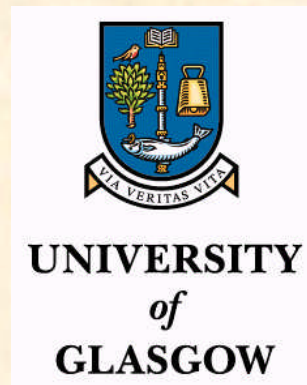


Matching under Preferences: Results Old and New

Rob Irving

Computing Science Department
University of Glasgow



Matching applicants to employers – an example

$a_1 : h_1 h_2 h_5$
 $a_2 : h_1 h_2 h_3$
 $a_3 : h_2 h_4 h_5$
 $a_4 : h_3 h_1 h_2$
 $a_5 : h_4 h_5 h_1$
 $a_6 : h_1 h_5 h_3$
 $a_7 : h_2 h_3 h_4$
 $a_8 : h_2 h_1 h_5$
 $a_9 : h_3 h_4 h_1$
 $a_{10} : h_1 h_3 h_2$

$h_1 : 2 : a_9 a_8 a_6 a_4 a_1 a_5 a_{10} a_2$
 $h_2 : 2 : a_4 a_3 a_8 a_{10} a_2 a_1 a_7$
 $h_3 : 2 : a_6 a_7 a_9 a_4 a_2 a_{10}$
 $h_4 : 2 : a_7 a_3 a_5 a_9$
 $h_5 : 2 : a_6 a_3 a_1 a_8 a_5$

applicants' preferences

employers' capacities and preferences

- Question: how should we match applicants to employers to best take account of the preferences?
 - what kind of optimality properties may be appropriate?

Matching applicants to employers – an example

a_1 : h_1 h_2 h_5
 a_2 : h_1 h_2 h_3
 a_3 : h_2 h_4 h_5
 a_4 : h_3 h_1 h_2
 a_5 : h_4 h_5 h_1
 a_6 : h_1 h_5 h_3
 a_7 : h_2 h_3 h_4
 a_8 : h_2 h_1 h_5
 a_9 : h_3 h_4 h_1
 a_{10} : h_1 h_3 h_2

h_1 : 2 : a_9 a_8 a_6 a_4 a_1 a_5 a_{10} a_2
 h_2 : 2 : a_4 a_3 a_8 a_{10} a_2 a_1 a_7
 h_3 : 2 : a_6 a_7 a_9 a_4 a_2 a_{10}
 h_4 : 2 : a_7 a_3 a_5 a_9
 h_5 : 2 : a_6 a_3 a_1 a_8 a_5

A maximum cardinality matching (size 10)

- all applicants are matched - but few participants are happy
- in particular there are *blocking pairs* – e.g. (a_6, h_1) (and others)
 - a_6 prefers h_1 and h_1 prefers a_6 (to at least one of its assignees)
 - a_6 and h_1 could come to a mutually beneficial private deal, and so disrupt the matching
- so the matching is *unstable*

Matching applicants to employers – an example

$a_1 : h_1 h_2 h_5$
 $a_2 : h_1 h_2 h_3$
 $a_3 : h_2 h_4 h_5$
 $a_4 : h_3 h_1 h_2$
 $a_5 : h_4 h_5 h_1$
 $a_6 : h_1 h_5 h_3$
 $a_7 : h_2 h_3 h_4$
 $a_8 : h_2 h_1 h_5$
 $a_9 : h_3 h_4 h_1$
 $a_{10} : h_1 h_3 h_2$

$h_1 : 2 : a_9 a_8 a_6 a_4 a_1 a_5 a_{10} a_2$
 $h_2 : 2 : a_4 a_3 a_8 a_{10} a_2 a_1 a_7$
 $h_3 : 2 : a_6 a_7 a_9 a_4 a_2 a_{10}$
 $h_4 : 2 : a_7 a_3 a_5 a_9$
 $h_5 : 2 : a_6 a_3 a_1 a_8 a_5$

a stable matching (size 8)

- there are no blocking pairs
- but a_2 and a_{10} are unmatched, h_4 and h_5 are undersubscribed
- all the same, stability is recognised as a key property for the success and acceptance of real-world centralised matching schemes

Matching applicants to employers – an example

$a_1 : h_1 h_2 h_5$
 $a_2 : h_1 h_2 h_3$
 $a_3 : h_2 h_4 h_5$
 $a_4 : h_3 h_1 h_2$
 $a_5 : h_4 h_5 h_1$
 $a_6 : h_1 h_5 h_3$
 $a_7 : h_2 h_3 h_4$
 $a_8 : h_2 h_1 h_5$
 $a_9 : h_3 h_4 h_1$
 $a_{10} : h_1 h_3 h_2$

$h_1 : 2 : a_9 a_8 a_6 a_4 a_1 a_5 a_{10} a_2$
 $h_2 : 2 : a_4 a_3 a_8 a_{10} a_2 a_1 a_7$
 $h_3 : 2 : a_6 a_7 a_9 a_4 a_2 a_{10}$
 $h_4 : 2 : a_7 a_3 a_5 a_9$
 $h_5 : 2 : a_6 a_3 a_1 a_8 a_5$

- a second stable matching (again size 8)
- again a_2 and a_{10} are unmatched, and h_4 and h_5 are undersubscribed

Classical stable matching problems

- The *Stable Marriage* (SM) problem - the one-to-one case
- The *Hospitals/Residents* (HR) problem - the many-to-one case
 - so called because of one its main areas of application
- There is always at least one stable matching, and there is a linear time algorithm to find such a matching – the *Gale-Shapley algorithm*
[Gale & Shapley 1962]
- All stable matchings have the same size, match exactly the same applicants, and fill the same number of places with each employer
[Gale & Sotomayor 1985, Roth 1986]
- The stable matchings form a *distributive lattice* under a natural partial order relation
[Knuth, Conway 1976]
- Top and bottom of this lattice are the *applicant-optimal* and *employer-optimal* stable matchings
 - the Gale-Shapley algorithm can be used to find both of these

Exploiting the lattice representation (for SM)

- a compact representation of the lattice can be constructed in *linear* time (whereas the lattice can take exponential time)

[Irving & Leather 1986]

- the *stable pairs* can be determined in *linear* time
 - the pairs that appear in some stable matching

[Gusfield 1987]

- *all the stable matchings* can be generated in *linear* time per matching

[Gusfield 1987]

- an *egalitarian* stable matching can be found in *quadratic* time

[Irving, Leather & Gusfield 1987]

- egalitarian is a form of 'balanced' optimality
- in $O(n^{3/2})$ time (n is the sum of the lengths of the preference lists)

[Feder 1994]

Applications of stable matching

- **The National Resident Matching Program (NRMP) in the US**
 - matches around 30000 medical graduates to hospitals annually
 - operational since 1952
 - uses a variant of the Gale-Shapley algorithm
- **Similar medical matching schemes in other countries**
 - including the Scottish Foundation Allocation Scheme (SFAS)
- **Matching of pupils to schools in many places**
 - Singapore
 - US cities such as New York and Boston
 - many English authorities
- **Matching of students to colleges and universities in many countries**
 - Spain, Hungary, Norway, Turkey . . .

Some more challenging problems – *Exchange stability*

a_1 : h_1 h_2 h_5
 a_2 : h_1 h_2 h_3
 a_3 : h_2 h_4 h_5
 a_4 : h_3 h_1 h_2
 a_5 : h_4 h_5 h_1
 a_6 : h_1 h_5 h_3
 a_7 : h_2 h_3 h_4
 a_8 : h_2 h_1 h_5
 a_9 : h_3 h_4 h_1
 a_{10} : h_1 h_3 h_2

h_1 : 2 : a_9 a_8 a_6 a_4 a_1 a_5 a_{10} a_2
 h_2 : 2 : a_4 a_3 a_8 a_{10} a_2 a_1 a_7
 h_3 : 2 : a_6 a_7 a_9 a_4 a_2 a_{10}
 h_4 : 2 : a_7 a_3 a_5 a_9
 h_5 : 2 : a_6 a_3 a_1 a_8 a_5

this stable matching (size 8) is not *exchange stable*
 for example, a_4 and a_7 would prefer to exchange
 their allocations

- A stable matching that is also exchange stable may not exist
- Determining the existence of a stable matching that is also exchange stable is NP-complete

- polynomial-time solvable if the applicants' lists are of length ≤ 3 [Irving 2005]
- but NP-complete if applicants' lists can be of length 4 [Irving 2005]

[McDermid et al 2007]

Some more challenging problems – *Size versus Stability*

a_1 : h_1 h_2 h_5
 a_2 : h_1 h_2 h_3
 a_3 : h_2 h_4 h_5
 a_4 : h_3 h_1 h_2
 a_5 : h_4 h_5 h_1
 a_6 : h_1 h_5 h_3
 a_7 : h_2 h_3 h_4
 a_8 : h_2 h_1 h_5
 a_9 : h_3 h_4 h_1
 a_{10} : h_1 h_3 h_2

h_1 : 2 : a_9 a_8 a_6 a_4 a_1 a_5 a_{10} a_2
 h_2 : 2 : a_4 a_3 a_8 a_{10} a_2 a_1 a_7
 h_3 : 2 : a_6 a_7 a_9 a_4 a_2 a_{10}
 h_4 : 2 : a_7 a_3 a_5 a_9
 h_5 : 2 : a_6 a_3 a_1 a_8 a_5

A maximum cardinality matching (size 10)

- This maximum matching has a total of 9 blocking pairs
- Is this best possible for a maximum cardinality matching?
- Finding a max cardinality matching with fewest blocking pairs is NP-hard
 - even if all preference lists have length ≤ 3
 - but polynomial-time solvable if preference lists on one side have length ≤ 2

[Biro, Manlove et al 2008]

Some more challenging problems – *Applicants with sizes*

$a_1 : 1 : h_1 \ h_2 \ h_5$
 $a_2 : 1 : h_1 \ h_2 \ h_3$
 $a_3 : 1 : h_2 \ h_4 \ h_5$
 $a_4 : 1 : h_3 \ h_1 \ h_2$
 $a_5 : 1 : h_4 \ h_5 \ h_1$
 $a_6 : 1 : h_1 \ h_5 \ h_3$
 $a_7 : 2 : h_2 \ h_3 \ h_4$
 $a_8 : 2 : h_2 \ h_1 \ h_5$

$h_1 : 2 : a_8 \ a_6 \ a_4 \ a_1 \ a_5 \ a_2$
 $h_2 : 2 : a_4 \ a_3 \ a_8 \ a_2 \ a_1 \ a_7$
 $h_3 : 2 : a_6 \ a_7 \ a_4 \ a_2$
 $h_4 : 2 : a_7 \ a_3 \ a_5$
 $h_5 : 2 : a_6 \ a_3 \ a_1 \ a_8 \ a_5$

a stable matching of size 9

- some applicants could be couples seeking a post together
- a stable matching need not exist in this case
- Determining the existence of a stable matching is NP-complete
 - even if all applicant sizes and employer capacities are 1 or 2 and all the preference lists are of length ≤ 3

[McDermid & Manlove 2008]

Some more challenging problems – *Ties*

- Stable Marriage with Ties (and Incomplete Lists) – *SMTI*
- Hospitals / Residents with Ties – *HRT*
 - each preference list can contain one or more *ties*
 - arguably a more realistic model in practical applications
 - large hospitals cannot reasonably differentiate among all of their many applicants
- Again, a pair blocks a matching only if both members of the pair would be better off if matched together
- To find a stable matching, break the ties arbitrarily and apply the Gale-Shapley algorithm
- But now – *stable matchings can have different sizes*
 - in practice we would like to find a stable matching of *maximum* size

SMTI - an illustration

m_1 : w_3 (w_1) (w_2) w_4
 m_2 : (w_1) w_3
 m_3 : (w_3) w_1
 m_4 : w_1 (w_3) (w_4) w_2

men's preferences

w_1 : (m_1) (m_2) m_3 m_4
 w_2 : (m_1) m_4
 w_3 : (m_4) (m_3) m_1 m_2
 w_4 : (m_4) m_1

women's preferences

(ties denoted by parentheses)

A stable matching of size 4: $\{(m_1, w_2), (m_2, w_1), (m_3, w_3), (m_4, w_4)\}$

A stable matching of size 2: $\{(m_1, w_1), (m_4, w_3)\}$

Ties make life difficult

For SMTI (and therefore HRT)

- it's NP-hard to find a maximum cardinality stable matching
- it's NP-complete to determine if a given pair is stable, or whether a given person is matched in any stable matching
- it's NP-hard to find an egalitarian stable matching
 - all of these, under very severe restrictions on the ties

[Iwama, Manlove et al 1999], [Manlove, Irving et al 2002]

- finding a maximum cardinality stable matching remains NP-hard
 - even if there are 'master' preference lists, eg if the applicants are ranked by some objective criterion

[Irving, Manlove and Scott 2007]

- and even if all preference lists are of length ≤ 3

[Irving, Manlove and O'Malley 2008]

Approximating maximum cardinality SMTI (and HRT)

- Trivially **2-approximable**
 - max cardinality $\leq 2 * \text{min cardinality}$
- But **APX-complete**

[Irving, Manlove et al 2003]
- And not approximable within **21/19** (unless P = NP)
- **13/7-approximable** if all ties are of length 2
- **8/5-approximable** if all ties are of length 2 and on just one side
- **10/7 approximable** (randomised) if, in addition, ≤ 1 tie per list

[Halldórsson et al 2003]
- **15/8 approximable** in the general case

[Iwama et al 2007]
- **5/3 approximable** if ties are on one side at the end (HRT also)

[Irving and Manlove 2007]

HR and HRT in Practice

- **Most real-world matching schemes solve instances of HR**
 - using some variant of the Gale-Shapley algorithm
- **Ties are either forbidden or are broken randomly by the matching scheme**
- **But breaking ties artificially places additional constraints on stable matchings**
 - and crucially this tends to reduce the cardinality
- **As far as we know the SFAS scheme is the only one that encourages the use of ties and makes use of heuristics (based on approximation algorithms) in an attempt to maximise the size of the stable matching**
 - for example in 2007 SFAS had 781 applicants
 - random tie-breaking led to a stable matching of size 736 after many iterations
 - we found stable matchings of size as small as 721
 - our heuristic gave a stable matching of size 744

Matching under *one-sided* preferences

Matching, say, applicants to posts, where only applicants have preferences

a_1	p_1	p_5	p_3	p_7
a_2	p_2	p_5	p_6	p_4
a_3	p_3	p_5	p_6	p_8
a_4	p_4	p_3	p_7	p_6
a_5	p_1	p_2	p_5	p_4
a_6	p_2	p_3	p_4	p_7
a_7	p_4	p_1	p_5	p_3
a_8	p_1	p_2	p_3	p_6

Applicants' preferences

Its *profile* is (1, 2, 2, 3)

- 1 applicant gets his 1st choice, 2 their 2nd choice, etc.

An example matching -

$\{(a_1, p_3), (a_2, p_5), (a_3, p_8), (a_4, p_7),$
 $(a_5, p_4), (a_6, p_2), (a_7, p_1), (a_8, p_6)\}$

It's a maximum matching

But is it a 'good' matching?

Matching under one-sided preferences – an example

a_1	:	p_1	p_5	p_3	p_7
a_2	:	p_2	p_5	p_6	p_4
a_3	:	p_3	p_5	p_6	p_8
a_4	:	p_4	p_3	p_7	p_6
a_5	:	p_1	p_2	p_5	p_4
a_6	:	p_2	p_3	p_4	p_7
a_7	:	p_4	p_1	p_5	p_3
a_8	:	p_1	p_2	p_3	p_6

Applicants' preferences

A second example matching -

$\{(a_1, p_5), (a_2, p_6), (a_3, p_3), (a_4, p_7), (a_6, p_2), (a_7, p_4), (a_8, p_1)\}$

It's not a maximum matching

But its profile seems better

Its *profile* is $(4, 1, 2, 0)$,
compared to $(1, 2, 2, 3)$

Rank-maximal matchings

Denote a profile by (x_1, x_2, \dots, x_r)

A matching is *rank-maximal* if

1. x_1 has the maximum possible value
2. subject to 1, x_2 has the maximum possible value,
3. subject to 1 and 2, x_3 has the maximum possible value,
4. etc.

Algorithmic solutions

[Irving et al 2004]

- For constant length preference lists, a rank-maximal matching can be found in $O(n^{3/2})$ time where n is the number of applicants
- In general, $O(\min(nm, rn^{1/2}m))$ time, where r is the maximum length of list and m the sum of the list lengths
- In both cases, even if there are ties in the preference lists

Popular matchings

- If $(a, p) \in M$ then we denote p by $M(a)$, and a by $M(p)$
- Applicant a **prefers** matching M to matching M' if
 - a is matched in M but not in M' , or
 - a prefers $M(a)$ to $M'(a)$
- A matching M is **popular** if there is no matching M' such that more applicants prefer M' to M than prefer M to M'
- Define the relation \rightarrow on matchings by $M \rightarrow M'$ if more applicants prefer M to M' than prefer M' to M
- A popular matching is a minimal element in this relation

Popular matchings – an illustration

a_1 :	p_1	p_5	p_3	p_7
a_2 :	p_2	p_5	p_6	p_4
a_3 :	p_3	p_5	p_6	p_8
a_4 :	p_4	p_3	p_7	p_6
a_5 :	p_1	p_2	p_5	p_4
a_6 :	p_2	p_3	p_4	p_7
a_7 :	p_4	p_1	p_5	p_3
a_8 :	p_1	p_2	p_3	p_6

Applicants' preferences

the rank-maximal matching

$$M = \{(a_1, p_5), (a_2, p_6), (a_3, p_3), (a_4, p_7), (a_6, p_2), (a_7, p_4), (a_8, p_1)\}$$

is not a popular matching

a_1 and a_2 prefer

$$M' = \{(a_1, p_1), (a_2, p_5), (a_3, p_3), (a_4, p_7), (a_6, p_2), (a_7, p_4), (a_8, p_6)\}$$

whereas only a_8 prefers M

Popular matchings may not exist

- \rightarrow is not an order relation

a_1	:	p_1	p_2	p_3
a_2	:	p_1	p_2	p_3
a_3	:	p_1	p_2	p_3

- The **blue** matching **M** shown is unique up to symmetry
- The **red** matching **M'** satisfies **M'** \rightarrow **M**

Popular matchings

Algorithmic solutions

[Abraham, Irving et al 2005]

Case of strict preferences:

- $O(m)$ algorithm to determine whether a popular matching exists, and if so to find a largest one
 - m is the sum of the lengths of the preference lists

Ties in the preference lists:

- $O(mn^{1/2})$ algorithm for the same problem
 - n is the number of applicants + the number of posts

Recently, for strict preferences:

- $O(m + kn)$ algorithm to generate all k popular matchings

[McDermid 2008]

A selection of open problems

- **Improved approximation** of maximum stable matching for SMTI / HRT
 - in theory (guarantees) or in practice (heuristics)
- **Generating** all stable matchings for SMTI / HRT
 - non-trivial even to solve the uniqueness problem [Scott 2004]
- **Generating** all rank-maximal matchings
 - can we find all rank-maximal matching, say in $O(n)$ time per matching after we've found one such matching?
- **Popular** matchings in SMI, i.e. when **preferences are on both sides**
 - a popular matching always exists – since a stable matching is provably popular
 - but there may be a popular matching of size greater than a stable matching
 - is there a polynomial-time algorithm to find a popular matching of maximum size?