Diseño de algoritmos

Trabajo Práctico 1 - Asignaciones Estables

Manuel Latorre FAI-1931
manuel.latorre@est.fi.uncoma.edu.ar

Segundo cuatrimestre 2022

Índice

1. Punto 1

Implementar el siguiente algoritmo de asignación de parejas

Problema:

Pensar en una extensión del Problema de Asignaciones Estables de Gale and Shapley en la que ciertos pares hombre-mujer están explícitamente prohibidos.

Dado un conjunto H de n hombres, un conjunto M de n mujeres y un conjunto de pares $P \subseteq H \times M$ que son aquellos que simplemente no están permitido emparejarse.

Cada hombre h ranquea a todas las mujeres m tal que $(h, m) \notin P$, y cada mujer m' ranquea a todos los hombres h' tal que $(h', m') \notin P$

Entonces decimos que un macheo S es estable si no presenta ninguno de los siguientes tipos de inestabilidad:

(I) *No existen dos pares (h, m) y (h', m') en S con la propiedad que si $(h, m') \notin P$, h prefiere a m' en lugar de m, y m' prefiere a h en lugar de h'. (Inestabilidad básica)*

(II) *Existe un par $(h, m) \in S$, y un hombre h', tal que h' no es parte de ningún par en el macheo, $(h', m) \in P$, y m prefiere a h' en lugar de h. (Hombre soltero más deseable y no prohibido)*

(III) *Existe un par $(h, m) \in S$, y una mujer m', tal que m' no es parte de ningún par en el macheo, $(h, m') \notin P$, y h prefiere a m' en lugar de m. (Mujer soltera más deseable y no prohibida)*

(IV) *Existe un hombre h y una mujer m, ninguno de los cuales forman parte de una pareja en el emparejamiento, es decir $(h, m) \notin P$. (Hay dos personas solteras sin nada que les impida casarse entre ellas)*

Se pide:

a) *Elegir las estructuras de datos adecuadas y justificar.*

Para representar las preferencias de parejas se usaran matrices en donde los indices de las filas representaran al identificador de cada hombre/mujer, los indices de columnas representaran la prioridad en cuanto a preferencia, siendo el 0 la mayor preferencia, y los valores almacenados representaran a los identificador de cada mujer (que también serán representadas con indices pero de otra matriz). En la figura 1 se puede ver un ejemplo de las matrices de preferencia tanto de hombres como de mujeres.

Para representar el conjunto de parejas que no están permitidos se utilizara una matriz donde los indices de las filas representaran a los hombres, los indices de las columnas a las mujeres y los valores almacenados serán 0 (Para indicar que una pareja con hombre de indice i y mujer de indice j no esta permitida) y 1 (Para indicar que una pareja con hombre de indice i y mujer de indice j esta permitida)

Para finalizar se utilizaran dos arreglos, uno para las mujeres y otro para los hombres, donde se guardaran las parejas asignadas de cada uno, en donde los indices representaran a cada persona, es decir los indices del arreglo de hombres representaran a cada hombre y los indices del arreglo de mujeres representaran a cada mujer y el valor sera el indice representativo de la pareja de sexo opuesto asignada.

En caso de que un hombre y mujer queden sin pareja a causa de las restricciones planteadas entonces sus valores en los arreglos de parejas tendrán el valor -1 para representar que justamente no se les pudo asignar una pareja.

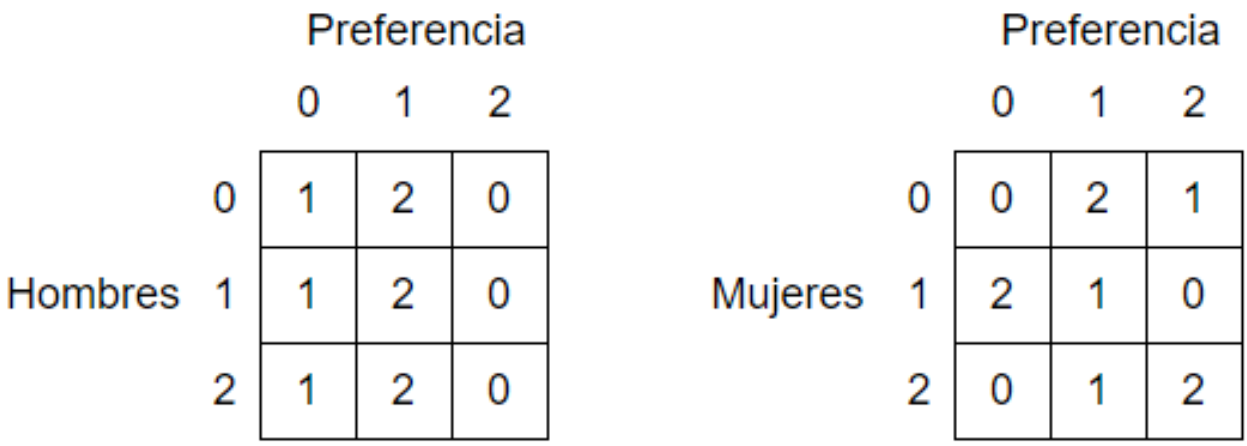

Figura 1: Matrices de preferencia de hombres y mujeres

Los arreglos que se muestran en la figura 3 representan el resultado del algoritmo sobre las matrices mostradas en las figuras 1 y 2, como se puede observar a causa de las restricciones planteadas el hombre 2 y la mujer 0 quedaron sin parejas asignadas.

Figura 2: Matriz de parejas no permitidas donde el hombre 0 y la mujer 2 es una pareja no permitida y el hombre 2 y la mujer 0 es otra pareja no permitida

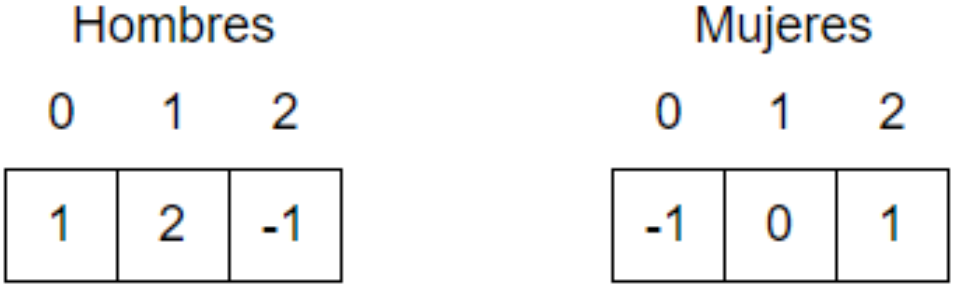

Figura 3: Arreglos de parejas asignadas

b) *Diseñar e Implementar en Java el algoritmo para resolver el problema*

Código 1 Implementacion problema de las parejas con asignaciones estables

```
1   public static void main(String[] args) {
2     int[][] menPref = { //Indice fila es el hombre, indice col es la
          preferencia, valor es el indice mujer
3           {1, 2, 0},
4           {1, 2, 0},
5           {1, 2, 0},
6     };
7     int[][] womenPref = { //Indice fila es el hombre, indice col es
          la preferencia, valor es el indice mujer
8           {0, 2, 1},
9           {2, 1, 0},
10          {0, 1, 2},
11    };
12    int[][] bannedCouples = { //Indice fila es el hombre, indice col
          mujeres, 1 = pueden ser pareja, 0 = no pueden ser pareja
13          {1, 1, 0},//Hombre 0 con mujer 2 no
14          {1, 1, 1},
15          {0, 1, 1}, //Hombre 2 con mujer 0 no
16    };
17
18    int[] womanCouples = new int[womenPref.length]; //indice hombres,
          -1 = sin pareja, otro valor indice de su pareja
19    int[] manCouples = new int[menPref.length];
20    assignCouples(menPref,womenPref,bannedCouples,manCouples,womanCouples);
21
22    for (int i = 0; i < menPref.length; i++) {
23      System.out.println("Hombre "+i+" --> "+" Mujer "+manCouples[i]);
24    }
25    for (int i = 0; i < womenPref.length; i++) {
26      System.out.println("Mujer "+i+" --> "+" Hombre
            "+womanCouples[i]);
27    }
28  }
29
30  public static int[] assignCouples(int[][] menPref, int[][]
        womenPref, int[][] bannedPartners,
        int[]manCouples,int[]womanCouples) {
31    int amountMen = menPref.length;
32    int freeMen = amountMen; //Cantidad de hombres restantes sin
        pareja
33
34    Arrays.fill(manCouples, -1);//Inicialmente no tienen pareja
35    Arrays.fill(womanCouples, -1);//Inicialmente no tienen pareja
36    while (freeMen > 0) {//Mientras queden hombres sin pareja
37      int indexMan = 0;
38      while (indexMan < amountMen && manCouples[indexMan] > -1)
          {//Busca al siguiente hombre sin pareja
39        indexMan++;
40      }
41      int indexPref = 0;
```

```
42        while (indexMan < amountMen && manCouples[indexMan] == -1 &&
              indexPref<womenPref.length) {
43          int selectedWoman = menPref[indexMan][indexPref];
44          if (bannedPartners[indexMan][selectedWoman] == 1) { //si
              pueden ser pareja
45            if (womanCouples[selectedWoman] == -1) {//si mujer NO tiene
                pareja, los empajera
46              womanCouples[selectedWoman] = indexMan;
47              manCouples[indexMan] = selectedWoman; //Los empareja
48              freeMen--; //Hay un hombre menos sin pareja
49            } else {//Si la mujer tiene pareja
50              int currentWomanCouple =
                  womanCouples[selectedWoman];//Busca la pareja de la mujer
51              if (betterCouple(womenPref, currentWomanCouple, indexMan))
                  {//Si el hombre actual es mejor pareja lo asigna
52                womanCouples[selectedWoman] = indexMan;
53                manCouples[currentWomanCouple] = -1; //La pareja actual
                    deja de serlo
54                manCouples[indexMan] = selectedWoman;//Se le asigna su
                    nueva pareja
55              }
56            }
57          }
58          indexPref++;//Busca al siguiente hombre
59        }
60        //Si no encontro pareja es porque no se lo permitio la
            restriccion, por lo tanto quedara sin pareja
61        //Para evitar bucle se decrementa en 1 la cantidad de hombres
            libres
62        if(indexMan<amountMen && manCouples[indexMan]==-1){
63          freeMen--;
64        }
65
66      }
67      return womanCouples;
68    }
69
70
71    public static boolean betterCouple(int[][] womenPref,int
        currentWomanCouple, int newCouple){
72      int indexCurrent =
          Arrays.asList(womenPref).indexOf(currentWomanCouple);
73      int indexNew =
          Arrays.asList(womenPref).indexOf(currentWomanCouple);
74      return indexNew < indexCurrent; //A menor indice mayor preferencia
75    }
```

c) *Calcular la eficiencia en tiempo y espacio.*

Se deberá operar mientras queden hombres sin pareja por lo tanto se hará un recorrido equivalente a la cantidad de filas de la matriz de preferencias de hombres, luego se deberá ir recorriendo el arreglo de parejas de hombres buscando a los diferentes hombres que no tengan pareja, como este arreglo es igual a la cantidad de filas de la matriz de preferencia

de los hombres entonces hasta el momento se tendrá $O(n^2)$.

Finalmente se recorrerá la matriz de preferencia para cada hombre, es decir solamente su fila correspondiente que sera igual al numero de columnas del arreglo de preferencia, para buscar una mujer que se ajuste a sus preferencia por prioridad, luego internamente en esta repetitiva se aplica la lógica para ver si una mujer debe o no cambiar de pareja en función a sus preferencias pero al trabajar con indices no sera necesario recorrer su matriz de preferencia, si no que como cada hombre y mujer esta representado con indices se podrá acceder directamente al indice correspondiente que se desee checkear por lo que finalmente se tendrá un orden de ejecución $O(n \cdot m)$ en donde n sera el numero de hombres y m el numero de mujeres. Si se supone que se tiene la misma (o menos) cantidad de hombres que de mujeres se puede asegurar que se tiene un $O(n^2)$

d) *Demostrar la correctitud (termina, es correcto y es estable).* La correctitud se demostrara a partir de las siguientes reglas

- **Propiedad 1:** Los hombres se declaran a las mujeres en orden decreciente de preferencia siempre y cuando no conformen una pareja que no esta permitida
- **Propiedad 2:** Una vez que una mujer se empareja, solo cambia de estado para mejorar de pareja

Terminacion:

- En cada iteración un hombre le propone salir a una mujer o como mínimo se lo propone, ya que para realizar la proposición primero debe verificarse que sea una pareja permitida
- Solo hay n^2 posibles comparaciones, suponiendo una misma cantidad de hombres que de mujeres
- Sea P(t) el conjunto de pares (h, m) en los que h le ha propuesto salir a una m al final de la iteración t. Para todo t, P(t+1) es estrictamente mayor que P(t). Como solo hay n^2 posibles comparaciones, P(t) puede aumentar n^2 veces, entonces nunca habrá mas de n^2 iteraciones. Por lo tanto queda demostrada la terminación del algoritmo

Perfección (Demostración por reducción al absurdo):

- Suponiendo que un hombre H no esta emparejado al terminar la ejecución del algoritmo
- Por lo anterior una mujer M no tendrá pareja
- Pero, por la propiedad 2, M no recibió ninguna proposición
- Sin embargo, H puede terminar sin pareja después de haberle propuesto (o haberse planteado hacerlo) salir a todas las mujeres

Estabilidad (Demostración por reducción al absurdo): Suponiendo que (H, M) es una pareja inestable en el emparejamiento devuelto por el algoritmo

1. H nunca le pidió salir a M entonces H prefiere a su pareja actual antes que a M o la propuesta generaría una pareja no permitida (propiedad 1) entonces (H, M) es estable.
2. H le pidió salir a M entonces M rechazo a H (directamente o después de encontrar a alguien mejor) entonces M prefiere a su pareja actual antes que a H (propiedad 2) entonces H, M es estable.
3. En cualquier caso, (H, M) es estable

e) *Explicar de qué estrategia algorítmica se trata.*

Se trata de una técnica de búsqueda local, es decir es que se puede tratar como un problema de optimización ya que se busca minimizar o maximizar algún valor (costo/beneficio), en este caso encontrar parejas con la mejor calidad posible en función a las preferencias, en este caso el método utilizado es mejora iterativa, este método lo que hace es comenzar con una posible solución que va mejorando a traves de aplicaciones repetidas de un paso simple. Funciona a partir de encontrar una primera solución relativamente fácil. Esta solución se podrá mejorar por una secuencia de cambios pequeños y finalizara cuando no se puedan hacer mas mejoras, por ejemplo las mujeres una vez que se les asigna una pareja luego pueden cambiarla por otra que se ajuste mas a sus preferencias