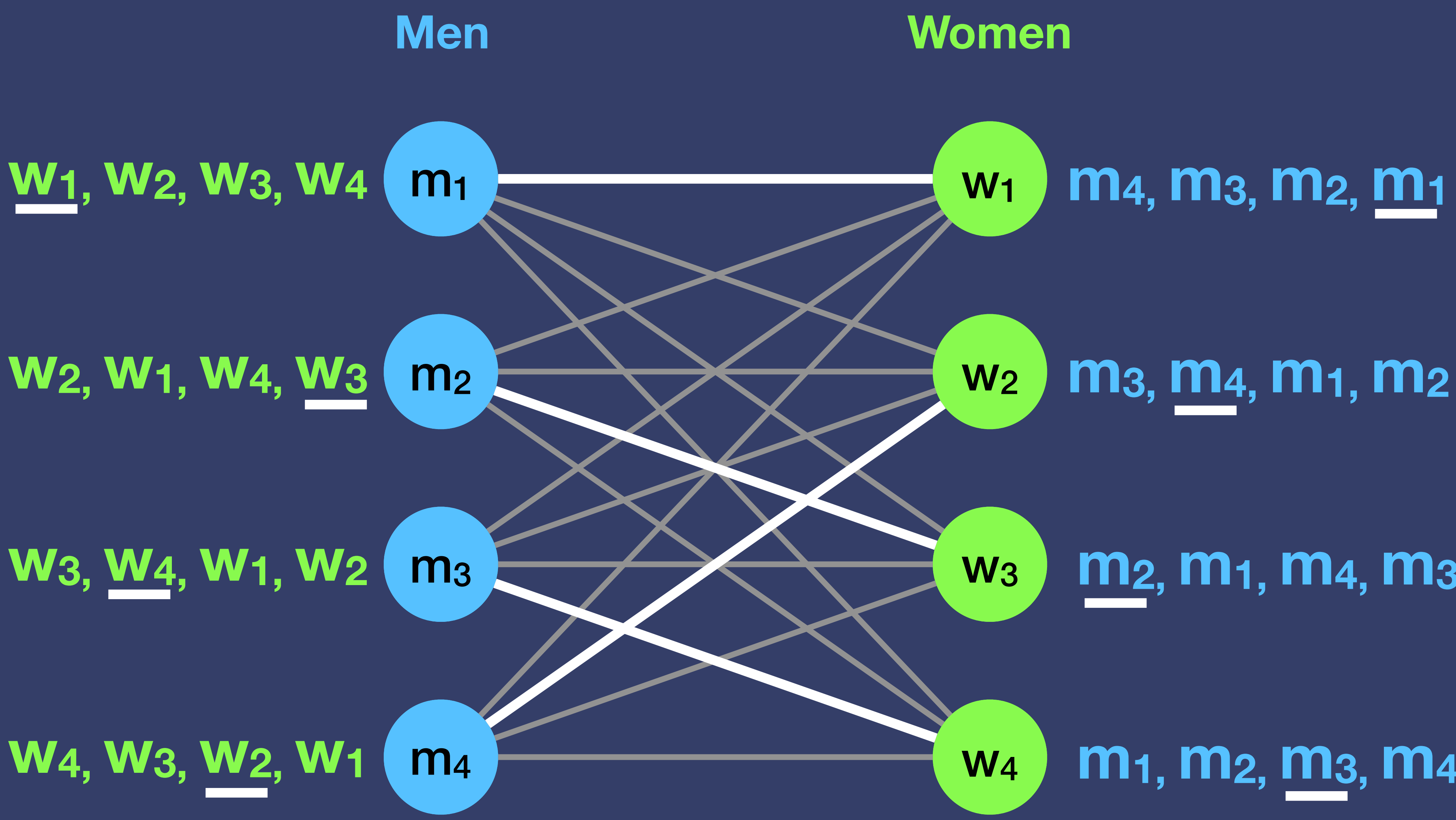
Stable Marriage



Stable Marriage

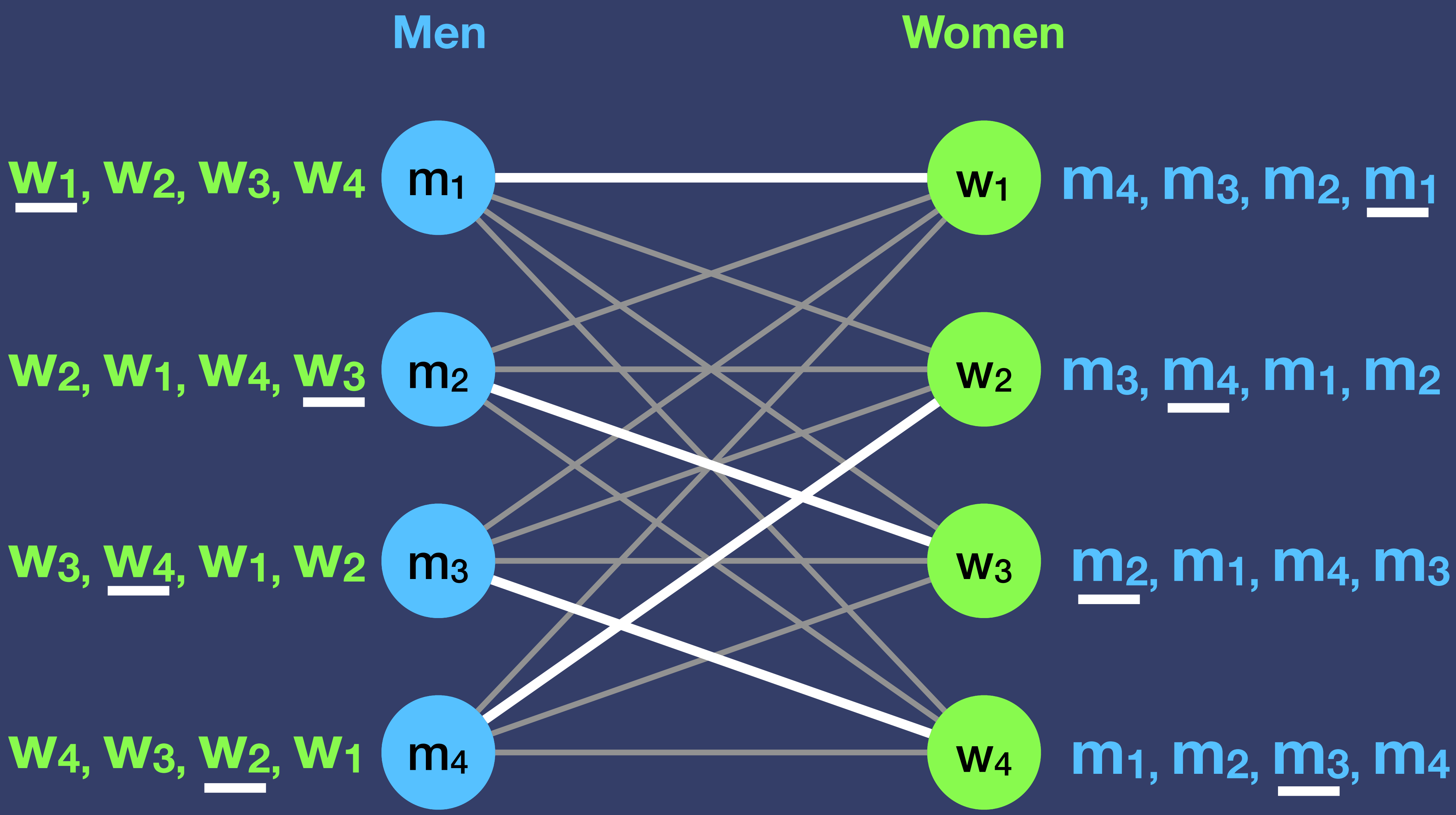


Stable Marriage



Rank

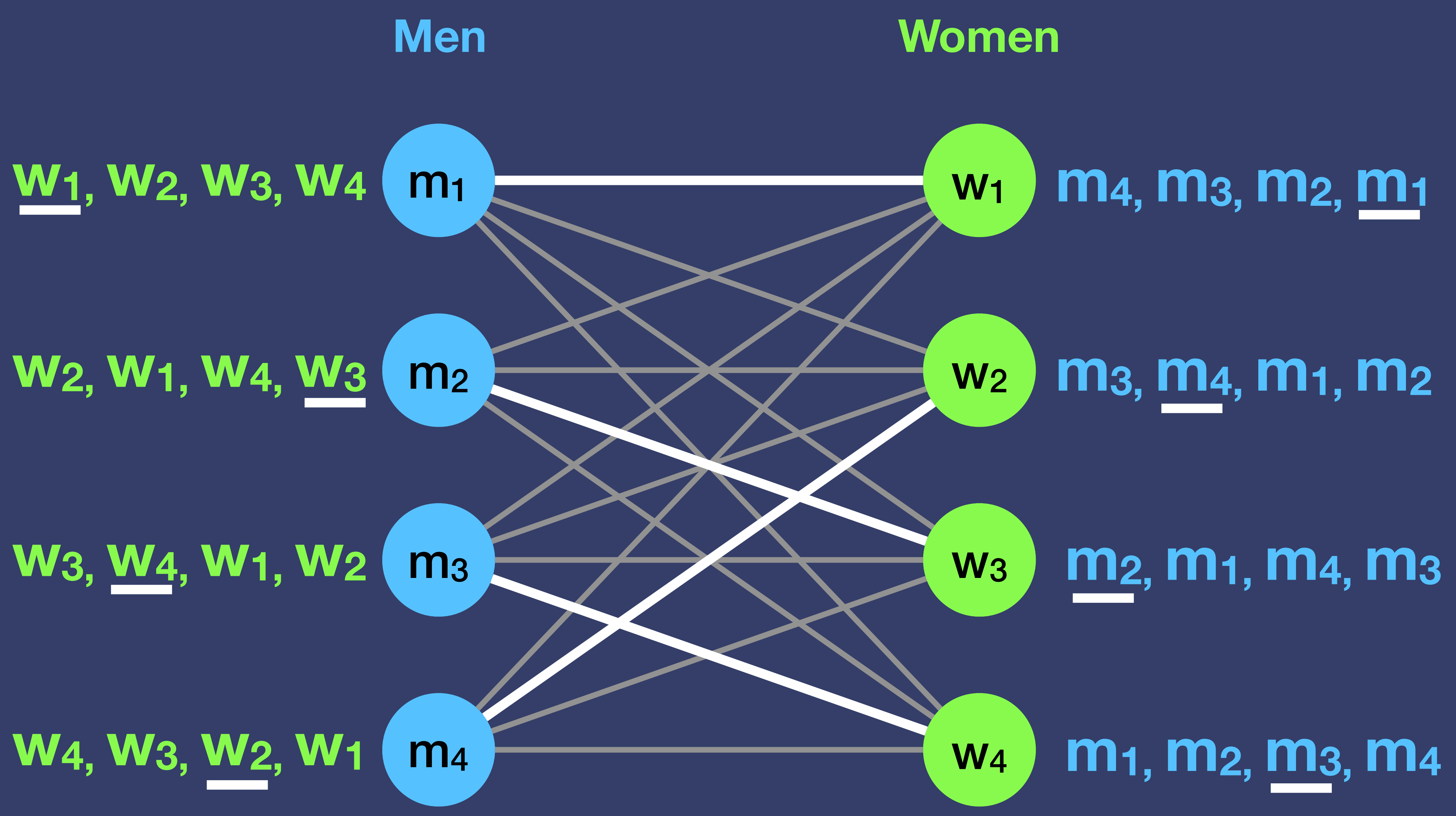
Stable Marriage



Rank

Degree

Stable Marriage

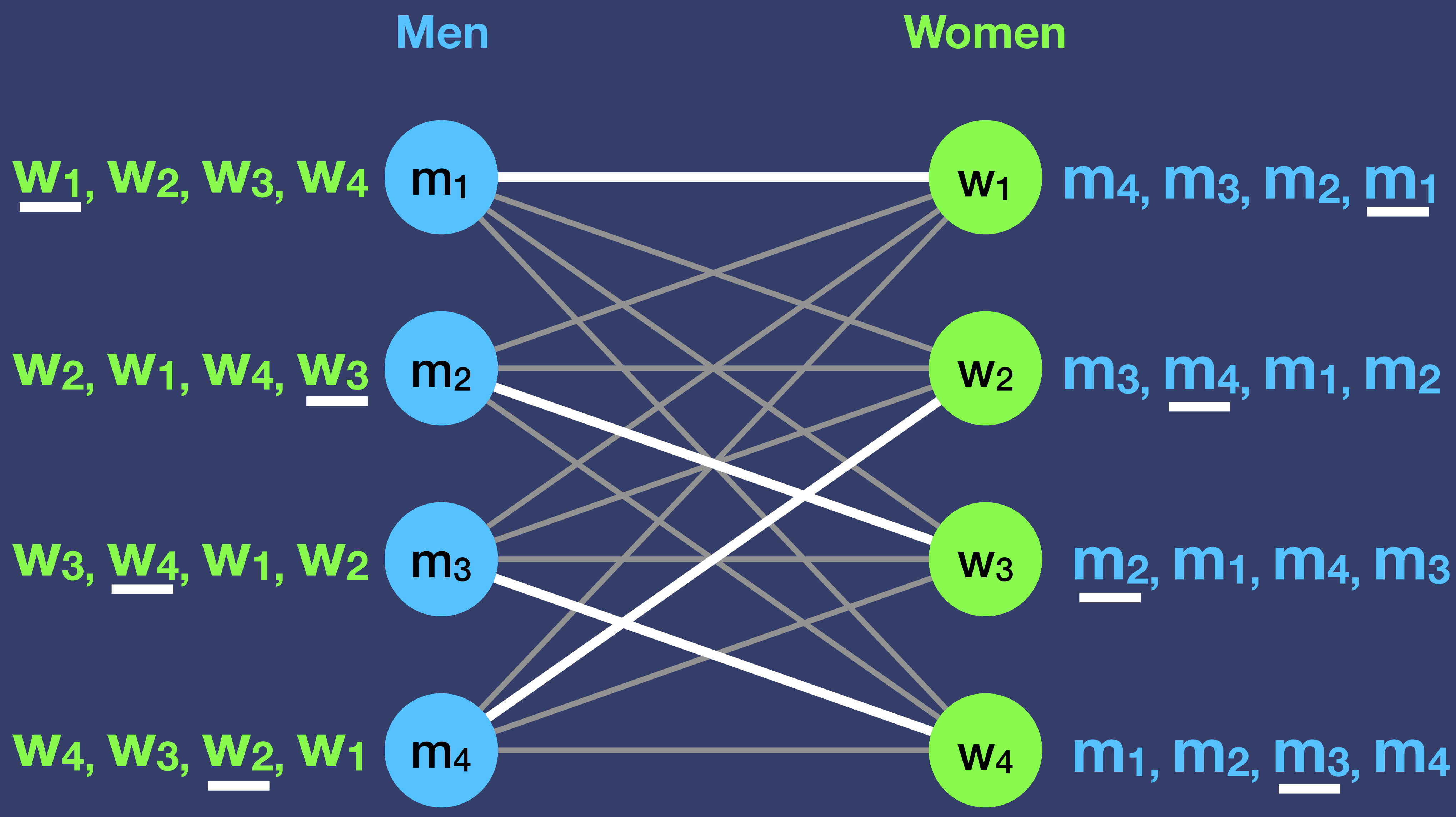


Rank

Degree

Profile <_, _, _, _>

Stable Marriage



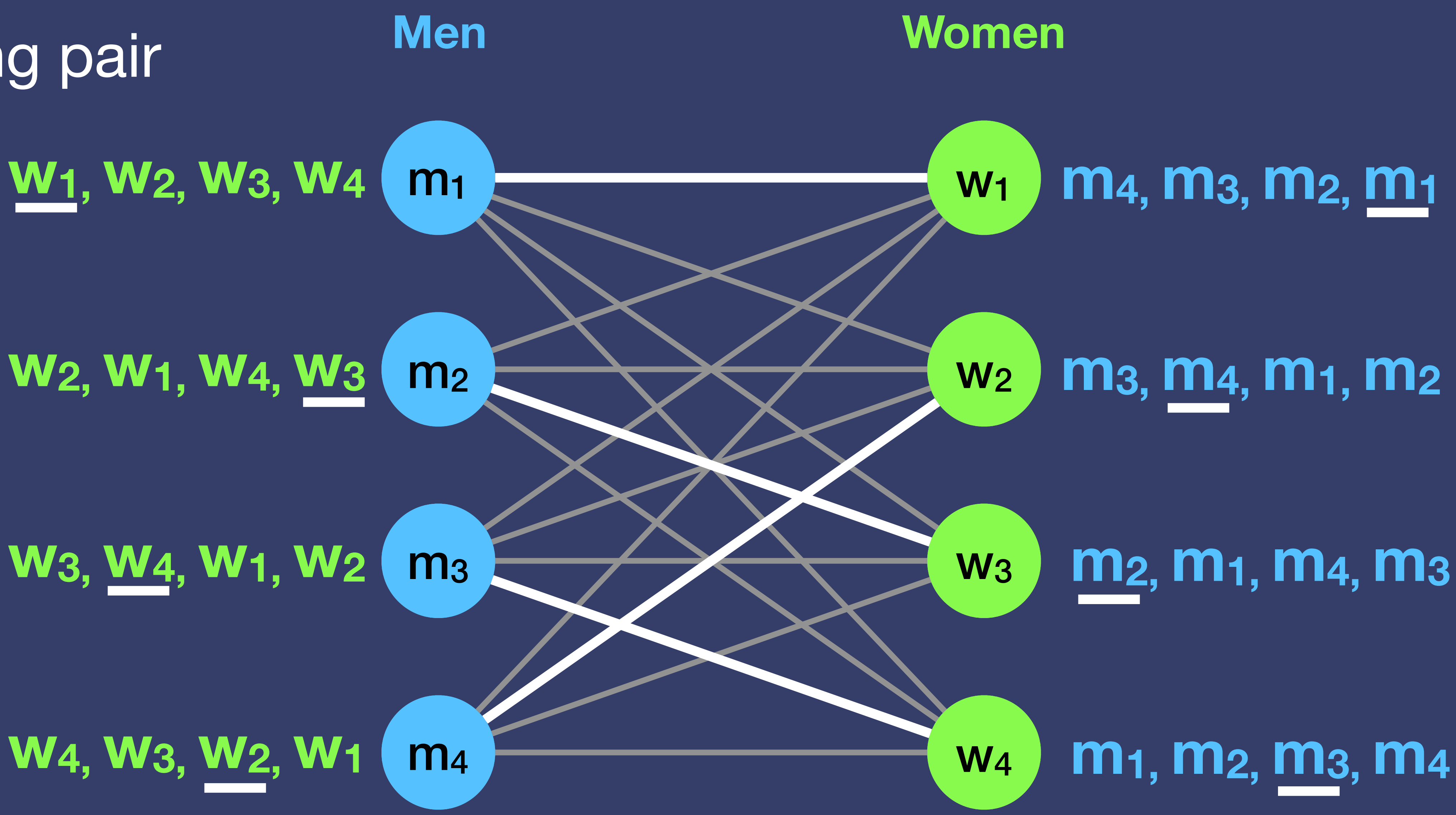
Stable Marriage

Rank

Degree

Profile <_, _, _, _>

Blocking pair



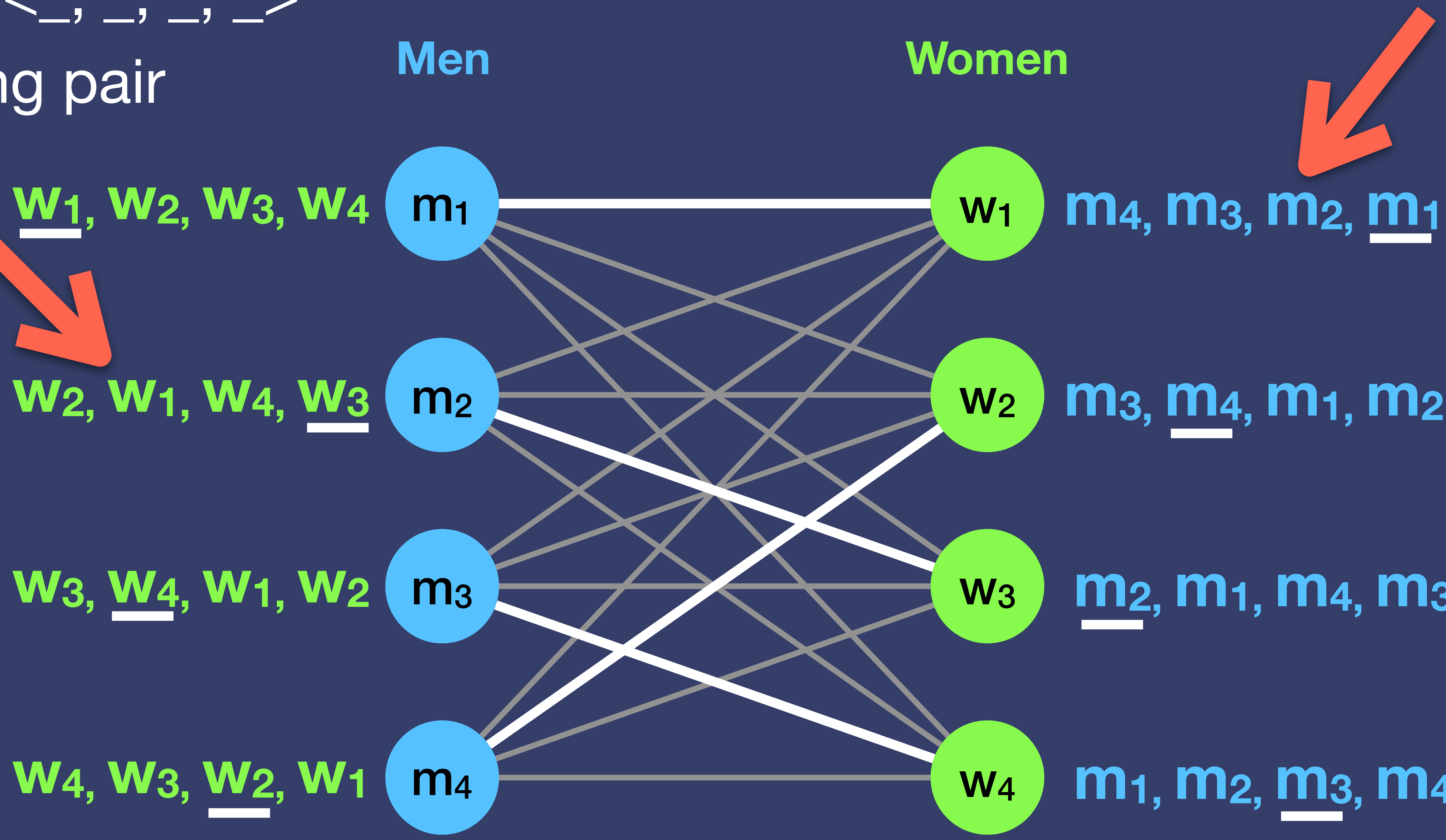
Stable Marriage

Rank

Degree

Profile <_, _, _, _>

Blocking pair



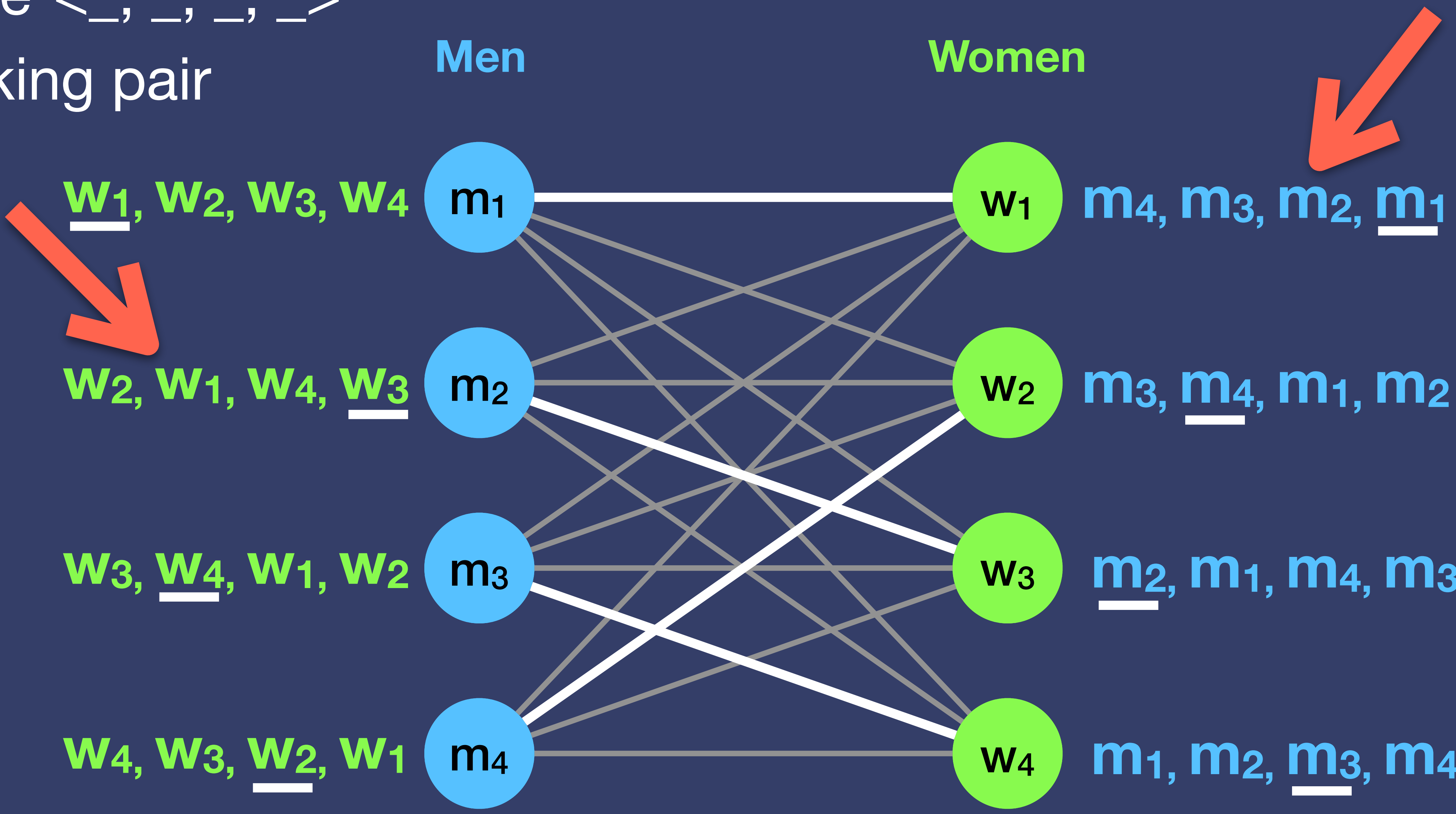
Stable Marriage

Rank

Degree

Profile <_, _, _, _>

Blocking pair



A **stable matching** is a matching with no blocking pairs

Fairness


CORN DOGS
CORN DOGS
SUPER DOG
SUPER DOG
LONG DOG ON A STICK
COLD DRINKS
DOG ON A STICK
CORN DOGS

Fairness

- Many stable matchings per instance

Fairness

- Many stable matchings per instance
- Want to find a stable matching that provides some kind of equality between men and women

Fairness

- Many stable matchings per instance
- Want to find a stable matching that provides some kind of equality between men and women
- Several different fairness measures

Fairness measures

Fairness measures

Sex-equal - minimise the differences between the sum of ranks for men and the sum of ranks for women - NP-hard

Complexity of the Sex-Equal Stable Marriage Problem; Japan Journal of Industrial and Applied Mathematics; 1993; Kato

Fairness measures

Sex-equal - minimise the differences between the sum of ranks for men and the sum of ranks for women - NP-hard

Complexity of the Sex-Equal Stable Marriage Problem; Japan Journal of Industrial and Applied Mathematics; 1993; Kato

Rank-maximal - maximises the number assigned to their first choice, and subject to that, their second, and so on (maximises profile lexicographically) - Polynomial

Fairness measures

Sex-equal - minimise the differences between the sum of ranks for men and the sum of ranks for women - NP-hard

Complexity of the Sex-Equal Stable Marriage Problem; Japan Journal of Industrial and Applied Mathematics; 1993; Kato

Rank-maximal - maximises the number assigned to their first choice, and subject to that, their second, and so on (maximises profile lexicographically) - Polynomial

Generous - minimises the number assigned to their last choice, and subject to that, their second to last, and so on (minimises reverse profile lexicographically) - Polynomial

Fairness measures

Sex-equal - minimise the differences between the sum of ranks for men and the sum of ranks for women - NP-hard

Complexity of the Sex-Equal Stable Marriage Problem; Japan Journal of Industrial and Applied Mathematics; 1993; Kato

→ **Rank-maximal** - maximises the number assigned to their first choice, and subject to that, their second, and so on (maximises profile lexicographically) - Polynomial

→ **Generous** - minimises the number assigned to their last choice, and subject to that, their second to last, and so on (minimises reverse profile lexicographically) - Polynomial

Fairness measures

Fairness measures

Sex-equal

Sum men ranks: 10

Sum women ranks: 10

Sex-equal difference: 0

Profile: <0, 4, 4, 0>

m_1: w_1, w_2, $\underline{w_3}$, w_4

m_2: w_2, $\underline{w_1}$, w_4, w_3

m_3: w_3, $\underline{w_4}$, w_1, w_2

m_4: w_4, w_3, $\underline{w_2}$, w_1

w_1: m_4, m_3, $\underline{m_2}$, m_1

w_2: m_3, $\underline{m_4}$, m_1, m_2

w_3: m_2, $\underline{m_1}$, m_4, m_3

w_4: m_1, m_2, $\underline{m_3}$, m_4

Fairness measures

Sex-equal

Sum men ranks: 10

Sum women ranks: 10

Sex-equal difference: 0

Profile: <0, 4, 4, 0>

m_1: $w_1, w_2, \underline{w_3}, w_4$
m_2: $w_2, \underline{w_1}, w_4, w_3$
m_3: $w_3, \underline{w_4}, w_1, w_2$
m_4: $w_4, w_3, \underline{w_2}, w_1$

w_1: $m_4, m_3, \underline{m_2}, m_1$
w_2: $m_3, \underline{m_4}, m_1, m_2$
w_3: $m_2, \underline{m_1}, m_4, m_3$
w_4: $m_1, m_2, \underline{m_3}, m_4$

Rank-maximal

Sum men ranks: 4

Sum women ranks: 16

Sex-equal difference: 12

Profile: <4, 0, 0, 4>

m_1: $\underline{w_1}, w_2, w_3, w_4$
m_2: $\underline{w_2}, w_1, w_4, w_3$
m_3: $\underline{w_3}, w_4, w_1, w_2$
m_4: $\underline{w_4}, w_3, w_2, w_1$

w_1: $m_4, m_3, m_2, \underline{m_1}$
w_2: $m_3, m_4, m_1, \underline{m_2}$
w_3: $m_2, m_1, m_4, \underline{m_3}$
w_4: $m_1, m_2, m_3, \underline{m_4}$

Fairness measures

Sex-equal

Sum men ranks: 10

Sum women ranks: 10

Sex-equal difference: 0

Profile: <0, 4, 4, 0>

m_1: w_1, w_2, $\underline{w_3}$, w_4

m_2: w_2, $\underline{w_1}$, w_4, w_3

m_3: w_3, $\underline{w_4}$, w_1, w_2

m_4: w_4, w_3, $\underline{w_2}$, w_1

w_1: m_4, m_3, $\underline{m_2}$, m_1

w_2: m_3, $\underline{m_4}$, m_1, m_2

w_3: m_2, $\underline{m_1}$, m_4, m_3

w_4: m_1, m_2, $\underline{m_3}$, m_4

Rank-maximal

Sum men ranks: 4

Sum women ranks: 16

Sex-equal difference: 12

Profile: <4, 0, 0, 4>

m_1: $\underline{w_1}$, w_2, w_3, w_4

m_2: $\underline{w_2}$, w_1, w_4, w_3

m_3: $\underline{w_3}$, w_4, w_1, w_2

m_4: $\underline{w_4}$, w_3, w_2, w_1

w_1: m_4, m_3, m_2, $\underline{m_1}$

w_2: m_3, m_4, m_1, $\underline{m_2}$

w_3: m_2, m_1, m_4, $\underline{m_3}$

w_4: m_1, m_2, m_3, $\underline{m_4}$

Generous

Sum men ranks: 8

Sum women ranks: 12

Sex-equal difference: 4

Profile: <0, 4, 4, 0>

m_1: w_1, $\underline{w_2}$, w_3, w_4

m_2: w_2, $\underline{w_1}$, w_4, w_3

m_3: w_3, $\underline{w_4}$, w_1, w_2

m_4: w_4, $\underline{w_3}$, w_2, w_1

w_1: m_4, m_3, $\underline{m_2}$, m_1

w_2: m_3, m_4, $\underline{m_1}$, m_2

w_3: m_2, m_1, $\underline{m_4}$, m_3

w_4: m_1, m_2, $\underline{m_3}$, m_4

Fairness measures

Sex-equal

Sum men ranks: 10

Sum women ranks: 10

Sex-equal difference: 0

Profile: <0, 4, 4, 0>

also generous

m_1: $w_1, w_2, \underline{w_3}, w_4$
m_2: $w_2, \underline{w_1}, w_4, w_3$
m_3: $w_3, \underline{w_4}, w_1, w_2$
m_4: $w_4, w_3, \underline{w_2}, w_1$

w_1: $m_4, m_3, \underline{m_2}, m_1$
w_2: $m_3, \underline{m_4}, m_1, m_2$
w_3: $m_2, \underline{m_1}, m_4, m_3$
w_4: $m_1, m_2, \underline{m_3}, m_4$

Rank-maximal

Sum men ranks: 4

Sum women ranks: 16

Sex-equal difference: 12

Profile: <4, 0, 0, 4>

m_1: $\underline{w_1}, w_2, w_3, w_4$
m_2: $\underline{w_2}, w_1, w_4, w_3$
m_3: $\underline{w_3}, w_4, w_1, w_2$
m_4: $\underline{w_4}, w_3, w_2, w_1$

w_1: $m_4, m_3, m_2, \underline{m_1}$
w_2: $m_3, m_4, m_1, \underline{m_2}$
w_3: $m_2, m_1, m_4, \underline{m_3}$
w_4: $m_1, m_2, m_3, \underline{m_4}$

Generous

Sum men ranks: 8

Sum women ranks: 12

Sex-equal difference: 4

Profile: <0, 4, 4, 0>

m_1: $w_1, \underline{w_2}, w_3, w_4$
m_2: $w_2, \underline{w_1}, w_4, w_3$
m_3: $w_3, \underline{w_4}, w_1, w_2$
m_4: $w_4, \underline{w_3}, w_2, w_1$

w_1: $m_4, m_3, \underline{m_2}, m_1$
w_2: $m_3, m_4, \underline{m_1}, m_2$
w_3: $m_2, m_1, \underline{m_4}, m_3$
w_4: $m_1, m_2, \underline{m_3}, m_4$

Finding a Rank-maximal stable matching

Using $O(n^5 \log n)$ Irving, Gusfield and Leather Approach

An efficient algorithm for the "optimal" stable marriage; Journal of the ACM; 1987; Irving, Gusfield, Leather

Rotations

Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

R_1 $\begin{matrix} m_1 & m_4 \\ w_2 & w_3 \end{matrix}$

Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

R_1	m_1	m_4
	w_2	w_3

M_1	m_1	m_2	m_3	m_4
	w_2	w_1	w_4	w_3

Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

R_1: m_1 m_4 / w_2 w_3

M_1: m_1 m_2 m_3 m_4 / w_2 w_1 w_4 w_3

↓ R_1

Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

R_1

m_1	m_4
w_2	w_3

M_1

m_1	m_2	m_3	m_4
w_2	w_1	w_4	w_3

↓ R_1

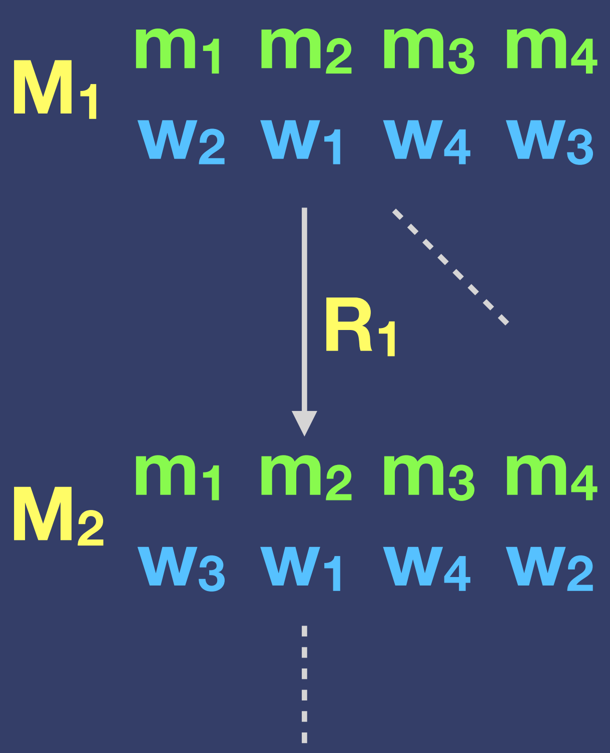
M_2

m_1	m_2	m_3	m_4
w_3	w_1	w_4	w_2

Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

R_1	m_1	m_4
	w_2	w_3

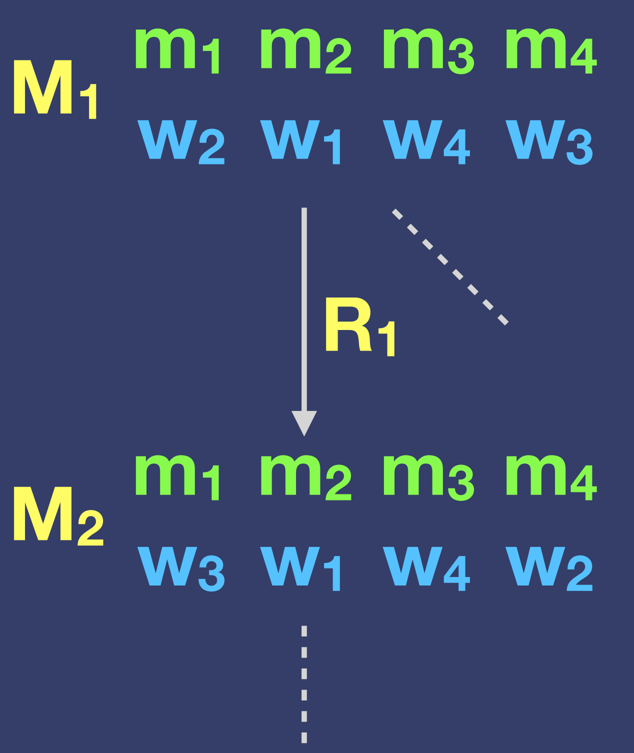


Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

R_1	m_1	m_4
	w_2	w_3

- Can only eliminate *exposed* rotations



Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

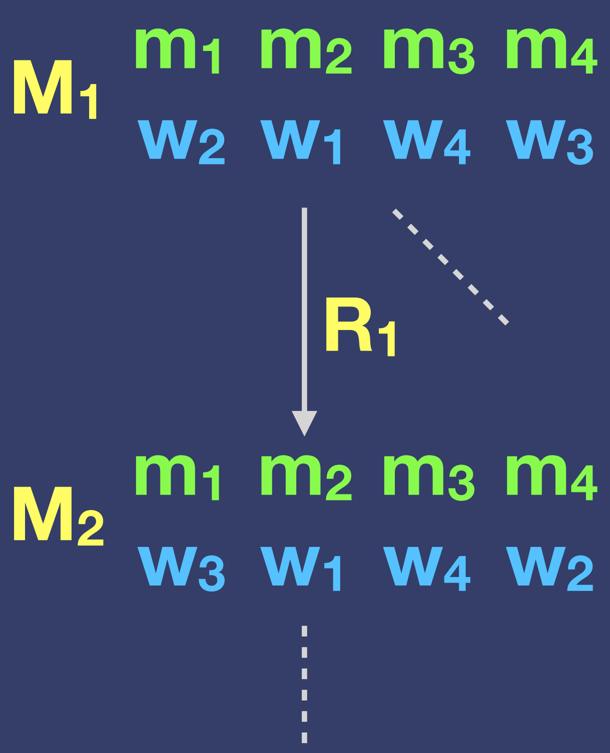
R_1

m_1	m_4
w_2	w_3

- Can only eliminate *exposed* rotations

R_2

m_1	m_2
w_1	w_2



Rotations

- Rotation - series of man-woman pairs that take us from one stable matching to another when permuted

R_1: m_1 m_4 / w_2 w_3

- Can only eliminate *exposed* rotations

R_2: m_1 m_2 / w_1 w_2

- $O(n^2)$ algorithm to find all rotations

Rotation weights

Rotation weights

Rotation profile

Rotation weights

Rotation profile

- Profile of a rotation is the change in profile of the matching if it is eliminated.

Rotation weights

Rotation profile

- Profile of a rotation is the change in profile of the matching if it is eliminated.
- $p = \langle p_1, p_2, \ldots p_n \rangle$

E.g. $p = \langle 2, -1, 0, -1 \rangle$

Rotation weights

Rotation profile

- Profile of a rotation is the change in profile of the matching if it is eliminated.

- $p = \langle p_1, p_2, \ldots p_n \rangle$ E.g. $p = \langle 2, -1, 0, -1 \rangle$

Rotation weight

Rotation weights

Rotation profile

- Profile of a rotation is the change in profile of the matching if it is eliminated.
- $p = \langle p_1, p_2, \ldots p_n \rangle$ E.g. $p = \langle 2, -1, 0, -1 \rangle$

Rotation weight

- convert rotation profiles to a single exponential number

Rotation weights

Rotation profile

- Profile of a rotation is the change in profile of the matching if it is eliminated.
- $p = \langle p_1, p_2, \ldots p_n \rangle$ E.g. $p = \langle 2, -1, 0, -1 \rangle$

Rotation weight

- convert rotation profiles to a single exponential number
- $w(p) = p_1 * n^{n-1} + p_2 * n^{n-2} + \ldots + p_n$ E.g. $w(p) = 111$

Frances Cooper

Rotation weights

Rotation profile

- Profile of a rotation is the change in profile of the matching if it is eliminated.
- $p = \langle p_1, p_2, \ldots p_n \rangle$ E.g. $p = \langle 2, -1, 0, -1 \rangle$

Rotation weight

- convert rotation profiles to a single exponential number
- $w(p) = p_1 * n^{n-1} + p_2 * n^{n-2} + \ldots + p_n$ E.g. $w(p) = 111$

If a rotation has a positive weight then we want to eliminate it if possible as it helps us find a rank-maximal matching.

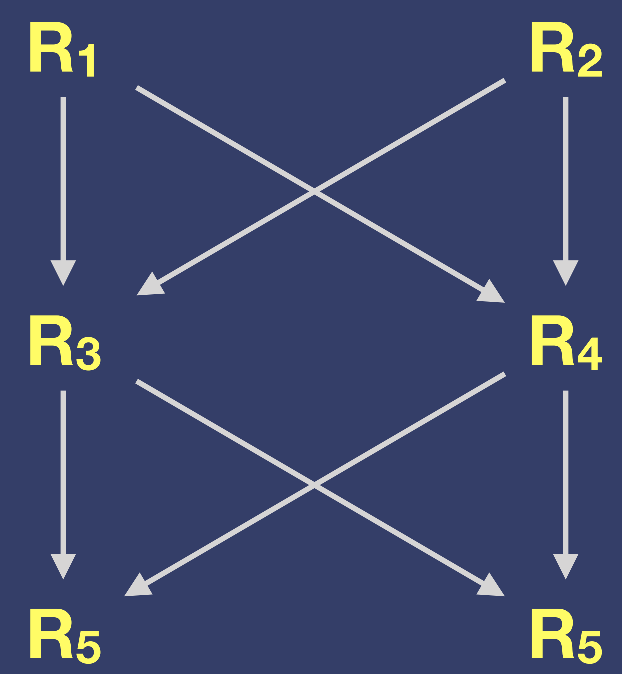
Rotation poset

Rotation poset

- Displays order in which rotations can be eliminated

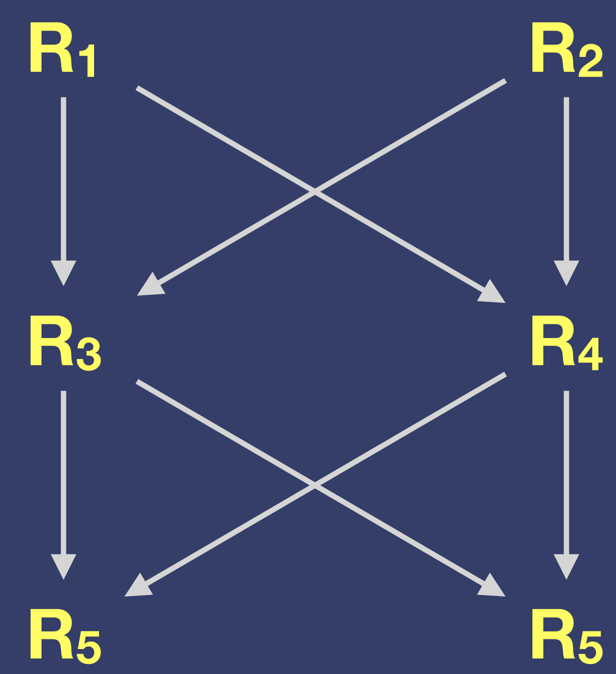
Rotation poset

- Displays order in which rotations can be eliminated



Rotation poset

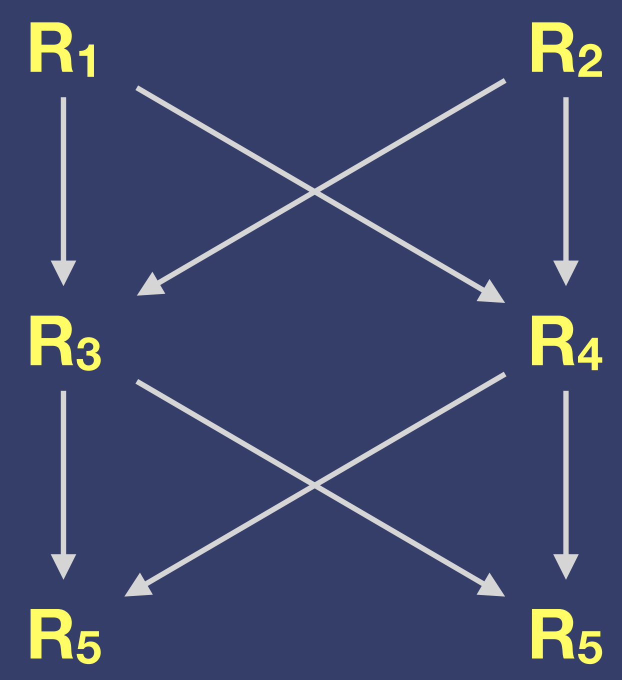
- Displays order in which rotations can be eliminated



- The set of stable matchings are in 1-1 correspondence with the closed subsets of the poset

Rotation poset

- Displays order in which rotations can be eliminated



- The set of stable matchings are in 1-1 correspondence with the closed subsets of the poset

- Want: Max weight closed subset of the rotation poset

Flow Network

Flow Network

- Build a flow network based on the rotation poset

Flow Network

- Build a flow network based on the rotation poset
- The max flow relates to the max closed subset

Flow Network

- Build a flow network based on the rotation poset
- The max flow relates to the max closed subset
- Use a min-cut max-flow algorithm to find max flow

Flow Network

- Build a flow network based on the rotation poset
- The max flow relates to the max closed subset
- Use a min-cut max-flow algorithm to find max flow
    - Sleator-Tarjan max-flow algorithm O(n^5 logn)

can be improved -
we'll look at this later

Steps

Steps

1. List rotations

Steps

1. List rotations
2. Build rotation poset

Steps

1. List rotations
2. Build rotation poset
3. Build a flow network

Steps

1. List rotations

2. Build rotation poset

3. Build a flow network

4. Find a minimum cut

Steps

1. List rotations
2. Build rotation poset
3. Build a flow network
4. Find a minimum cut
5. Maximum closed subset of the rotation poset

Steps

1. List rotations
2. Build rotation poset
3. Build a flow network
4. Find a minimum cut
5. Maximum closed subset of the rotation poset

$O(n^5 \log n)$ using Irving, Gusfield and Leather approach

An efficient algorithm for the "optimal" stable marriage; Journal of the ACM; 1987; Irving, Gusfield, Leather

Problem

Problem

Weights are exponentially large:

Problem

Weights are exponentially large:

- Calculations may cause overflow / inaccuracies for primitive types

Problem

Weights are exponentially large:

- Calculations may cause overflow / inaccuracies for primitive types
- Memory issues for types that can store arbitrarily large numbers

Rank-maximal stable matching using a vector-based approach

Combinatorial approach

Combinatorial approach

- We present a vector-based combinatorial approach - no need to use exponential weights

Combinatorial approach

- We present a vector-based combinatorial approach - no need to use exponential weights
- Why does this help?

Combinatorial approach

- We present a vector-based combinatorial approach - no need to use exponential weights
- Why does this help?
    - distribution of rotation profiles are non uniform

Combinatorial approach

- We present a vector-based combinatorial approach - no need to use exponential weights
- Why does this help?
    - distribution of rotation profiles are non uniform
    - vectors can be *compressed*

Combinatorial approach

- We present a vector-based combinatorial approach - no need to use exponential weights
- Why does this help?
  - distribution of rotation profiles are non uniform
  - vectors can be *compressed*

  <1, 4, 0, 0, -2, 0, 0, 0, 0, 0, 0, -3, 0, 0, 0, 0 >

Combinatorial approach

- We present a vector-based combinatorial approach - no need to use exponential weights
- Why does this help?
    - distribution of rotation profiles are non uniform
    - vectors can be *compressed*

<1, 4, 0, 0, -2, 0, 0, 0, 0, 0, 0, -3, 0, 0, 0, 0 >

Combinatorial approach

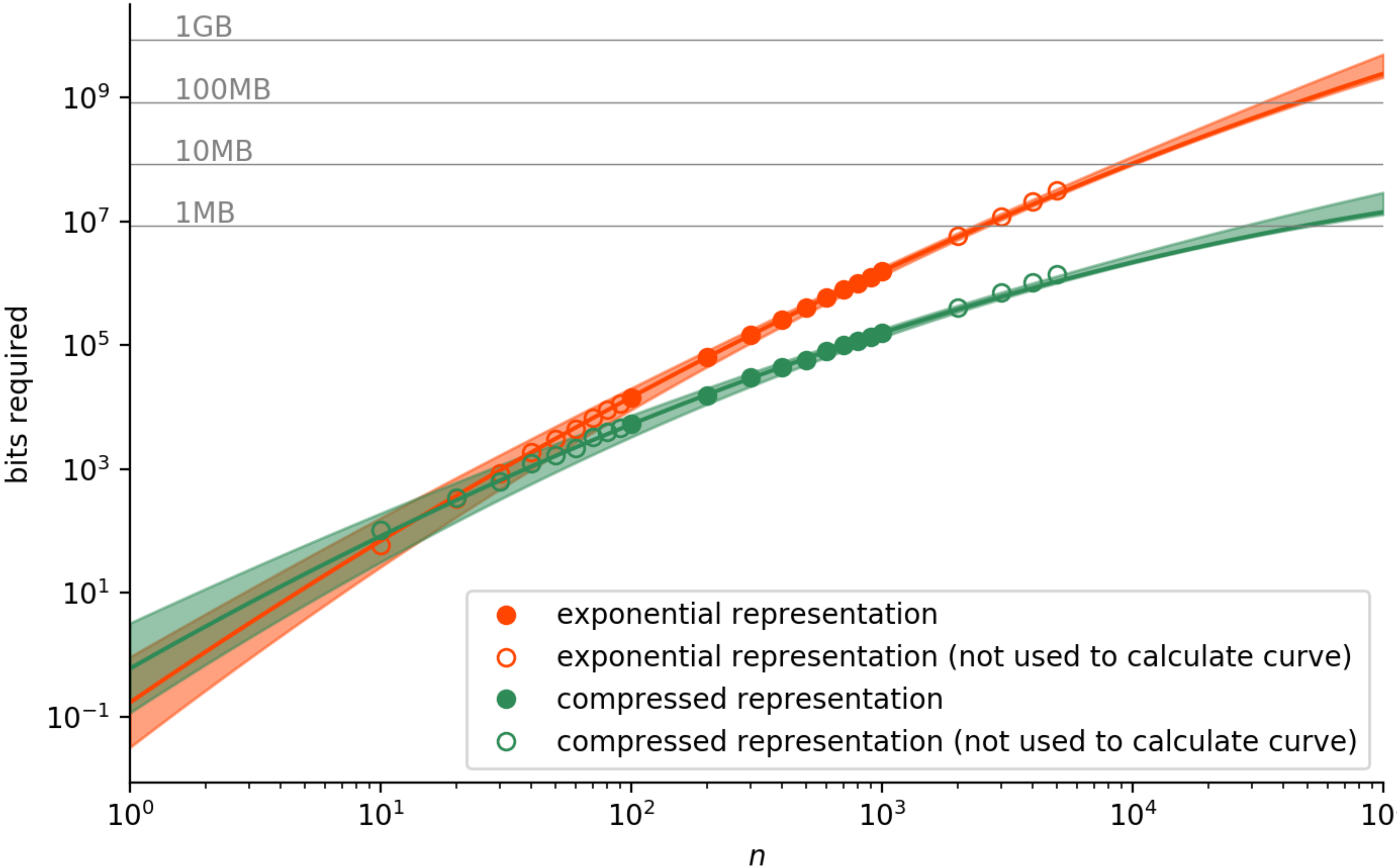
- We present a vector-based combinatorial approach - no need to use exponential weights
- Why does this help?
  - distribution of rotation profiles are non uniform
  - vectors can be *compressed*

<1, 4, 0, 0, -2, 0, 0, 0, 0, 0, 0, -3, 0, 0, 0, 0 >

save the index and value of non zero elements (lossless)

<(0,1), (1,4), (4,-2), (11,-3)>

Combinatorial approach



Steps

1. List rotations
2. Build rotation poset
3. Build a flow network
4. Find the minimum cut
5. Maximum closed subset of the rotation poset

Steps

1. List rotations
2. Build rotation poset
3. Build a flow network
4. Find the minimum cut
5. Maximum closed subset of the rotation poset

Vector-based approach

Vector-based approach

- Flow network

Vector-based approach

- Flow network
    - Vector-based capacities and flows

Vector-based approach

- Flow network
    - Vector-based capacities and flows
    - E.g. $<p_1, p_2, \ldots p_n>$

Vector-based approach

- Flow network
  - Vector-based capacities and flows
  - E.g. <p_1, p_2, … p_n>
  - We don’t convert to an exponential

Vector-based approach

- Flow network
    - Vector-based capacities and flows
    - E.g. $<p_1, p_2, \ldots p_n>$
    - We don’t convert to an exponential
- adds complications

Vector-based approach

- Flow network
    - Vector-based capacities and flows
    - E.g. $<p_1, p_2, \ldots p_n>$
    - We don't convert to an exponential
- adds complications
- Had to define our own arithmetic over these vectors

Proofs

Proofs

1. Eliminating the max *profile* closed subset of the rotation poset finds us the rank-maximal stable matching

Proofs

1. Eliminating the max *profile* closed subset of the rotation poset finds us the rank-maximal stable matching

2. Correspondence between high-weight and vector-based flows and capacities

Proofs

1. Eliminating the max *profile* closed subset of the rotation poset finds us the rank-maximal stable matching

2. Correspondence between high-weight and vector-based flows and capacities

3. Proved Max Flow-Min Cut Theorem extends to vector-based setting

Proofs

1. Eliminating the max *profile* closed subset of the rotation poset finds us the rank-maximal stable matching

2. Correspondence between high-weight and vector-based flows and capacities

3. Proved Max Flow-Min Cut Theorem extends to vector-based setting

4. Proved could adapt the Sleator-Tarjan Max-Flow algorithm to use vector capacities and flows

Proofs

1. Eliminating the max *profile* closed subset of the rotation poset finds us the rank-maximal stable matching
2. Correspondence between high-weight and vector-based flows and capacities
3. Proved Max Flow-Min Cut Theorem extends to vector-based setting
4. Proved could adapt the Sleator-Tarjan Max-Flow algorithm to use vector capacities and flows

★ Can find a rank-maximal stable matching in $O(n^5 \log n)$ using vectors - matches the exponential approach (but with added bonus of vector compression) ★

Generous stable matching

Generous stable matching

- We want to minimise the reverse profile

Generous stable matching

- We want to minimise the reverse profile
    - Same as maximising the reverse profile where each element is negated!

Generous stable matching

- We want to minimise the reverse profile
  - Same as maximising the reverse profile where each element is negated!
  - If $p = \langle p_1, p_2, \dots p_n \rangle$, then $p' = \langle -p_n, -p_{n-1}, \dots, -p_1 \rangle$

Three Fast Algorithms for Four Problems in Stable Marriage; Siam Journal of Computing; 1987; Gusfield

Generous stable matching

- We want to minimise the reverse profile
  - Same as maximising the reverse profile where each element is negated!
  - If $p = \langle p_1, p_2, \ldots p_n \rangle$, then $p' = \langle -p_n, -p_{n-1}, \ldots, -p_1 \rangle$
- **Minimum-regret stable matching** - minimises degree of the matching - $O(n^2)$

Three Fast Algorithms for Four Problems in Stable Marriage; Siam Journal of Computing; 1987; Gusfield

Generous stable matching

- We want to minimise the reverse profile
  - Same as maximising the reverse profile where each element is negated!
  - If $p = <p_1, p_2, \dots p_n>$, then $p' = <-p_n, -p_{n-1}, \dots, -p_1>$
- **Minimum-regret stable matching** - minimises degree of the matching - $O(n^2)$

★ Generous stable matching can be found in $O(n^2d^3 \log n)$ time - competitive when d is small ★

d is degree of the minimum-regret stable matching

Experimental results

Methodology

Methodology

- Instances size {10, 20, …, 100, 200, …, 1000}, complete preference lists, 1000 instance per size.

Methodology

- Instances size {10, 20, …, 100, 200, …, 1000}, complete preference lists, 1000 instance per size.
- looked at properties over several types of optimal stable matching (rank-maximal, generous, median, egalitarian and sex-equal)

Methodology

- Instances size {10, 20, …, 100, 200, …, 1000}, complete preference lists, 1000 instance per size.
- looked at properties over several types of optimal stable matching (rank-maximal, generous, median, egalitarian and sex-equal)
- GS algorithm twice + digraph → all stable matchings

Methodology

- Instances size {10, 20, …, 100, 200, …, 1000}, complete preference lists, 1000 instance per size.
- looked at properties over several types of optimal stable matching (rank-maximal, generous, median, egalitarian and sex-equal)
- GS algorithm twice + digraph → all stable matchings
- Java, Python, Bash, GNU parallel

Methodology

- Instances size {10, 20, …, 100, 200, …, 1000}, complete preference lists, 1000 instance per size.
- looked at properties over several types of optimal stable matching (rank-maximal, generous, median, egalitarian and sex-equal)
- GS algorithm twice + digraph → all stable matchings
- Java, Python, Bash, GNU parallel
- Correctness

Methodology

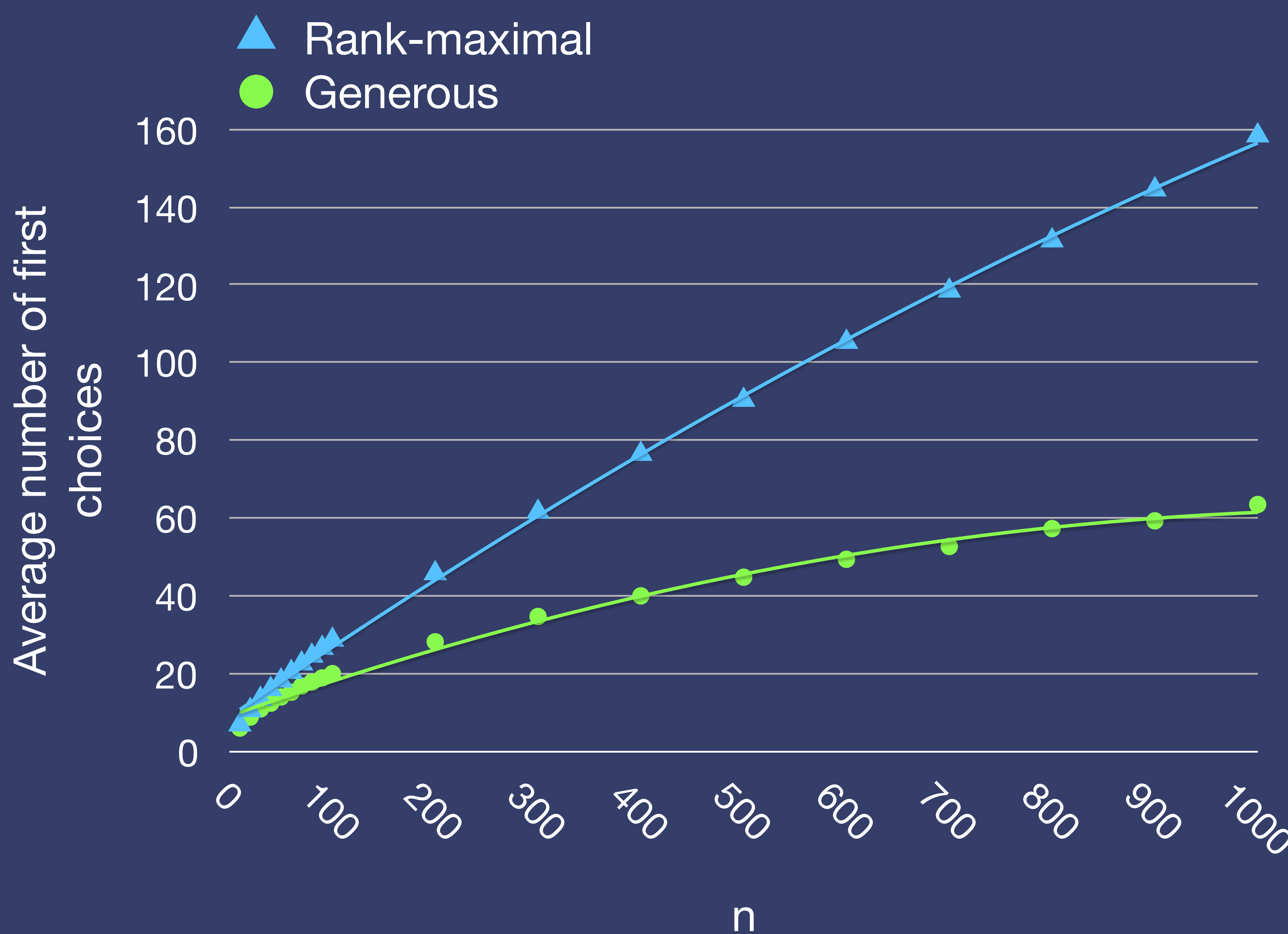
- Instances size {10, 20, …, 100, 200, …, 1000}, complete preference lists, 1000 instance per size.
- looked at properties over several types of optimal stable matching (rank-maximal, generous, median, egalitarian and sex-equal)
- GS algorithm twice + digraph → all stable matchings
- Java, Python, Bash, GNU parallel
- Correctness
    - all stable matchings found were stable

Methodology

- Instances size {10, 20, …, 100, 200, …, 1000}, complete preference lists, 1000 instance per size.
- looked at properties over several types of optimal stable matching (rank-maximal, generous, median, egalitarian and sex-equal)
- GS algorithm twice + digraph → all stable matchings
- Java, Python, Bash, GNU parallel
- Correctness
    - all stable matchings found were stable
    - CPLEX up to size n = 60 for the number of stable matchings

Average number of first choices

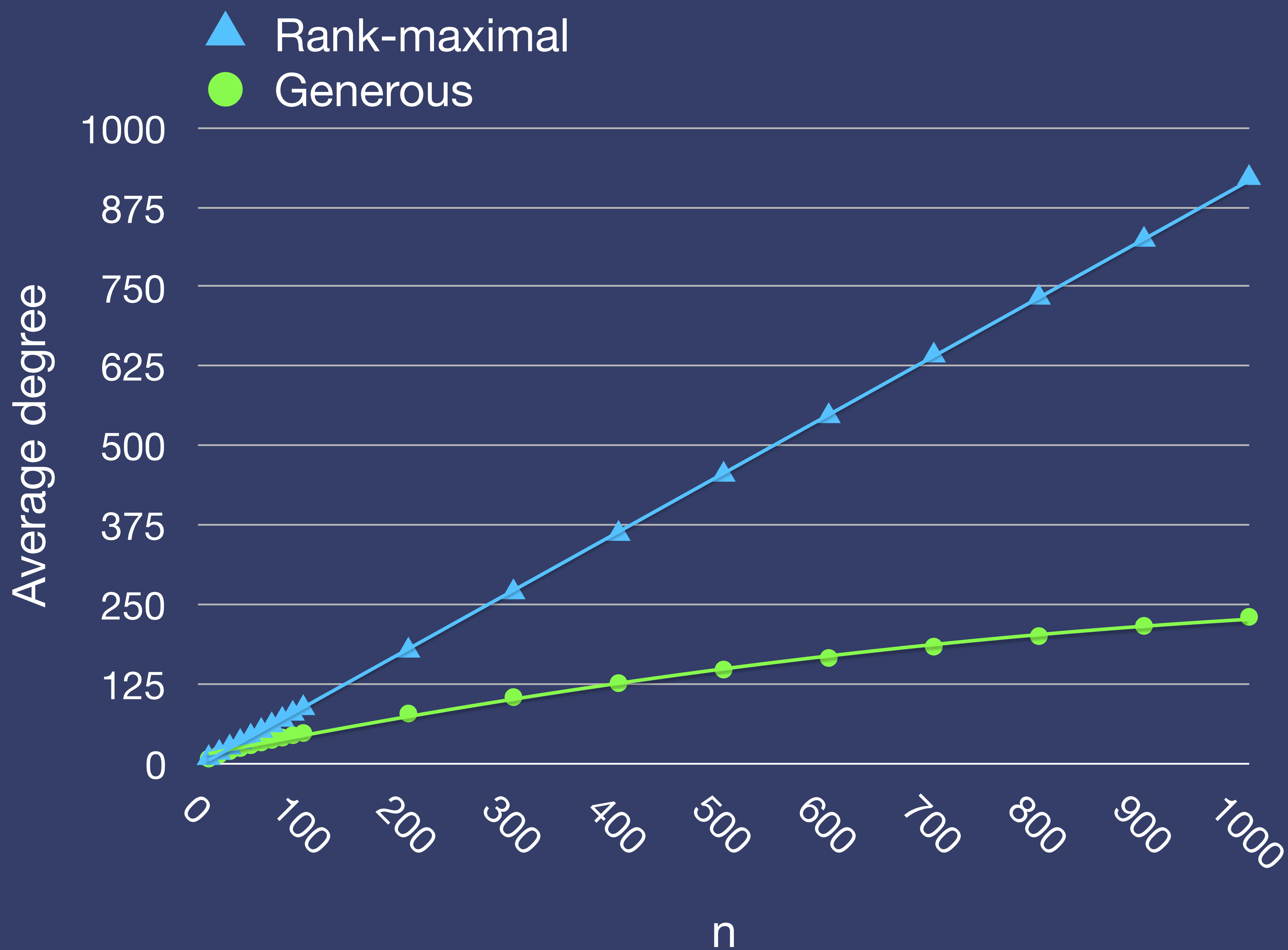
Average number of first choices



Frances Cooper

Average degree

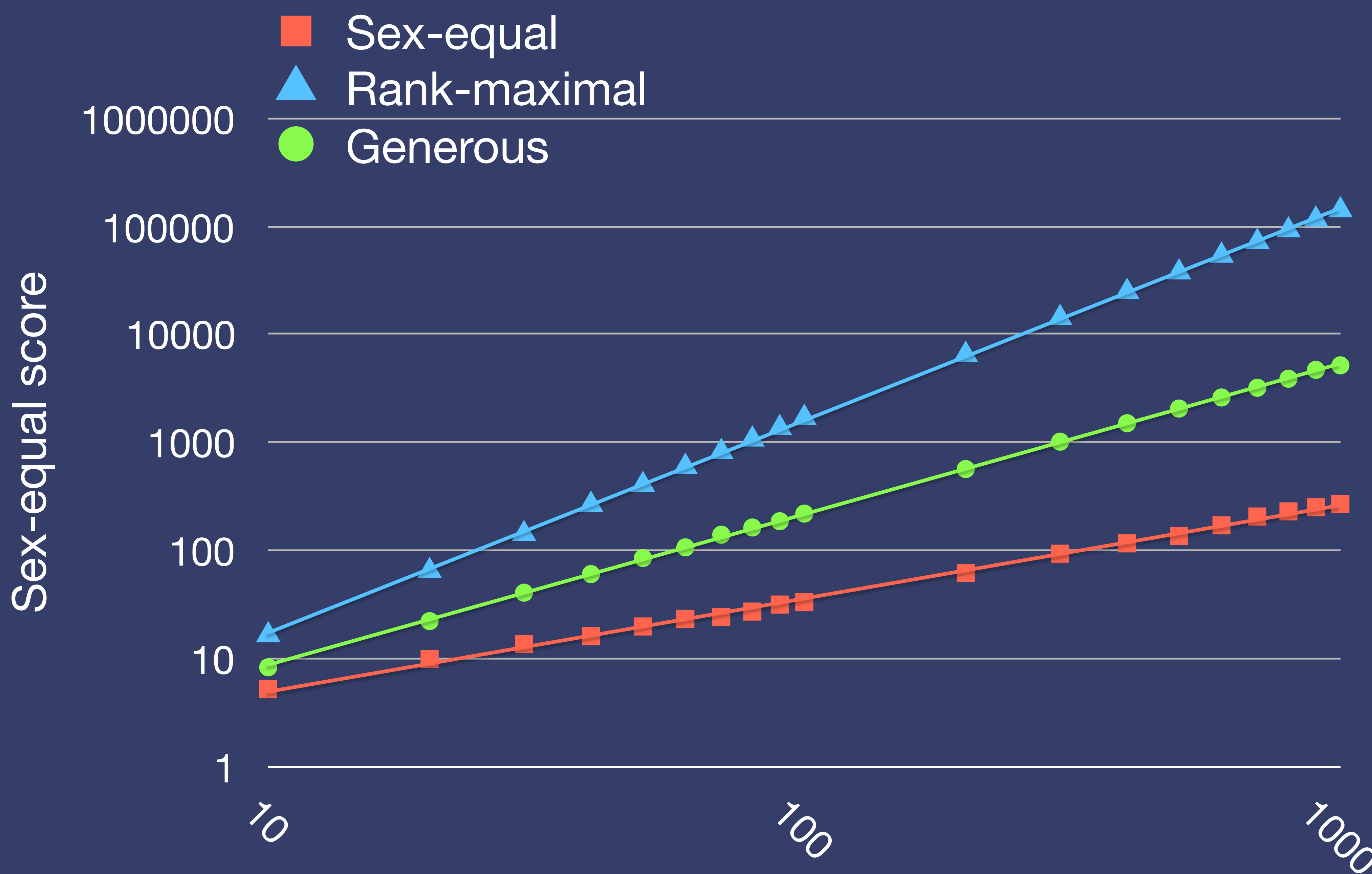
Average degree



Frances Cooper

Sex-equal score

Sex-equal score



Frances Cooper

Future Work

Future Work

- Adapt Orlin's Max Flow algorithm

Max Flows in O(nm) Time, or Better; Association for Computing Machinery; 2013; Orlin

Future Work

- Adapt Orlin's Max Flow algorithm

Max Flows in O(nm) Time, or Better; Association for Computing Machinery; 2013; Orlin

  - Would get $O(n^5)$

improvement automatically available in the exponential case

Future Work

- Adapt Orlin's Max Flow algorithm
    - Would get $O(n^5)$

Max Flows in O(nm) Time, or Better; Association for Computing Machinery; 2013; Orlin

improvement automatically available in the exponential case

- Adapt Feder's technique

A new fixed point approach for stable networks and stable marriages; Journal of Computer and System Sciences; 1992; Feder

Future Work

- Adapt Orlin's Max Flow algorithm
  - Would get $O(n^5)$

Max Flows in O(nm) Time, or Better; Association for Computing Machinery; 2013; Orlin

improvement automatically available in the exponential case

- Adapt Feder's technique
  - Based on weighted SAT

A new fixed point approach for stable networks and stable marriages; Journal of Computer and System Sciences; 1992; Feder

Future Work

- Adapt Orlin's Max Flow algorithm
  - Would get $O(n^5)$

Max Flows in O(nm) Time, or Better; Association for Computing Machinery; 2013; Orlin

improvement automatically available in the exponential case

- Adapt Feder's technique
  - Based on weighted SAT
  - Would get $O(n^{4.5})$

A new fixed point approach for stable networks and stable marriages; Journal of Computer and System Sciences; 1992; Feder

Thank you

Summary

- Matching problems
- Fairness
- Finding fair stable matchings
- Experiments
- Future work: adapting algorithms to vector-based setting for improved time complexity

University of Glasgow

f.cooper.1@research.gla.ac.uk
http://fmcooper.github.io

EPSRC

Engineering and Physical Sciences Research Council

EPSRC Doctoral Training Account