Performance Evaluation of Stochastic Non-Bipartite Matching Models

Céline Comte — c.m.comte@tue.nl — Eindhoven University of Technology
Jan-Pieter Dorsman — j.l.dorsman@uva.nl — University of Amsterdam

ASMTA 2021 — December 14, 2021
Stochastic bipartite matching model

Bipartite graph $G = (\mathcal{I}, \mathcal{K}, \mathcal{E})$
Stochastic bipartite matching model

Bipartite graph $G = (\mathcal{I}, \mathcal{K}, \mathcal{E})$ with

- $\mathcal{I} \sim \text{“customer” or “demand” classes}$
Stochastic bipartite matching model

Bipartite graph $G = (\mathcal{I}, \mathcal{K}, \mathcal{E})$ with
- $\mathcal{I} \rightsquigarrow$ “customer” or “demand” classes
- $\mathcal{K} \rightsquigarrow$ “server” or “supply” classes
Stochastic bipartite matching model

Bipartite graph $G = (I, K, E)$ with
- $I \rightsquigarrow$ “customer” or “demand” classes
- $K \rightsquigarrow$ “server” or “supply” classes
- $E \rightsquigarrow$ authorized matchings
Infinite bipartite matching model

- Sequence of i.i.d. customer classes: class \( i \) with probability \( \lambda_i \), \( i \in I \)
- Sequence of i.i.d. server classes: class \( k \) with probability \( \mu_k \), \( k \in K \)
- First-come-first-matched policy
Infinite bipartite matching model

- Sequence of i.i.d. customer classes: class $i$ with probability $\lambda_i, i \in I$

\[
1 \quad 1 \quad 2 \quad 3 \quad 2 \quad 4 \quad 1 \quad \ldots
\]
Infinite bipartite matching model

• Sequence of i.i.d. customer classes: class $i$ with probability $\lambda_i$, $i \in I$
• Sequence of i.i.d. server classes: class $k$ with probability $\mu_k$, $k \in K$
Infinite bipartite matching model

- Sequence of i.i.d. customer classes: class $i$ with probability $\lambda_i$, $i \in \mathcal{I}$
- Sequence of i.i.d. server classes: class $k$ with probability $\mu_k$, $k \in \mathcal{K}$
- First-come-first-matched policy
Infinite bipartite matching model

- Sequence of i.i.d. customer classes: class $i$ with probability $\lambda_i, i \in \mathcal{I}$
- Sequence of i.i.d. server classes: class $k$ with probability $\mu_k, k \in \mathcal{K}$
- First-come-first-matched policy
Infinite bipartite matching model

- Sequence of i.i.d. customer classes: class $i$ with probability $\lambda_i$, $i \in I$
- Sequence of i.i.d. server classes: class $k$ with probability $\mu_k$, $k \in K$
- First-come-first-matched policy
Infinite bipartite matching model

- Sequence of i.i.d. customer classes: class $i$ with probability $\lambda_i$, $i \in I$
- Sequence of i.i.d. server classes: class $k$ with probability $\mu_k$, $k \in K$
- First-come-first-matched policy
Infinite bipartite matching model

- Sequence of i.i.d. customer classes: class $i$ with probability $\lambda_i, i \in I$
- Sequence of i.i.d. server classes: class $k$ with probability $\mu_k, k \in K$
- First-come-first-matched policy
What a queueing theorist sees...

At each time slot, reveal the next customer and the next server:

- The customer belongs to class $i$ with probability $\lambda_i$.
- The server belongs to class $k$ with probability $\mu_k$.

First-come-first-matched service policy.
What a queueing theorist sees...

State \( c = (1, 2, 1) \)

\[
\begin{array}{ccc}
1 & 2 & 1 \\
D & D & D \\
\end{array}
\]

State \( d = (D, D, D) \)
What a queueing theorist sees...

State $c = (1, 2, 1)$

State $d = (D, D, D)$

- At each time slot, reveal the next customer and the next server:
  - The customer belongs to class $i$ with probability $\lambda_i$.
  - The server belongs to class $k$ with probability $\mu_k$. 
What a queueing theorist sees...

State $c = (1, 2, 1)$

State $d = (D, D, D)$

- At each time slot, reveal the next customer \textit{and} the next server:
  - The customer belongs to class $i$ with probability $\lambda_i$.
  - The server belongs to class $k$ with probability $\mu_k$.
- First-come-first-matched service policy.
What a queueing theorist sees...

State $c = (1, 2, 1)$

State $d = (D, D, D)$

• At each time slot, reveal the next customer and the next server:
  • The customer belongs to class $i$ with probability $\lambda_i$.
  • The server belongs to class $k$ with probability $\mu_k$.
• First-come-first-matched service policy.
What a queueing theorist sees...

State $c = (1, 2, 1)$

State $d = (D, D, D)$
What a queueing theorist sees...

State \( c = (1, 2, 1) \)

State \( d = (D, D, D) \)

• There are always as many customers as servers in the queue.
What a queueing theorist sees...

State $c = (1, 2, 1)$

State $d = (D, D, D)$

- There are always as many customers as servers in the queue.
- The set $\mathcal{A}$ of unmatched item classes satisfies:
  - $\mathcal{A}$ is an independent set of the graph $G$
  - $\mathcal{A} \cap \mathcal{I} \neq \emptyset$ if and only if $\mathcal{A} \cap \mathcal{K} \neq \emptyset$
What a queueing theorist sees...

State $c = (1, 2, 1)$

State $d = (D, D, D)$

- There are always as many customers as servers in the queue.
- The set $A$ of unmatched item classes satisfies:
  - $A$ is an independent set of the graph $G$
  - $A \cap I \neq \emptyset$ if and only if $A \cap K \neq \emptyset$
Related works and contributions

• Model introduction (Caldentey, Kaplan, and Weiss, 2009)
Related works and contributions

- Model introduction (Caldentey, Kaplan, and Weiss, 2009)
- Necessary and sufficient stability condition (Bušić, Gupta, and Mairesse, 2013)
Related works and contributions

• Model introduction (Caldentey, Kaplan, and Weiss, 2009)
• Necessary and sufficient stability condition (Bušić, Gupta, and Mairesse, 2013)
• Performance evaluation
  • (Adan and Weiss, 2012)
  • (Adan, Bušić, Mairesse, and Weiss, 2017)
Related works and contributions

- Model introduction (Caldentey, Kaplan, and Weiss, 2009)
- Necessary and sufficient stability condition (Bušić, Gupta, and Mairesse, 2013)
- Performance evaluation
  - (Adan and Weiss, 2012)
  - (Adan, Bušić, Mairesse, and Weiss, 2017)
- Optimization and learning (Cadas, 2021)
Performance evaluation

- **Stationary distribution** of the set of unmatched item classes

\[
\Delta(A)\pi(A) = \mu(A \cap K) \sum_{i \in A \cap I} \lambda_i \pi(A \{i\}) + \lambda(A \cap I) \sum_{k \in A \cap K} \mu_k \pi(A \{k\})
\]

\[
+ \sum_{i \in A \cap I} \sum_{k \in A \cap K} \lambda_i \mu_k \pi(A \{i, k\}), \quad \text{if } A \text{ is non-empty},
\]

where \(\Delta(A) = \mu(K(A \cap I))\lambda(I(A \cap K)) - \lambda(A \cap I)\mu(A \cap K)\).
Performance evaluation

- **Stationary distribution** of the set of unmatched item classes

\[
\Delta(\mathcal{A})\pi(\mathcal{A}) = \mu(\mathcal{A} \cap \mathcal{K}) \sum_{i \in \mathcal{A} \cap \mathcal{I}} \lambda_i \pi(\mathcal{A}\{i\}) + \lambda(\mathcal{A} \cap \mathcal{I}) \sum_{k \in \mathcal{A} \cap \mathcal{K}} \mu_k \pi(\mathcal{A}\{k\}) \\
+ \sum_{i \in \mathcal{A} \cap \mathcal{I}} \sum_{k \in \mathcal{A} \cap \mathcal{K}} \lambda_i \mu_k \pi(\mathcal{A}\{i, k\}), \quad \text{if } \mathcal{A} \text{ is non-empty},
\]

where \(\Delta(\mathcal{A}) = \mu(\mathcal{K}(\mathcal{A} \cap \mathcal{I})) \lambda(\mathcal{I}(\mathcal{A} \cap \mathcal{K})) - \lambda(\mathcal{A} \cap \mathcal{I}) \mu(\mathcal{A} \cap \mathcal{K}).\)

The value of the **normalization constant** \(\pi(\emptyset)\) follows by normalization.
Performance evaluation

- **Stationary distribution** of the set of unmatched item classes

\[ \Delta(A)\pi(A) = \mu(A \cap K) \sum_{i \in A \cap I} \lambda_i \pi(A \{i\}) + \lambda(A \cap I) \sum_{k \in A \cap K} \mu_k \pi(A \{k\}) \]

\[ + \sum_{i \in A \cap I} \sum_{k \in A \cap K} \lambda_i \mu_k \pi(A \{i, k\}), \text{ if } A \text{ is non-empty,} \]

where \( \Delta(A) = \mu(K(A \cap I))\lambda(I(A \cap K)) - \lambda(A \cap I)\mu(A \cap K) \). The value of the **normalization constant** \( \pi(\emptyset) \) follows by normalization.

- Similar expressions for waiting probability, mean waiting time...
Discussion

• **Time complexity.** \( O(I \cdot K \cdot ((I + K) \cdot M) + N) \), where
  • \( I \) = number of customer classes,
  • \( K \) = number of server classes,
  • \( M \) = number of maximal independent sets,
  • \( N \) = number of independent sets.
Discussion

• **Time complexity.** $O(I \cdot K \cdot ((I + K) \cdot M) + N)$, where
  - $I =$ number of customer classes,
  - $K =$ number of server classes,
  - $M =$ number of maximal independent sets,
  - $N =$ number of independent sets.

• **Flexibility.** This approach can be easily adapted to derive other performance metrics (e.g., matching rates, mean length of a busy sequence).
Numerical results

\[ \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4} \]

\[ \mu_A = \rho \]
\[ \mu_B = \mu_C = \mu_D = \frac{1}{4} \]
\[ \mu_E = \frac{1}{4} - \rho \]

Average Waiting Probability

<table>
<thead>
<tr>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
<th>Class 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.2</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>0.8</td>
<td>1</td>
<td>1.2</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Mean Waiting Time

\[ 5, 10, 15, 20 \]
Numerical results

\[ \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4} \]

\[ \mu_A = \frac{\rho}{4}, \quad \mu_B = \mu_C = \mu_D = \frac{1}{4}, \quad \mu_E = \frac{1 - \rho}{4} \]
Numerical results

\[ \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4} \]

\[ \mu_A = \frac{\rho}{4}, \quad \mu_B = \mu_C = \mu_D = \frac{1}{4}, \quad \mu_E = \frac{1-\rho}{4} \]

<table>
<thead>
<tr>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
<th>Class 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.4</td>
<td>0.6</td>
<td>0.8</td>
</tr>
<tr>
<td>0.6</td>
<td>0.8</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>

Waiting probability

Mean waiting time
Numerical results

\[ \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4} \]

\[ \mu_A = \frac{\rho}{4} \quad \mu_B = \mu_C = \mu_D = \frac{1}{4} \quad \mu_E = \frac{1 - \rho}{4} \]

The graphs show the waiting probability and mean waiting time for different classes. The legend indicates the different classes represented by different lines.
Numerical results

\[ \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4} \]

\[ \mu_A = \frac{\rho}{4}, \quad \mu_B = \mu_C = \mu_D = \frac{1}{4}, \quad \mu_E = \frac{1 - \rho}{4} \]

**Average**

**Class 1**

**Class 2**

**Class 3**

**Class 4**
Numerical results

\[ \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4} \]

\[ \mu_A = \frac{\rho}{4}, \quad \mu_B = \mu_C = \mu_D = \frac{1}{4}, \quad \mu_E = \frac{1-\rho}{4} \]

Average

Class 1

Class 2

Class 3

Class 4

Waiting probability

Mean waiting time
Numerical results

$\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4}$

$\mu_A = \frac{\rho}{4}$, $\mu_B = \mu_C = \mu_D = \frac{1}{4}$, $\mu_E = \frac{1 - \rho}{4}$

Average, Class 1, Class 2, Class 3, Class 4
Numerical results

\[ \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4} \]

\[ \mu_A = \frac{\rho}{4}, \quad \mu_B = \mu_C = \mu_D = \frac{1}{4}, \quad \mu_E = \frac{1 - \rho}{4} \]

<table>
<thead>
<tr>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
<th>Class 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.2</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>0.2</td>
<td>0.4</td>
<td>0.6</td>
<td>0.8</td>
</tr>
<tr>
<td>0.4</td>
<td>0.6</td>
<td>0.8</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Average

Mean waiting time

Waiting probability

Graphs showing waiting probability and mean waiting time for classes 1 to 4.
Numerical results

$$\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4}$$

$$\mu_A = \frac{\rho}{4}, \quad \mu_B = \mu_C = \mu_D = \frac{1}{4}, \quad \mu_E = \frac{1 - \rho}{4}$$

<table>
<thead>
<tr>
<th>Class</th>
<th>Waiting probability</th>
<th>Mean waiting time</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.8</td>
<td>0.0</td>
</tr>
<tr>
<td>B</td>
<td>0.6</td>
<td>0.2</td>
</tr>
<tr>
<td>C</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>D</td>
<td>0.2</td>
<td>0.6</td>
</tr>
<tr>
<td>E</td>
<td>0.0</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Graphs showing the relationship between waiting probability and mean waiting time for classes A to E.
Conclusion

• New closed-form expressions for performance metrics in the stochastic bipartite matching model.
Conclusion

• New closed-form expressions for performance metrics in the stochastic bipartite matching model.
• Numerical evaluations on toy examples.
Conclusion

• New closed-form expressions for performance metrics in the stochastic bipartite matching model.
• Numerical evaluations on toy examples.
• Self-advertising 😊 → (Comte, Stochastic Models, 2021) Similar expressions for the stochastic non-bipartite matching model (with additional comments on order-independent queues!)