

Contents

1	Stable Maching problema	2
1.1	Algoritmo Gale-Shapley	2
1.2	Alternativas	2
1.2.1	Diferentes cantidades de oferentes que requeridos	2
1.2.2	Preferencias incompletas	3
1.2.3	Preferencias con empates	4
1.2.4	Agrupacion de 1 a muchos	5
1.2.5	Agrupacion de muchos a 1	5
1.2.6	Agrupacion de y a x	5

1 Stable Matching problema

1.1 Algoritmo Gale-Shapley

Este algoritmo al terminar de ejecutarse se encuentra un matching perfecto si:

- Si existen n solicitantes con diferentes preferencias.
- Si existen n requeridos con diferentes preferencias.

Eligiendo las estructuras correctamente se puede plantear en $O(n)$.

```
1  Inicialmente M=Vacio
2
3  Mientras existe un solicitante sin pareja que no aun se haya postulado a todas las
   parejas
4
5      Sea s un solicitante sin pareja
6      Sea r el requerido de su mayor preferencia al que no le
           solicito previamente
7
8
9      if r esta desocupado
10         M = M U (s,r)
11         s esta ocupado
12     else
13         Sea s' tal que (s', r) pertenece a M
14
15         si r prefiere a s sobres s'
16             M = M - {(s', r)} U (s,r)
17             s esta ocupado
18             s' esta libre
19
20 Retornar M
```

Listing 1: Algoritmo de Gale-Shapley

1.2 Alternativas

1.2.1 Diferentes cantidades de oferentes que requeridos

Dado n oferentes y m requeridos, con $m < n$, no se puede encontrar un matching stable.

Entonces, tenemos que redefinir el concepto de estable. Una pareja (s,r) es **estable** si:

- No existe requerido r' sin pareja al que s prefiera a su actual pareja.
- No existe un requerido r' en pareja, tal que s y r' se prefieran sobre sus respectivas parejas.
- No existe solicitante s' sin pareja al que r prefiera a su actual pareja.
- No existe un solicitante s' en pareja tal que r y s' se prefieran sobre sus respectivas parejas.

Por lo tanto un matching es estable si:

- No tienen parejas inestables bajo la condicion anterior.
- Que no queden requeridos y solicitantes sin pareja.

Soluciones para ajustar al modelo de Gale-Shapley:

1. Inventar $|n - m|$ elementos ficticios

- Los elementos ficticios se pondran en las listas de preferencias con menos elementos.
- Estos elementos ficticios se agregan al final y deben ser los menos preferidos.

- Luego ejecutar Gale-Shapley
- Por ultimo, eliminar las parejas con elementos ficticios. Estos seran los requeridos que quedan sin pareja.

2. Adecuar el Algoritmo

- Si hay mas **solicitantes** que requeridos, quitar de la *lista de solicitantes* sin parejas a aquellos que agotaron sus propuestas.
- Si hay mas **requeridos** que solicitantes, quitar de la *lista de parejas* a aquellas donde el requerido quedo sin pareja.

1.2.2 Preferencias incompletas

Las listas de preferencias de los oferentes y los requeridos son un subset de las contrapartes.

Son parejas **acceptables** de un elemento a aquellas contrapartes que figuran en su lista de preferencias.

Una pareja (s,r) es **estable** si:

- Son *acceptables* entre ellos.
- No existe requerido *acceptable* r' sin pareja al que s prefiera a su actual pareja.
- No existe un requerido *acceptable* r' en pareja, tal que s y r' se prefieran sobre sus respectivas parejas.
- No existe solicitante *acceptable* s' sin pareja al que r prefiera a su actual pareja.
- No existe un solicitante *acceptable* s' en pareja tal que r y s' se prefieran sobre sus respectivas parejas.

Un matching es estable si no tiene parejas inestables bajo la condicion anteriores.

```

1 Inicialmente M=Vacio
2
3 #Iterea mientras no haya acotado su sublista de preferencias
4 Mientras existe un solicitante sin pareja
5     'que no aun se haya postulado a todas las parejas'
6
7     Sea s un solicitante sin pareja
8     Sea r el requerido de su mayor preferencia al que no le
9         solicito previamente
10
11     # se condiera si es acceptable
12     if r considera 'acceptable' a s
13
14         if r esta desocupado
15             M = M U (s,r)
16             s esta ocupado
17         else
18             Sea s' tal que (s', r) pertenece a M
19             si r prefiere a s sobres s'
20                 M = M - {(s', r)} U (s,r)
21                 s esta ocupado
22                 s' esta libre
23
24 # Retornar solo parejas aceptables
25 Retornar M

```

Listing 2: Algoritmo para parejas incompletas

1.2.3 Preferencias con empates

INDIFERENCIA Y PREFERENCIA ESTRICTA

1. X es **indiferente** a "y" y a "z" si en su lista de preferencias estan el la misma posicion.
2. X es **prefiere estrictamente** a "y" sobre "z" si en su lista de preferencias no le son indiferentes y "y" se encuentra antes que "z" en la misma.

ESTABILIDAD DEBIL

Una pareja (s,r) es debilmente estable si no existe una pareja (s' y r') talque:

- s prefiere estrictamente a r' sobre r (*pareja actual de s*)
- r' prefiere estrictamente a s sobre s' (*pareja actual de r'*)

```
1  Inicialmente M=Vacio
2
3  #Iterea mientras no haya acotado su sublista de preferencias
4  Mientras existe un solicitante sin pareja
5      'que no aun se haya postulado a todas las parejas'
6
7      Sea s un solicitante sin pareja
8      Sea r el requerido de su mayor preferencia al que no le
9          solicito previamente
10
11     if r esta desocupado
12         M = M U (s,r)
13         s esta ocupado
14     else
15         Sea s' tal que (s', r) pertenece a M
16
17         # prefiere estrictamente
18         si r prefiere estrictamente a s sobres s'
19             M = M - {(s', r)} U (s,r)
20             s esta ocupado
21             s' esta libre
22
23     Retornar M
```

Listing 3: Algoritmo para parejas incompletas

En caso de que sea empate, se mantendra con su pareja actual.

ESTABILIDAD FUERTE

Una pareja (s,r) es debilmente estable si no existe una pareja (s' y r') talque:

- s prefiere estrictamente o le es indiferente a r' sobre r (*pareja actual de s*)
- r' prefiere estrictamente o le es indiferente a s sobre s' (*pareja actual de r'*)

Puede no existir un matching perfecto.

```
1  Inicialmente M=Vacio
2
3  Mientras existe un solicitante sin pareja y no exista solicitante que agoto sus
4  parejas
5
6      Sea s un solicitante sin pareja
7      Sea r el requerido de su mayor preferencia al que pueda proponer
8      Por cada sucesor s' a s en la lista de preferencias de r
9          if (s',r) pertenece a M
10              M = M - {(s',r)}
```

```

10         s' esta libre
11         quitar s' de la lista de preferencias de r
12         quitar r de la lista de preferencias de s'
13
14     Por cada requerido r' que tiene multiples parejas
15     Por cada pareja s' en pareja con r'
16         M = M - {(s',r')}
17         quitar s' de la lista de preferencias de r'
18         quitar r' de la lista de preferencias de s'
19
20 if estan todos en pareja
21     Retornar M
22 else
23     No existe ningun matching super estable

```

Listing 4: Algoritmo para parejas super estables

En caso de que sea empate, se mantendra con su pareja actual.

1.2.4 Agrupacion de 1 a muchos

El solicitante puede tener varios cupos por lo tanto:

- Existen m requeridos, donde un requerido puede estar unicamente con 1 pareja.
- Existen n solicitantes, donde cada solicitante puede tener c cupos para armar parejas.

Existe un matching estable si la cantidad de requeridos es igual a la cantidad de solicitantes por la cantidad de cupos.

$$m = n * c \quad (1)$$

No cambia la definici3n de Gale Shampey para **matching estable**

```

1     Inicialmente M=Vacio
2
3     Mientras exista un solicitante con cupo disponible
4
5         Sea s un solicitante sin pareja
6         Sea r el requerido de su mayor preferencia al que no le
7             solicito previamente
8
9         if r esta desocupado
10             M = M U (s,r)
11             s decremente su disponibilidad de parejas
12         else
13             Sea s' tal que (s', r) pertenece a M
14
15             si r prefiere a s sobres s'
16                 M = M - {(s', r)} U (s,r)
17                 s decremente su disponibilidad de parejas
18                 s' incrementa su disponibilidad de parejas
19     Retornar M

```

Listing 5: Algoritmo de solicitantes con cupos

La complejidad algoritmica no se modifica porque solo se agrega un contador.

1.2.5 Agrupacion de muchos a 1

El requerido puede tener varios cupos por lo tanto:

- Existen m requeridos, donde cada solicitante puede tener z cupos para armar parejas.
- Existen n solicitantes, donde un requerido puede estar unicamente con 1 pareja.

Existe un matching estable si la cantidad de solicitantes es igual a la cantidad de requeridos por la cantidad de cupos.

$$n = m * z \quad (2)$$

No cambia la definición de Gale Shampey para **matching estable**

```
1  Inicialmente M=Vacio
2
3  Mientras exista un solicitante con cupo disponible
4
5      Sea s un solicitante sin pareja
6      Sea r el requerido de su mayor preferencia al que no le
7          solicito previamente
8
9      if r tiene cupo
10         M = M U (s,r)
11         s esta ocupado
12         r decrementa su disponibilidad de parejas
13     else
14         Sea s' tal que (s', r) pertenece a M y
15             s' es el menos preferidos de las parejas r
16
17         si r prefiere a s sobre s'
18             M = M - {(s', r)} U (s,r)
19             s esta ocupado
20             s' esta libre
21
22  Retornar M
```

Listing 6: Algoritmo de requeridos con cupos

La complejidad algoritmica si se modifica.

Para conocer el solicitante de menor preferencia podemos utilizar un heap de minimos. Como el cupo es de z , la complejidad algoritmica para actualizar el heap es $\log(z)$.

1.2.6 Agrupacion de y a x

- Existen n solicitantes, donde cada solicitante puede tener c cupos para armar parejas.
- Existen m requeridos, donde cada requerido puede tener z cupos para armar parejas.

Existe un matching estable si:

$$n * c = m * z \quad (3)$$