Centre for
Data Analytics

Insight

Revisiting Two-Sided Stability Constraints

Mohamed Siala Barry O'Sullivan

May 18, 2016

DCU NUI Galway OÉ Gaillimh UCC UCD Dublin sfi

Context

Context

- Matching under preferences & Constraint Programming?

Context

- Matching under preferences & Constraint Programming?
- Few CP formulations exist in the literature for stable matching

Context

- Matching under preferences & Constraint Programming?
- Few CP formulations exist in the literature for stable matching
- Not much about local consistency levels

Context

- Matching under preferences & Constraint Programming?
- Few CP formulations exist in the literature for stable matching
- Not much about local consistency levels
- Global constraints for stable matching problems?

Matching Under Preferences

- They are everywhere!
- For instance, assigning students to universities/residents to hospitals/ workers to firms . . .

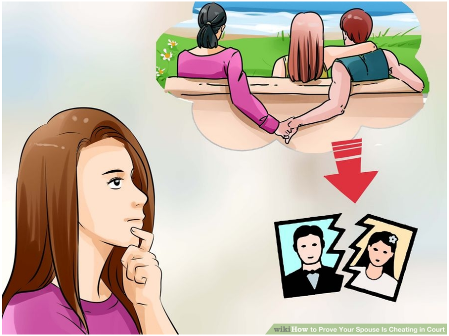
Matching Under Preferences

- They are everywhere!
- For instance, assigning students to universities/residents to hospitals/ workers to firms . . .
- Bipartie structure with two sided preferences
- Bipartie structure with one sided preferences
- Non-bipartie structure

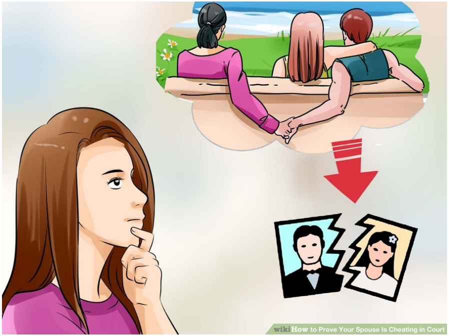
Matching Under Preferences

- They are everywhere!
- For instance, assigning students to universities/residents to hospitals/ workers to firms . . .
- Bipartie structure with two sided preferences
- Bipartie structure with one sided preferences
- Non-bipartie structure

Stable Marriage [Gale and Shapley, 1962]

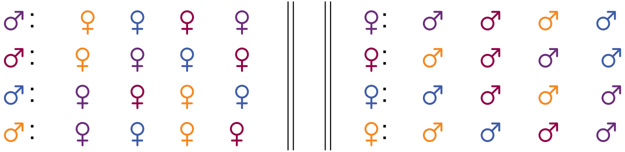


Stable Marriage [Gale and Shapley, 1962]

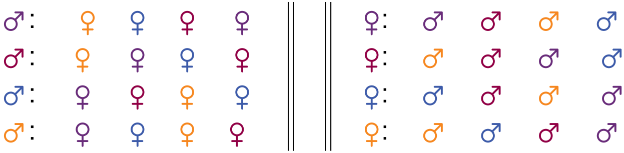


- ♂, ♂, ♂, . . .
- ♀, ♀, ♀, . . .
- Two sided preferences
- A matching *M* is stable when no blocking pair exists
- A pair (♂,♀) is blocking a matching M if ♂/♀ prefer each other to their situation in *M*

[Gale and Shapley, 1962]

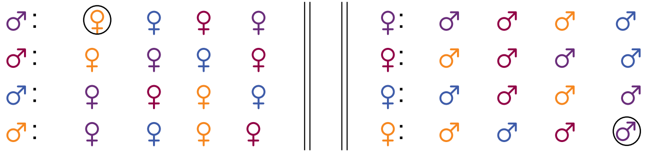


[Gale and Shapley, 1962]



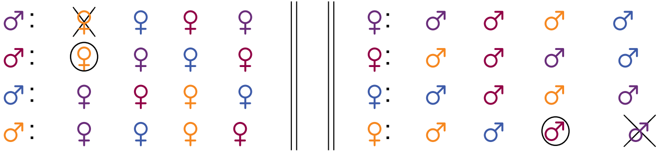
- Male proposals

[Gale and Shapley, 1962]



- Male proposals
  (♂,♀)

[Gale and Shapley, 1962]



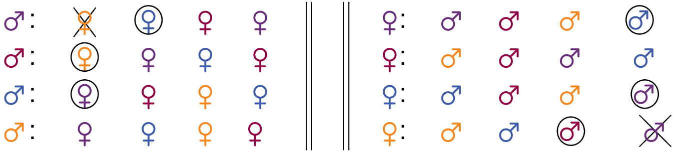
- Male proposals
  (♂,♀), (♂,♀)

[Gale and Shapley, 1962]

♂:	~~♀~~	♀	♀	♀	‖ ‖	♀:	♂	♂	♂	♂
♂:	♀	♀	♀	♀	‖ ‖	♀:	♂	♂	♂	♂
♂:	♀	♀	♀	♀	‖ ‖	♀:	♂	♂	♂	♂
♂:	♀	♀	♀	♀	‖ ‖	♀:	♂	♂	♂	~~♂~~

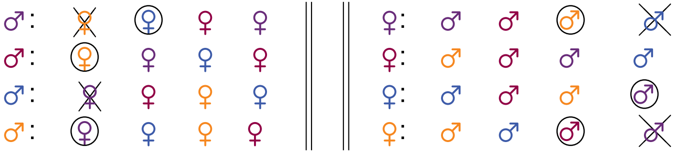
- Male proposals
  (♂,♀), (♂,♀), (♂,♀)

[Gale and Shapley, 1962]



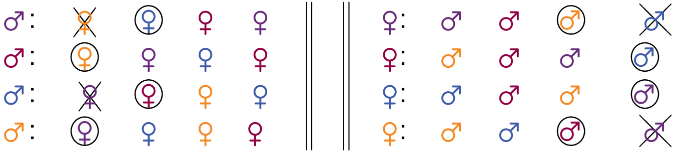
- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀)

[Gale and Shapley, 1962]



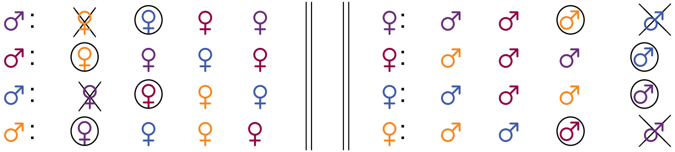
- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀), (♂,♀)

[Gale and Shapley, 1962]



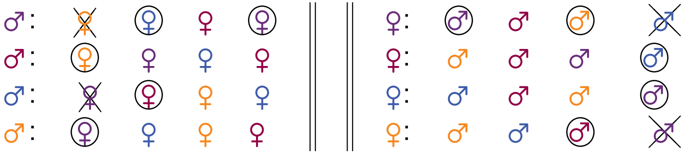
- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀), (♂,♀) ,(♂,♀)

[Gale and Shapley, 1962]



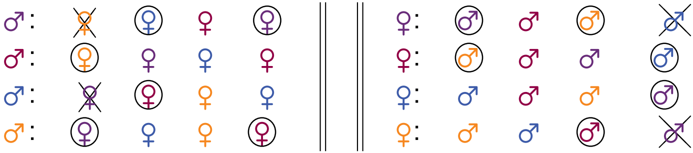
- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀), (♂,♀) ,(♂,♀)
- Female proposals

[Gale and Shapley, 1962]



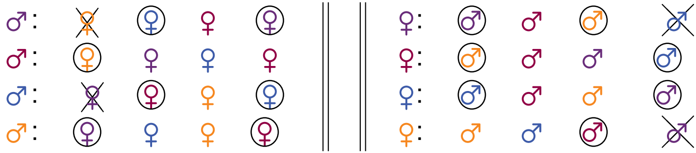
- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀), (♂,♀) ,(♂,♀)
- Female proposals
  (♀,♂)

[Gale and Shapley, 1962]



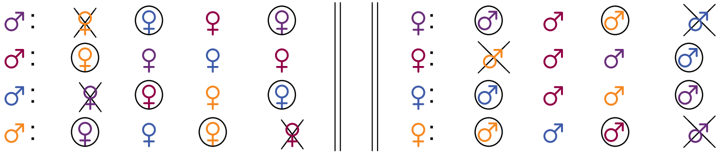
- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀), (♂,♀) ,(♂,♀)
- Female proposals
  (♀,♂) , (♀,♂)

[Gale and Shapley, 1962]



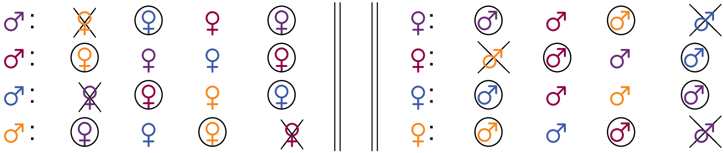
- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀), (♂,♀) ,(♂,♀)
- Female proposals
  (♀,♂) , (♀,♂), (♀,♂)

[Gale and Shapley, 1962]



- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀), (♂,♀) ,(♂,♀)
- Female proposals
  (♀,♂) , (♀,♂), (♀,♂) , (♀,♂)

[Gale and Shapley, 1962]



- Male proposals
  (♂,♀), (♂,♀), (♂,♀) ,(♂,♀), (♂,♀) ,(♂,♀)
- Female proposals
  (♀,♂) , (♀,♂), (♀,♂) , (♀,♂) , (♀,♂)

The Hospital/Resident Problem

The Hospital/Resident Problem

- Many to one extension of SM

The Hospital/Resident Problem

- Many to one extension of SM
- Two sets of agents: residents $r_1, r_2, \ldots$ and hospitals $h_1, h_2, \ldots$

The Hospital/Resident Problem

- Many to one extension of SM
- Two sets of agents: residents $r_1, r_2, \ldots$ and hospitals $h_1, h_2, \ldots$
- Preferences without ties

The Hospital/Resident Problem

- Many to one extension of SM
- Two sets of agents: residents $r_1, r_2, \dots$ and hospitals $h_1, h_2, \dots$
- Preferences without ties
- Each hospital h_j has a capacity c_j
- Notation: i better than j w.r.t. a list L: $i \underset{L}{\prec\prec} j$ or $j \underset{L}{\succ\succ} i$

The Hospital/Resident Problem

- Many to one extension of SM
- Two sets of agents: residents $r_1, r_2, \ldots$ and hospitals $h_1, h_2, \ldots$
- Preferences without ties
- Each hospital h_j has a capacity c_j
- Notation: i better than j w.r.t. a list L: $i \underset{L}{\prec\prec} j$ or $j \underset{L}{\succ\succ} i$
- Find a stable matching?

Previous CP Approaches

- Some global constraints exist in the literature [Gent et al., 2001, Unsworth and Prosser, 2005, Manlove et al., 2007, Unsworth and Prosser, 2013]

Previous CP Approaches

- Some global constraints exist in the literature [Gent et al., 2001, Unsworth and Prosser, 2005, Manlove et al., 2007, Unsworth and Prosser, 2013]
- They are all equivalent in terms filtering (even for SM)

Previous CP Approaches

- Some global constraints exist in the literature [Gent et al., 2001, Unsworth and Prosser, 2005, Manlove et al., 2007, Unsworth and Prosser, 2013]
- They are all equivalent in terms filtering (even for SM)
- They all are equivalent to the GL-Lists

Previous CP Approaches

- Some global constraints exist in the literature [Gent et al., 2001, Unsworth and Prosser, 2005, Manlove et al., 2007, Unsworth and Prosser, 2013]
- They are all equivalent in terms filtering (even for SM)
- They all are equivalent to the GL-Lists
- What is the level of consistency?

CP Model for Hospital/Resident Problem (Γ)

Variables

- x_i: index of the hospital assigned to r_i
- $y_{j,k}$: index of the resident assigned to the k^{th} position in h_j

Constraints

$$y_{j,k} < y_{j,k+1} \; (\forall j \in [1, n_H], \forall k \in [1, c_j - 1]) \tag{1}$$

$$y_{j,k} \geq q_{i,j} \implies x_i \leq p_{i,j} \; (\forall j \in [1, n_H], \forall k \in [1, c_j], \forall i \in \mathcal{H}_j) \tag{2}$$

$$x_i \neq p_{i,j} \implies y_{j,k} \neq q_{i,j} \; (\forall i \in [1, n_R], \forall j \in \mathcal{R}_i, \forall k \in [1, c_j]) \tag{3}$$

$$(x_i \geq p_{i,j} \wedge y_{j,k-1} < q_{i,j}) \implies y_{j,k} \leq q_{i,j} \; (\forall i \in [1, n_R], \forall j \in \mathcal{R}_i, \forall k \in [1, c_j]) \tag{4}$$

$$y_{j,c_j} < q_{i,j} \implies x_i \neq p_{i,j} \; (\forall j \in [1, n_H], \forall i \in \mathcal{H}_j) \tag{5}$$

Example

$$
\begin{array}{ll|l}
\mathcal{R}_1 = [3,2,1] & & \mathcal{H}_1 = [1,2,4] \\
\mathcal{R}_2 = [4,1,3,2] & & \mathcal{H}_2 = [2,1,3] \\
\mathcal{R}_3 = [2,4,3] & & \mathcal{H}_3 = [3,2,4,1] \\
\mathcal{R}_4 = [1,3,4] & & \mathcal{H}_4 = [4,3,2]
\end{array}
$$

Initial Domain

- $\mathcal{D}(x_1) = \mathcal{D}(x_3) = \mathcal{D}(x_4) = \{1,2,3,5\}$
- $\mathcal{D}(x_2) = \{1,2,3,4,5\}$
- $\mathcal{D}(y_{1,0}) = \mathcal{D}(y_{2,0}) = \mathcal{D}(y_{3,0}) = \mathcal{D}(y_{4,0}) = \{0\}$
- $\mathcal{D}(y_{1,1}) = \mathcal{D}(y_{2,1}) = \mathcal{D}(y_{4,1}) = \{1,2,3,5\}$
- $\mathcal{D}(y_{3,1}) = \{1,2,3,4,5\}$

2-SidedStability$(\mathcal{X}, \mathcal{A}, \mathcal{B}, \mathcal{C})$

- $\mathcal{X}$ is the set of variables $x_1, \ldots, x_{n_R}$ defined the same way in Γ,
- $\mathcal{A} = \{\mathcal{R}_1, \ldots, \mathcal{R}_{n_R}\}$
- $\mathcal{B} = \{\mathcal{H}_1, \ldots, \mathcal{H}_{n_H}\}$
- $\mathcal{C} = \{c_1, \ldots, c_{n_H}\}$

2-SidedStability$(\mathcal{X}, \mathcal{A}, \mathcal{B}, \mathcal{C})$

- $\mathcal{X}$ is the set of variables $x_1, \ldots, x_{n_R}$ defined the same way in Γ,
- $\mathcal{A} = \{\mathcal{R}_1, \ldots, \mathcal{R}_{n_R}\}$
- $\mathcal{B} = \{\mathcal{H}_1, \ldots, \mathcal{H}_{n_H}\}$
- $\mathcal{C} = \{c_1, \ldots, c_{n_H}\}$

We show that AC on Γ enforces BC(D) on any domain $\mathcal{D}$

Local Consistency

Local Consistency

Definitions

- A support of a constraint C in a domain $\mathcal{D}$ is an assignment of the variables in $\mathcal{D}$ that satisfies C

Local Consistency

Definitions

- A support of a constraint C in a domain $\mathcal{D}$ is an assignment of the variables in $\mathcal{D}$ that satisfies C
- A constraint in arc consistent in $\mathcal{D}$ iff for every variable x in the scope of C, every value in $\mathcal{D}(x)$ has a support in $\mathcal{D}$

Local Consistency

Definitions

- A support of a constraint C in a domain $\mathcal{D}$ is an assignment of the variables in $\mathcal{D}$ that satisfies C
- A constraint in arc consistent in $\mathcal{D}$ iff for every variable x in the scope of C, every value in $\mathcal{D}(x)$ has a support in $\mathcal{D}$
- A constraint in bound(D) consistent in $\mathcal{D}$ iff for every variable x in the scope of C, $min(\mathcal{D}(x))$ and $max(\mathcal{D}(x))$ have a support in $\mathcal{D}$

Local Consistency

Definitions

- A support of a constraint C in a domain $\mathcal{D}$ is an assignment of the variables in $\mathcal{D}$ that satisfies C
- A constraint in arc consistent in $\mathcal{D}$ iff for every variable x in the scope of C, every value in $\mathcal{D}(x)$ has a support in $\mathcal{D}$
- A constraint in bound(D) consistent in $\mathcal{D}$ iff for every variable x in the scope of C, $min(\mathcal{D}(x))$ and $max(\mathcal{D}(x))$ have a support in $\mathcal{D}$

Bound(D) consistency is stronger than the classic bound consistency property

Arc Consistency using Γ?

Arc Consistency using Γ?

- AC removes 5 from $\mathcal{D}(x_1), \mathcal{D}(x_2), \mathcal{D}(x_3), \mathcal{D}(x_4)$
- AC removes 5 from $\mathcal{D}(y_{1,1}), \mathcal{D}(y_{2,1}), \mathcal{D}(y_{3,1}), \mathcal{D}(y_{4,1})$
- No more propagation

Arc Consistency using Γ?

- AC removes 5 from $\mathcal{D}(x_1), \mathcal{D}(x_2), \mathcal{D}(x_3), \mathcal{D}(x_4)$
- AC removes 5 from $\mathcal{D}(y_{1,1}), \mathcal{D}(y_{2,1}), \mathcal{D}(y_{3,1}), \mathcal{D}(y_{4,1})$
- No more propagation
- Assigning 3 to x_2 has no solution
- Γ hinders propagation!

Theorem

[Gusfield and Irving, 1989]

Theorem

[Gusfield and Irving, 1989]

- The number of assigned residents per hospital is the same in all stable matchings

Theorem

[Gusfield and Irving, 1989]

- The number of assigned residents per hospital is the same in all stable matchings
- If a resident r_i is unassigned in one stable matching then it is unassigned in all stable matchings.

Theorem

[Gusfield and Irving, 1989]

- The number of assigned residents per hospital is the same in all stable matchings
- If a resident r_i is unassigned in one stable matching then it is unassigned in all stable matchings.
- If a hospital h_j is under-subscribed in one stable matching then it is assigned exactly the same residents in all stable matching

Preprocessing

- Compute the GS_lists and prune the domain accordingly ($O(L)$)

Preprocessing

- Compute the GS_lists and prune the domain accordingly (O(L))
- Notation: ***HFull*** the set of hospitals fully subscribed in any stable matching

Necessary and Sufficient Condition for Stability

Theorem

2-SidedStability($\mathcal{X}, \mathcal{A}, \mathcal{B}, \mathcal{C}$) is satisfiable iff

$$\forall\, 1 \leq j \leq n_H,\ \sum_{i=1}^{n_R} (\mathcal{R}_i[x_i] == j) \leq c_j \wedge$$

$$\forall\, 1 \leq i \leq n_R,\ \forall\, 1 \leq j \leq n_H + 1, x_i = j \implies$$

$$\forall k \in [1, j[$$

$$\textbf{if}\, h = \mathcal{R}_i[k] \textbf{then}$$

$$\sum_{m=1}^{n_R} (\mathcal{R}_m[x_m] == h) = c_h$$

$$\wedge$$

$$\forall l \underset{\mathcal{H}_h}{\succ\succ} i,\ \mathcal{R}_l[x_l] \neq h$$

Lemma

If Γ is AC then assigning all variables to their minimum value is solution.

Lemma
If Γ is AC then assigning all variables to their minimum value is solution.

Lemma
If Γ is AC then assigning all variables to their maximum value is solution.

Lemma

If Γ is AC then assigning all variables to their minimum value is solution.

Lemma

If Γ is AC then assigning all variables to their maximum value is solution.

Theorem

Enforcing AC on Γ makes $\text{2-SidedStability}(\mathcal{X}, \mathcal{A}, \mathcal{B}, \mathcal{C})$ Bound($\mathcal{D}$) consistent.

Revisiting Bound(D) Consistency

- Best BC(D) algorithm runs in $O(c \times L)$ [Manlove et al., 2007]
- We propose an optimal algorithm running in $O(L)$

Revisiting Bound(D) Consistency

- Best BC(D) algorithm runs in $O(c \times L)$ [Manlove et al., 2007]
- We propose an optimal algorithm running in $O(L)$

Theorem

2-SidedStability($\mathcal{X}, \mathcal{A}, \mathcal{B}, \mathcal{C}$) is BC(D) iff assigning every variable to its maximum is a solution and assigning every variable to its minimum is a solution.

Lower bound changes

- Assume we have a domain that is BC(D)
- Let r_i be a resident whose lower bound has changed
- Let h_j be the hospital corresponding to the new lower bound

Step 1

- Let h be any hospital ‘between’ the old and the new lower bound
- Make sure that any resident worse (in $\mathcal{H}_h$) than r_i cannot be assigned to h

Step 1

- Let h be any hospital 'between' the old and the new lower bound
- Make sure that any resident worse (in $\mathcal{H}_h$) than r_i cannot be assigned to h

$$\mathcal{H}_h \quad r_{10} \quad r_3 \quad r_1 \quad r_7 \quad r_{14} \quad r_i \quad r_4 \quad r_6 \qquad \underset{\substack{\Uparrow \\ lastLeft_h}}{r_2} \qquad \cancel{r_8} \quad \cancel{r_{11}}$$

Step 1

- Let h be any hospital 'between' the old and the new lower bound
- Make sure that any resident worse (in $\mathcal{H}_h$) than r_i cannot be assigned to h

$$\mathcal{H}_h \quad r_{10} \quad r_3 \quad r_1 \quad r_7 \quad r_{14} \quad r_i \quad r_4 \quad r_6 \quad \underset{\substack{\Uparrow \\ lastLeft_h}}{\cancel{r_2}} \quad \cancel{r_8} \quad \cancel{r_{11}}$$

Step 1

- Let h be any hospital 'between' the old and the new lower bound
- Make sure that any resident worse (in $\mathcal{H}_h$) than r_i cannot be assigned to h

$$\mathcal{H}_h \quad r_{10} \quad r_3 \quad r_1 \quad r_7 \quad r_{14} \quad r_i \quad r_4 \quad \underset{\substack{\Uparrow \\ lastLeft_h}}{\cancel{r_6}} \quad \cancel{r_2} \quad \cancel{r_8} \quad \cancel{r_{11}}$$

Step 1

- Let h be any hospital 'between' the old and the new lower bound
- Make sure that any resident worse (in $\mathcal{H}_h$) than r_i cannot be assigned to h

$$\mathcal{H}_h \quad r_{10} \quad r_3 \quad r_1 \quad r_7 \quad r_{14} \quad r_i \quad \cancel{r_4} \quad \cancel{r_6} \quad \cancel{r_2} \quad \cancel{r_8} \quad \cancel{r_{11}}$$

$$\Uparrow$$

$$lastLeft_h$$

Step 1

- Let h be any hospital 'between' the old and the new lower bound
- Make sure that any resident worse (in $\mathcal{H}_h$) than r_i cannot be assigned to h

$$\mathcal{H}_h \quad r_{10} \quad r_3 \quad r_1 \quad r_7 \quad r_{14} \quad \underset{\underset{lastLeft_h}{\Uparrow}}{\cancel{r_i}} \quad \cancel{r_4} \quad \cancel{r_6} \quad \cancel{r_2} \quad \cancel{r_8} \quad \cancel{r_{11}}$$

Step 1

- Let h be any hospital 'between' the old and the new lower bound
- Make sure that any resident worse (in $\mathcal{H}_h$) than r_i cannot be assigned to h

$$\mathcal{H}_h \quad r_{10} \quad r_3 \quad r_1 \quad r_7 \quad \underset{\substack{\Uparrow \\ lastLeft_h}}{r_{14}} \quad \cancel{r_i} \quad \cancel{r_4} \quad \cancel{r_6} \quad \cancel{r_2} \quad \cancel{r_8} \quad \cancel{r_{11}}$$

Step 2

Make sure that the new hospital h_j has no more than c_j residents whose lower bound corresponds to h_j.

- MIN_{h_j}: The variables whose minimum corresponds to h_j
- If $|MIN_{h_j}| = c_{h_j} + 1$: Remove the worst resident r from MIN_{h_j} and prune h_j from the domain of r

Step 2

Make sure that the new hospital h_j has no more than c_j residents whose lower bound corresponds to h_j.

- MIN_{h_j}: The variables whose minimum corresponds to h_j
- If $|MIN_{h_j}| = c_{h_j} + 1$: Remove the worst resident r from MIN_{h_j} and prune h_j from the domain of r

Example

$$\mathcal{H}_{h_j} \quad r_2 \quad r_8 \quad r_7 \quad r_{11} \quad r_9 \quad r_5 \quad r_6 \quad r_3 \quad r_1 \quad \underset{\substack{\Uparrow \\ lastLeft_h}}{r_8} \quad \cancel{r_6} \quad \cancel{r_{12}}$$

Step 2

Make sure that the new hospital h_j has no more than c_j residents whose lower bound corresponds to h_j.

- MIN_{h_j}: The variables whose minimum corresponds to h_j
- If $|MIN_{h_j}| = c_{h_j} + 1$: Remove the worst resident r from MIN_{h_j} and prune h_j from the domain of r

Example

$$\mathcal{H}_{h_j} \quad r_2 \quad r_8 \quad r_7 \quad r_{11} \quad r_9 \quad r_5 \quad r_6 \quad r_3 \quad r_1 \quad \underset{\underset{lastLeft_h}{\Uparrow}}{r_8} \quad \cancel{r_6} \quad \cancel{r_{12}}$$

Step 2

Make sure that the new hospital h_j has no more than c_j residents whose lower bound corresponds to h_j.

- MIN_{h_j}: The variables whose minimum corresponds to h_j
- If $|MIN_{h_j}| = c_{h_j} + 1$: Remove the worst resident r from MIN_{h_j} and prune h_j from the domain of r

Example

$$\mathcal{H}_{h_j} \quad r_2 \quad r_8 \quad r_7 \quad r_{11} \quad r_9 \quad r_5 \quad r_6 \quad r_3 \quad \underset{\substack{\Uparrow \\ lastLeft_h}}{r_1} \quad r_8 \quad \cancel{r_6} \quad \cancel{r_{12}}$$

Step 2

Make sure that the new hospital h_j has no more than c_j residents whose lower bound corresponds to h_j.

- MIN_{h_j}: The variables whose minimum corresponds to h_j
- If $|MIN_{h_j}| = c_{h_j} + 1$: Remove the worst resident r from MIN_{h_j} and prune h_j from the domain of r

Example

$$\mathcal{H}_{h_j} \quad r_2 \quad r_8 \quad r_7 \quad r_{11} \quad r_9 \quad r_5 \quad r_6 \qquad \underset{\underset{lastLeft_h}{\Uparrow}}{\cancel{r_3}} \qquad r_1 \quad r_8 \quad \cancel{r_6} \quad \cancel{r_{12}}$$

Step 2

Make sure that the new hospital h_j has no more than c_j residents whose lower bound corresponds to h_j.

- MIN_{h_j}: The variables whose minimum corresponds to h_j
- If $|MIN_{h_j}| = c_{h_j} + 1$: Remove the worst resident r from MIN_{h_j} and prune h_j from the domain of r

Example

$$\mathcal{H}_{h_j} \quad r_2 \quad r_8 \quad r_7 \quad r_{11} \quad r_9 \quad r_5 \quad r_6 \quad \cancel{r_3} \quad r_1 \quad r_8 \quad \cancel{r_6} \quad \cancel{r_{12}}$$

$$\Uparrow$$

$$lastLeft_h$$

Upper bound changes

- MAX_h: The indexes of residents where the maximum corresponds to h
- $maxofMAX_h = \max(MAX_h)$

Lemma

Assigning all variables to their maximum is a solution iff $\forall\ h \in \boldsymbol{HFull}$, $|MAX_h| = c_h$, and $\forall\ i \leq maxofMAX_h$, let $r = \mathcal{H}_h[i]$, and $l = \mathcal{R}_r{}^{-1}[h]$, then $i \notin MAX_h \implies \max(x_r) < l$.

Upper bound changes

- Assume we have a domain that is BC(D)
- Let r_i be a resident whose upper bound has changed
- Let h be the hospital corresponding to the previous upper bound
- Find a 'replacement' for r

Upper bound changes

- Assume we have a domain that is BC(D)
- Let r_i be a resident whose upper bound has changed
- Let h be the hospital corresponding to the previous upper bound
- Find a 'replacement' for r

Example

$$\mathcal{H}_{h_j} \quad . \quad . \quad r_2 \quad . \quad . \quad r_7 \quad . \quad . \quad \cancel{r_5} \quad r_6 \quad r_{11} \quad r_3 \quad . \quad .$$

Upper bound changes

- Assume we have a domain that is BC(D)
- Let r_i be a resident whose upper bound has changed
- Let h be the hospital corresponding to the previous upper bound
- Find a 'replacement' for r

Example

$$\mathcal{H}_{h_j} \quad . \quad . \quad r_2 \quad . \quad . \quad r_7 \quad . \quad . \quad \cancel{r_5} \quad \underset{\underset{maxofMAX_h}{\Uparrow}}{r_6} \quad r_{11} \quad r_3 \quad . \quad .$$

Upper bound changes

- Assume we have a domain that is BC(D)
- Let r_i be a resident whose upper bound has changed
- Let h be the hospital corresponding to the previous upper bound
- Find a 'replacement' for r

Example

$$\mathcal{H}_{h_j} \quad . \quad . \quad r_2 \quad . \quad . \quad r_7 \quad . \quad . \quad \cancel{r_5} \quad r_6 \qquad \underset{\substack{\Uparrow \\ maxofMAX_h}}{r_{11}} \qquad r_3 \quad . \quad .$$

Upper bound changes

- Assume we have a domain that is BC(D)
- Let r_i be a resident whose upper bound has changed
- Let h be the hospital corresponding to the previous upper bound
- Find a 'replacement' for r

Example

$$\mathcal{H}_{h_j} \quad . \quad . \quad r_2 \quad . \quad . \quad r_7 \quad . \quad . \quad \cancel{r_5} \quad r_6 \quad r_{11} \qquad \underset{\substack{\Uparrow \\ maxofMAX_h}}{r_3} \qquad . \quad .$$

Arc Consistency

Arc Consistency

- Enforce BC(D)

Arc Consistency

- Enforce BC(D)
- For any variable x_i, we suppose that the upper bound is removed

Arc Consistency

- Enforce BC(D)
- For any variable x_i, we suppose that the upper bound is removed
- Enforce BC(D) on the new domain

Arc Consistency

- Enforce BC(D)
- For any variable x_i, we suppose that the upper bound is removed
- Enforce BC(D) on the new domain
- Any value between the old and the new upper bound does not have a support

Arc Consistency

- Enforce BC(D)
- For any variable x_i, we suppose that the upper bound is removed
- Enforce BC(D) on the new domain
- Any value between the old and the new upper bound does not have a support
- Repeat Until the lower bound of x_i

Arc Consistency

- Enforce BC(D)
- For any variable x_i, we suppose that the upper bound is removed
- Enforce BC(D) on the new domain
- Any value between the old and the new upper bound does not have a support
- Repeat Until the lower bound of x_i
- Complexity: $O(n_R \times L)$

Arc Consistency

- Enforce BC(D)
- For any variable x_i, we suppose that the upper bound is removed
- Enforce BC(D) on the new domain
- Any value between the old and the new upper bound does not have a support
- Repeat Until the lower bound of x_i
- Complexity: $O(n_R \times L)$
- Not incremental

Beyond Constraint Propagation

Beyond Constraint Propagation

Hospital/Resident Problem with Forced and Forbidden Pairs

- Unknown complexity
- Straightforward to solve! Just enforce BC(D) on the domain
- $O(L)$ to solve with our approach

Beyond Constraint Propagation

Hospital/Resident Problem with Forced and Forbidden Pairs

- Unknown complexity
- Straightforward to solve! Just enforce BC(D) on the domain
- $O(L)$ to solve with our approach

Stable Marriage Problem with Forced and Forbidden Pairs

- Best complexity $O(n^4)$ [Dias et al., 2003]
- Particular case of the above approach
- $O(n^2)$ to solve

Experiments

Problem description

- Some couples prefer to be matched together
- Same preferences for these couples
- Find a stable matching maximizing the number of such couples who are matched together

Experiments

Protocol

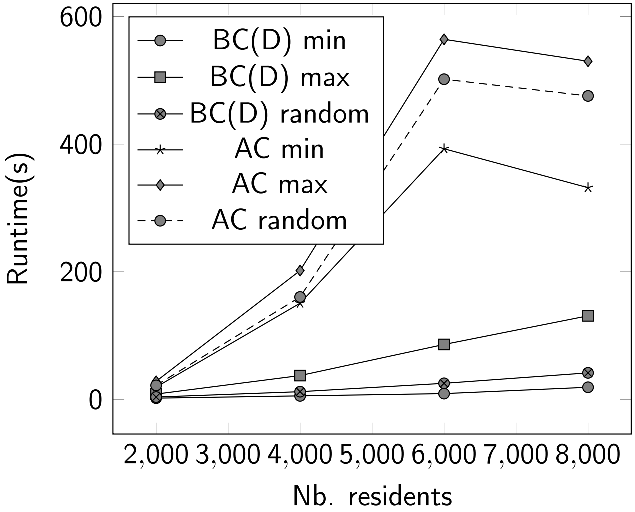
- Random instances: $2k, 4k, \ldots, 8k$ residents; $100, 200, \ldots 500$ hospitals; and different capacities $c \in \{100 + 50 + k | k \in [0, 8]\}$
- BC(D) and AC Implemented in Mistral-2.0
- LEX variable branching + (min/max/random min max) value branching
- Geometric restarts
- 5 different seeds for “random min max”
- 20 minutes time cutoff

Summary of the Results

Set	BC(D)-min			AC-min			BC(D)-max		
	Time	Nodes	Opt	Time	Nodes	Opt	Time	Nodes	Opt
2k	2	16.56	**100**	19	13.33	**100**	8	256.38	**100**
4k	5	18.14	**100**	151	14.69	**100**	37	394.86	**100**
6k	9	18.08	**100**	393	14.89	93	86	648.82	**100**
8k	19	18.16	**100**	332	15.80	79	131	491.50	**100**

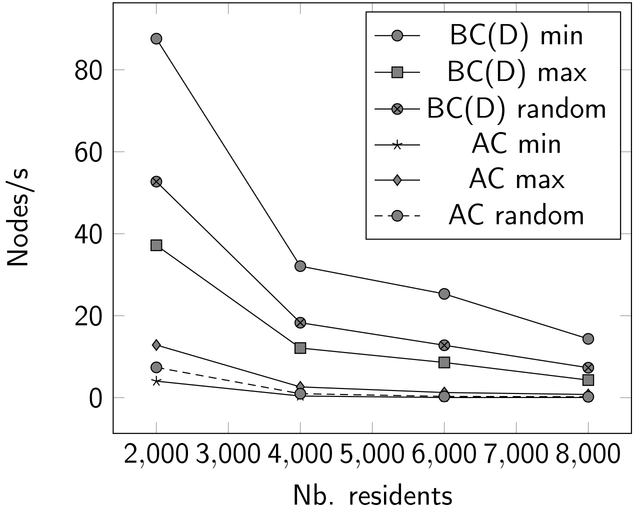
	AC-max			BC(D)-rand			AC-rand		
	Time	Nodes	Opt	Time	Nodes	Opt	Time	Nodes	Opt
2k	28	204.53	**100**	4	67.36	**100**	12	81.91	**100**
4k	202	320.33	**100**	12	81.91	**100**	160	65.61	**100**
6k	564	535.65	85	25	120.58	**100**	502	99.20	92
8k	530	432.87	69	42	98.88	**100**	475	88.11	77

Runtime



600
400
200
0
Runtime(s)
BC(D) min
BC(D) max
BC(D) random
AC min
AC max
AC random
2,000
3,000
4,000
5,000
6,000
7,000
8,000
Nb. residents

Speed of Exploration



Conclusion and Future Research

Contributions

- Previous CP propositions maintain only BC(D)
- A better implementation of BC(D) in $O(L)$ time
- An adaptation of BC(D) to achieve AC
- First polynomial algorithm to solve the Hospital/Resident problem with forced and forbidden pairs
- Improving the worst case complexity to solve SM with forced and forbidden pairs by a factor of $O(n^2)$

Conclusion and Future Research

Contributions

- Previous CP propositions maintain only BC(D)
- A better implementation of BC(D) in $O(L)$ time
- An adaptation of BC(D) to achieve AC
- First polynomial algorithm to solve the Hospital/Resident problem with forced and forbidden pairs
- Improving the worst case complexity to solve SM with forced and forbidden pairs by a factor of $O(n^2)$

Future Research

- Better implementation for AC?
- New global constraints are coming for different stable matching problems..

Thank you.

Picture taken from The New York Times

References I

Dias, V. M. F., da Fonseca, G. D., de Figueiredo, C. M. H., and Szwarcfiter, J. L. (2003).
The stable marriage problem with restricted pairs.
Theor. Comput. Sci., 306(1-3):391–405.

Gale, D. and Shapley, L. S. (1962).
College admissions and the stability of marriage.
American mathematical monthly, pages 9–15.

Gent, I. P., Irving, R. W., Manlove, D., Prosser, P., and Smith, B. M. (2001).
A constraint programming approach to the stable marriage problem.
In *Proceedings of CP*, pages 225–239.

Gusfield, D. and Irving, R. W. (1989).
The Stable marriage problem - structure and algorithms.
Foundations of computing series. MIT Press.

Manlove, D., O'Malley, G., Prosser, P., and Unsworth, C. (2007).
A constraint programming approach to the hospitals / residents problem.
In *Proceedings of CPAIOR*, pages 155–170.

References II

Unsworth, C. and Prosser, P. (2005).
A specialised binary constraint for the stable marriage problem.
In *Abstraction, Reformulation and Approximation, 6th International Symposium, SARA 2005, Airth Castle, Scotland, UK, July 26-29, 2005, Proceedings*, pages 218–233.

Unsworth, C. and Prosser, P. (2013).
An n-ary constraint for the stable marriage problem.
CoRR, abs/1308.0183.