Appendix A.0: Approximating other performance measures

Alternative definition of service level and approximation. The *waiting time* is defined as the minimum of virtual waiting time and patience. Define an alternative service level, $\mathrm{SL}^{(2)}_{j,p}(\mathbf{x})$, as the long-run fraction of calls that are not lost and whose waiting time is less than the AWT, denoted τ throughout Appendix A.0 (regardless of class). This is the fraction defining $g_{j,p}$, modified by adding to both enumerator and denominator the expected number of arrivals that wait less than τ and abandon (that is, no arrivals are excluded in the denominator). This implies that $g_{j,p} = (\mathrm{SL}^{(2)}_{j,p} - \theta_j)/(1-\theta_j)$, where θ_j is the long-run fraction of class-j calls that wait less than τ and abandon. Our approximation of $\mathrm{SL}^{(2)}_{j,p}$, denoted $\widehat{\mathrm{SL}}^{(2)}_{j,p}$, is analogous to the LDA approximation, except that we replace the function D_{A} in (4) by the approximated probability that a call is lost or its waiting time exceeds τ, conditional upon overflow to its last station; this is the right side in (4) modified by multiplying the second term by the probability that patience exceeds τ, i.e., $e^{-\eta\tau}$.

A conceptually better approximation of $g_{j,p}$. As a better approximation of our original SL we could consider $\hat{g}_{j,p} = (\widehat{\mathrm{SL}}^{(2)}_{j,p} - \hat{\theta}_j)/(1-\hat{\theta}_j)$, where

$$\hat{\theta}_j = (\gamma_{\ell(j),j}/\lambda_j) \sum_{k=0}^{c-1} \pi_{s+k}(\mu^*) \int_0^\tau p_k(\mu^*, t)\eta e^{-\eta t} dt$$

approximates θ_j (this is obtained by conditioning on the patience time and employing the usual analysis, i.e., class-j calls queue up at station $\ell(j)$). This $\hat{g}_{j,p}$ is conceptually better than our LDA approximation because it is consistent with the SL definition (1): it excludes the quickly-abandoning calls and focuses on the waiting time. In our examples, the absolute difference between these two approximations was no more than 0.5% when aggregated over all classes. Using $\hat{g}_{j,p}$ instead of the standard approximation in the staffing algorithm did not lead to noticeably better staffing solutions. However, $\hat{g}_{j,p}$ is considerably more expensive to compute (it requires numerical integration to compute $\hat{\theta}_j$), so we view it as less attractive.

Other performance measures. The long-run fraction of class-j calls that abandon is $(\gamma_{\ell(j),j}/\lambda_j)A_{\ell(j)}$, where $A_{\ell(j)}$ is the long-run fraction of delay calls that abandon at the station $\ell(j)$, as derived in Section 3.2.The distribution of the (stationary) class-j waiting time is a mixture: with probability $\gamma_{\ell(j),j}/\lambda_j$, it is their waiting time at station $\ell(j)$; otherwise, it is zero. Conditional

on being positive, the waiting time exceeds τ with probability equal to the second term on the right side of (4) multiplied by $e^{-\eta\tau}$. Thus, the class-j expected waiting time follows by integrating this function of τ on $(0, \infty)$. The type-j service-completion rate at station i is $f_{i,j} = \gamma_{i,j}(1 - A_i)$ if $i = \ell(j)$, and $\gamma_{i,j}(1 - B_i)$ otherwise.

Appendix A.1: Pseudocodes

Figures 3 to 8 list pseudocodes for the procedures referred to in Section 4. Procedure `Search` controls the Stage-1 search as shown in the outline in Figure 2.In the descriptions that follow, $\mathcal{S}_i = \{j : i \in \mathcal{R}_j\}$ is the skill set of type-i agents; q^* is the smallest-known infeasible move size for agent removal; and NormalTermination is a boolean variable indicating whether normal termination has occurred; $|\cdot|$ denotes set cardinality; round(x) is the integer closest to x; median$(\mathbf{x})$ is the median of the elements of $\mathbf{x}$. `RandUnif`(P), where P is a finite set, denotes an element of P chosen randomly, uniformly over P. `LD`$(\mathbf{x})$ signifies a call to the loss-delay approximation for staffing vector $\mathbf{x}$; this returns the object `LD`; `LD.isFeas`, `LD.globSL`, `LD.classSL`, and `LD.served` denote the feasibility indicator, global SL, SL for all classes, and the service-completion rate for all i and j, respectively. For approximation LDN, `LD.indeterminate` is an empty set if a solution was found; otherwise, it is the station that was declared indeterminate. In the simulation-based adjustment (Figures 7 and 8), calls to `Simulate`$(\mathbf{x})$ return an object `SIM` whose fields are analogous to those of object `LD`, except that they are simulation-based estimates.

Appendix A.2: Effect of initialization

We compared several initialization methods in our standard examples. The methods are now described; in these descriptions, references are made to steps of procedure `Init` in Figure 3.

1. As in Figure 3, Alternative 5A. Favor specialist agents and require a fixed *pseudo*-SL (parameter ξ_0) in step 4. Unless indicated otherwise, $\xi_0 = 0.8$.

2. (Geometric). Minor variation of method 1; replace step 2 as follows: scan stations by increasing cost, allocating a fraction β of the yet-unallocated arrival rate; the last station receives

PROCEDURE: `Init`
OUTPUT: $\mathbf{x} = (x_i)_{i=1}^m$, an E-feasible staffing vector

CALLS: `LD`; function D (no aband.) or its counterpart D_{A} (aband.).

1. $i_j^* = \arg\min_{i \in \mathcal{R}_j} c_i, \quad j \in \mathcal{N}$ (cheapest feasible agent type, by call class)
2. Split arrival rate, by agent type and call class:

$$\tilde{f}_{i,j} = \left\{ \begin{array}{ll} \beta\lambda_j & \text{if } i = i_j^* \\ (1-\beta)\lambda_j/(|\mathcal{R}_j| - 1) & \text{if } i \in \mathcal{R}_j \setminus \{i_j^*\} \end{array} \right\}, \quad j \in \mathcal{N}$$

3. Aggregated parameters, by agent type:

$$\tilde{f}_i = \sum_{j \in \mathcal{S}_i} \tilde{f}_{i,j}, \quad \frac{1}{\tilde{\mu}_i} = \left(\sum_{j \in \mathcal{S}_i} \frac{\tilde{f}_{i,j}}{\tilde{f}_i} \frac{1}{\mu_{i,j}} \right), \quad \tilde{\tau}_i = \sum_{j \in \mathcal{S}_i} \frac{\tilde{f}_{i,j}}{\tilde{f}_i} \tau_j, \quad i \in \mathcal{M}$$

4A. $\xi = \xi_0$

4B. $x_i = \min\{x : 1 - D(\tilde{\tau}_i; x, 0, \tilde{f}_i, 0, \tilde{\mu}_i) \geq \xi\}, \quad i \in \mathcal{M}; \quad \mathbf{x} = (x_i)_{i=1}^m$
`LD` $=$ `LD`$(\mathbf{x})$; $\quad g =$ `LD.globSL`; $\quad (g_j)_{j=1}^n =$ `LD.classSL`

Alternative 5A. Ensure E-feasibility:
while (`LD.isFeas` = False)
  if (`LD.indeterminate` $= \emptyset$)
    $(f_{i,j})_{j \in \mathcal{S}_i, i \in \mathcal{M}} =$ `LD.served`; $\quad g =$ `LD.globSL`; $\quad (g_j)_{j=1}^n =$ `LD.classSL`
    if $(g < l)$
      $i = \arg\max_{i \in \mathcal{M}} \frac{\sum_{j \in \mathcal{S}_i} \frac{f_{i,j}}{\mu_{i,j}}}{x_i c_i}$
    else
      $j^* = \arg\max_{j \in \mathcal{N}} (l_j - g_j); \quad \mathcal{C}_{j^*} = \mathcal{R}_{j^*} \cap \{i : x_i \geq 1\}; \quad i = \arg\max_{i \in \mathcal{C}_{j^*}} \frac{\frac{f_{i,j^*}}{\mu_{i,j^*}}}{\sum_{j \in \mathcal{S}_i} \frac{f_{i,j}}{\mu_{i,j}}}$
    end if
  else
    $i =$ `LD.indeterminate`
  end if
  $\mathbf{x} = \mathbf{x} + \mathbf{e}_i; \quad$ `LD` $=$ `LD`$(\mathbf{x})$
end while

OR

Alternative 5B. Ensure E-feasibility (control constraint slack):
while $(g < l + (1-l)\upsilon$ or $(g_j < l_j + (1-l_j)\upsilon$ for some $j \in \mathcal{N}))$
  $\xi = \xi + (1-\xi)\zeta$
  Execute step 4B
end while

Figure 1: Initialization. Steps 5A and 5B are alternatives of ensuring the staffing is E-feasible.

PROCEDURE: `Search`
$k = 1; \quad \mathbf{x}^{(1)} = \mathbf{x}; \quad g^* = \texttt{LD.globSL}(\mathbf{x}^{(1)}); \quad q^* = \infty; \quad q_i^* = \infty, \quad \forall i \in \mathcal{M}$
NormalTermination= False
while (NormalTermination= False and User-defined termination = False)
  Select a positive integer move size $q \leq \max_i x_i^{(k)}$
  if $q < q^*$
    $[k, \mathbf{x}^{(k)}, g^*, q^*] \leftarrow \texttt{Remove}(q, k, \mathbf{x}^{(k)}, g^*, q^*)$
  else
    $[k, \mathbf{x}^{(k)}, g^*, (q_i^*)_{i\in\mathcal{M}}, \text{NormalTermination}] \leftarrow \texttt{Switch}(q, k, \mathbf{x}^{(k)}, g^*, (q_i^*)_{i\in\mathcal{M}})$
  end if
end while

Figure 2: Search integration.

$[k, \mathbf{x}^{(k)}, g^*, q^*] =$ function $\texttt{Remove}(q, k, \mathbf{x}^{(k)}, g^*, q^*)$
$\mathcal{T} = \{i : x_i^{(k)} \geq q\}$ (set of candidates)
if $(\mathcal{T} = \emptyset)$
  $q^* = \min(q^*, q)$ (update the smallest-known infeasible move size)
  return
end if
$\texttt{LD}_i = \texttt{LD}(\mathbf{x}^{(k)} - q\mathbf{e}_i),\ i \in \mathcal{T}$ (evaluate candidates)
$F = \mathcal{T} \cap \{i : \texttt{LD}_i\texttt{.isFeas} = \text{True}\}$ (set of feasible candidates)
if $(F \neq \emptyset)$
  $\tilde{g}_i = \texttt{LD}_i\texttt{.globSL}, \quad i \in F$
  $\iota_i = (g^* - \tilde{g}_i)/c_i, \quad i \in F$
  $I^* = \arg\min_{i\in F} \iota_i$ (best candidate)
  $k = k + 1; \quad \mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} - q\mathbf{e}_{I^*}; \quad g^* = \tilde{g}_{I^*}; \quad q^* = \infty$
else
  $q^* = \min(q^*, q)$ (update the smallest-known infeasible move size)
end if

Figure 3: Agent removal.

$[k, \mathbf{x}^{(k)}, g^*, (q_i^*)_{i\in\mathcal{M}}, \text{NormalTermination}] = \text{function } \texttt{Switch}(q, k, \mathbf{x}^{(k)}, g^*, (q_i^*)_{i\in\mathcal{M}})$

$\mathcal{P} = \{i : x_i \geq q,\ q_i^* > q\}$ (set of pivot candidates)

if $\mathcal{P} = \emptyset$

  if $q > 1$, return; else, NormalTermination= True; return; end if

end if

$P = \texttt{RandUnif}(\mathcal{P})$ (selected pivot)

$S = \{i : c_i < c_P\}$ (set of candidates to switch in)

if $S = \emptyset$

  $q_P^* = \min(q_P^*, q)$ (update the smallest-known infeasible move size)

  return

end if

$\texttt{LD}_i = \texttt{LD}(\mathbf{x}^{(k)} - q\mathbf{e}_P + q\mathbf{e}_i), \quad i \in S$ (evaluate candidates)

$F = S \cap \{i : \texttt{LD}_i.\texttt{isFeas} = \text{True}\}$ (set of feasible candidates)

if $(F \neq \emptyset)$

  $\tilde{g}_i = \texttt{LD}_i.\texttt{globSL}, \quad i \in F$

  $\iota_i = (g^* - \tilde{g}_i)/(c_P - c_i), \quad i \in F$

  $I^* = \arg\min_{i\in F} \iota_i$ (best candidate)

  $k = k + 1; \quad \mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} - q\mathbf{e}_P + q\mathbf{e}_{I^*}; \quad g^* = \tilde{g}_{I^*}; \quad q_i^* = \infty,\ \forall i \in \mathcal{M}$

else

  $q_P^* = \min(q_P^*, q)$ (update the smallest-known infeasible move size)

end if

Figure 4: Agent switching.

PROCEDURE: `SIMAdd`

$\texttt{SIM} = \texttt{Simulate}(\mathbf{x})$

while $(\texttt{SIM.isFeas} = \text{False})$

  $(f_{i,j})_{j\in\mathcal{S}_i, i\in\mathcal{M}} = \texttt{SIM.served}$

  $g = \texttt{SIM.globSL}; \quad (g_j)_{j=1}^n = \texttt{SIM.classSL}$

  if $(\max_{j\in\mathcal{N}}\{l_j - g_j\} > 0)$ (a constraint for some class is violated)

    $j^* = \arg\max_{j\in\mathcal{N}}(l_j - g_j)$ (j^* is class with maximum constraint violation)

    $\mathcal{C}_{j^*} = \mathcal{R}_{j^*} \cap \{i : x_i \geq 1\}$

    $i = \arg\max_{i\in\mathcal{C}_{j^*}} \frac{f_{i,j^*}/\mu_{i,j^*}}{\sum_{j\in\mathcal{S}_i} f_{i,j}/\mu_{i,j}}$ (i is agent type maximally dedicated to class j^*)

  else if $(g < l)$ (the global constraint is violated)

    $i = \arg\max_{i\in\mathcal{M}} \frac{\sum_{j\in\mathcal{S}_i} f_{i,j}/\mu_{i,j}}{x_i c_i}$ (i is agent type maximizing occupancy-to-cost ratio)

  end if

  $\mathbf{x} = \mathbf{x} + \mathbf{e}_i$

  $\texttt{SIM} = \texttt{Simulate}(\mathbf{x})$

end while

Figure 5: Adjusting a SIM-infeasible solution $\mathbf{x}$ for feasibility.

PROCEDURE: `SIMRemove`

$\mathcal{L} = \{i : x_i \geq 1\}$ (set of candidates for removal)
$k = 0$; $\mathbf{x}^{(0)} = \mathbf{x}$; `SIM` = `Simulate`($\mathbf{x}$)
while (`SIM.isFeas` = True and $\mathcal{L} \neq \emptyset$)
  $(f_{i,j})_{j \in \mathcal{S}_i, i \in \mathcal{M}}$ = `SIM.served`
  $(g_j)_{j=1}^n$ = `SIM.classSL`
  $w_{i,j} = \frac{f_{i,j}/\mu_{i,j}}{\sum_{k \in \mathcal{S}_i} f_{i,k}/\mu_{i,k}}$, $j \in \mathcal{S}_i$, $i \in \mathcal{L}$ (busy-time fractions, by agent type and call class)
  $\chi_i = \sum_{j \in \mathcal{S}_i} w_{i,j}(g_j - l_j)$, $i \in \mathcal{L}$ (excess capacity, by agent type
  Let π index $\mathcal{L}$ by decreasing $c\chi$: $c_{\pi(1)}\chi_{\pi(1)} \geq c_{\pi(2)}\chi_{\pi(2)} \geq \ldots \geq c_{\pi(|\mathcal{L}|)}\chi_{\pi(|\mathcal{L}|)}$.
  for $i = 1$ to $|\mathcal{L}|$
    $\mathbf{x} = \mathbf{x}^{(k)} - \mathbf{e}_{\pi(i)}$; `SIM` = `Simulate`($\mathbf{x}$)
    if (`SIM.isFeas` = True)
      $k = k + 1$; $\mathbf{x}^{(k)} = \mathbf{x}$
      if ($x_{\pi(i)} = 0$)
        $\mathcal{L} = \mathcal{L} \setminus \pi(i)$ (remove this type from candidate list)
      end if
      continue while
    else
      $\mathcal{L} = \mathcal{L} \setminus \pi(i)$ (remove this type from candidate list)
    end if
  end for
end while

Figure 6: Cost reduction of a SIM-feasible solution $\mathbf{x}$.

the remainder up to λ_j. Set $\xi_0 = 0.8$.

3. (Overstaffing). Allocate a large number x for each agent type.

4. (Random). The type-i agent count is a random fraction F_i of a random total staffing Y (except for rounding to the nearest integer). Replace steps 1-4 by the following:

   4.1 $(F_1, F_2, \ldots F_m) \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_m)$ with $\alpha_i = x$ for all i. (This distribution arises from independent random variables $Z_1, Z_2, \ldots, Z_m$, where Z_i has the χ^2 distribution with $2\alpha_i$ degrees of freedom; the distribution of $(Z_1, Z_2 \ldots, Z_m)/\sum_{j=1}^m Z_j$ is $\text{Dirichlet}(\alpha_1, \ldots, \alpha_m)$.)

   4.2 Y is set to ρ times U, where $\rho = \sum_{j \in \mathcal{N}} \lambda_i/\mu_i$ is the aggregate load and U is a uniform random variable on a range $[l, u]$

   4.3 $x_i = \text{round}(YF_i)$, $i \in \mathcal{M}$

   Note that $\alpha_i = x$ (all i) ensures symmetry across i, so no agent type is favored over others.

In methods 1, 2, and 4, if the solution at step 4 is E-infeasible, then E-feasibility is ensured at step 5A; in our large call center examples, this gave initial solutions with an unusually large fraction of expensive agents *relative* to the total. The methods that follow are simple remedies to this problem. The idea is to control explicitly the agent counts; in methods 5 and 6, this is done via the arrival-rate splitting step 2, and then by increasing the pseudo-SL ξ until a given constraint slack is met; in method 7, the agent counts are set as random fractions of a total staffing initialized randomly, but determined iteratively until a minimum constraint slack is met.

5. Remedy for method 1, replacing 5A by 5B. The parameter $\upsilon = 0.05$ controls the required constraint slack. The parameter $\zeta = 0.03$ controls the increase in ξ per iteration.

6. Remedy for method 2, replacing 5A by 5B. We set $\upsilon = 0.05$, $\zeta = 0.03$.

7. Remedy for method 4, replacing 5A as follows, with $\upsilon = 0.05$:

   while $(g < l + (1-l)\upsilon$ or $(g_j < l_j + (1-l_j)\upsilon$ for some $j \in \mathcal{N}))$

   $Y = 1.02Y$; Compute $\mathbf{x}$ as in 4B; $g = \texttt{LD}(\mathbf{x})\texttt{.globSL}$; $(g_j)_{j=1}^n = \texttt{LD}(\mathbf{x})\texttt{.classSL}$

   end while

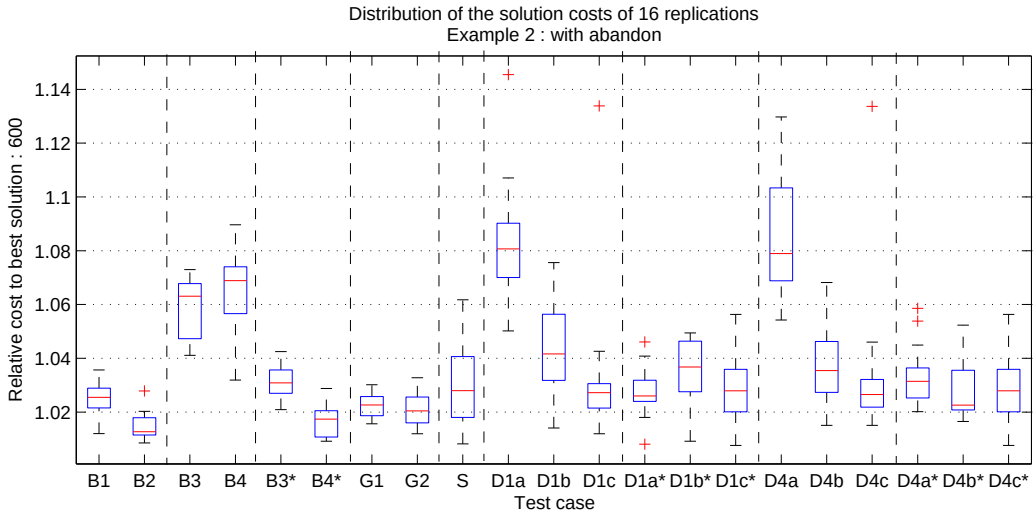

Figure 7: Boxplots of gap-to-empirical-optimum with various initializations for problem CC2A based on 16 runs. From left to right: method 1 with $\beta = 0.8$ (B1); method 1 with $\beta = 0.6$ (B2); method 1 with $\xi_0 = 0.3$ and $\beta = 0.8$ (B3); method 1 with $\xi_0 = 0.3$ and $\beta = 0.6$ (B4); method 5 with $\beta = 0.8$ (B3*); method 5 with $\beta = 0.6$ (B4*); method 2 with $\beta = 0.8$ (G1); method 2 with $\beta = 0.6$ (G2); method 3 with $x = 100$ (S); method 4 with all $a_i = 1$ and $[l, u]$ varying as $[0.8, 1]$ (D1a), $[1, 1.5]$ (D1b), and $[1.5, 2]$ (D1c); method 7, remedies of the cases of method 4 (D1a*, D1b*, D1c*, resp.); method 4, counterparts of earlier cases of method 4, changing to all $a_i = 4$ (D4a, D4b, and D4c); and method 7, counterparts of the latter three cases of method 4 (D4a*, D4b*, and D4c*).

We examined the gap-to-empirical-optimum for examples CC1L, CC1A, CC2L, and CC2A. This will be shown below via boxplots of the final cost for *all* runs (i.e., we include those where the final solution was infeasible). The average infeasibility gap $\bar{G}$ is small (as discussed in the paper), so little is lost by considering these instead of restricting results to feasible solutions. The noise due to stage 2 was controlled to be negligible compared to other sources of noise. In problems CC1L, CC1A, "Overstaffing" and "Random" did slightly worse than the others, but the gap to better initializations was small: about 0.5% in CC1L, less than 1% for CC1A.

However, large sensitivity was seen in problems CC2L and CC2A. We discuss problem CC2A; similar observations hold for CC2L. In Figure 9, we show the gap-to-empirical-optimum via boxplots. Each box has lines at the lower quartile, median, and upper quartile values; whiskers extend from the box out to the most extreme data value within 1.5 times the inter-quartile range (i.e., box height); and outliers are values beyond the ends of the whiskers, marked "+". In the four poor performers that stand out, the common theme is that in the initial staffing, there were too many expensive (many-skill) agents relative to the total. Methods 5 and 7 are effective remedies to their

counterparts.

Our results suggest: (a) there exist problem instances and initializations where our approach delivers poor solutions; (b) favoring specialist agents over more expensive ones appears to be an effective general-purpose initialization heuristic (methods 1 and 2 with ξ_0 determined by trial and error to ensure E-feasibility; and methods 5 and 6); and (c) the methods in (b) yield better final solutions than overstaffing (method 3) and randomized allocation (methods 4 and 7).

Appendix A.3: Effect of randomized search move size

The search neighborhoods depend on the move size q. We evaluated experimentally the following choices: $q = 1$; q is a random variable with uniform distribution on $\{1, 2, \ldots, \max_{i\in\mathcal{M}}(x_i^{(k)})\}$, i.e., the set of nontrivial move sizes for incumbent $\mathbf{x}^{(k)}$ (Uniform); and $q = \max(1, \text{round}(X))$, where X is an exponential random variable with mean x times the median of the components of the incumbent solution ($x*$med). We summarize empirical results over examples CC1A, CC2L, and CC2A. We turned off multistart; initialization was via method 1 with $\xi_0 = 0.8$ on all examples, $\beta = 0.7$ for CC1, and $\beta = 0.8$ for CC2. Work was controlled by a stage-1 time limit of 12 CPU hours and by setting $T = 640$ for CC1, $T = 320$ for CC2L, and $T = 160$ for CC2A. Based on 32 runs, $q = 1$ slightly underperformed in CC1A; Choices "Uniform" and "5*med" slightly underperformed in CC2L. The difference in median gap-to-empirical-optimum between the best and worst method was about 0.5% in CC1A and CC2L, and smaller in the other examples. The largest difference in minimum gap-to-empirical-optimum occurred in CC2A, where "1*med" was better (lower) than "Uniform" by about 1%. Over all examples, "0.5*med" and "1*med" were slightly better choices.

Appendix A.4: Proof of Proposition 2

Membership of μ in $\mathcal{I}$ implies $\lambda_D < s\mu$, so h is well-defined in $\mathcal{I}$. For the existence of a root, it is easy to see that h is continuous with $\lim_{\mu\to\mu_1} h(\mu) < 0$ and $h(\mu_2) > 0$. To show uniqueness of the root, first observe that $B(\mu) = \left[1 + s!\,(1 - \lambda_D/(s\mu)) \sum_{k=0}^{s-1} \left((\lambda_D + \lambda_L)/\mu\right)^{k-s}/k!\right]^{-1}$ (by simplification), so B has derivative $B'(\mu) < 0$ on $\mathcal{I}$. Differentiation of h gives $h'(\mu) = w'(\mu)(1/\mu_D - 1/\mu_L) + 1/\mu^2$, where $w'(\mu) = \lambda_D\lambda_L B'(\mu)/\{\mu_D + \mu_L[1 - B(\mu)]\}^2 < 0$ on $\mathcal{I}$; the assumption $\mu_D > \mu_L$ now implies

$h'(\mu) > 0$ on $\mathcal{I}$. □

Appendix A.5: Parameters used for the cutting-plane algorithm

CP is initialized by solving a set-covering problem that provides a solution for which the agents can cover a fraction α_j of the load of call class j for each j (in a fluid model), for some constants α_j chosen heuristically. It is recommended to use $\alpha_j = 1$ when there are low abandonments, and slightly smaller numbers if there are moderate abandonments. We followed these recommendations. For instances with abandonment, this yielded an initial solution with global SL above the target and affected negatively the solution quality; in these cases, α was iteratively decreased by 0.1 until the initial solution had a global SL below the target; this gave $\alpha = 0.9$ for example 1 and $\alpha = 0.8$ for example 2. In our experience, α was the most influential parameter. CP was also sensitive to d_i, $i \in \mathcal{M}$, the number of agents of type i added to the current solution when estimating subgradients. Cezik and Lecuyer (2008) recommend using $d_i > 1$ if the global SL is low or the simulation is not very long (is noisy). We set $d_i = 1$, with the following exceptions that gave better results: $d_i = 2$ when $T \leq 50$; in most cases, a higher d_i did not change the results significantly. In CC2A, $d_i = 1$ gave poor results, so we set $d_i = 2$ for all T.