A Staffing Algorithm for Call Centers with Skill-Based Routing

Ward Whitt

IEOR Department, Columbia University

http://www.columbia.edu/~ww2040
Joint work with:

Rodney B. Wallace

IBM and George Washington University

Thesis: Performance Modelling and Design of Call Centers with Skill-Based Routing

Advisors: William A. Massey (Princeton), Thomas A. Mazzuchi (GW) and Ward Whitt (Columbia)
Multiple Types of Calls and Agents

skill-based routing

call types

server pools
Multiple Types of Calls and Agents

Special case: The service-time distribution does not depend on the call type or the agent.
First Contribution:

Routing and Provisioning Algorithm

Minimize the Required Staff and Telephone Lines
While Meeting the Service level Agreement (SLA)

\[ P(\text{Delay} \leq 30 \text{ seconds}) \geq 0.80 \]
\[ P(\text{Blocking}) \leq 0.005 \]

(service level may depend on call type)
Second Contribution:

Demonstrate Resource-Pooling Phenomenon

A small amount of cross training (multiple skills) produces almost the same performance as if all agents had all skills (as in the single-type case).

Simulation Experiments
Precedents
"A little bit of flexibility goes a long way."

Joining One of Many Queues
- Azar, Broder, Karlin and Upfal (1994)
- Vvedenskaya, Dobrushin and Karpelovich (1996)
- Mitzenmacher (1996) and
- Mitzenmacher and Vöcking (1999)

Flexible Manufacturing: Chaining
- Jordan and Graves (1995)
- Hopp and Van Oyen (2003)
- Gurumurthi and Benjaafar (2004)
Outline

1. SBR Call-Center Model
2. Resource-Pooling Experiment
3. Provisioning Algorithm
4. Simulation to Show Performance
1. $C$ agents, $C + K$ telephone trunklines, and $n$ call types.

2. **Non-preemptive Priorities (NPrPr)** - Calls are processed in priority order. Calls are worked to completion once they are handed to an agent.

3. **Longest-Idle-Agent Routing (LIAR) Policy** - Calls are forwarded to the agent who has been waiting the longest since his last job completion and has the highest skill to handle the request.
Agent-Skill Matrix - $C \times n$

4. Agent-Skill Profile - Predefined in an agent-skill matrix $A \equiv (a_{ij})$ as

$$a_{ij} = \begin{cases} 
    k & \text{when agent } i \text{ supports call type } k \\
    0 & \text{at priority level } j \text{ (primary, secondary, etc),}
\end{cases}$$

where $i = 1, \ldots, C$, $1 \leq k \leq n$, and $1 \leq j \leq n$.

Examples:

$$A_{5 \times 1} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}, \quad A_{3 \times 2}^{(1)} = \begin{pmatrix} 1 & 0 \\ 2 & 0 \\ 2 & 0 \end{pmatrix}, \quad A_{4 \times 2} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 2 & 1 \\ 2 & 1 \end{pmatrix}, \quad A_{6 \times 4} = \begin{pmatrix} 3 & 4 & 1 & 0 \\ 1 & 4 & 0 & 0 \\ 2 & 3 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 3 & 1 & 2 & 4 \\ 1 & 0 & 4 & 0 \end{pmatrix}$$
What to do when an Arrival occurs

Arrival Event

Update System States
Determine Call Type
Schedule Next arrival

Are all agents busy?

Yes

Search Idle Agent Queue
for First Available Agent

Primary Skill
Agent Idle?

No

route_cust_to_agent()

Return

No

place_cust_in_q()

n-th Skill Level
Agent Idle?

Yes
What to do when an Agent becomes free

Agent $i$

Departure Event

Are all the queues empty?

Yes

No

Is Primary Skill Queue Empty?

Yes

No

Check each Supported Queue in Priority Order for First Waiting Customer

Is $n$th Skill Level Queue Empty?

Yes

No

make_server_idle()

get_waiting_cust()

Return
Resource-Pooling Experiment
Model Assumptions

1. **Arrival Process** - \( n \) types of calls arrive at the call center according to \( n \) mutually independent Poisson processes with rate \( \lambda_i, 1 \leq i \leq n. \)
   \[ n = 6, \lambda_i = 1.40 \text{ for all } i \]

2. **Service Time Process** - Call holding (service) times are mutually independent exponential random variables with mean \( 1/\mu_i \) which are independent of the arrival process, \( 1 \leq i \leq n. \)
   \[ 1/\mu_i = 1/\mu = 10 \text{ minutes for all } i \]

3. **Offered Loads** - \( \alpha_i = \lambda_i/\mu_i \)
   \[ \alpha_i = 14 \text{ for all } i, \text{ so the total offered load is } \alpha = 84 \]

4. **Agents and Telephone Lines**
   \[ C = 90 \text{ and } K = 30 \] (\( C + K = 120 \))
Agents are given $k$ skills, $1 \leq k \leq 6$

Three Loads: Normal (84), Light (77.4), Heavy (90)
Provisioning Algorithm

Find C, K and A

So that each agent has at most 2 skills and all performance constraints are met.
How do we know it works?

The optimal values of $C$ and $K$ are almost the same as for $M/M/C/K$ which occurs with a single call type.
Balanced Example

M/M/C/K: $C = 90$ and $K = 19$

SBR: $C = 91$ and $K = 20$
SBR Balanced Provisioning Example

- Call volume is $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \lambda_5 = \lambda_6 = 1.375$,

- Service times are $1/\mu_1 = \ldots = 1/\mu_6 = 10$ mins

- Agents Skill Profile: Agents have 2 skills each.

- Service level targets
  1. Blocking service level target is 0.5%.
  2. 80% of the calls are answered within $\tau = 0.5$ minute.

- Square-root safety method for distributing agents into work groups is used.

- It is known that the total number of agents required is between 90 (best-case) and 106 (worse-case). Similarly, the telephone trunkline capacity is between 111 and 156.
Unbalanced Example

M/M/C/K: C = 90 and K = 19

SBR: C = 91 and K = 21
SBR Unbalanced Provisioning Example

- Call volume is $\lambda_1 = \lambda_2 = 0.425$, $\lambda_3 = 1.05$, $\lambda_4 = 1.375$, $\lambda_5 = 1.925$, and $\lambda_6 = 3.05$ calls/min.

- Service times are $1/\mu_1 = \ldots = 1/\mu_6 = 10$ mins

- Agents Skill Profile: Agents have 2 skills each.

- Service level targets
  1. Blocking service level target is 0.5%.
  2. 80% of the calls are answered within $\tau = 0.5$ minute.

- Square-root safety method for distributing agents into work groups is used.

- It is known that the total number of agents required is between 90 (best-case) and 106 (worse-case). Similarly, the telephone trunkline capacity is between 111 and 156.
Unbalanced SBR Provisioning Example Summary

<table>
<thead>
<tr>
<th>(C, C + K)</th>
<th>Best Case</th>
<th>Actual Perf.</th>
<th>Worst Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>(90, 109)</td>
<td>(91, 111)</td>
<td>(106, 156)</td>
<td></td>
</tr>
<tr>
<td>Workgroup 1 C_1</td>
<td>7</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Workgroup 2 C_2</td>
<td>7</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Workgroup 3 C_3</td>
<td>13</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>Workgroup 4 C_4</td>
<td>15</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Workgroup 5 C_5</td>
<td>21</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>Workgroup 6 C_6</td>
<td>28</td>
<td>36</td>
<td></td>
</tr>
</tbody>
</table>
SBR Provisioning

• Solves the problem of determining the minimum number of agents \( C \) and the minimum number of telephone trunklines \( C + K \) needed to meet service level targets.

• Exploits resource pooling results.

• Exploits \( M/M/C/K \) results to determine initial estimate for \((C, K)\).

• Uses fair agent skill assignment scheme to construct agent skill matrix satisfying general agent skill profile.

• Simulation runs are performed to make improvements on the initial assignment using a heuristic search algorithm.
Determining Primary Skills

\[ C_k = \alpha_k + x \sqrt{\alpha_k} \]

\[ x = \frac{(C-\alpha)}{\sum_{i=1}^{n} \sqrt{\alpha_i}} \]

and round
Determining Secondary Skills

\[ C_{i,k} = \frac{C_i C_k}{C - C_i} \]

and round
## Initial SBR Provisioning Algorithm

<table>
<thead>
<tr>
<th>Performance Measure</th>
<th>Number of Iterations (Agents)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 (90)</td>
</tr>
<tr>
<td>1. Blocking (%)</td>
<td>0.53</td>
</tr>
<tr>
<td>4. $P(\text{Delay} \leq 0.5</td>
<td>entry)$</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_1 \leq 0.5</td>
<td>entry)$</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_2 \leq 0.5</td>
<td>entry)$</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_3 \leq 0.5</td>
<td>entry)$</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_4 \leq 0.5</td>
<td>entry)$</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_5 \leq 0.5</td>
<td>entry)$</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_6 \leq 0.5</td>
<td>entry)$</td>
</tr>
</tbody>
</table>
## Refined SBR Provisioning Algorithm

<table>
<thead>
<tr>
<th>Performance Measure</th>
<th>Number of Iterations (Agents)</th>
<th>4 (93)</th>
<th>5 (92)</th>
<th>6 (92)</th>
<th>7 (91)</th>
<th>8 (91)</th>
<th>9 (90)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Blocking (%)</td>
<td></td>
<td>0.30</td>
<td>0.35</td>
<td>0.36</td>
<td>0.43</td>
<td>0.44</td>
<td>0.54</td>
</tr>
<tr>
<td>4. $P(\text{Delay} \leq 0.5</td>
<td>\text{entry})$</td>
<td></td>
<td>88.8</td>
<td>86.5</td>
<td>86.2</td>
<td>83.4</td>
<td>82.9</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_1 \leq 0.5</td>
<td>\text{entry})$</td>
<td></td>
<td>80.5</td>
<td>78.0</td>
<td>81.6</td>
<td>78.6</td>
<td>82.6</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_2 \leq 0.5</td>
<td>\text{entry})$</td>
<td></td>
<td>80.3</td>
<td>77.6</td>
<td>81.4</td>
<td>78.6</td>
<td>81.9</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_3 \leq 0.5</td>
<td>\text{entry})$</td>
<td></td>
<td>88.0</td>
<td>86.1</td>
<td>85.8</td>
<td>83.6</td>
<td>83.4</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_4 \leq 0.5</td>
<td>\text{entry})$</td>
<td></td>
<td>88.8</td>
<td>87.2</td>
<td>87.0</td>
<td>83.2</td>
<td>82.6</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_5 \leq 0.5</td>
<td>\text{entry})$</td>
<td></td>
<td>89.8</td>
<td>87.7</td>
<td>86.7</td>
<td>84.6</td>
<td>83.1</td>
</tr>
<tr>
<td>5. $P(\text{Delay}_6 \leq 0.5</td>
<td>\text{entry})$</td>
<td></td>
<td>90.9</td>
<td>88.0</td>
<td>86.9</td>
<td>84.1</td>
<td>82.9</td>
</tr>
</tbody>
</table>