Stochastic Dynamic Matching in Graphs

Céline Comte — c.m.comte@tue.nl
Eindhoven University of Technology

Inria — DYOGENE Seminar — January 17, 2022

Outline

Model and notation

Performance under the first-come-first-matched policy
Comte, Stochastic Models (2022)

Matching rates under an arbitrary policy
Comte, Mathieu, and Bušić, arXiv:2112.14457 (2022)

Outline

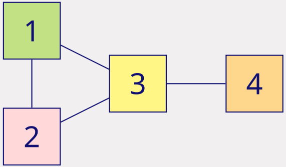
Model and notation

Performance under the first-come-first-matched policy
Comte, Stochastic Models (2022)

Matching rates under an arbitrary policy
Comte, Mathieu, and Bušić, arXiv:2112.14457 (2022)

Compatibility graph

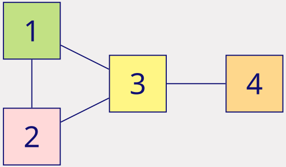
Graph $G = (V, E)$ undirected, connected, without loop



Compatibility graph

Graph $G = (V, E)$ undirected, connected, without loop

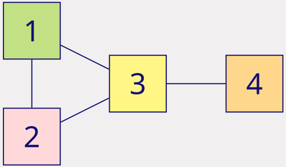
- Nodes $V = \{1, 2, \ldots, n\} \to$ items



Compatibility graph

Graph $G = (V, E)$ undirected, connected, without loop

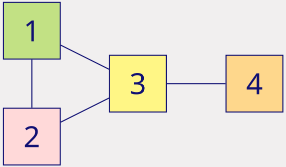
- Nodes $V = \{1, 2, \ldots, n\} \rightarrow$ items
- Edges $E = \{1, 2, \ldots, m\} \rightarrow$ possible matches



Compatibility graph

Graph $G = (V, E)$ undirected, connected, without loop

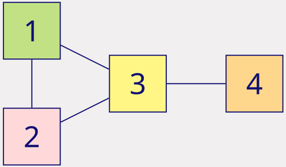
- Nodes $V = \{1, 2, \ldots, n\} \rightarrow$ item classes
- Edges $E = \{1, 2, \ldots, m\} \rightarrow$ possible matches



Compatibility graph

Graph $G = (V, E)$ undirected, connected, without loop

- Nodes $V = \{1, 2, \dots, n\} \to$ item classes
- Edges $E = \{1, 2, \dots, m\} \to$ possible matches

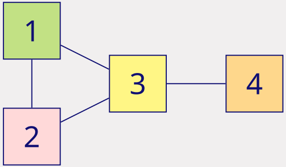


- $V_i = \{\text{neighbors of node } i\}$
  $V(U) = \bigcup_{i \in U} V_i$ for each $U \subseteq V$

Compatibility graph

Graph $G = (V, E)$ undirected, connected, without loop

- Nodes $V = \{1, 2, \dots, n\}$ $\rightarrow$ item classes
- Edges $E = \{1, 2, \dots, m\}$ $\rightarrow$ possible matches

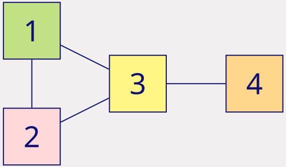


- $V_i = \{\text{neighbors of node } i\}$
  $V(U) = \bigcup_{i \in U} V_i$ for each $U \subseteq V$
- $E_i = \{\text{edges containing node } i\}$

Compatibility graph

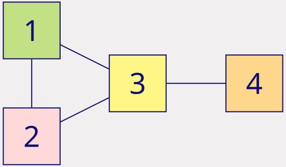
Graph $G = (V, E)$ undirected, connected, without loop

- Nodes $V = \{1, 2, \dots, n\}$ → item classes
- Edges $E = \{1, 2, \dots, m\}$ → possible matches

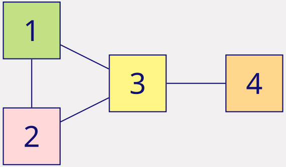


- $V_i = \{\text{neighbors of node } i\}$
  $V(U) = \bigcup_{i \in U} V_i$ for each $U \subseteq V$
- $E_i = \{\text{edges containing node } i\}$
- Independent sets
  $\mathbb{I} = \{\{1\}, \{2\}, \{3\}, \{4\}, \{1, 4\}, \{2, 4\}\}$
  $\mathbb{I}_0 = \mathbb{I} \cup \{\emptyset\}$

Random dynamics

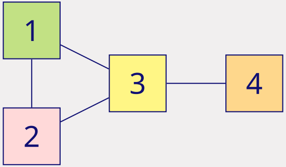


Random dynamics



Class-i items arrive as a Poisson process with rate μ_i

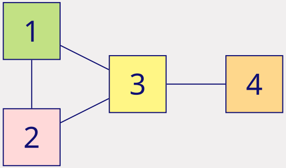
Random dynamics



Class-i items arrive as a Poisson process with rate μ_i

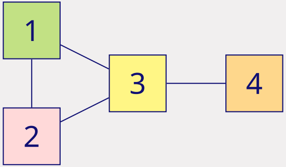
TU/e

Random dynamics

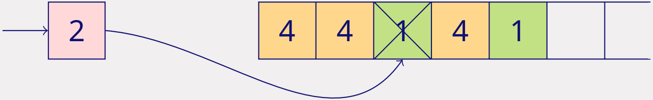


Class-*i* items arrive as a Poisson process with rate μ_i

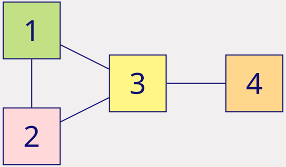
Random dynamics



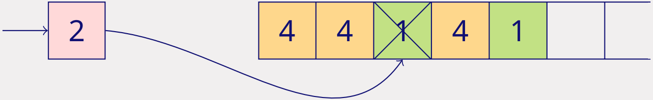
Class-i items arrive as a Poisson process with rate μ_i



Random dynamics



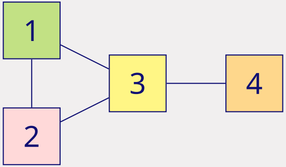
Class-i items arrive as a Poisson process with rate μ_i



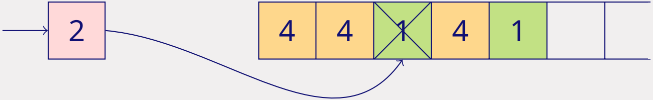
The system dynamics depend on:

- the graph $G = (V, E)$,
- the vector $\mu = (\mu_1, \mu_2, \ldots, \mu_n)$,
- the matching policy.

Random dynamics



Class-i items arrive as a Poisson process with rate μ_i

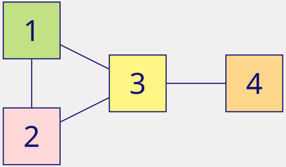


The system dynamics depend on:

- the graph $G = (V, E)$,
- the vector $\mu = (\mu_1, \mu_2, \ldots, \mu_n)$,
- the matching policy.

Additional notation:

- Arrival rate $\mu(U) = \sum_{i \in U} \mu_i$, $U \subseteq V$
- Load $\rho(I) = \frac{\mu(I)}{\mu(V(I))}$, $I \in \mathbb{I}$



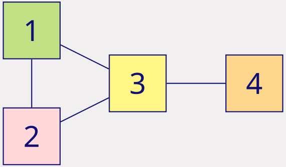
“Stabilizability”

(Bušić, Gupta, and Mairesse, 2013) (Mairesse and Moyal, 2016)

- The matching problem (G, μ) is stabilizable

"Stabilizability"

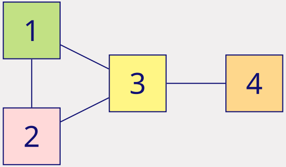
(Bušić, Gupta, and Mairesse, 2013) (Mairesse and Moyal, 2016)



- The matching problem (G, μ) is stabilizable if and only if $\rho(I) < 1$ for each $I \in \mathbb{I}$.

“Stabilizability”

(Bušić, Gupta, and Mairesse, 2013) (Mairesse and Moyal, 2016)

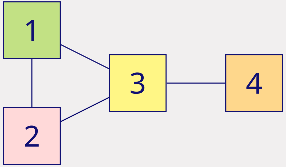


- The matching problem (G, μ) is stabilizable if and only if $\rho(I) < 1$ for each $I \in \mathbb{I}$.

$$\begin{cases} \rho(\{1\}) = \dfrac{\mu_1}{\mu_2 + \mu_3} & \rho(\{2\}) = \dfrac{\mu_2}{\mu_1 + \mu_3} & \rho(\{3\}) = \dfrac{\mu_3}{\mu_1 + \mu_2 + \mu_4} \\ \rho(\{4\}) = \dfrac{\mu_4}{\mu_3} & \rho(\{1,4\}) = \dfrac{\mu_1 + \mu_4}{\mu_2 + \mu_3} & \rho(\{2,4\}) = \dfrac{\mu_2 + \mu_4}{\mu_1 + \mu_3} \end{cases}$$

“Stabilizability”

(Bušić, Gupta, and Mairesse, 2013) (Mairesse and Moyal, 2016)



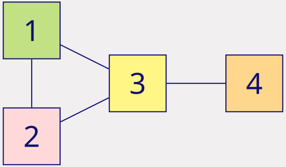
- The matching problem (G, μ) is stabilizable if and only if $\rho(I) < 1$ for each $I \in \mathbb{I}$.

$$\begin{cases} \rho(\{1\}) = \dfrac{\mu_1}{\mu_2 + \mu_3} & \rho(\{2\}) = \dfrac{\mu_2}{\mu_1 + \mu_3} & \rho(\{3\}) = \dfrac{\mu_3}{\mu_1 + \mu_2 + \mu_4} \\ \rho(\{4\}) = \dfrac{\mu_4}{\mu_3} & \rho(\{1,4\}) = \dfrac{\mu_1 + \mu_4}{\mu_2 + \mu_3} & \rho(\{2,4\}) = \dfrac{\mu_2 + \mu_4}{\mu_1 + \mu_3} \end{cases}$$

- The compatibility graph G is stabilizable

"Stabilizability"

(Bušić, Gupta, and Mairesse, 2013) (Mairesse and Moyal, 2016)



- The matching problem (G, μ) is stabilizable if and only if $\rho(I) < 1$ for each $I \in \mathbb{I}$.

$$\begin{cases} \rho(\{1\}) = \dfrac{\mu_1}{\mu_2 + \mu_3} & \rho(\{2\}) = \dfrac{\mu_2}{\mu_1 + \mu_3} & \rho(\{3\}) = \dfrac{\mu_3}{\mu_1 + \mu_2 + \mu_4} \\ \rho(\{4\}) = \dfrac{\mu_4}{\mu_3} & \rho(\{1,4\}) = \dfrac{\mu_1 + \mu_4}{\mu_2 + \mu_3} & \rho(\{2,4\}) = \dfrac{\mu_2 + \mu_4}{\mu_1 + \mu_3} \end{cases}$$

- The compatibility graph G is stabilizable if and only if G is non-bipartite.

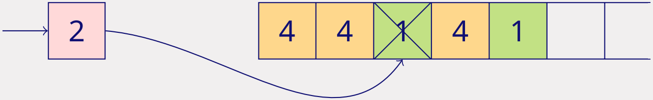
Outline

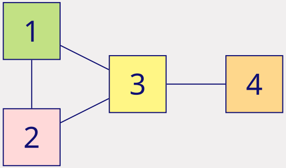
Model and notation

Performance under the first-come-first-matched policy
Comte, Stochastic Models (2022)

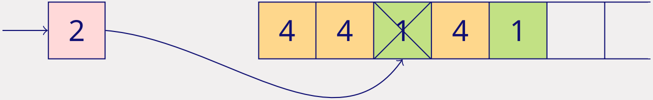
Matching rates under an arbitrary policy
Comte, Mathieu, and Bušić, arXiv:2112.14457 (2022)

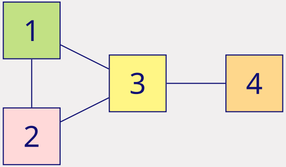
First-come-first-matched policy





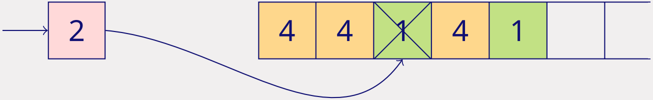
First-come-first-matched policy

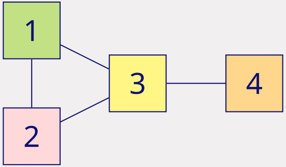




- Perceived as “fair”, greedy, easy to implement, easy to analyze.

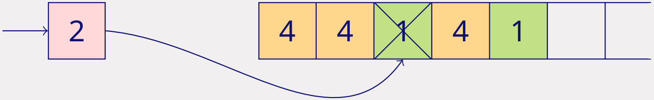
First-come-first-matched policy

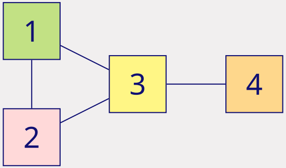




- Perceived as “fair”, greedy, easy to implement, easy to analyze.
- (Moyal, Bušić, and Mairesse, 2021) derives:
    - the necessary and sufficient stability condition,
    - the product-form stationary distribution of the “detailed” state.

First-come-first-matched policy

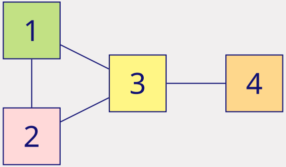




- Perceived as “fair”, greedy, easy to implement, easy to analyze.
- (Moyal, Bušić, and Mairesse, 2021) derives:
    - the necessary and sufficient stability condition,
    - the product-form stationary distribution of the “detailed” state.

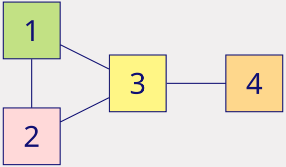
What is the long-term performance under first-come-first-matched?

Calculate long-term performance metrics



- This is an order-independent loss queue!

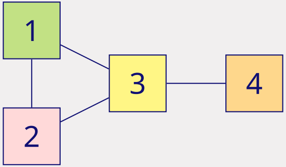
Calculate long-term performance metrics



- This is an order-independent loss queue!
- Stationary distribution of the set of unmatched classes:

$$\pi(I) = \frac{\rho(I)}{1 - \rho(I)} \left(\sum_{i \in I} \frac{\mu_i}{\mu(I)} \pi(I \setminus \{i\}) \right), \quad I \in \mathbb{I}.$$

Calculate long-term performance metrics

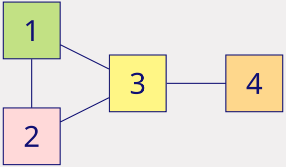


- This is an order-independent loss queue!
- Stationary distribution of the set of unmatched classes:

$$\pi(I) = \frac{\rho(I)}{1 - \rho(I)} \left(\sum_{i \in I} \frac{\mu_i}{\mu(I)} \pi(I \setminus \{i\}) \right), \quad I \in \mathbb{I}.$$

The value of $\pi(\emptyset)$ follows by normalization.

Calculate long-term performance metrics



- This is an order-independent loss queue!
- Stationary distribution of the set of unmatched classes:

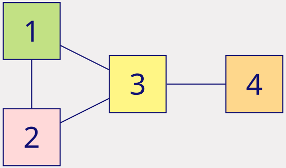
$$\pi(I) = \frac{\rho(I)}{1 - \rho(I)} \left(\sum_{i \in I} \frac{\mu_i}{\mu(I)} \pi(I \setminus \{i\}) \right), \quad I \in \mathbb{I}.$$

The value of $\pi(\emptyset)$ follows by normalization.

- Waiting probability of class *i*:

$$\omega_i = \sum_{\substack{I \in \mathbb{I}_0: \\ i \notin V(I)}} \pi(I),$$

Calculate long-term performance metrics



- This is an order-independent loss queue!
- Stationary distribution of the set of unmatched classes:

$$\pi(I) = \frac{\rho(I)}{1 - \rho(I)} \left(\sum_{i \in I} \frac{\mu_i}{\mu(I)} \pi(I \setminus \{i\}) \right), \quad I \in \mathbb{I}.$$

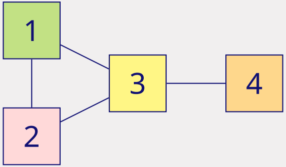
  The value of $\pi(\emptyset)$ follows by normalization.

- Waiting probability of class i:

$$\omega_i = \sum_{\substack{I \in \mathbb{I}_0: \\ i \notin V(I)}} \pi(I), \quad \text{which implies} \quad \frac{\sum_{i \in V} \mu_i \omega_i}{\sum_{i \in V} \mu_i} = \frac{1}{2}.$$

TU/e

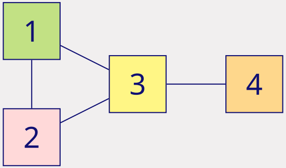
Calculate long-term performance metrics



- Mean number of unmatched items:

$$L = \sum_{I \in \mathbb{I}} \ell(I), \quad \text{with} \quad \ell(I) = \frac{\pi(I)}{1 - \rho(I)} + \frac{\rho(I)}{1 - \rho(I)} \left(\sum_{i \in I} \frac{\mu_i}{\mu(I)} \ell(I \setminus \{i\}) \right).$$

Calculate long-term performance metrics

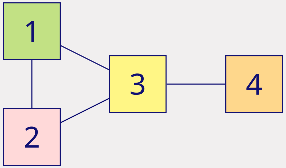


- Mean number of unmatched items:

$$L = \sum_{I \in \mathbb{I}} \ell(I), \quad \text{with} \quad \ell(I) = \frac{\pi(I)}{1 - \rho(I)} + \frac{\rho(I)}{1 - \rho(I)} \left(\sum_{i \in I} \frac{\mu_i}{\mu(I)} \ell(I \setminus \{i\}) \right).$$

The mean waiting time of an item follows using Little's law.

Calculate long-term performance metrics

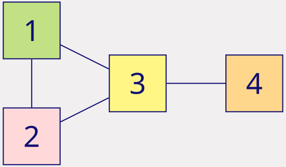


- Mean number of unmatched items:

$$L = \sum_{I \in \mathbb{I}} \ell(I), \quad \text{with} \quad \ell(I) = \frac{\pi(I)}{1 - \rho(I)} + \frac{\rho(I)}{1 - \rho(I)} \left(\sum_{i \in I} \frac{\mu_i}{\mu(I)} \ell(I \setminus \{i\}) \right).$$

  The mean waiting time of an item follows using Little's law.
- More detailed formulas for the per-class performance.

Calculate long-term performance metrics

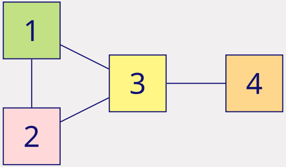


- Mean number of unmatched items:

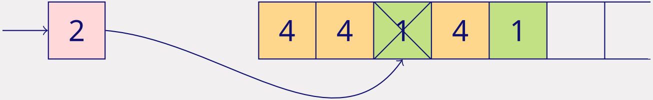
$$L = \sum_{I \in \mathbb{I}} \ell(I), \quad \text{with} \quad \ell(I) = \frac{\pi(I)}{1 - \rho(I)} + \frac{\rho(I)}{1 - \rho(I)} \left(\sum_{i \in I} \frac{\mu_i}{\mu(I)} \ell(I \setminus \{i\}) \right).$$

  The mean waiting time of an item follows using Little's law.
- More detailed formulas for the per-class performance.
- Similar results for stochastic bipartite matching model (Comte and Dorsman, ASMTA, 2021).

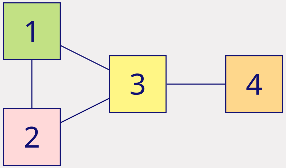
Calculate long-term performance metrics



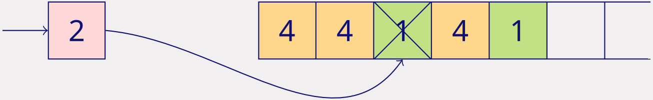
- Matching rate along edge $k = \{i, j\}$: mean number of matches per time unit between classes i and j.



Calculate long-term performance metrics

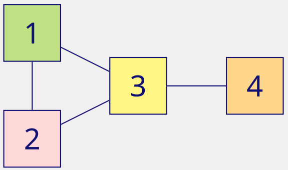


- Matching rate along edge $k = \{i, j\}$: mean number of matches per time unit between classes i and j.

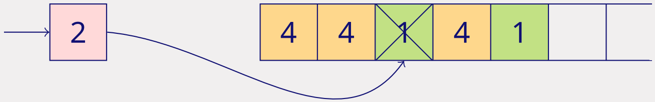


- Closed-form expression: consider a finer partition of the state space.

Calculate long-term performance metrics

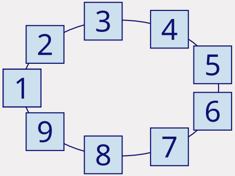


- Matching rate along edge $k = \{i, j\}$: mean number of matches per time unit between classes i and j.

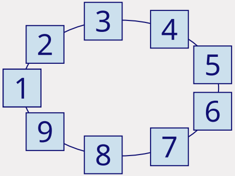


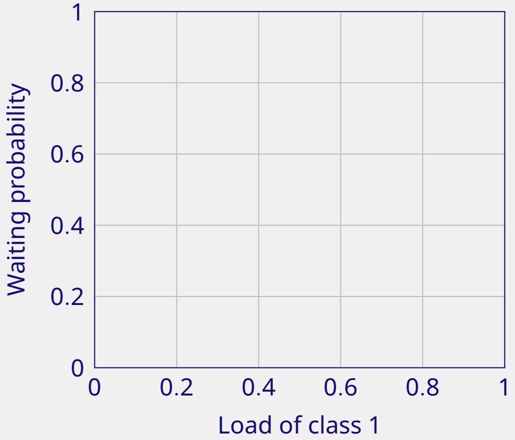
- Closed-form expression: consider a finer partition of the state space.
- More in a few slides...

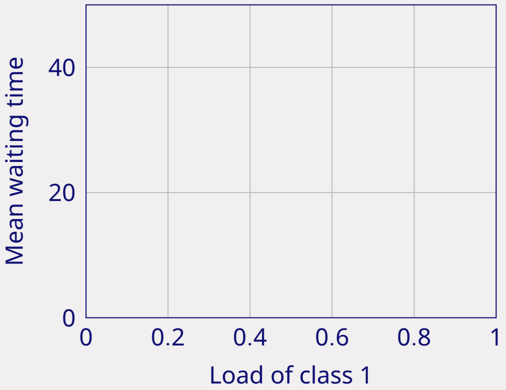
Numerical results: Cycle



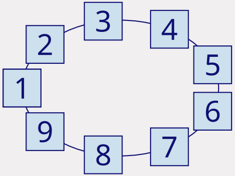
Numerical results: Cycle

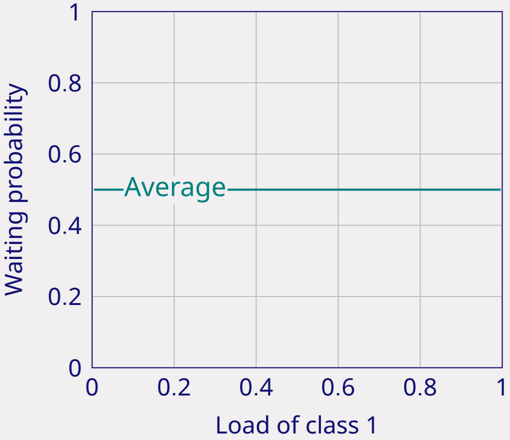


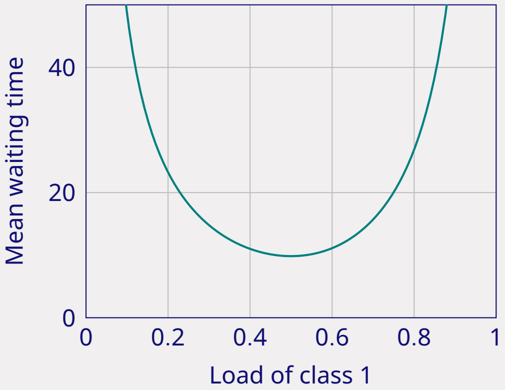




Numerical results: Cycle

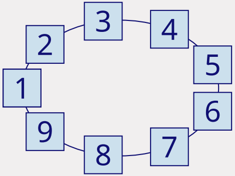


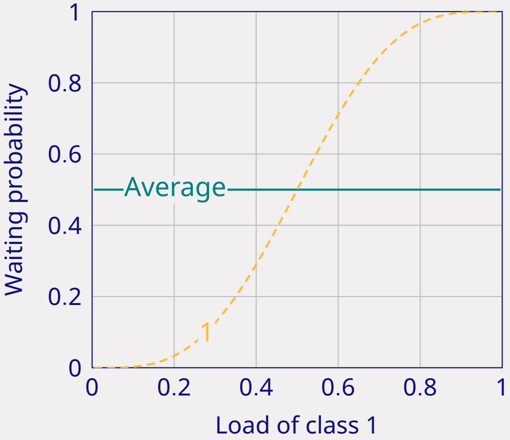


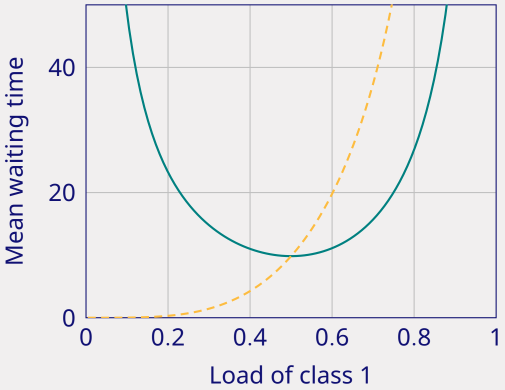


TU/e

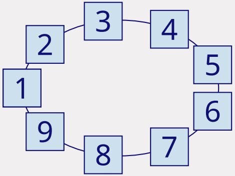
Numerical results: Cycle

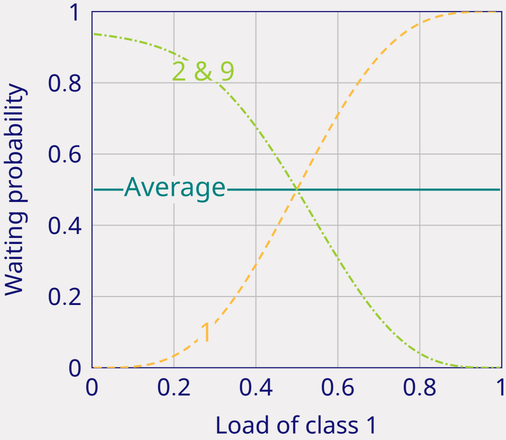


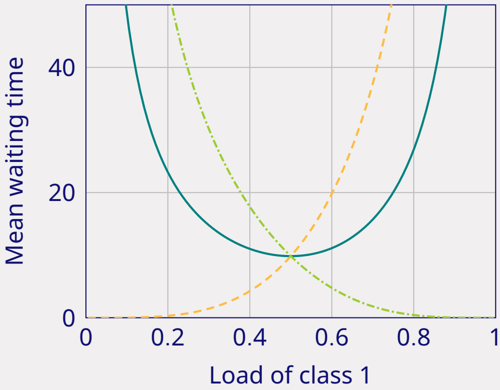




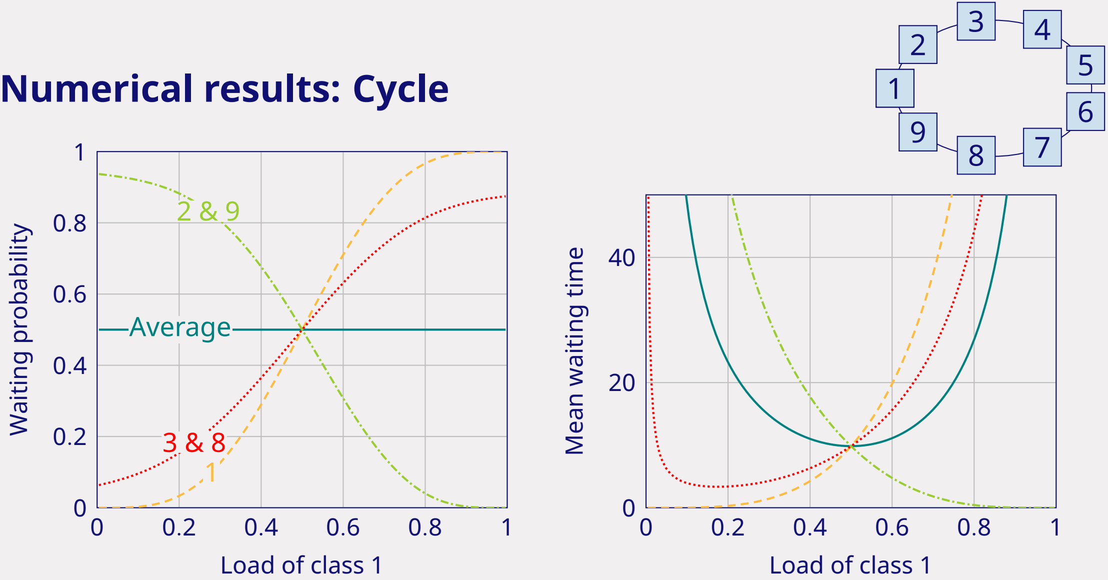
Numerical results: Cycle



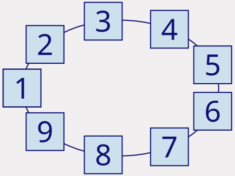


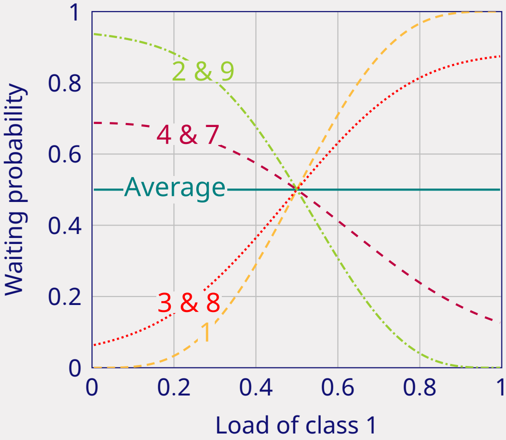


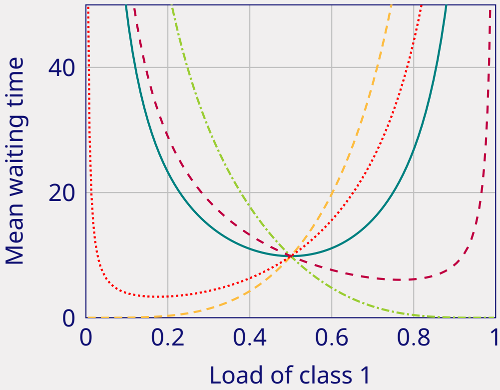
Numerical results: Cycle



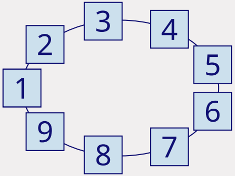
Numerical results: Cycle

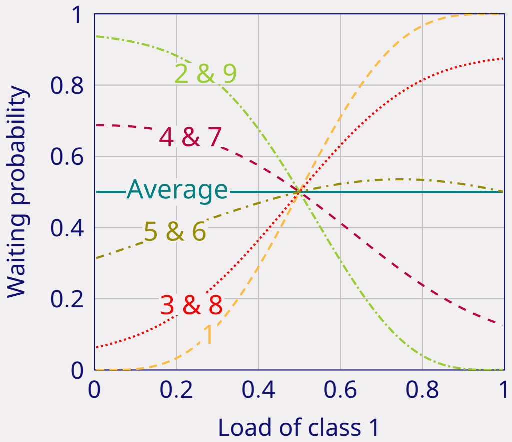


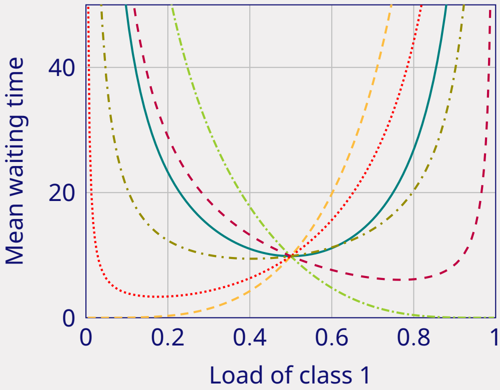




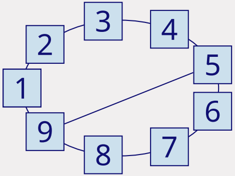
Numerical results: Cycle



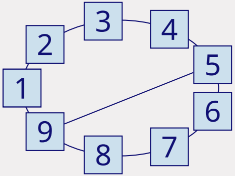


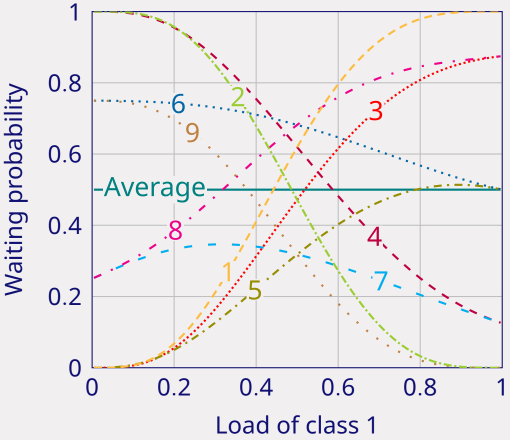


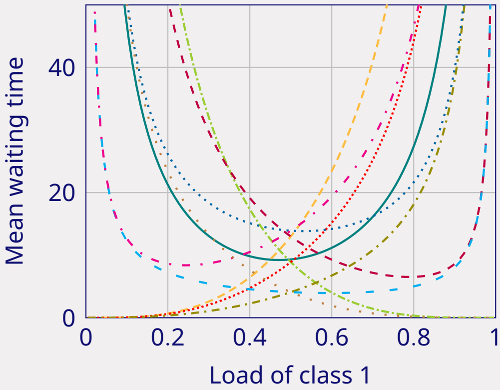
Numerical results: Cycle with a chord



Numerical results: Cycle with a chord







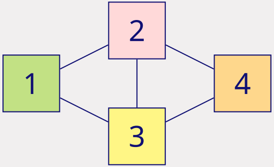
Outline

Model and notation

Performance under the first-come-first-matched policy
Comte, Stochastic Models (2022)

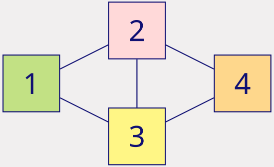
Matching rates under an arbitrary policy
Comte, Mathieu, and Bušić, arXiv:2112.14457 (2022)

Matching rates



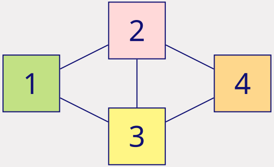
- Matching rate λ_k along edge $k = \{i, j\}$:
  mean number of matches per time unit between classes i and j.

Matching rates



- Matching rate λ_k along edge $k = \{i, j\}$:
  mean number of matches per time unit between classes i and j.
- Matching rates are particularly interesting:
  - We often want to optimize a function of these matching rates.
  - They give intuition about the long-term impact of the matching policy.

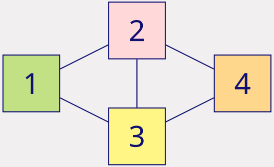
Matching rates



- Matching rate λ_k along edge $k = \{i, j\}$:
  mean number of matches per time unit between classes i and j.
- Matching rates are particularly interesting:
  - We often want to optimize a function of these matching rates.
  - They give intuition about the long-term impact of the matching policy.

Given a graph $G = (V, E)$ and a vector $\mu = (\mu_1, \mu_2, \ldots, \mu_n)$ of arrival rates, what is the set of "feasible" vectors $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ of matching rates?

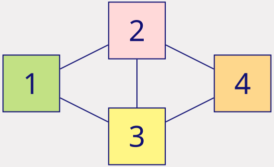
Conservation equation



The matching rates satisfy the conservation equation

$$\sum_{k \in E_i} \lambda_k = \mu_i, \quad i \in \{1, 2, \dots, n\}.$$

Conservation equation

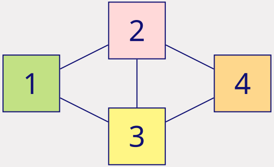


The matching rates satisfy the conservation equation

$$\sum_{k \in E_i} \lambda_k = \mu_i, \quad i \in \{1, 2, \ldots, n\}.$$

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{2,4} + \lambda_{3,4} = \mu_4 \end{cases}$$

Conservation equation



The matching rates satisfy the conservation equation

$$\sum_{k \in E_i} \lambda_k = \mu_i, \quad i \in \{1, 2, \ldots, n\},$$

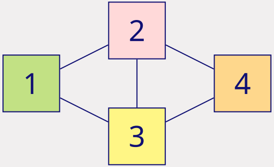
that is, in matrix form,

$$A\lambda = \mu,$$

where $A = (a_{i,k})$ is the incidence matrix of the compatibility graph.

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{2,4} + \lambda_{3,4} = \mu_4 \end{cases}$$

Conservation equation



The matching rates satisfy the conservation equation

$$\sum_{k \in E_i} \lambda_k = \mu_i, \quad i \in \{1, 2, \dots, n\},$$

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{2,4} + \lambda_{3,4} = \mu_4 \end{cases}$$

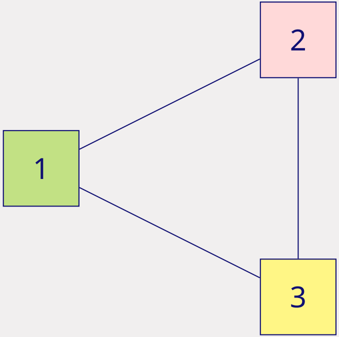
that is, in matrix form,

$$A\lambda = \mu,$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

where $A = (a_{i,k})$ is the incidence matrix of the compatibility graph.

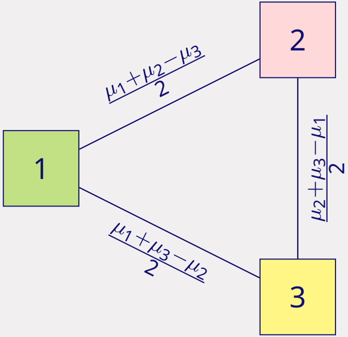
Example: Triangle graph



$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} = \mu_3 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{bmatrix}$$

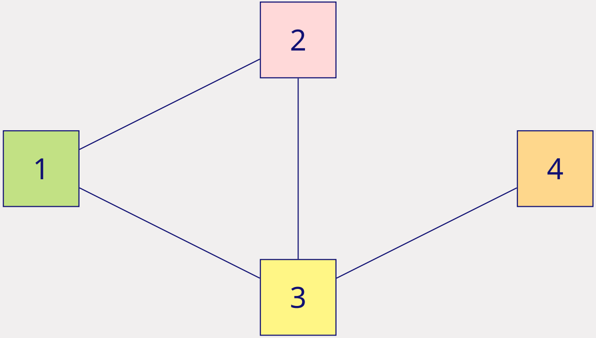
Example: Triangle graph



$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} = \mu_3 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{bmatrix}$$

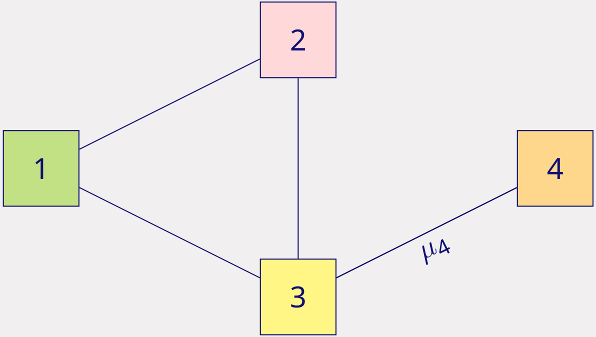
Example: Paw graph



$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{3,4} = \mu_4 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

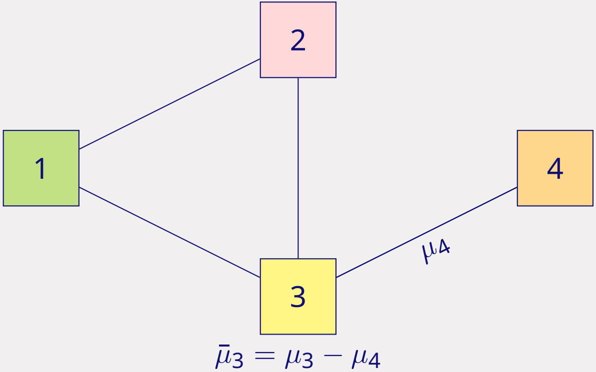
Example: Paw graph



$$
\begin{cases}
\lambda_{1,2} + \lambda_{1,3} = \mu_1 \\
\lambda_{1,2} + \lambda_{2,3} = \mu_2 \\
\lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\
\lambda_{3,4} = \mu_4
\end{cases}
$$

$$
\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{3,4} \end{bmatrix}
=
\begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}
$$

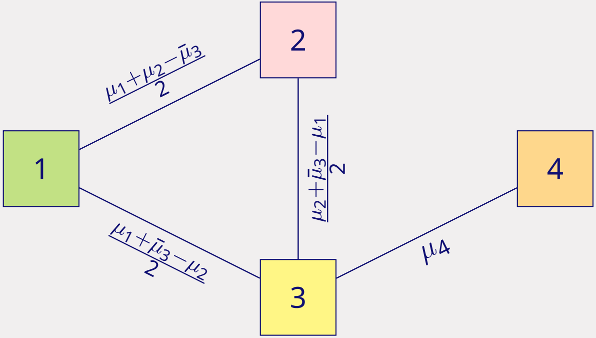
Example: Paw graph



$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{3,4} = \mu_4 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

Example: Paw graph

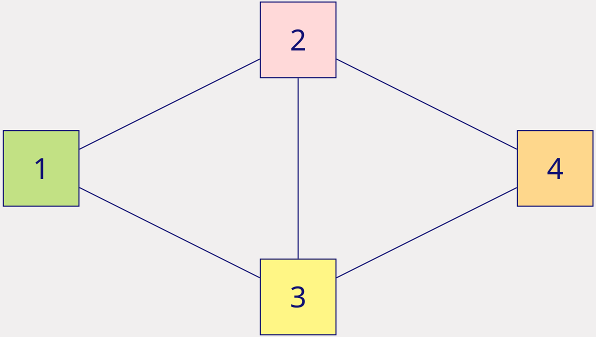


$$\bar{\mu}_3 = \mu_3 - \mu_4$$

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{3,4} = \mu_4 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

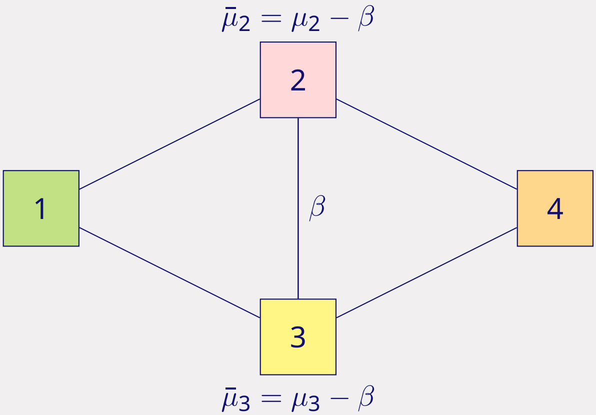
Example: Diamond graph



$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{2,4} + \lambda_{3,4} = \mu_4 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

Example: Diamond graph

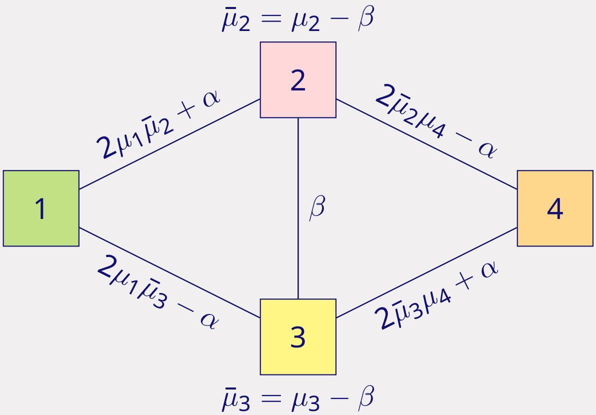


$$\beta = \frac{1}{2}(\mu_2 + \mu_3 - \mu_1 - \mu_4)$$
$$\mu_1 + \mu_4 = \bar{\mu}_2 + \bar{\mu}_3 = \frac{1}{2}$$

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{2,4} + \lambda_{3,4} = \mu_4 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

Example: Diamond graph

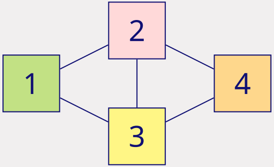


$$\beta = \frac{1}{2}(\mu_2 + \mu_3 - \mu_1 - \mu_4)$$
$$\mu_1 + \mu_4 = \bar{\mu}_2 + \bar{\mu}_3 = \frac{1}{2}$$

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{2,4} + \lambda_{3,4} = \mu_4 \end{cases}$$

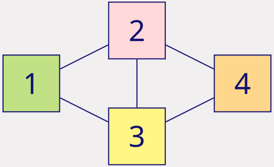
$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

Surjectivity, injectivity, and bijectivity



- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.

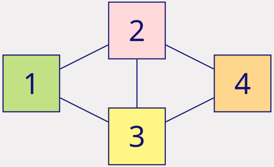
Surjectivity, injectivity, and bijectivity



- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.

- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.

Surjectivity, injectivity, and bijectivity

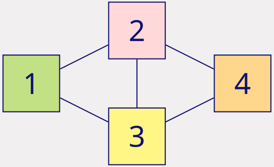


- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.

- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.

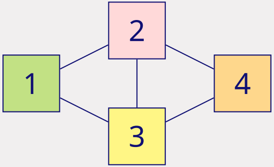
- The compatibility graph G is bijective if G is surjective and injective.

Surjectivity, injectivity, and bijectivity



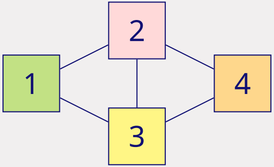
- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.
  - The conservation equation $A\lambda = \mu$ has at least one solution, for each $\mu \in \mathbb{R}^n$.
- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.
  - The conservation equation $A\lambda = \mu$ has at most one solution, for each $\mu \in \mathbb{R}^n$.
- The compatibility graph G is bijective if G is surjective and injective.

Surjectivity, injectivity, and bijectivity



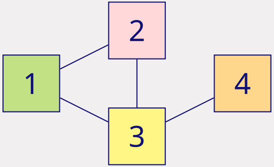
- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.
  - The conservation equation $A\lambda = \mu$ has at least one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G is non-bipartite (i.e., contains at least one odd cycle).
- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.
  - The conservation equation $A\lambda = \mu$ has at most one solution, for each $\mu \in \mathbb{R}^n$.
- The compatibility graph G is bijective if G is surjective and injective.

Surjectivity, injectivity, and bijectivity



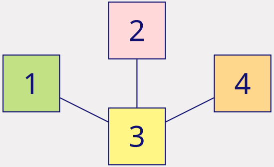
- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.
  - The conservation equation $A\lambda = \mu$ has at least one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G is non-bipartite (i.e., contains at least one odd cycle).
- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.
  - The conservation equation $A\lambda = \mu$ has at most one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G contains at most one cycle and this cycle is odd.
- The compatibility graph G is bijective if G is surjective and injective.

Surjectivity, injectivity, and bijectivity



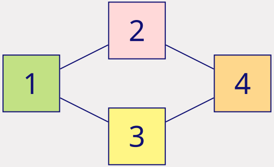
- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.
  - The conservation equation $A\lambda = \mu$ has at least one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G is non-bipartite (i.e., contains at least one odd cycle).
- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.
  - The conservation equation $A\lambda = \mu$ has at most one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G contains at most one cycle and this cycle is odd.
- The compatibility graph G is bijective if G is surjective and injective.

Surjectivity, injectivity, and bijectivity



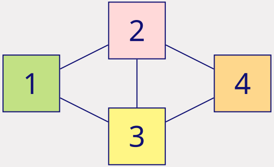
- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.
  - The conservation equation $A\lambda = \mu$ has at least one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G is non-bipartite (i.e., contains at least one odd cycle).
- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.
  - The conservation equation $A\lambda = \mu$ has at most one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G contains at most one cycle and this cycle is odd.
- The compatibility graph G is bijective if G is surjective and injective.

Surjectivity, injectivity, and bijectivity



- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.
  - The conservation equation $A\lambda = \mu$ has at least one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G is non-bipartite (i.e., contains at least one odd cycle).
- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.
  - The conservation equation $A\lambda = \mu$ has at most one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G contains at most one cycle and this cycle is odd.
- The compatibility graph G is bijective if G is surjective and injective.

Surjectivity, injectivity, and bijectivity

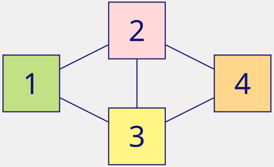


- The compatibility graph G is surjective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is surjective.
  - The conservation equation $A\lambda = \mu$ has at least one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G is non-bipartite (i.e., contains at least one odd cycle).

- The compatibility graph G is injective if
  - The linear application $\lambda \in \mathbb{R}^m \mapsto A\lambda \in \mathbb{R}^n$ is injective.
  - The conservation equation $A\lambda = \mu$ has at most one solution, for each $\mu \in \mathbb{R}^n$.
  - The compatibility graph G contains at most one cycle and this cycle is odd.

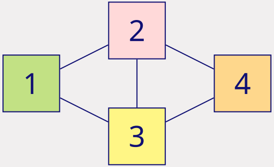
- The compatibility graph G is bijective if G is surjective and injective.

"Stabilizability"



$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

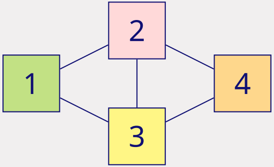
"Stabilizability"



- A matching problem (G, μ) is stabilizable if and only if $\rho(I) < 1$ for each $I \in \mathbb{I}$.

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

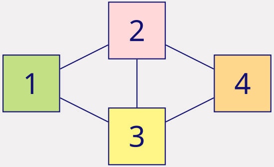
“Stabilizability”



- A matching problem (G, μ) is stabilizable
  if and only if the conservation equation $A\lambda = \mu$ has a solution $\lambda > 0$.

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

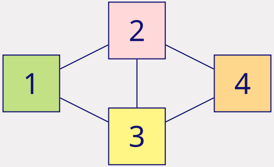
"Stabilizability"



- A matching problem (G, μ) is stabilizable
  if and only if the conservation equation $A\lambda = \mu$ has a solution $\lambda > 0$.
  ☺ The time complexity to verify this condition is polynomial in n and m.

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

"Stabilizability"

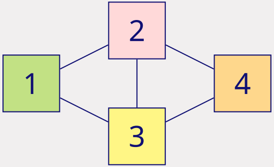


- A matching problem (G, μ) is stabilizable
  if and only if the conservation equation $A\lambda = \mu$ has a solution $\lambda > 0$.
  ☺ The time complexity to verify this condition is polynomial in n and m.

- A compatibility graph G is stabilizable if and only if G is non-bipartite.

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

"Stabilizability"

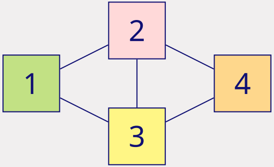


- A matching problem (G, μ) is stabilizable
  if and only if the conservation equation $A\lambda = \mu$ has a solution $\lambda > 0$.
  ☺ The time complexity to verify this condition is polynomial in n and m.

- A compatibility graph G is stabilizable if and only if G is surjective.

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

"Stabilizability"



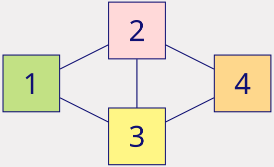
- A matching problem (G, μ) is stabilizable
  if and only if the conservation equation $A\lambda = \mu$ has a solution $\lambda > 0$.
  ☺ The time complexity to verify this condition is polynomial in n and m.

- A compatibility graph G is stabilizable if and only if G is surjective.
  ☺ The rank of matrix A is n.
  The nullity of matrix A is $d = m - n$
  (according to the rank-nullity theorem).

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

TU/e

Affine space of solutions

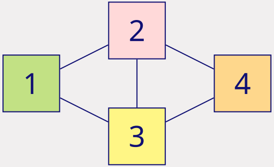


- The solution set of the conservation equation is

$$\Lambda = \left\{ \lambda^\circ + \alpha_1 b_1 + \alpha_2 b_2 + \ldots + \alpha_d b_d : \alpha \in \mathbb{R}^d \right\}$$

where λ° is a particular solution of the conservation equation and $\{b_1, b_2, \ldots, b_d\}$ is a basis of Ker(A), of cardinality $d = m - n$.

Affine space of solutions

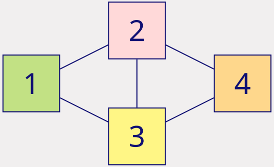


- The solution set of the conservation equation is

$$\Lambda = \left\{ \lambda^\circ + \alpha_1 b_1 + \alpha_2 b_2 + \ldots + \alpha_d b_d : \alpha \in \mathbb{R}^d \right\}$$

  where λ° is a particular solution of the conservation equation and $\{b_1, b_2, \ldots, b_d\}$ is a basis of Ker(A), of cardinality $d = m - n$.
- We borrowed an algorithm from (Doob, 1973) to build a basis of Ker(A).

Affine space of solutions

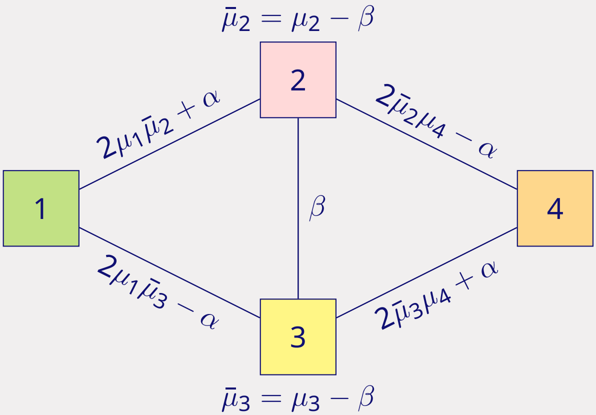


- The solution set of the conservation equation is

$$\Lambda = \left\{\lambda^\circ + \alpha_1 b_1 + \alpha_2 b_2 + \ldots + \alpha_d b_d : \alpha \in \mathbb{R}^d\right\}$$

  where λ° is a particular solution of the conservation equation and $\{b_1, b_2, \ldots, b_d\}$ is a basis of Ker(A), of cardinality $d = m - n$.
- We borrowed an algorithm from (Doob, 1973) to build a basis of Ker(A).
- We use two coordinate systems:
  - Edge coordinates $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m) \in \mathbb{R}^m$.
  - Kernel coordinates $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_d) \in \mathbb{R}^d$.

Example: Diamond graph

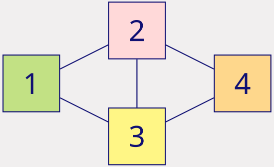


$$\beta = \frac{1}{2}(\mu_2 + \mu_3 - \mu_1 - \mu_4)$$
$$\mu_1 + \mu_4 = \bar{\mu}_2 + \bar{\mu}_3 = \frac{1}{2}$$

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = \mu_1 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = \mu_2 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = \mu_3 \\ \lambda_{2,4} + \lambda_{3,4} = \mu_4 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{bmatrix}$$

Convex polytope of non-negative solutions

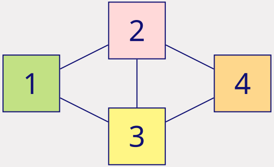


- The set of non-negative solutions of the conservation equation is

$$\Lambda_{\geq 0} = \Lambda \cap \mathbb{R}_+^m \approx \left\{ \alpha \in \mathbb{R}^d : \lambda^\circ + \alpha_1 b_1 + \alpha_2 b_2 + \ldots + \alpha_d b_d \geq 0 \right\}.$$

This is a *d*-dimensional convex polytope.

Convex polytope of non-negative solutions



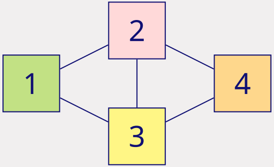
- The set of non-negative solutions of the conservation equation is

$$\Lambda_{\geq 0} = \Lambda \cap \mathbb{R}_+^m \approx \left\{ \alpha \in \mathbb{R}^d : \lambda^\circ + \alpha_1 b_1 + \alpha_2 b_2 + \ldots + \alpha_d b_d \geq 0 \right\}.$$

  This is a *d*-dimensional convex polytope.

- The subgraph restricted to the support of a vertex of $\Lambda_{\geq 0}$ is injective

Convex polytope of non-negative solutions



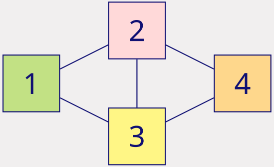
- The set of non-negative solutions of the conservation equation is

$$\Lambda_{\geq 0} = \Lambda \cap \mathbb{R}_+^m \approx \left\{ \alpha \in \mathbb{R}^d : \lambda^\circ + \alpha_1 b_1 + \alpha_2 b_2 + \ldots + \alpha_d b_d \geq 0 \right\}.$$

  This is a d-dimensional convex polytope.

- The subgraph restricted to the support of a vertex of $\Lambda_{\geq 0}$ is injective:
  - If this subgraph is bijective, we can reach this vertex by applying any stable matching policy on this subgraph.

Convex polytope of non-negative solutions



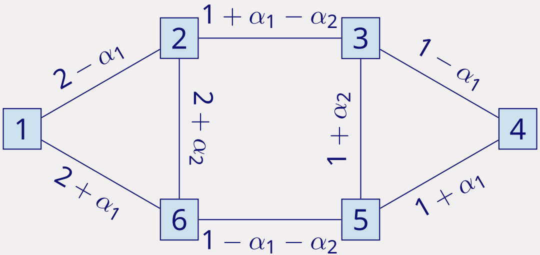
- The set of non-negative solutions of the conservation equation is

$$\Lambda_{\geq 0} = \Lambda \cap \mathbb{R}_+^m \approx \left\{ \alpha \in \mathbb{R}^d : \lambda^\circ + \alpha_1 b_1 + \alpha_2 b_2 + \ldots + \alpha_d b_d \geq 0 \right\}.$$

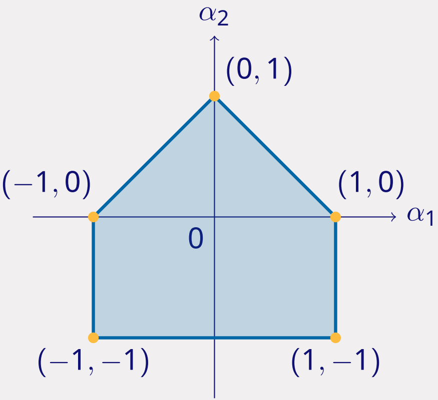
  This is a *d*-dimensional convex polytope.

- The subgraph restricted to the support of a vertex of $\Lambda_{\geq 0}$ is injective:
  - If this subgraph is bijective, we can reach this vertex by applying any stable matching policy on this subgraph.
  - If this subgraph is injective but not surjective, it's more complicated...

Example: Codomino graph 1

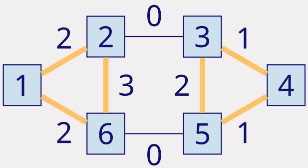


(a) Solution of the conservation equation $A\lambda = \mu$ with $\mu = (4, 5, 3, 2, 3, 5)$.

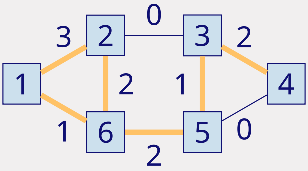


(b) Polytope $\Lambda_{\geq 0}$ in kernel coordinates.

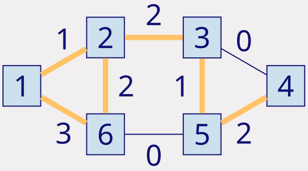
Example: Codomino graph 1



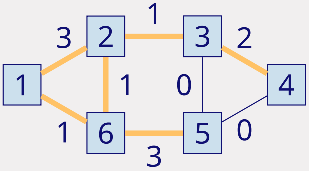
(a) Vertex $(0, 1)$.



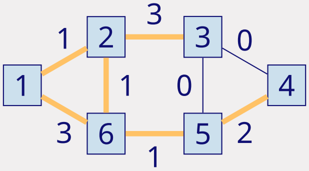
(b) Vertex $(-1, 0)$.



(c) Vertex $(1, 0)$.

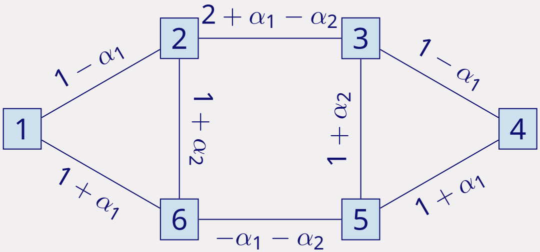


(d) Vertex $(-1, -1)$.

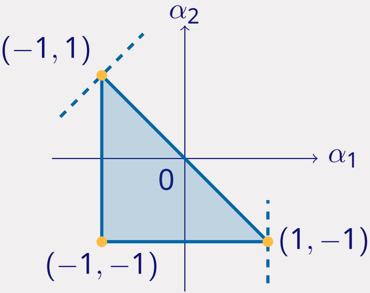


(e) Vertex $(1, -1)$.

Example: Codomino graph 2

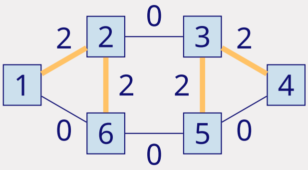


(a) Solution of the conservation equation $A\lambda = \mu$ with $\mu = (2, 4, 4, 2, 2, 2)$.

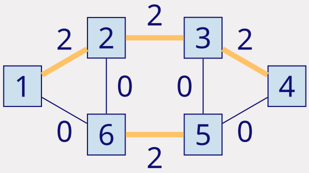


(b) Polytope $\Lambda_{\geq}$ in kernel coordinates.

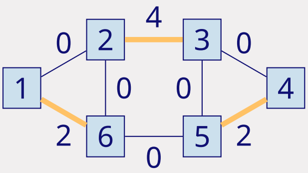
Example: Codomino graph 2



(a) Vertex $(-1, 1)$.

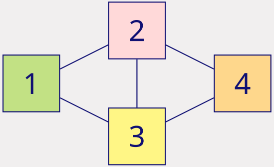


(b) Vertex $(-1, -1)$.



(c) Vertex $(1, -1)$.

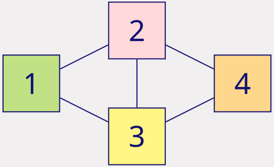
Conclusion



Take-away

- Stochastic dynamic matching problem associated with organ transplant programs and assembly systems.
- Performance evaluation under the first-come-first-matched policy.
- Analysis of the matching rates under an arbitrary matching policy.

Conclusion



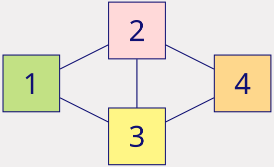
Take-away

- Stochastic dynamic matching problem associated with organ transplant programs and assembly systems.
- Performance evaluation under the first-come-first-matched policy.
- Analysis of the matching rates under an arbitrary matching policy.

Future works

- More realistic model: hypergraph? state-dependent arrival rates?
- Optimization and learning: graph structure? arrival rates? policy?

References

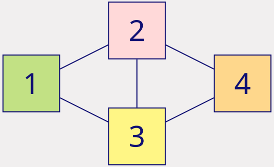


C. Comte. "Stochastic non-bipartite matching models and order-independent loss queues". *Stochastic Models* 38.1 (Jan. 2022), pp. 1–36

C. Comte and J.-P. Dorsman. "Performance Evaluation of Stochastic Bipartite Matching Models". *Performance Engineering and Stochastic Modeling*. Lecture Notes in Computer Science. Springer, 2021, pp. 425–440

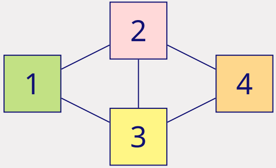
C. Comte, F. Mathieu, and A. Bušić. "Stochastic dynamic matching: A mixed graph-theory and linear-algebra approach". (Jan. 2022). arXiv: 2112.14457

Basis of the kernel of the matrix A



- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.

Basis of the kernel of the matrix A

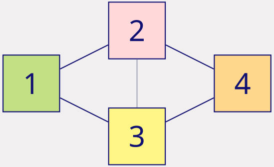


- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Basis of the kernel of the matrix A

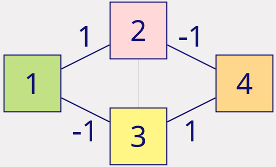


- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Basis of the kernel of the matrix A

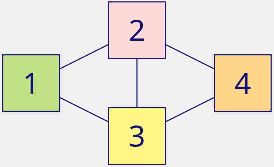


- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Basis of the kernel of the matrix A

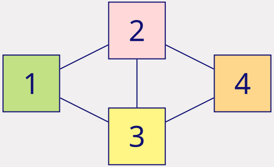


- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Basis of the kernel of the matrix A

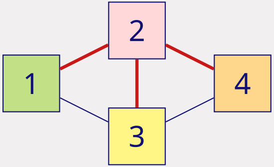


- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.
- Algorithm to construct a basis of Ker(A) (Doob, 1973)

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Basis of the kernel of the matrix A

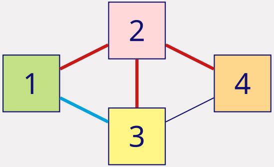


- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.
- Algorithm to construct a basis of Ker(A) (Doob, 1973)
  1. Build a spanning tree **T** of G.

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Basis of the kernel of the matrix A

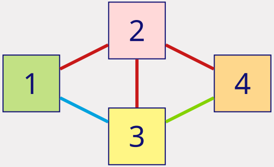


- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.
- Algorithm to construct a basis of Ker(A) (Doob, 1973)
  1. Build a spanning tree **T** of G.
  2. Identify an edge **k** $\notin$ **T** such that **T** $\cup \{$**k**$\}$ contains an odd cycle.

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Basis of the kernel of the matrix A

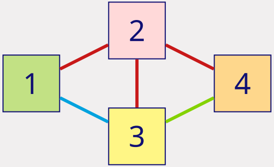


- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.
- Algorithm to construct a basis of Ker(A) (Doob, 1973)
  1. Build a spanning tree **T** of G.
  2. Identify an edge **k** $\notin$ **T** such that **T** $\cup$ {**k**} contains an odd cycle.
  3. For each edge **l** $\notin$ (**T** $\cup$ {**k**}), build a kernel vector with support {**l**} $\subseteq S \subseteq$ **T** $\cup$ {**k**, **l**}

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Basis of the kernel of the matrix A



- A vector $\lambda \in \mathbb{R}^m$ belongs to Ker(A) if and only if $A\lambda = 0$.
- Algorithm to construct a basis of Ker(A) (Doob, 1973)
  1. Build a spanning tree **T** of G.
  2. Identify an edge **k** $\notin$ **T** such that **T** $\cup \{$**k**$\}$ contains an odd cycle.
  3. For each edge **l** $\notin$ (**T** $\cup \{$**k**$\}$), build a kernel vector with support $\{$**l**$\} \subseteq S \subseteq$ **T** $\cup \{$**k**, **l**$\}$
- The matching rate along an edge is unique if and only if this edge doesn't belong to any "generalized even cycle".

$$\begin{cases} \lambda_{1,2} + \lambda_{1,3} = 0 \\ \lambda_{1,2} + \lambda_{2,3} + \lambda_{2,4} = 0 \\ \lambda_{1,3} + \lambda_{2,3} + \lambda_{3,4} = 0 \\ \lambda_{2,4} + \lambda_{3,4} = 0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{1,2} \\ \lambda_{1,3} \\ \lambda_{2,3} \\ \lambda_{2,4} \\ \lambda_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$