Multi-skill Call Center Routing Using Weights, Call Waiting Times and Agent Idle Times

Wyean Chan[†], Ger Koole[‡], Pierre L'Ecuyer[†]

[†] Université de Montréal, Canada
[‡] VU University Amsterdam, The Netherlands

2011 INFORMS Simulation Society Research Workshop
Simulation in Complex Service Systems
HEC, Montréal, July 19, 2011

What is a Call Center ?

- A **call center** is a set of resources for communication between an organization and its customers over the phone.
- Common call centers: toll free 1-800 numbers, emergency centers, government offices, banks, ...
- A client is categorized by its required service, named **call type**.
- Clients are served by **agents** or *customer service representatives*.
- The type of calls that an agent can serve is given by his **skill set**, and an **agent group** contains agents with the same skill set.
- **Routing problem:**
  When an agent becomes idle, which call should he serve next?
  When a new call arrives, which idle agent should serve it?
- **Goal:** Optimize the routing policy subject to some performance measures constraints.
  We consider general black-box type objective functions and we use simulation-based optimization methods.

Example of a Call Center Model

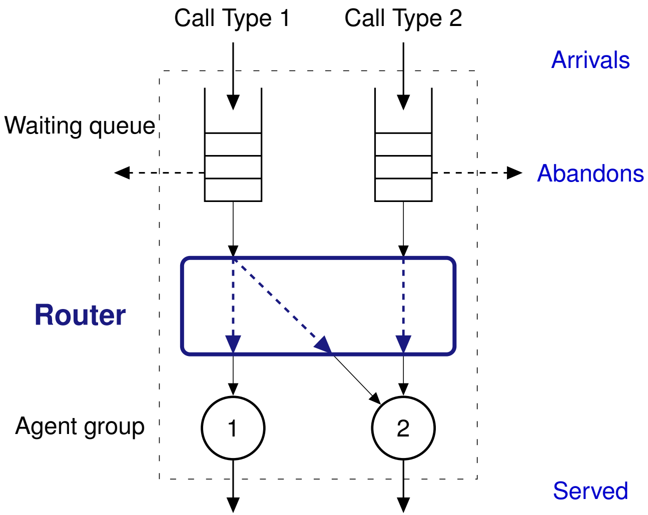


Figure: A N-model call center. Group 2 can serve both call types. A call exits the center either after service completion or by abandon.

Performance Measures

Examples of performance measures:

- The service level (SL):

$$S(\pi, \tau) = \frac{\mathbb{E}[X(\pi, \tau)]}{\mathbb{E}[N - B(\pi, \tau)]},$$

  where τ: acceptable waiting time (AWT), $X(\pi, \tau)$: # of calls served that waited at most τ, N: # of arrivals, $B(\pi, \tau)$: # of calls that abandoned after waiting at most τ and π: the routing policy. $\mathbb{E}$ is the expectation operator.

- The ratio of abandonments:

$$A(\pi) = \mathbb{E}[Z(\pi)]/\mathbb{E}[N],$$

  where $Z(\pi)$: number of abandonments.

- The agent occupancy ratio of group g:

$$O_g(\pi) = \frac{1}{y_g T}\mathbb{E}\left[\int_0^T G_g(\pi, t)dt\right],$$

  where y_g: number of agents in group g, $G_g(\pi, t)$: number of busy agents at time t and T is the time horizon.

Objective Functions

Performance measure constraints are defined as penalty cost functions.
Suppose K call types and G agent groups.
Here are examples of penalty functions to **minimize**:

- Service levels:

$$F_S(\pi) = \sum_{k=1}^{K} a_k \max(t_k - S_k(\pi, \tau_k), 0)^2,$$

  where t_k is the SL target of call type k and a_k is a parameter.

- Service levels and ratio of abandonments :

$$F_{SA}(\pi) = F_S(\pi) + \sum_{k=1}^{K} b_k \max(A_k(\pi) - u_k, 0)^2,$$

  where u_k is the acceptable ratio of abandonments of call type k and b_k is a parameter.

- Service levels and agent occupancy fairness :

$$F_{SO}(\pi) = F_S(\pi) + \sum_{j=1}^{G} c_g \left| O_g(\pi) - \bar{O} \right|^2,$$

  where $\bar{O}$ is the average group occupancy and c_g is a parameter.

Routing policies

Basic fairness rules that apply for all routing policies:

- First-come-first-served (FCFS) for calls of the same type.
- Longest-idle-server-first (LISF) for agents of the same group.

We will compare with some common routing strategies:

- **Priority routing (P)**: Also called **overflow routing**. When an agent completes a call, he selects the next call by following the order of his *group-to-type* preference list. When a new call arrives, it searches for the first available agent following the *type-to-group* preference list.
- **Delays (D)**: A call of type k must have waited $d_{k,g}$ seconds before it can be answered by an agent of group g.
- **Minimum number of idle agents (M)**: Agents of group g can serve a call of type k only if there are more than $m_{k,g}$ idle agents. Reduce temporarily the skill set of an agent group if the number of idle agents is low.

We propose a new routing policy based on **weights**, **call waiting times** and **agent idle times**, called weight-based routing (W).

Weight-Based Routing Policy (W)

Assume K call types, G agent groups and $\mathcal{S}_g \subseteq \{1, \ldots, K\}$ as the skill set of group g.

- There is a weight $c_{k,g}$ for each skill (group g and call type $k \in \mathcal{S}_g$):

$$c_{k,g} = q_{k,g} + a_{k,g} w_k + b_{k,g} v_g,$$

- where $q_{k,g}, a_{k,g}, b_{k,g} \in \mathbb{R}$ are parameters (total of $3 \sum_{g=1}^{G} |\mathcal{S}_g|$ parameters),
- $w_k \geq 0$ is the waiting time of the oldest call of type k in the queue,
- $v_g \geq 0$ is the idle time of the current longest idle agent in group g.
- If **no idle agent** in group g, then $c_{k,g} = -\infty, \forall k$. If there are **no calls** of type g waiting, then $c_{k,g} = -\infty, \forall g$.

Note: The weight $c_{k,g}$ is not restricted to a linear function, it can also be dependent on the state of the call center.

How Does the Weight-Based Policy Work ?

The call and agent are matched as follows:

1. The router monitors and updates the $c_{k,g}$ regularly.
2. If all $c_{k,g} < 0$, then do nothing.
3. If there is a $c_{k,g} \geq 0$, then select call type k^* and group g^* such that:
$$c_{k^*,g^*} = \max_{k,g} \{c_{k,g}\}.$$
4. Assign the longest waiting call of type k^* to the longest idle agent of group g^*.

Policy W Can Approximate Other Routing Policies

The weight-based routing policy can approximate the simpler policies.

Set parameters to approximate the policies:

- **global FCFS**: Set $q_{k,g} = 0$ and $a_{k,g} = 1, b_{k,g} = \epsilon$, for all k, g, where ϵ is a small positive number.
- **P**: Set $q_{k,g} \geq 0$ accordingly to the type-to-group lists (for new calls). Set $a_{k,g} \geq 0$ accordingly to the group-to-type lists (for idle agents). $b_{k,g} = \epsilon$. The parameters $a_{k,g}$ have no effect when a new call arrives since it has 0 waiting time. The parameter $a_{k,g}$ must dominate $q_{k,g}$ when there are calls waiting.
- **P and D**: Set the priority in the same way as for the policy P. In addition, use $q_{k,g} < 0$ to set the delay and adjust the $a_{k,g}$.

We can except the weight-based routing policy to perform no worse than those policies.

Routing Optimization

- Compare the routing policies with their **optimal parameters**.
- Use **simulation** to get more accurate estimations.
- Consider the **objective function** as a **black-box function**.
- We implemented and adapted two heuristic algorithms: a **stochastic gradient descent (SGD)** and a **modified genetic algorithm (MGA)**.

Heuristic algorithms used:

- **Priority rules**: Exhaustive enumeration (for very small problem) or MGA.
- **Delays**: SGD or MGA.
- **Minimun number of idle agents**: MGA.
- **Weight-based**: SGD or MGA.

Because of the difficulty of the optimization problems, we execute the implemented algorithms and take the best solution found.

Stochastic Gradient Descent (SGD)

Use the well-known stochastic gradient descent.

- Estimate the gradient by central (or forward) finite difference using a simulator.
- Combine with a line search algorithm (golden section search).
- Execute a number of restarts to increase the chance of avoiding local optima.
- Stop when no improvement for consecutive restarts.

Alternatively, use a quasi-Newton method.

Modified Genetic Algorithm (MGA)

Simplified version of the *estimation of distribution algorithms* (EDA) or the *cross-entropy* (CE) for optimization.

- No cross-over and mutation operators.
- For each of the N variables, select a probability distribution function Φ_n with parameter vector $\boldsymbol{\theta}_n$.
- Generate randomly the population of solutions from $\Phi_n(\boldsymbol{\theta}_n)$.
- Consider the variables to be independent of each other.
- Start with distribution functions that cover a large set of solutions (and hopefully the optimum).
- Goal: maximize the probability density of the optimal solution.

Modified Genetic Algorithm (MGA)

Suppose N parameters to optimize.

Input: $\Phi_1, \ldots, \Phi_N, \boldsymbol{\theta}_1^{(0)}, \ldots, \boldsymbol{\theta}_N^{(0)}$, maxIt, P (pop. size), Q (# elites), f (obj. function)
Output: best solution $\mathbf{x}^*$ found
begin

    $\boldsymbol{\theta}_n = \boldsymbol{\theta}_n^{(0)}, n = 1, \ldots, N$
    for $i = 1$ *to* maxIt **do**
        for $p = 1$ *to* P **do**
            for $n = 1$ *to* N **do**
                $x_n^{(p)}$ = Generate a random value from probability distribution $\Phi_n(\boldsymbol{\theta}_n)$.
            end
            $f^{(p)} = f(\mathbf{x}^{(p)})$ `// simulate solution`
        end
        Sort $\mathbf{x}^{(p)}$ by order of $f^{(p)}$ and keep the Q best (elite) solutions.
        Update $\mathbf{x}^*$ if found a better solution.
        Update $\boldsymbol{\theta}_n, \forall n$ (e.g., by maximum likelihood) from the set of Q best solutions.
        if (*variance of* $\Phi_n(\boldsymbol{\theta}_n) < \epsilon, \forall n$) **then stop**
    end
end

Numerical Examples

We test the following routing policies:

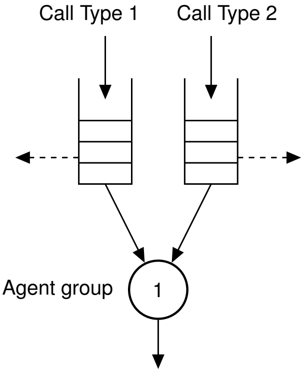
- **G**: Global FCFS and LISF rules.
- **P**: Priority routing.
- **PD**: Priority routing with delays.
- **PM**: Priority routing with minimum number of idle agents.
- **PDM**: Priority routing with delays and minimum number of idle agents.
- **W**: Weight-based routing. (our new policy)

For each example, we simulate by using *common random number* (CRN) to solve the same sample average problem.

V-model example

For all call types, service rate: 10/hour, patience rate: 10/hour, acceptable waiting time $\tau_k = 20$ seconds.

Param	case 1	case 2	case 3
λ_1	50	50	100
λ_2	50	50	10
y	12	12	13
t_1	80%	70%	70%
t_2	80%	90%	90%



λ_k: arrival rate per hour of call type k,
y: number of agents,
t_k: service level target of call type k.

V-model example results

Objective function : $F_S(\pi) = \max(t_1 - S_1(\pi, 20), 0)^2 + \max(t_2 - S_2(\pi, 20), 0)^2$.

Routing policy	Case 1			Case 2			Case 3		
	ΔS_1	ΔS_2	f^*	ΔS_1	ΔS_2	f^*	ΔS_1	ΔS_2	f^*
G	-3.4	-3.5	24	6.6	-13.5	182	6.0	-14.0	195
P	-4.0	0.9	16	6.0	-9.1	82	5.9	-5.3	28
PD	-3.2	-1.2	12	2.7	-7.1	51	4.4	-2.4	6
PM	1.0	-4.0	16	-4.5	-0.3	20	-4.0	8.4	16
PDM	-1.1	-3.2	11	-2.8	-2.3	13	4.4	-2.5	6
W	-0.9	-0.9	**2**	-0.9	-1.6	**3**	3.6	0.1	**0**

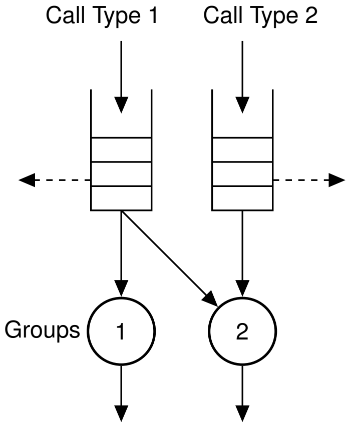
$\Delta S_k = (S_k - t_k)$ of call type k. f^*: Best cost found. All performance are measured in %.

The lowest penalty costs are obtained with the weight-based policy.

N-model example

For all call types, service rate: 10/hour, patience rate: 10/hour, acceptable waiting time $\tau_k = 20$ seconds.

Param	case 1	case 2
λ_1	100	100
λ_2	100	100
y_1	11	5
y_2	12	18
t_1	80%	85%
t_2	80%	85%



λ_k: arrival rate per hour of call type k,
y_g: number of agents in group g,
t_k: service level target of call type k.

N-model example results

Objective function: $F_{SA}(\pi) = \sum_{k=1}^{2} \max(t_k - S_k(\pi, 20), 0)^2 + \sum_{k=1}^{2} A_k(\pi)^2$.

Routing policy	Case 1					Case 2				
	ΔS_1	ΔS_2	O_1	O_2	f^*	ΔS_1	ΔS_2	O_1	O_2	f^*
G	9.4	-21.6	74	89	1494	-1.9	-8.4	80	85	194
P	7.7	-16.1	79	86	505	-3.5	-3.7	80	85	134
PD	-7.6	-8.8	82	82	137	-3.5	-3.7	80	85	134
PM	-6.2	-5.7	82	82	71	-3.5	-3.7	80	85	134
PDM	-3.9	-6.8	81	82	**63**	-6.5	-2.5	80	83	95
W	-4.8	-6.5	81	82	66	-3.8	-1.7	84	84	**18**

$\Delta S_k = S_k - t_k$. A_k: abandonment ratio(%) of call type k. f^*: Best score found.

Policy PDM performs a little better than the policy W for the N-model when the staffing is more *balanced* with the volume of calls.

Larger call center example

- 8 call types, 10 agent groups,
- Arrival rates (/ hour): $\boldsymbol{\lambda} = (250, 200, 100, 80, 50, 20, 15, 10)$,
- Service rates (/ hour): $\boldsymbol{\mu} = (10, 6, 6, 10, 6, 6, 8, 10)$,
- Patience rates (/ hour): $\boldsymbol{\nu} = (10, 8, 10, 12, 6, 10, 12, 10)$,
- Staffing vector: $\mathbf{y} = (21, 12, 14, 8, 16, 5, 3, 7, 8, 9)$,
- Group skill sets: $\mathcal{S}_1 = \{1, 4\}$, $\mathcal{S}_2 = \{2, 5\}$, $\mathcal{S}_3 = \{3, 4, 7\}$, $\mathcal{S}_4 = \{4, 6, 8\}$, $\mathcal{S}_5 = \{2, 5\}$, $\mathcal{S}_6 = \{6, 7, 8\}$, $\mathcal{S}_7 = \{1, 3, 7\}$, $\mathcal{S}_8 = \{2, 4, 8\}$, $\mathcal{S}_9 = \{1, 3, 4, 8\}$, $\mathcal{S}_{10} = \{2, 7, 8\}$.
- SL target: $t_k = 80\%$ and $\tau_k = 20$ seconds for all call types k.

Larger example results

Objective function: F_S with $a_k = 1, \forall k$.

Routing policy	ΔS max	ΔS med	ΔS min	S agg	A max	A med	A min	A agg	f^*
G	16.8	-0.2	-17.1	73	5.3	3.0	0.5	4.4	638
P, PM (*)	16.0	3.1	-11.7	75	5.9	2.0	0.6	4.3	267
PD, PDM (*)	15.4	-0.4	-10.9	75	5.5	3.3	0.8	4.3	219
W	-0.5	-1.8	-5.0	77	11.9	6.5	4.1	5.1	**58**

Objective function: F_{SA} with $a_k = b_k = 1, u_k = 0, \forall k$.

Routing policy	ΔS max	ΔS med	ΔS min	S agg	A max	A med	A min	A agg	f^*
B	16.8	2.0	-17.1	73	5.4	3.0	0.5	4.4	745
P, PM (*)	16.0	3.3	-11.5	75	5.6	2.0	0.7	4.4	350
PD, PDM (*)	14.1	2.3	-10.8	75	5.5	2.5	1.1	4.4	303
W	12.4	-1.0	-8.9	76	5.0	4.0	1.4	4.4	**233**

$\Delta S_k = S_k - t_k$. A_k: ratio of abandonments. f^*: best cost found.
agg: aggregate measure.
(*): We found the best solution with the simpler policy.

Conclusion

Summary:

- We propose a routing policy using weights, call waiting times and agent idle times.
- We presented a multiplicative and addictive weight rule, but it can take different expressions.
- The weight-based policy had the best cost for most examples.

Future work:

- Improve the optimization methods.
- Try alternative weight rules.