Centro de
Investigación en
Computación

Introduction: Some representative problems

Algunos problemas representativos

Tema 1

Course

Analysis and design of algorithms

Instructor

Acosta Bermejo Raúl et al.

Lecture notes

2024-B
27 de agosto del 2024

Instituto
Politécnico
Nacional

Table of contents (outline)

Tabla de contenido

Some basic definitios

Algunas definiciones básicas

An algorithm is:

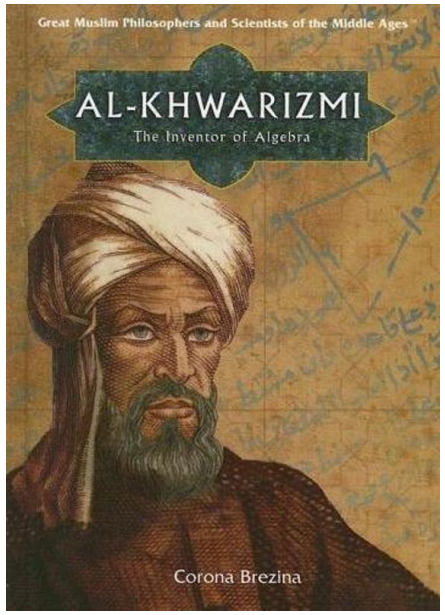
- (Informally) Any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output.
- A sequence of computational steps that transform the input into the output.
- A tool for solving a well-specified computational problem.
  - The statement of the problem specifies in general terms the desired input/output relationship.
  - The algorithm describes a specific computational procedure for achieving that input/output relationship.

Some basic definitios

Algunas definiciones básicas

Al-Juarismi

Algorithm



The name comes from a Persian mathematician

Abu Ja'far Muhammad ibn Musa Khwarizmi:

Abu Abdallah Muḥammad ibn Mūsā al-Jwārizmī

- Author of the first reference algebraic text.
- A Moon crater is dedicated to him.

Algorithm vs Program vs Proceso

- A algorithm describes (at high level) a computation procedure which when executed produces a result.
- A program is the implementation of a algorithm by means of a programming language.
- Un proceso es un programa en ejecución, por lo tanto tiene Data+Code+Stack.

Some basic definitios

Algunas definiciones básicas

Secuencia de pasos

Sorting

input
secuencia
<31, 41, 59, 26, 41, 58>

Algorithm

output
secuencia
<26, 31, 41, 41, 58, 59>

Programa computacional

Secuencia de Instrucciones

Conjunto de Entradas
input1.txt , ..., inputN.txt

Entrada estándar

Casos de Prueba

Conjunto de Salidas
output1.txt , ..., outputN.txt

Salida estándar

Some basic definitios

Algunas definiciones básicas

Correctness

- An algorithm is correct iff for all problem instances i ∈ I it terminates and produces the correct output o ∈ O (i.e. the pair (i,o) satisfies R).
- An incorrect algorithm either does not terminate or terminates and produces a wrong output for at least one input.

Specification

- An algorithm can be specified in various ways using:
  - Natural languages,
  - Informal pseudo-codes
  - Formal specifications,
  - Programming languages.

Some basic definitios

Algunas definiciones básicas

Analysis

Evaluates the amount of resources required by an algorithm:

- Processor (execution) time.
- Memory.
- Exchenge of messages, etc.

Testing

- Debugging (error correction), and
- Profiling (deriving performance profiles of algorithms).

Some basic definitios

Algunas definiciones básicas

Types of proof

Correctness (correctitud)

- It is said that the algorithm is correct with respect to a specification.
- Functional correctness refers to the input-output behaviour of the algorithm (i.e., for each input it produces the expected output).
- A distinction is made between *total correctness*, which additionally requires that the algorithm terminates, and *partial correctness*, which simply requires that if an answer is returned it will be correct.
- *Hoare logic* is a specific formal system for reasoning rigorously about the correctness of computer programs. It uses pre and post conditions.
- A proof would have to be a mathematical proof, assuming both the algorithm and specification are given formally.

Soundness (validez)

An algorithm is sound if, anytime it returns an answer, that answer is true. An algorithm is complete if it guarantees to return a correct answer for any arbitrary input (or, if no answer exists, it guarantees to return failure).

Some basic definitios

Algunas definiciones básicas

Other properties

Safety (seguridad)

- Lamport 1977
- “Nothig bad happens” like deadlock.

Liveness (vivacidad/viveza, concurrency)

- “Some good thing will occur” like termination.
- Ausencia de deadlock o starvation (recursos en SO).

Some basic definitios

Algunas definiciones básicas

Other properties, for example, search algorithms

- An algorithm is complete if whenever there is at least one solution, the algorithm is guaranteed to find it within a finite amount of time.

- An algorithm is optimal if when it finds a solution, it is the best one (e.g., the shortest).

Representative problems

Problemas representativos

Try to identify the pattern

Five representative problems

Cinco problemas representativos

We'll see the following problems:

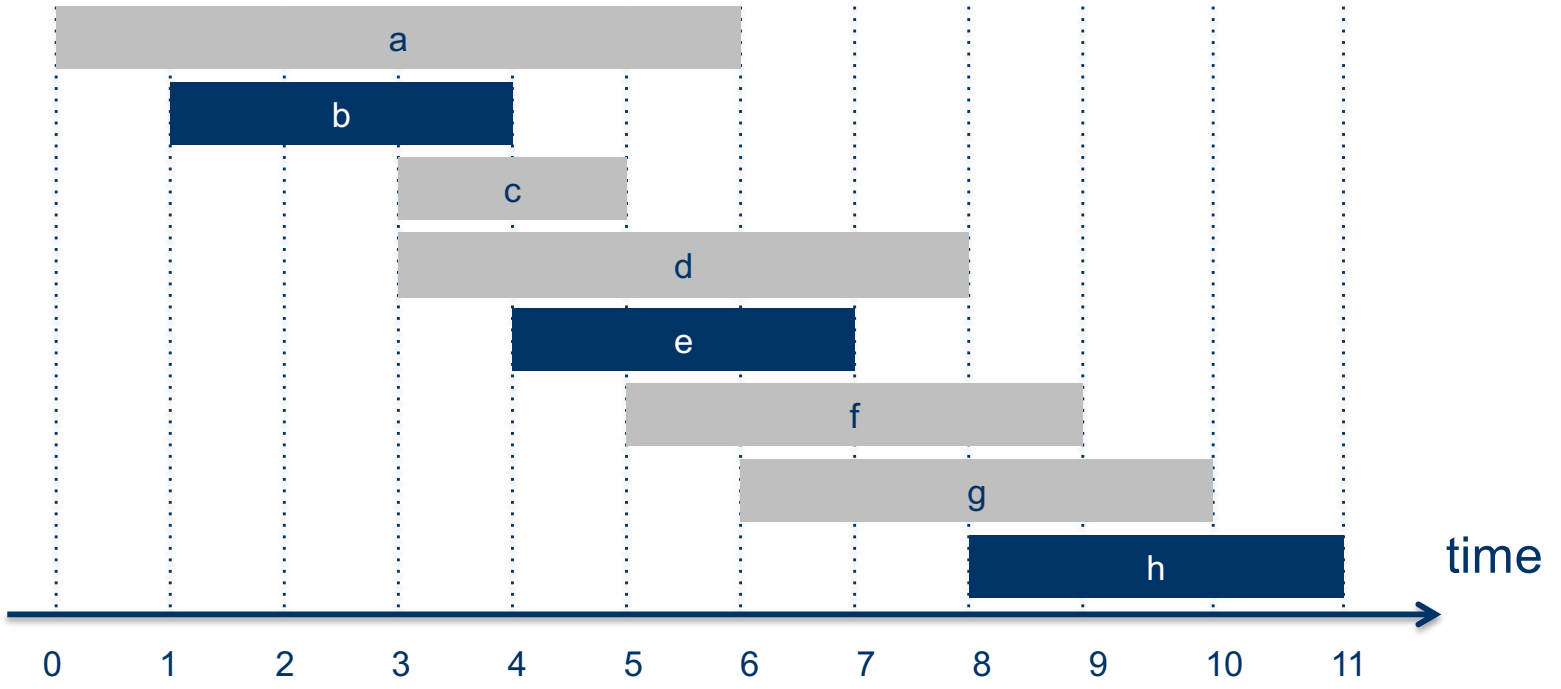
1. Interval scheduling
2. Weighted interval scheduling
3. Bipartite matching
4. Independent set
5. Competitive facility location

Problem 1

Interval scheduling (1.2.1)

Input. Set of jobs with start times and finish times.

Goal. Find maximum cardinality subset of mutually compatible jobs (jobs don't overlap).

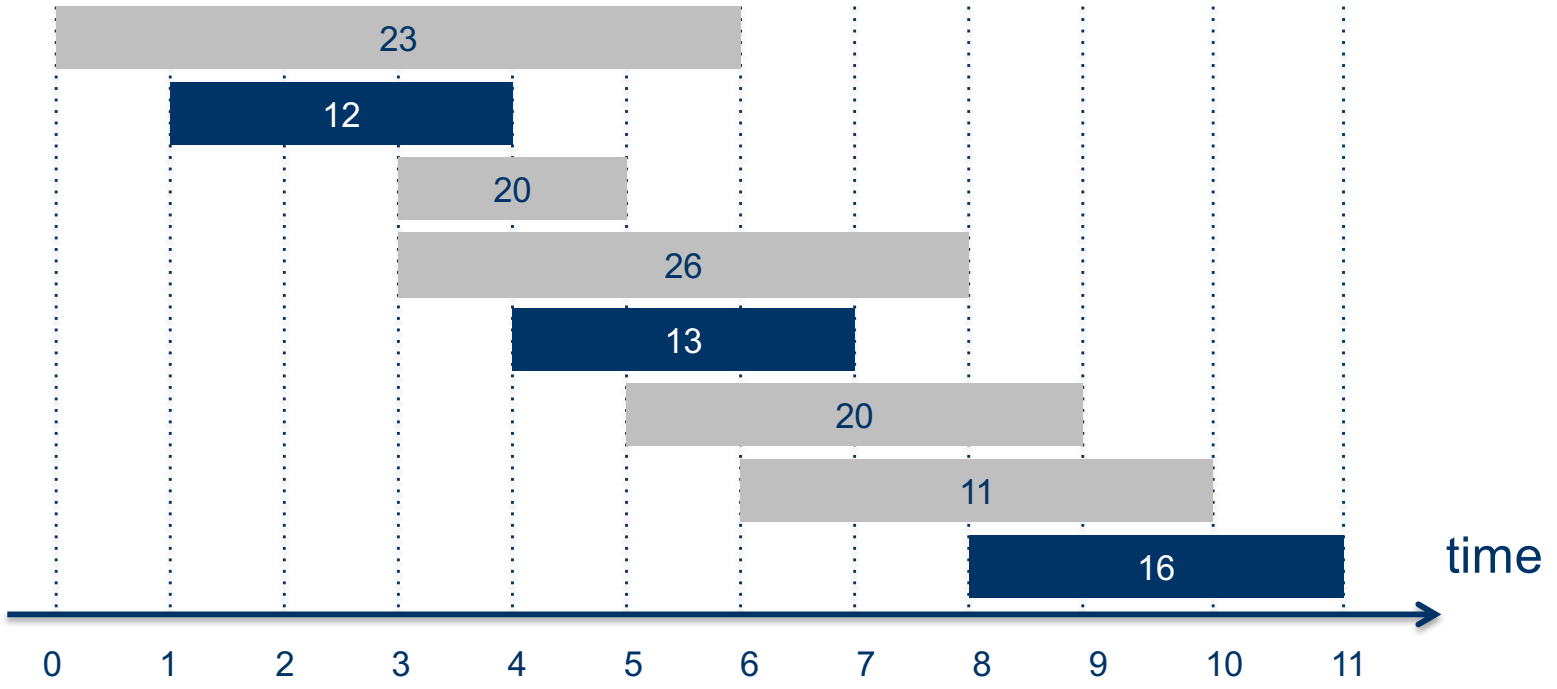


Problem 2

Weighted interval scheduling (1.2.2)

Input. Set of jobs with start times, finish times, and weights.

Goal. Find maximum weight subset of mutually compatible jobs.

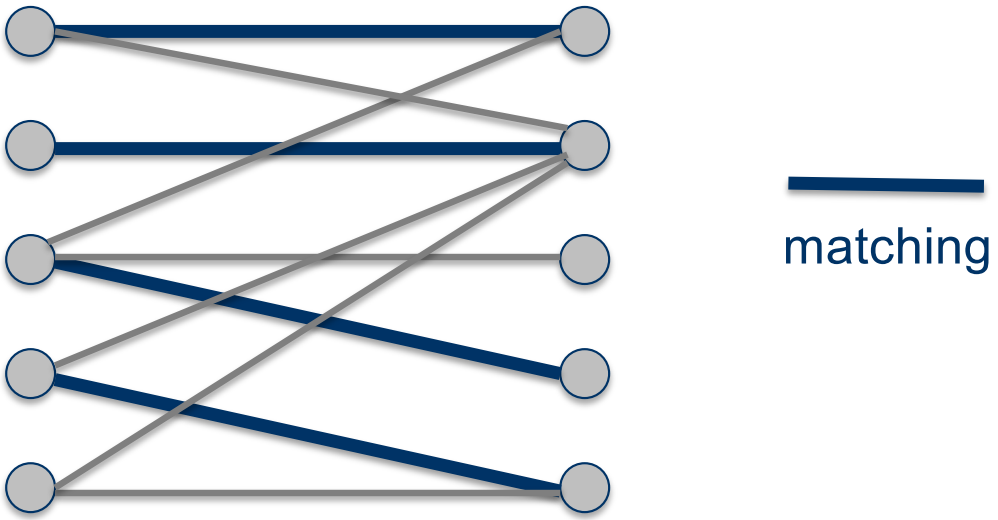


Problem 3

Bipartite matching (1.2.3)

Problem. Given a bipartite graph $G = (L \cup R, E)$, find a max cardinality matching.

Def. A subset of edges $M \subseteq E$ is a matching if each node appears in exactly one edge in M.

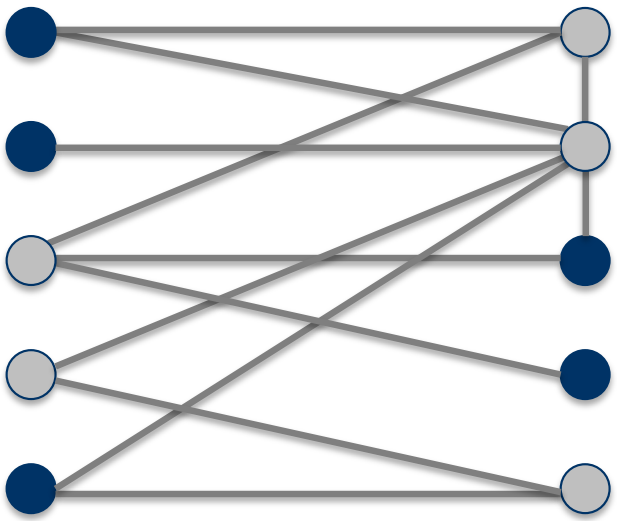


Problem 4

Independent set (1.2.4)

Problem. Given a graph $G = (V, E)$, find a max cardinality independent set.

Def. A subset $S \subseteq V$ is independent if for every $(u, v) \in E$, either $u \notin S$ or $v \notin S$ (or both).



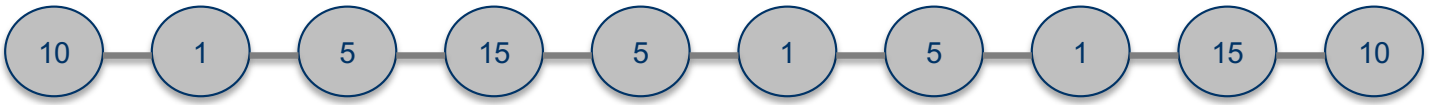
It is a set of vertices in a graph, no two of which are adjacent.

independent set

Problem 5

Competitive facility location

Input. Graph with weight on each node.
Game. Two competing players alternate in selecting nodes.
Not allowed to select a node if any of its neighbors have been selected.
Goal. Select a maximum weight subset of nodes.



Second player can guarantee 20 but not 25.

Five representative problems

Cinco problemas representativos

The complexity of these problems is:

1. Interval scheduling: $O(n \log n)$ Greedy algorithm.
2. Weighted interval scheduling: $O(n \log n)$ Dinamic programming algorithm.
3. Bipartite matching: $O(n^k)$ max-flow based algorithm.
4. Independent set: NP-complete.
5. Competitive facility location: PSPACE-complete.

Centro de
Investigación en
Computación

Stable Matching

Emparejamiento Estable

Definición

Algoritmo

Demostración

Instituto
Politécnico
Nacional

Stable matching

Emparejamiento estable

The problem has been defined in 2 ways (goal):

1. Hospitals assignments

   Given a set of preferences among hospitals and med-school students, design a self-reinforcing admissions process.

2. Stable matching

   Given a set of *n* men and a set of *n* women, find a "suitable" matching.

Additional sentences:

- Participants rank members of opposite sex.
- Each man lists women in order of preference from best to worst.
- Each woman lists men in order of preference from best to worst.

Stable matching

Emparejamiento estable

Formalizations (mathematics)

Set n men SM = $\{m_1 , m_2 , \ldots., m_n\}$
Set n women SW = $\{w_1 , w_2 , \ldots., w_n\}$

Conjuntos y operaciones sobre conjuntos: Relaciones (funciones)

A matching M is a bijection between the men and women.
We say a (man, woman) pair (m,w) blocks M if:

- m prefers w to his partner in M, and
- w prefers m to her partner in M.

A matching that admits no blocking pair is said to be stable

- Can't improve by making an arrangement outside the matching.

Stable matching

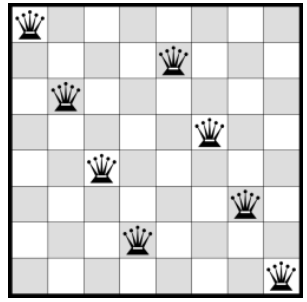
Emparejamiento estable

The problem SM has been tackle like a

Constraint Satisfaction Problem (CSP) like those of IO

Problema de Satisfacción de Restricciones (Programación con Restricciones)

Examples of simple problems that can be modeled as a CSP



- Eight queens puzzle
- Map coloring problem
- Sudoku, Futoshiki, Kakuro (Cross Sums), Numbrix, Hidato and many other logic puzzles.

La idea principal es usar el conocimiento de las restricciones para eliminar del dominio de las variables todos aquellos valores que no pueden formar parte de una solución al problema. De esta forma, es posible resolver un problema reduciendo los dominios de sus variables hasta conseguir aproximaciones muy cercanas al valor solución, o reducir el dominio de una o más variables hasta eliminar todos sus posibles valores (en cuyo caso el problema no tiene solución)

Constraint Satisfaction Problem

Formal definition

Formally, a constraint satisfaction problem is defined as a triple (X,D,C) where:

$X = \{X_1, \ldots, X_n\}$ is a set of variables,
$D = \{D_1, \ldots, D_n\}$ is a set of the respective domains of values, and
$C = \{C_1, \ldots, C_m\}$ is a set of constraints.

Conjuntos y operaciones sobre conjuntos: Relaciones (funciones)

Each variable X_i can take on the values in the nonempty domain D_i. Every constraint $C_j \in C$ is in turn a pair (t_j,R_j), where $t_j \subset X$ is a subset of k variables and R_j is an k-ary relation on the corresponding subset of domains D_j. An evaluation of the variables is a function from a subset of variables to a particular set of values in the corresponding subset of domains. An evaluation v satisfies a constraint (t_j,R_j) if the values assigned to the variables t_j satisfies the relation R_j.

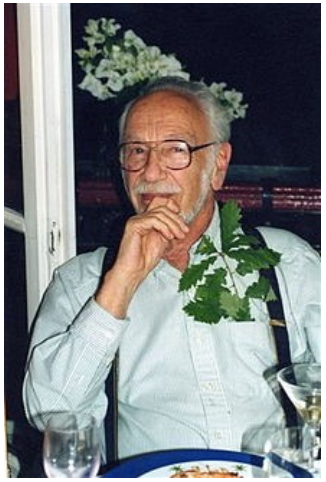
An evaluation is: 1) consistent if it does not violate any of the constraints, 2) complete if it includes all variables, 3) a solution if it is consistent and complete; such an evaluation is said to solve the constraint satisfaction problem.

Resolution: Backtracking technique (recursive algorithm).

Stable matching

Authors

In 1962, David Gale and Lloyd Shapley proved that, for any equal number of men and women, it is always possible to solve the SMP and make all marriages stable. They presented an algorithm to do so in 1962 and it was the first time to be formalised.



David Gale
1921-2008
mathematician and economist

Lloyd Shapley
1923 (92 años)
mathematician

Stable matching

Definiciones y ejemplos

	1st	2nd	3rd
Xavier	Amy	Bertha	Clare
Yancey	Bertha	Amy	Clare
Zeus	Amy	Bertha	Clare

Men's preference list

	1st	2nd	3rd
Amy	Yancey	Xavier	Zeus
Bertha	Xavier	Yancey	Zeus
Clare	Xavier	Yancey	Zeus

Women's preference list

Def. A matching S is a set of ordered pairs (m,w) with $m \in M$ and $w \in W$ s.t.

- Each man $m \in M$ appears in at most one pair of S.
- Each woman $w \in W$ appears in at most one pair of S.

Def. A matching S is perfect if $|S| = |M| = |W| = n$.

Stable matching

Definiciones y ejemplos

Men's	1st	2nd	3rd
Xavier	Amy	Bertha	Clare
Yancey	Bertha	Amy	Clare
Zeus	Amy	Bertha	Clare

Women's	1st	2nd	3rd
Amy	Yancey	Xavier	Zeus
Bertha	Xavier	Yancey	Zeus
Clare	Xavier	Yancey	Zeus

A perfect matching S = { (X,C), (Y,B), (Z,A) }

Def. Given a perfect matching S, man m and woman w are unstable if:

- m prefers w to his current partner. Example Xavier a Amy.
- w prefers m to her current partner. Amy a Xavier.

Key point. An unstable pair (m,w) could each improve partner by joint action

Stable matching

Definiciones y ejemplos

Def. A stable matching is a perfect matching with no unstable pairs.

Stable matching problem. Given the preference lists of n men and n women, find a stable matching (if one exists).

- Natural, desirable, and self-reinforcing condition.
- Individual self-interest prevents any man–woman pair from eloping (fugarse con la novia).

Men's	1st	2nd	3rd
Xavier	Amy	Bertha	Clare
Yancey	Bertha	Amy	Clare
Zeus	Amy	Bertha	Clare

Women's	1st	2nd	3rd
Amy	Yancey	Xavier	Zeus
Bertha	Xavier	Yancey	Zeus
Clare	Xavier	Yancey	Zeus

A perfect matching S = { (X,A), (Y,B), (Z,C) }

Stable matching

Algoritmo

GALE–SHAPLEY (preference lists for men and women)

INITIALIZE S to empty matching.
WHILE (some man m is unmatched and hasn't proposed to every woman)
    w ← first woman on m's list to whom m has not yet proposed.
    IF (w is unmatched)
        Add pair m–w to matching S.
    ELSE IF (w prefers m to her current partner m')
        Remove pair m'–w from matching S.
        Add pair m–w to matching S.
    ELSE
        w rejects m.

RETURN stable matching S.

Notas de implementación

1. Tipos de datos de:
   S: Conjunto como Lista
   $list_m$ y $list_w$: Listas / Tablas
2. Que significa:
   "No ha sido propuesto(a)"
   Implica llevar un registro.

Stable matching

Análisis del problema

Q. Do stable matchings always exist?

A1. Not obvious a priori.

A2. In fact, for a given problem instance, there may be several stable matchings.

Ver slides para demo

Observations (some need **proof**):

- Men propose to women in decreasing order of preference.
- Once a woman is matched, she never becomes unmatched, she only "trades up".
- Algorithm terminates after at most n^2 iterations of while loop.

Properties

- Termination.
- She never become unmatched.

Stable matching

Análisis del algoritmo

Q. How to implement GS algorithm efficiently?

- We need ordered lists (e.g. of women) with a pointer to the next proposal.
- A list of free men (in a stack or queue).
- Two arrays wife[m] y husband[w] with 0 (unmatched) or
  wife[m]=w, husband[w]=m.

Q. If there are multiples stable matchings, which one does GS find?

Q. Do all executions of GS algorithm yield the same stable matching?

- If so, which one?

Los participantes pueden engañar al algoritmo
Para salir beneficiados? No/Si (dar ejemplo)

Variants of Stable Matching Problem

Variantes del SMP

List of problems

- Stable Matching Problem (SMP) original
- Stable-Roommate Problem (SRP)
- Stable Marriage with Indifference (SMI)
- Stable Marriage with
  - Incomplete lists
  - Ties (ataduras, ligar)
    - Empates (más de un candidato en la misma posición)
  - Unacceptabe partners
  - Cheating strategies (manipulation of matching algorithms)

Veremos los 2 primeros

Variants of Stable Matching Problem

Variantes del SMP

The **Stable-Roommate Problem** (SRP)

- This is distinct from the SMP in that the SRP allows matches between any two elements, not just between classes of “men” and “women”.
- It is the problem of finding a stable matching—a matching in which there is no pair of elements where both members prefer their partner in a different matching over their partner in the stable matching.
- An efficient algorithm was given in:

  Irving, Robert W. (1985, Journal of Algorithms)

  "An efficient algorithm for the "stable roommates" problem"
- The algorithm will determine, for any instance of the problem, whether a stable matching exists, and if so, will find such a matching.
- **Irving**’s algorithm has $O(n^2)$ complexity, provided suitable data structures are used to implement the necessary manipulation of the preference lists and identification of rotations.

Variants of Stable Matching Problem

Variantes del SMP

Stable Marriage with Indifference

- In the classical version of the problem, each person must rank the members of the opposite sex in strict order of preference. However, in a real-world setting, a person may prefer two or more persons as equally favorable partner. Such tied preference is termed as **indifference**.
- A matching will be called **weakly stable** unless there is a couple each of whom strictly prefers the other to his/her partner in the matching.
- **Irving** has extended Gale–Shapley algorithm to provide such weakly stable matching in $O(n^2)$ time where n is size of stable marriage problem.
- A matching is **super-stable** if there is no couple each of whom either strictly prefers the other to his/her partner or is indifferent between them. **Irving** has modified the previous algorithm to check whether such super stable matching exists and outputs matching in $O(n^2)$ time if it exists.

Stable Matching Problem

Referencias

Implementaciones

- En varios lenguajes:
  - https://rosettacode.org/wiki/Stable_marriage_problem
  - http://www.geeksforgeeks.org/stable-marriage-problem/
  - Java: http://www.sanfoundry.com/java-program-gale-shapley-algorithm/
- Matlab
  - Tiene la función galeshapley.
  - https://www.mathworks.com/matlabcentral/fileexchange/44262-gale-shapley-stable-marriage-algorithm
- Python librería
  - https://pypi.org/project/matching/0.1.1/
- Github
  - Multiples versiones
  - https://github.com/topics/gale-shapley-algorithm
- Generadores
  - http://sephlietz.com/gale-shapley/

Several topics

Varios temas

Notas históricas

Problemas abiertos

Tareas

URLs

A recordar

Recommended reading and Web sites

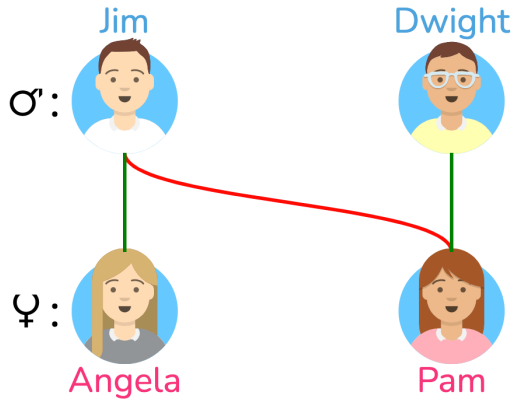
Lecturas recomendadas

Hay varios articulos interesantes sobre el algoritmo de GS:

- http://cran.at.r-project.org/web/packages/matchingMarkets/vignettes/matching.pdf.

Algunos simuladores en linea:

- Gráfico: http://mathsite.math.berkeley.edu/smp/smp.html
- Sencillo: http://sephlietz.com/gale-shapley/
- Explicación visual
  - https://uw-cse442-wi20.github.io/FP-cs-algorithm/



Homeworks: programming

Tareas: programación

Tarea

Obligatoria

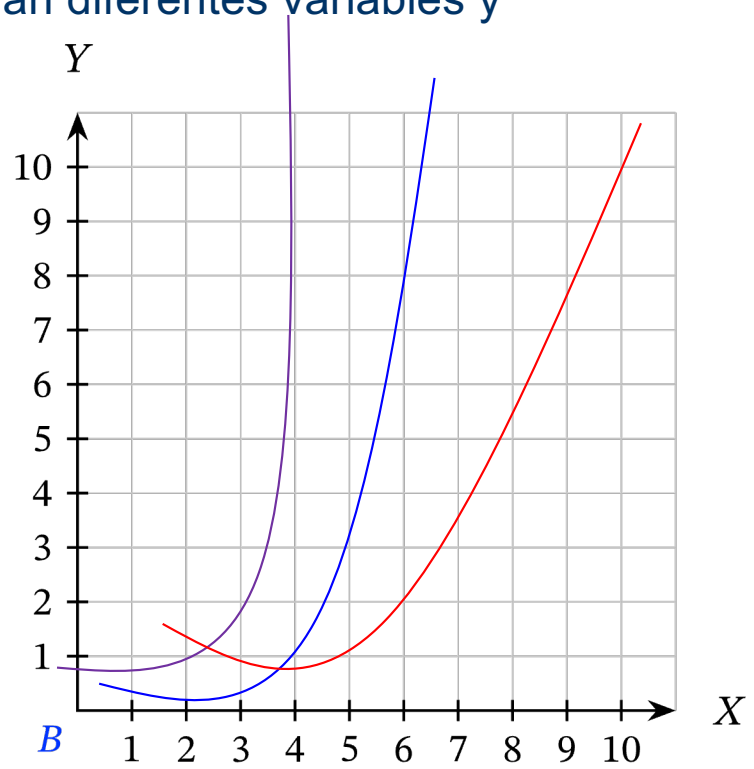
- Implementar el algoritmo de SM (Gale & Shapley):
  - Hacer modificaciones y probar con diferentes variantes.
  - Trabajo individual.
  - Usar un lenguaje de alto nivel: Java, C#, etc.
- Ejecutar el algoritmo "varias veces" para medir experimentalmente el tiempo de ejecución.
  - Hay que hacer varias ejecuciones con diferentes valores de entrada del algoritmo.
    - Número de hombres/mujeres. Lo van a incrementar de manera exponencial.
      Formas de crecimiento: lineal, logaritmica, etc.
    - La elección de los TADs utilizados.
  - Usar alguna libreria de graficación como gplot. Medir el tiempo de ejecución. Opcionalmente: la cantidad de memoria.
  - Grafica: En el eje de las X los N hombres/mujeres. En el eje de las Y el tiempo.

Homeworks: programming

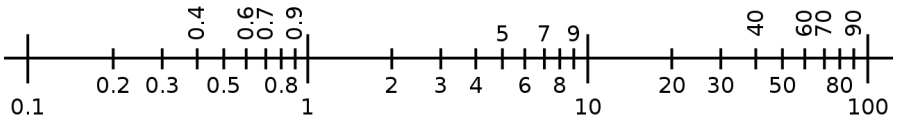
Tareas: programación

Benchmark

- Evaluación experimental:
  - Varias ejecuciones en las cuales se exploran diferentes variables y versiones.
- Graficación
  - Eje X: número de datos n.
  - Eje Y: tiempo de ejecución.
  - Experimentos:
    - E1: Algoritmo 1
    - E2: Algoritmo 2
    - E3: A1 con arreglos
    - E4: A1 con listas.



Eje X o Y Escala logarítmica



Recommended reading and Web sites

Lecturas recomendadas

- Leer capítulo 1 del libro de Kleinberg
- Leer artículos
  - Hacer resumen de: SOTA, Linea del tiempo, otros análisis.
  - Hacer una análisis de los artículos en cuanto:
    - Es un artículo Indexado?
    - Número de citas?

Lessons learned

Que se debe recordar

Lo más importante es:

1. Modelación matemática de los problemas.
    1. Hacer pruebas matemáticas (Proof of)
    2. Correctitud (correctness) y Completitud.
2. Conocimiento básico de Estructuras de Datos: pilas, colas. etc.

Stable matching

Algoritmos

Versiones

- Del libro de Kleinberg
- Geeks for Geeks
  - http://www.geeksforgeeks.org/stable-marriage-problem/
- Wikipedia.

Stable matching

Algoritmo

GALE–SHAPLEY (preference lists for men and women)

INITIALIZE S to empty matching.
WHILE (some man m is unmatched and hasn't proposed to every woman)
    w ← first woman on m's list to whom m has not yet proposed.
    IF (w is unmatched)
        Add pair m–w to matching S.
    ELSE IF (w prefers m to her current partner m')
        Remove pair m'–w from matching S.
        Add pair m–w to matching S.
    ELSE
        w rejects m.

RETURN stable matching S.

1 while
2 if conditions

Stable matching

Algoritmos

Kleinberg
Pag. 11

Initially all m∈M and w∈W are free

While there is a man m who is free and hasn't proposed to every woman
  Choose such a man m
  Let w be the highest-ranked woman in m's preference list
    to which m has not yet proposed
  If w is free then
    (m,w) become engaged
  Else w is currently engaged to m′
    If w prefers m′ to m then
      m remains free
    Else w prefers m to m′
      (m,w) become engaged
      m′ becomes free
    Endif
  Endif
Endwhile

Return the set S of engaged pairs

Stable matching

Algoritmos

Geeks for Geeks

```
Initialize all men and women to free
while there exist a free man m who still has a woman w to propose to {
    w = m's highest ranked such woman to whom he has not yet proposed
    if w is free
        (m, w) become engaged
    else some pair (m', w) already exists
            if w prefers m to m’
                (m, w) become engaged
                 m' becomes free
            else
                (m', w) remain engaged
}
```

Stable matching

Algoritmos

Wikipedia

```
function stableMatching {
    Initialize all m ∈ M and w ∈ W to free
     while ∃ free man m who still has a woman w to propose to {
          w = first woman on m's list to whom m has not yet proposed
          if w is free
              (m, w) become engaged
          else some pair (m', w) already exists
               if w prefers m to m'
                    m' becomes free
                    (m, w) become engaged
               else
                    (m', w) remain engaged
      }
}
```

The end

Contacto

Raúl Acosta Bermejo

http:www.cic.ipn.mx
http://www.ciseg.cic.ipn.mx/

racostab@ipn.mx
racosta@cic.ipn.mx

57-29-60-00
Ext. 56652