Assignments and Matchings

AU4606: Network Optimization

Xiaoming Duan
Department of Automation
Shanghai Jiao Tong University

December 18, 2023

Last few lectures

- Submodular function optimization
  - Definition of submodularity
  - Monotone submodular maximization and greedy
  - Lovász extension and unconstrained submodular minimization

Today

1. What is a matching problem?
2. Matching in bipartite graphs
3. Stable marriage problem

Today

1. What is a matching problem?
2. Matching in bipartite graphs
3. Stable marriage problem

Matching: formal definitions

Matching

Given an undirected graph $G = (N, E)$, a matching M is a subset of arcs with the property that no two arcs of M are incident to the same node.

Matching: formal definitions

Matching

Given an undirected graph $G = (N, E)$, a matching M is a subset of arcs with the property that no two arcs of M are incident to the same node.

Perfect matching

Given an undirected graph $G = (N, E)$, a perfect matching M is a matching where every node is incident to exactly one arc in the matching.

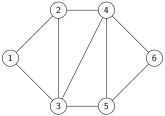
Matching: formal definitions

Matching

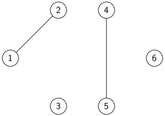
Given an undirected graph $G = (N, E)$, a matching M is a subset of arcs with the property that no two arcs of M are incident to the same node.

Perfect matching

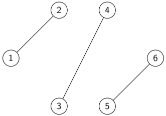
Given an undirected graph $G = (N, E)$, a perfect matching M is a matching where every node is incident to exactly one arc in the matching.



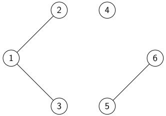
(a) Graph G



(b) matching



(c) Perfect mat.



(d) Not a mat.

Matching: problems of interest

A matching is a subset of arcs, optimality is defined w.r.t.

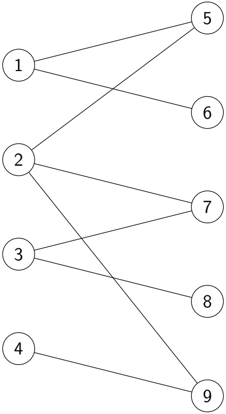
- Cardinality: find a matching that has the largest number of arcs
- Weights: find a matching with max/min total edge weights
- Weights/restricted: find a perfect matching with max/min weights
- Stability: find a "stable" perfect matching

Matching in different graph types

- Matching in bipartite graphs $(N_1 \cup N_2, E)$
- Matching in general undirected graphs (N, E)

Today

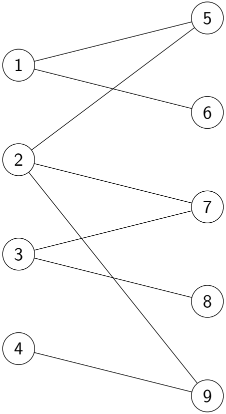
Bipartite cardinality matching



Given a bipartite graph $G = (N_1 \cup N_2, E)$

- Find a matching that contains the most number of arcs

Bipartite cardinality matching

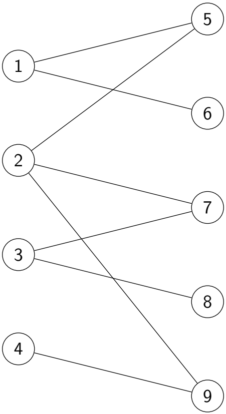


Given a bipartite graph $G = (N_1 \cup N_2, E)$

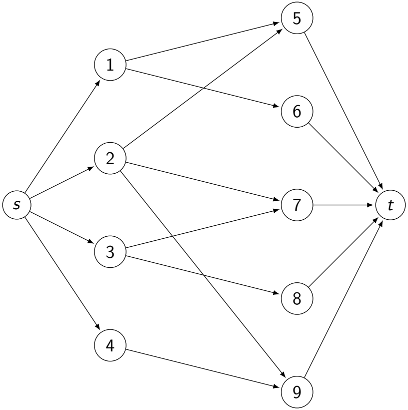
- Find a matching that contains the most number of arcs

Can we solve this problem using ideas we have already studied?

Bipartite cardinality matching & maximum flow



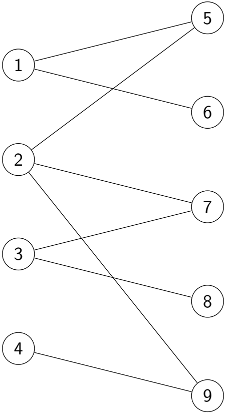
(a) Bipartite matching



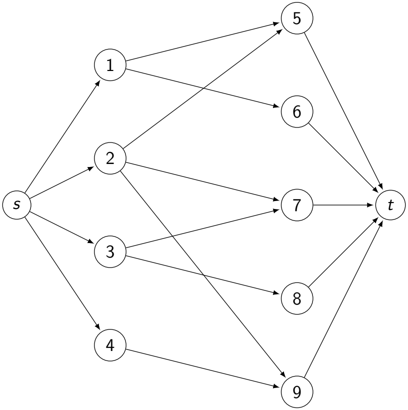
(b) Maximum flow

- Construct a maximum flow problem with unit capacities on arcs
- A maximum matching consists of arcs having unit flows in a max flow

Bipartite cardinality matching & maximum flow



(a) Bipartite matching



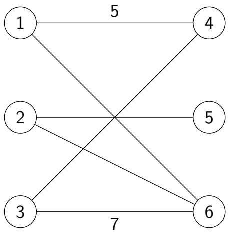
(b) Maximum flow

Characteristics of the flow network (unit capacity simple network)

- Each arc has unit capacity constraint
- Each node ($\neq$ s, t) has at most 1 incoming or at most 1 outgoing arc

Faster maximum flow algorithms are available ($O(m\sqrt{n})$)

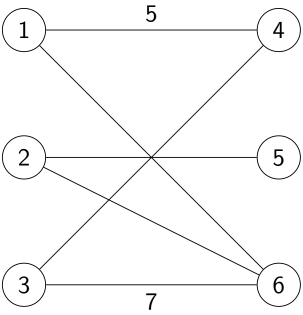
Bipartite weighted matching: formulation



Given a weighted bipartite graph $G = (N_1 \cup N_2, E)$ (assignment problem)

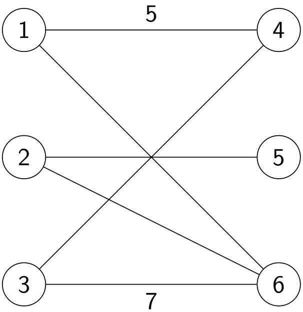
- Find a perfect matching such that total edge weights is minimized

Bipartite weighted matching: formulation



$$
\begin{aligned}
\text{minimize} \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} \\
\text{subject to} \quad & \sum_{j:(i,j)\in A} x_{ij} = 1 \quad \text{for } i \in N_1 \\
& \sum_{j:(j,i)\in A} x_{ji} = 1 \quad \text{for } i \in N_2 \\
& x_{ij} \geq 0 \quad \text{for each } (i,j) \in A.
\end{aligned}
$$

Bipartite weighted matching: formulation



$$\begin{aligned} \text{minimize} \quad & \sum_{(i,j)\in A} c_{ij}x_{ij} \\ \text{subject to} \quad & \sum_{j:(i,j)\in A} x_{ij} = 1 \quad \text{for } i \in N_1 \\ & \sum_{j:(j,i)\in A} x_{ji} = 1 \quad \text{for } i \in N_2 \\ & x_{ij} \geq 0 \quad \text{for each } (i,j) \in A. \end{aligned}$$

Minimum cost flow problem

$$\begin{aligned} \text{minimize} \quad & \sum_{(i,j)\in A} c_{ij}f_{ij} \\ \text{subject to} \quad & \sum_{j:(i,j)\in A} f_{ij} - \sum_{j:(j,i)\in A} f_{ji} = b(i) \\ & 0 \leq f_{ij} \leq u_{ij} \quad \text{for each } (i,j) \in A. \end{aligned}$$

Minimum weight bipartite perfect matching is minimum cost flow.

Bipartite weighted matching: an old example

Problem statement

Given

1. a set $S = \{a_1, \ldots, a_n\}$ of n unit-time tasks;
2. a set of n integer deadlines $d(a_1), \ldots, d(a_n)$;
3. a set of nonnegative penalties $w(a_1), \ldots, w(a_n)$;

Find a schedule for S that minimizes total penalties of missed deadlines.

Bipartite weighted matching: an old example

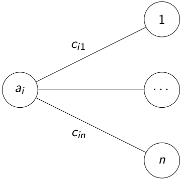
Problem statement

Given

1. a set $S = \{a_1, \dots, a_n\}$ of n unit-time tasks;
2. a set of n integer deadlines $d(a_1), \dots, d(a_n)$;
3. a set of nonnegative penalties $w(a_1), \dots, w(a_n)$;

Find a schedule for S that minimizes total penalties of missed deadlines.

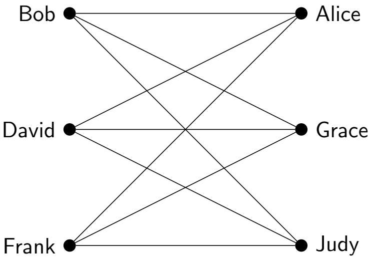
It is a bipartite weighted matching problem!



- $c_{ij} = 0$ if $d(a_i) \geq j$
- $c_{ij} = w(a_i)$ if $d(a_i) < j$

Today

Stable marriage problem: formulation



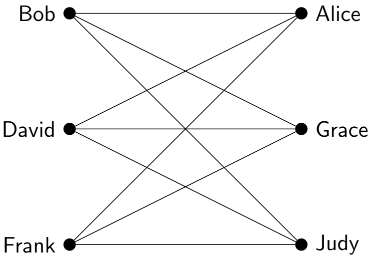
Bob: Grace $>$ Alice $>$ Judy
David: Grace $>$ Judy $>$ Alice
Frank: Alice $>$ Judy $>$ Grace

Alice: Bob $>$ David $>$ Frank
Grace: David $>$ Bob $>$ Frank
Judy: Frank $>$ David $>$ Bob

Given bipartite graph $G = (N_1 \cup N_2, E)$, $|N_1| = |N_2|$ and $E = N_1 \times N_2$

- Each node in N_i ranks nodes in N_{3-i} according to its preference
    - e.g., Bob may prefer Grace to Alice, and he prefers Alice to Judy

Stable marriage problem: formulation



Bob: Grace > Alice > Judy
David: Grace > Judy > Alice
Frank: Alice > Judy > Grace

Alice: Bob > David > Frank
Grace: David > Bob > Frank
Judy: Frank > David > Bob

Given bipartite graph $G = (N_1 \cup N_2, E)$, $|N_1| = |N_2|$ and $E = N_1 \times N_2$

- Each node in N_i ranks nodes in N_{3-i} according to its preference
    - e.g., Bob may prefer Grace to Alice, and he prefers Alice to Judy

Find a *stable* perfect matching

Stable perfect matching

A perfect matching is stable if no pair of unmatched nodes that prefer each other to their matched partner.

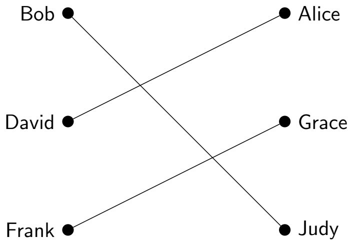
Stable marriage problem: examples

Bob: Grace > Alice > Judy
David: Grace > Judy > Alice
Frank: Alice > Judy > Grace

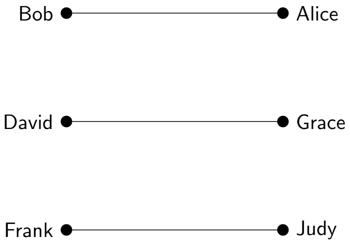
Alice: Bob > David > Frank
Grace: David > Bob > Frank
Judy: Frank > David > Bob

Stable perfect matching

A perfect matching is stable if no unmatched pair of nodes that prefer each other to their matched partner.



(a) Unstable matching (Alic, Bob)



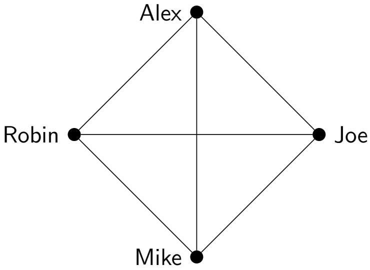
(b) Stable matching

Stable marriage problem: existence of solution

Does there always exist a stable perfect matching?

Stable marriage problem: existence of solution

Does there always exist a stable perfect matching?



Alex: Joe $>$ Robin $>$ Mike
Joe: Robin $>$ Alex $>$ Mike
Robin: Alex $>$ Joe $>$ Mike

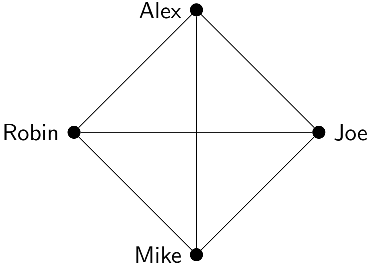
Mike: does not matter

Buddy matching

There is no stable buddy matching among the four people the above figure.

Stable marriage problem: existence of solution

Does there always exist a stable perfect matching?



Alex: Joe > Robin > Mike
Joe: Robin > Alex > Mike
Robin: Alex > Joe > Mike

Mike: does not matter

Buddy matching

There is no stable buddy matching among the four people the above figure.

Stable marriage

There is always a stable perfect matching for the stable marriage problem.

Stable marriage problem: algorithms

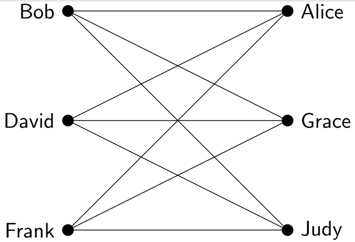
Propose and reject (iterative):

- Each man proposes to his most preferred woman
- Each woman rejects all proposals except her most preferred man
- If a man gets rejected, he proposes to his next preferred woman
- If a woman receives new proposals, she compares them to current one

We will show

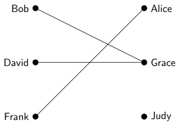
- The algorithm terminates
- The algorithm terminates with a stable perfect matching

Stable marriage problem: running example



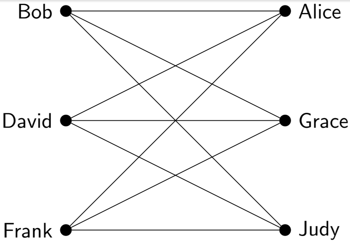
Bob: Grace > Alice > Judy
David: Grace > Judy > Alice
Frank: Alice > Judy > Grace

Alice: Bob > David > Frank
Grace: David > Bob > Frank
Judy: Frank > David > Bob



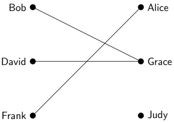
(a) Propose

Stable marriage problem: running example

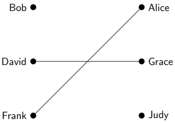


Bob: Grace > Alice > Judy
David: Grace > Judy > Alice
Frank: Alice > Judy > Grace

Alice: Bob > David > Frank
Grace: David > Bob > Frank
Judy: Frank > David > Bob

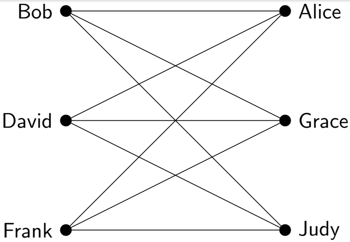


(a) Propose



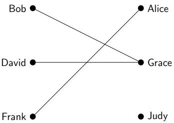
(b) Reject

Stable marriage problem: running example

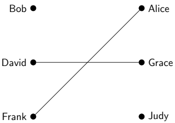


Bob: Grace > Alice > Judy
David: Grace > Judy > Alice
Frank: Alice > Judy > Grace

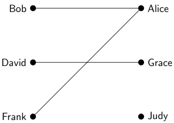
Alice: Bob > David > Frank
Grace: David > Bob > Frank
Judy: Frank > David > Bob



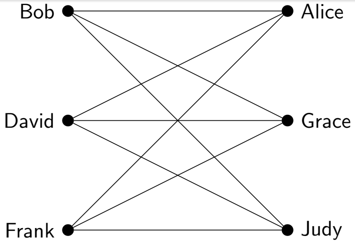
(a) Propose



(b) Reject

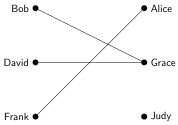


(c) Propose

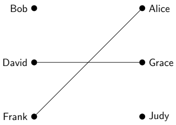


Bob: Grace > Alice > Judy
David: Grace > Judy > Alice
Frank: Alice > Judy > Grace

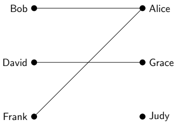
Alice: Bob > David > Frank
Grace: David > Bob > Frank
Judy: Frank > David > Bob



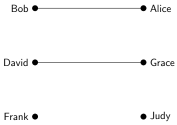
(a) Propose



(b) Reject

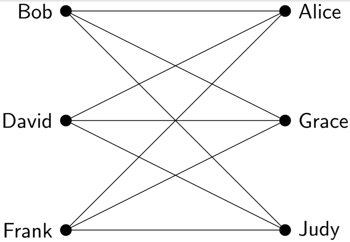


(c) Propose



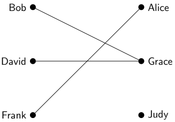
(d) Reject

Stable marriage problem: running example

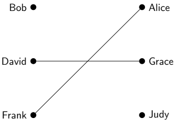


Bob: Grace > Alice > Judy
David: Grace > Judy > Alice
Frank: Alice > Judy > Grace

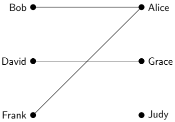
Alice: Bob > David > Frank
Grace: David > Bob > Frank
Judy: Frank > David > Bob



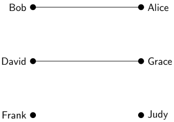
(a) Propose



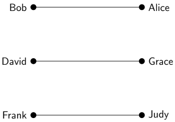
(b) Reject



(c) Propose



(d) Reject



(e) Propose

Propose and reject

Algorithm Propose and reject

$L \leftarrow N_1$
while $L \neq \emptyset$ **do**
  Get a man m from L
  m proposes to his most preferred woman w that has not rejected him before
  if w accepts m and rejects the currently mathced m' **then**
    Remove m from L
    Add m' to L
  end if
end while

Complexity

The propose and reject algorithm terminates in $O(n^2)$.

Correctness

The propose and reject algorithm terminates with a stable perfect matching.

Properties of the stable perfect matching

Note there could be multiple stable perfect matching

Optimal (pessimal) spouse

Given a stable marriage problem, an optimal (pessimal) spouse for a person is the most (least) preferred partner that the person can be matched to over all stable perfect matching.

Properties of the stable perfect matching

Note there could be multiple stable perfect matching

Optimal (pessimal) spouse

Given a stable marriage problem, an optimal (pessimal) spouse for a person is the most (least) preferred partner that the person can be matched to over all stable perfect matching.

Men are matched to optimal spouses

The propose and reject algorithm constructs a stable perfect matching where men obtain their optimal spouses.

Women are matched to pessimal spouses

The propose and reject algorithm constructs a stable perfect matching where women obtain their pessimal spouses.

Upcoming

Week 1-8 (AU4606 & AI4702):

- Introduction (1 lecture)
- Preparations (3 lectures)
  - basics of graph theory
  - algorithm complexity and data structure
  - graph search algorithm
- Shortest path problems (3 lectures)
- Maximum flow problems (5 lectures)
- Minimum cost flow problems (3 lectures)
- Introduction to multi-agent systems (1 lecture)
- Introduction to cloud networks (1 lecture)

Week 9-16 (AU4606):

- Simplex and network simplex methods (1 lecture)
- Global minimum cut problems (1.5 lectures)
- Minimum spanning tree problems (1.5 lectures)
- Submodular function optimization (2 lectures)
- Optimal assignments and matching (2 lectures)