Service and Capacity Allocation in $M/G/c/c$ State Dependent Queueing Networks

F. R. B. Cruz*, J. MacGregor Smith
Department of Mechanical & Industrial Engineering,
University of Massachusetts, Amherst, MA 01003, USA

D. C. Queiroz
Departamento de Estatística,
Universidade Federal de Minas Gerais,
31270-901 - Belo Horizonte - MG, Brazil

E-mail: {fcruz,jmsmith}@ecs.umass.edu, denise@est.ufmg.br

October 7, 2003

Abstract — **The problem of service and capacity allocation in state dependent $M/G/c/c$ queueing networks is analyzed and algorithms are developed to compute the optimal allocation c. The model is applied to the modeling of pedestrian circulation systems and basic series, merge, and split topologies are examined. Also of interest are applications to problems of evacuation planning in buildings. Computational experiments assert the algorithm's speed, robustness, and effectiveness. The results obtained indicate that the pattern of the optimal capacity surprisingly repeats over different topologies and it is also heavily dependent upon the arrival rate. Additional computational simulation results are provided to show the accuracy of the approach in all configurations tested.**

Keywords — Queueing networks, buffer allocation, finite capacity, state dependent.

1 Introduction

Many application problems including those in telecommunication, transportation, manufacturing, and service industries are most appropriately modeled as queueing networks with finite capacity and state dependent service rates. Often, the finite capacities in the queues and state dependent service rates further increase the complexity of solutions of these systems. In other cases, these assumptions may be relaxed. However, this paper is about applications for which it is fundamental to take into account finite capacities and state dependent services for the sake of accuracy. Of particular interest are $M/G/c/c$ state dependent queueing networks, *i.e.*, following Kendall's notation, queues with Markovian arrivals, general state dependent service rates, c parallel servers, and the total capacity c, including the servers.

Figure 1 illustrates a queueing network configured as a generic topology. Notice that the classical symbolic representation for each stochastic node is being used in this figure, as well as throughout this paper. However, it is worthwhile mentioning here that actually there is no queue at each $M/G/c/c$ node since the number of parallel servers equals the total capacity.

The use of queueing theory for the analysis of congestion has a long and storied existence. In the past, queueing networks have been important tools in the study of traffic light synchronization, in the analysis of vehicles at intersections [1], and in the evaluation of traffic flow by using a simplistic deterministic approach [2]. Nowadays, probably due to the increasing speed and reduced costs of the modern computer systems, more sophisticated models have been developed. These models have been used in applications such as pedestrian/vehicular network traffic analysis [3, 4, 5], synthesis [6, 7, 8, 9], and accumulating conveyor systems [10].

The main reason of this paper is to extend the development of algorithms for optimal service and capacity allocation in $M/G/c/c$ state dependent queueing networks, for a fixed generic network topology. In particular, the interest lies in pedestrian network applications, configured as a generic combination of basic series, merge, and split topologies as illustrated in Figure 2, but the extensions to other networks with state dependent service rates should be obvious.

The rest of this paper is organized as follows. Section 2 presents a mathematical programming formulation for the service and capacity allocation (SCA) problem.

*On sabbatical leave from the *Departamento de Estatística*, *Universidade Federal de Minas Gerais*, 31270-901 - Belo Horizonte - MG, Brazil.

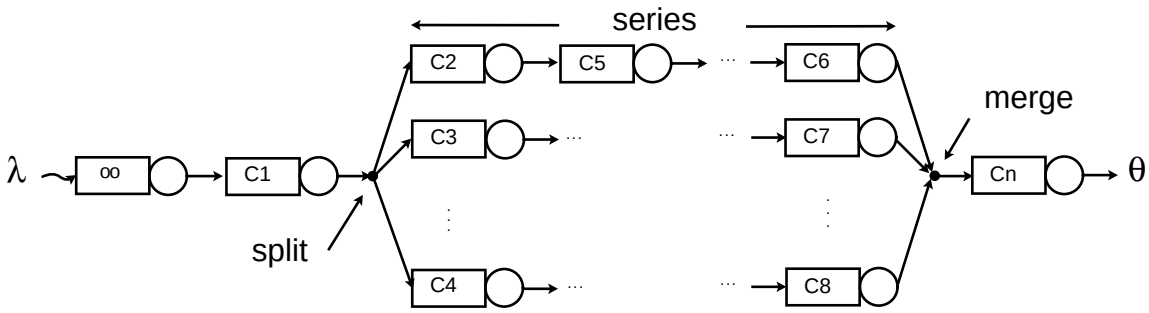


Figure 1: A generic network topology.

Section 3 presents the analytical stochastic model used to describe pedestrian flows and Section 4 describes the proposed algorithm. Computational experiments with the proposed algorithm are presented in Section 5. In order to illustrate the usefulness of the optimization algorithm in evacuation problems, Section 6 is dedicated to show results for a ten-story building evacuation network. Finally, Section 7 closes the paper with a summary and concluding remarks.

2 Problem Statement

2.1 Notation

The notation used throughout the text is provided below:

c	corridor capacity in number of occupants;
$\mathbf{c}$	capacity vector $(c_1, c_2, \ldots)^{\mathrm{T}}$;
f_i	cost per unit of capacity;
l	corridor length in meters;
w	corridor width in meters;
$\mathbf{w}$	width vector $(w_1, w_2, \ldots)^{\mathrm{T}}$;
V_n	average walking speed in a corridor for n occupants in m/s;
V_1	average lone occupant walking speed in m/s;
$f(n)$	V_n/V_1, service rate for n occupants;
V_a	average walking speed when pedestrian density equals 2 ped/m^2 in m/s;
a	$2 \times l \times w$, number of pedestrians in the corridor assuming a density of 2 ped/m^2;
V_b	average walking speed when pedestrian density equals 4 ped/m^2 in m/s;
b	$4 \times l \times w$, number of pedestrians in the corridor assuming a density of 4 ped/m^2;
γ, β	shape and scale parameters for the exponential model;
λ	pedestrian arrival rate in ped/s;
N	random variable that denotes the number of occupants in a corridor;
n	actual number of occupants in the corridor;
p_n	$\Pr\{\mathrm{N} = \mathrm{n}\}$, probability of n occupants in the corridor, for $n = 1, 2, \ldots, c$;
p_0	$\Pr\{\mathrm{N} = 0\}$, empty corridor probability;
p_c	$\Pr\{\mathrm{N} = \mathrm{c}\}$, blocking probability;
θ	throughput in ped/s;
L	$E[N]$, mean (expected) number of occupants in the corridor (known as work-in-process);
W	$E[T]$, mean waiting time (expected service time) in the corridor in seconds;
$E[T_1]$	mean waiting time for a lone occupant in the corridor in seconds.

2.2 Mathematical Programming Formulation

Assume that the topology of the network of queues is known beforehand and that it is defined as a graph $G(V, A)$, in which V is a finite set of nodes (corridors) and A is a finite set of arcs (connections between pair of corridors). The service and capacity allocation (SCA) problem is concerned with how much capacity must be provided in the nodes so that the blocking probability is below a specific threshold. In other words, the SCA problem is to find the smallest integers $c_i \geq 0$ for which $p_{c_i} = \Pr\{\mathrm{N_i} = \mathrm{c_i}\} \leq \varepsilon$, for all $i \in V$.

Thus, a possible integer mathematical programming formulation for the SCA problem is as follows:

$$z = \min \Big[g(\mathbf{c}) = \sum_{\forall\, i \in V} f_i c_i \Big], \tag{1}$$

s.t.:

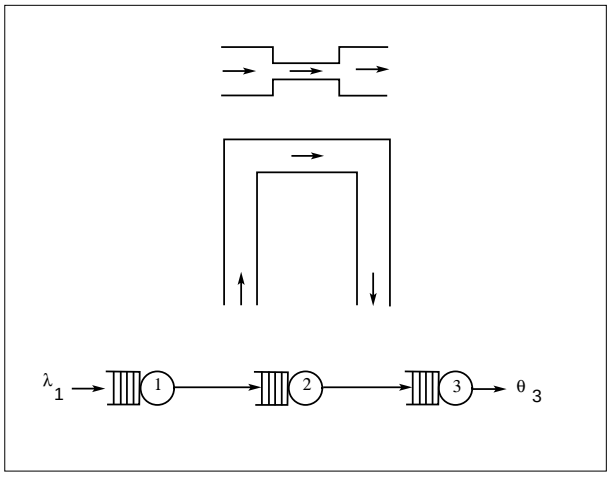
$$p_{c_i} \leq \varepsilon, \quad \forall\, i \in V, \tag{2}$$

$$c_i \in \{1, \ldots\}, \quad \forall\, i \in V, \tag{3}$$

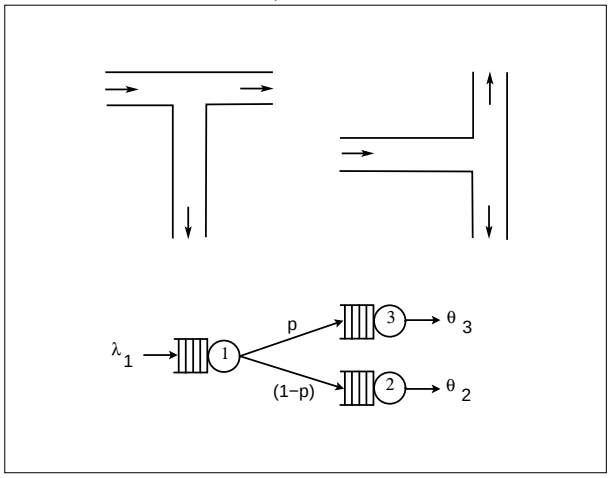
which minimizes the overall allocation cost $\sum_i f_i c_i$, constrained to provide a maximum blocking probability threshold ε, for all nodes $i \in V$, in which f_i is the cost per unit of capacity.

In spite of the linearity of the objective function, the SCA problem has inherent complications. One serious aspect to deal with from a practical point of view is the intractability of the expressions for p_{c_i} in closed form for any given topology. Notice that, in a topology such as the one seen in Figure 1, the blocking probability at the ith node depends on all upstream incoming flows and also on the blocking probabilities of all downstream nodes.

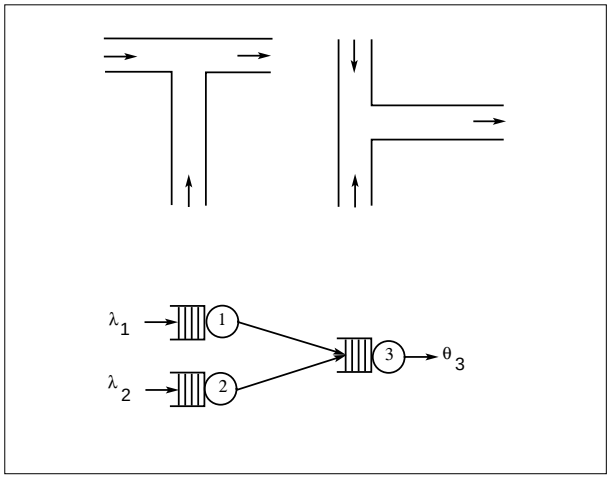
In fact, one can see in Figure 3 how complex are the blocking probabilities, p_{c_1} and p_{c_2}, as a function of ca-



a) series



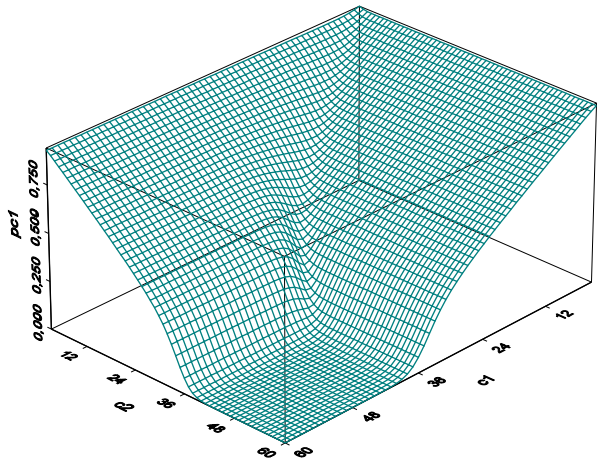
b) split



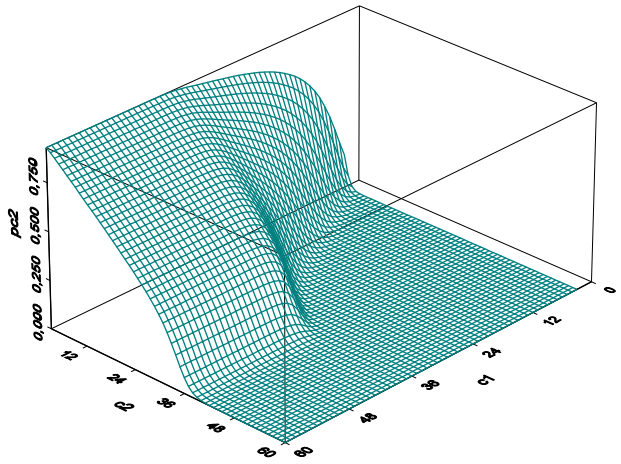
c) merge

Figure 2: Corridors in basic topologies.

pacities c_1 and c_2, even in a simple 2-node tandem configuration. Notice that the "flat" areas of the curves corresponds to the feasible regions, *i.e.*, points for which $p_{c_1} \leq \varepsilon$ and $p_{c_2} \leq \varepsilon$ hold.



a) node 1



b) node 2

Figure 3: Blocking probabilities p_{c_1} and p_{c_2} in a 2-node tandem network.

3 Flow Modeling

3.1 Analytical Model for a Single Traffic Link

An $M/G/c/c$ state dependent model is a reasonable and a most appropriate conceptual model to describe a single traffic link because of its finite capacity and of its very general service mechanism. The limiting probabilities for the number of users in an $M/G/c/c$ queueing system have been developed [3] and showed [4] to be stochastic equivalent to a *pure* Markovian $M/M/c/c$ queueing model. Thus, these probabilities can be written as follows:

$$p_n = \left\{ \frac{\left[\lambda E[T_1]\right]^n}{n! f(n) \cdots f(2) f(1)} \right\} p_0, \tag{4}$$

for $n = 1, 2, \ldots, c$, in which

$$p_0^{-1} = 1 + \sum_{i=1}^{c} \left\{ \frac{\left[\lambda E[T_1]\right]^i}{i! f(i) \cdots f(2) f(1)} \right\},$$

is the empty system probability, λ is the arrival rate in ped/s, $E[T_1]$ is the mean waiting (service) time for a lone occupant in seconds, and $f(n) = V_n/V_1$ is the service rate. From Eq. (4), one can derive all performance measures of interest:

$$\begin{cases} \theta &= \lambda\Big(1 - p_c\Big), \\ L &= E[N] = \sum_{n=1}^{c} n p_n, \\ W &= E[T] = L/\theta, \end{cases} \tag{5}$$

in which p_c is the blocking probability, θ is the throughput in ped/s, L is the mean number of customers in the system (also known as work-in-process, WIP), and W, derived from Little's formula, is the mean waiting (service) time in seconds.

3.2 Congestion Models

Finally, we derive the service rate $f(n)$ as to be used in Eq. 4. For instance, a corridor connecting locations i to j may be considered as a service mechanism for its occupants since it provides the path for moving from point i to point j. The number of servers in parallel equals the nodal capacity which also represents the total number of users simultaneously allowed in the system, that is:

$$c = \lfloor k \times l \times w \rfloor, \tag{6}$$

in which l is the nodal length, w is its width, k is the maximum pedestrian density in ped/m^2, and $\lfloor x \rfloor$ is the largest integer not superior to x. Notice that c must be integer so that it works in the probability calculations earlier shown.

In accordance to Tregenza's empirical studies [11], curves (a) through (f), in Figure 4, the average speed that a user will move through a corridor depends on several factors but mainly this speed is a function of the number of occupants therein. Also in accordance to the empirical data from Figure 4, $k = 5$ ped/m^2 represents a reasonable maximum pedestrian density above which we could assume that the pedestrian average walking speed is essentially zero.

Based on these remarks, Yuhaski and MacGregor Smith [3] developed linear and exponential congestion models for the average pedestrian walking speed in traffic links:

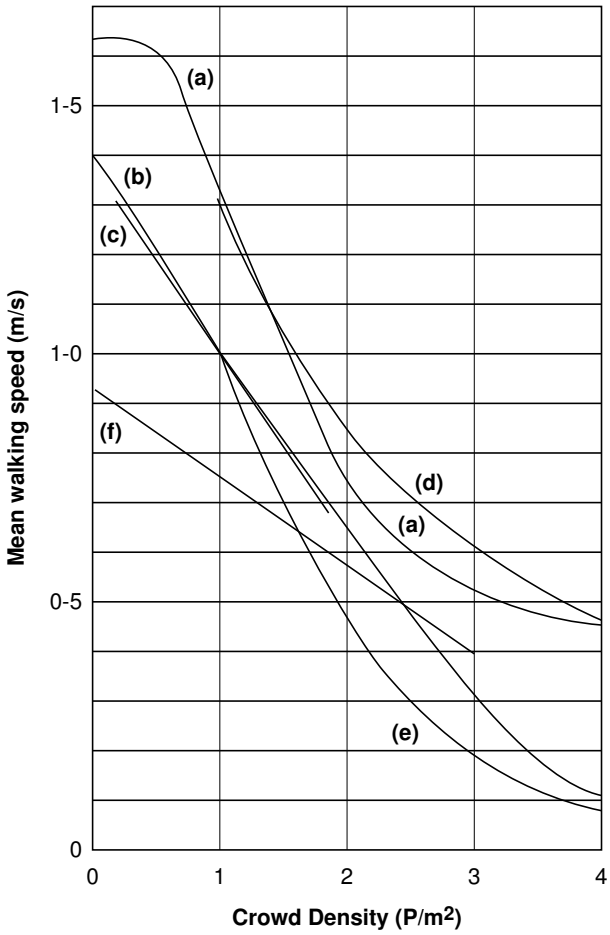


Figure 4: Average walking speed for pedestrians.

$$V_n = V_1 \frac{c + 1 - n}{c}, \tag{7}$$

and

$$V_n = V_1 \exp\left[-\left(\frac{n-1}{\beta} \right)^{\gamma} \right], \tag{8}$$

in which

$$\gamma = \ln\left[\frac{\ln(V_a/V_1)}{\ln(V_b/V_1)} \right] / \ln\left(\frac{a-1}{b-1} \right),$$

$$\beta = \frac{a-1}{[\ln(V_1/V_a)]^{1/\gamma}} = \frac{b-1}{[\ln(V_1/V_b)]^{1/\gamma}},$$

V_1 is the average speed for a lone occupant, assumed 1.5 m/s, V_a is the average speed in m/s when the crowd density is 2 ped/m^2, $a = 2lw$, V_b is the average speed when the crowd density is 4 ped/m^2, and $b = 4lw$.

Yuhaski and MacGregor Smith [3] also pointed out that the exponential model could be adjusted based on 3 points averaged over the six curves presented in Figure 4. Other possibilities also exists, *e.g.* non-linear regression or piece-wise linear approximations, but the results would not differ significantly. Throughout this paper, it is assumed that $V_a = 0.64$ m/s and $V_b = 0.25$

```
algorithm
    read G(V, A)
    read routing probabilities p_ij, ∀ (i, j) ∈ A
    read arrival rates λ_i and f_i, ∀ i ∈ V
        /* find an initial feasible vector c */
    for ∀ i ∈ V do
        c_i^opt ← c_i^sup ← 2^M
    end for
        /* search optimum solution */
    iter ← 0
    repeat
        iter ← iter + 1
        for ∀ i ∈ V do
            /* optimizes ith queue */
            OptQueue(i,c^sup)
        end for
            /* update best solution */
        if g(c^sup) < g(c^opt) then
            c^opt ← c^sup
            unmark all nodes
        else
            exit
        end if
    end repeat
    write c^opt
end algorithm
```

a) network optimization

```
algorithm OptQueue(i,c^sup)
        /* recursive labeling step */
    for ∀ (j, i) ∈ A do
        if node j is unmarked then
            OptQueue(j,c^sup)
            mark node j
        end if
    end for
        /* isolate optimum */
    j ← 1
    c_i^inf ← c_i^sup ← 2^j
    while p_(c_i^sup) ≰ ε, ∀ i ∈ V
        j ← j + 1
        c_i^sup ← 2^j
    end while
        /* narrow interval */
    c^can ← c^sup
    while (c_i^sup − c_i^inf) > 1
        c_i^can ← (c_i^inf + c_i^sup)/2
        if p_(c_i^can) ≤ ε, ∀ i ∈ V then
            c_i^sup ← c_i^can;
        else
            c_i^inf ← c_i^can;
        end if
    end while
end algorithm
```

b) single queue optimization

Figure 5: SCA problem resolution algorithm.

m/s. A discrete-event digital simulation model [12] has confirmed the accuracy of these settings. Additionally, it is worthwhile mentioning that Cheah and MacGregor Smith [4] successfully extended the exponential model to represent bi-directional and multi-directional pedestrian flows by using slightly different values for V_a and V_b.

3.3 The Generalized Expansion Method

It seems unlikely that exact analytical methods will ever be available to treat complex topologies such as those presented in Figure 1. Thus, one is forced to use some type of approximation such as the Generalized Expansion Method (GEM), proposed by Kerbache and MacGregor Smith [13, 14]. The GEM is a combination of repeated trials and node-by-node decomposition approximation methods, with a key characteristic that an artificial holding node is added preceding each finite queue in the network in order to register blocked customers that attempt to enter a finite node when it is at capacity. By the addition of holding nodes, the queueing network is "expanded" into an equivalent Jackson network, in which each node can then be analyzed separately.

Since its development, the GEM has been applied successfully to many finite capacity systems [3, 7, 8, 9]. Details on how the GEM is adapted to $M/G/c/c$ state dependent queueing networks will not be given here but can be found in the literature [4, 9].

4 Proposed Algorithm

In the vast literature on buffer allocation problems, many of the approaches already described could be adapted to solve the SCA problem. The algorithm proposed here is inspired by these approaches [15]. The algorithm is shown in Figure 5 in pseudo-code.

Figure 5-a show the main algorithm that implements a variation of the derivative free coordinate search method. All settings are read and an initial feasible solution is sought which is to find a large enough capacity to all nodes that will make sure that no queue will be blocked at all, *i.e.*, that all the constraints are satisfied. For convenience, since it will help the local search algorithm as described as follows, the initial feasible capacity is in the form 2^{M}, in which M is an integer big enough to ensure feasibility.

The local search is presented in Figure 5-b. This single queue optimization algorithm first isolates the optimum by coming up with an interval whose inferior limit $c_i^{\inf}$ is infeasible and superior limit $c_i^{\sup}$ is feasible. The following steps are to successively halve the initial interval up to that point they differ by the unity. Then, the superior limit is the smallest capacity that complies with the blocking probability requirements for the whole network and the inferior limit is the largest infeasible capacity.

As a last but not least important note concerning the local search algorithm, Figure 5-b, it is necessary to ensure that no node is locally optimized unless all of its predecessor nodes were already optimized. After much experimentation, we observed that the GEM tends to underestimate the blocking probabilities in situations in

Table 1: Tandem (series) topology results.

			$\max p_{c_i}$			
				simulation		
λ	**c** (ped)	**w** (m)	GEM	average	95% CI	cpu
1.0	41 → 44 → 45	1.04 → 1.11 → 1.14	0.0024	0.0021	[0.0013 ;0.0029]	0m 55s
	42 → 45 → 46	1.06 → 1.14 → 1.16	0.0010*	0.0014	[-0.0001;0.0028]	1m 47s
	43 → 46 → 47	1.09 → 1.16 → 1.19	0.0004	0.0004	[0.0001 ;0.0006]	2m 1s
	41 → 44 → 45→ 46 → 47	1.04 → 1.11 → 1.14→ 1.16 → 1.19	0.0024	0.0021	[0.0013 ;0.0029]	6m 22s
	42 → 45 → 46→ 47 → 48	1.06 → 1.14 → 1.16→ 1.19 → 1.21	0.0010*	0.0024	[-0.0003;0.0051]	3m 56s
	43 → 46 → 47→ 48 → 49	1.09 → 1.16 → 1.19→ 1.21 → 1.24	0.0004	0.0004	[0.0001 ;0.0006]	3m 6s
2.0	78 → 78 → 81	1.96 → 1.96 → 2.04	0.0022	0.0128	[0.0017 ;0.0240]	5m 16s
	79 → 79 → 82	1.99 → 1.99 → 2.06	0.0010*	0.0029	[-0.0002;0.0059]	4m 59s
	80 → 80 → 83	2.01 → 2.01 → 2.09	0.0004	0.0095	[-0.0042;0.0231]	5m 1s
	78 → 78 → 81→ 82 → 82	1.96 → 1.96 → 2.04→ 2.06 → 2.06	0.0023	0.0128	[0.0017 ;0.0240]	10m 19s
	79 → 79 → 82→ 83 → 83	1.99 → 1.99 → 2.06→ 2.09 → 2.09	0.0010*	0.0029	[-0.0002;0.0059]	9m 36s
	80 → 83 → 83→ 84 → 84	2.01 → 2.01 → 2.09→ 2.11 → 2.11	0.0004	0.0095	[-0.0042;0.0231]	9m 53s
4.0	150 → 151 → 154	3.76 → 3.79 → 3.86	0.0021	0.0000	[0.0000 ;0.0000]	16m 49s
	151 → 152 → 155	3.79 → 3.81 → 3.89	0.0010*	0.0000	[0.0000 ;0.0000]	17m 4s
	152 → 153 → 156	3.81 → 3.84 → 3.91	0.0005	0.0000	[0.0000 ;0.0000]	16m 46s
	150 → 151 → 154→ 156 → 157	3.76 → 3.79 → 3.86→ 3.91 → 3.94	0.0021	0.0000	[0.0000 ;0.0000]	41m 42s
	151 → 152 → 155→ 157 → 158	3.79 → 3.81 → 3.89→ 3.94 → 3.96	0.0010*	0.0000	[0.0000 ;0.0000]	40m 24s
	152 → 153 → 156→ 158 → 159	3.81 → 3.84 → 3.91→ 3.96 → 3.99	0.0005	0.0000	[0.0000 ;0.0000]	40m 55s

* optimization algorithm best solution

which there is a severe bottleneck in the final nodes of the network. As a consequence, if the local search proceeds randomly over the nodes, the optimal capacity vector computed may be infeasible. Thus, a labeling initial step must be programmed, Figure 5-b, in order to avoid such an undesirable situation. The labeling algorithm is inspired by Dijkstra's shortest path algorithm [16], which is used here in its recursive version.

5 Computational Experiments

The proposed algorithms were coded in C++, a flexible and efficient programming language. All computational experiments were carried out on a PC, CPU Pentium II 400 MHz, 256 MB RAM, under Windows NT 4.0 operating system.

For the sake of the argument, only the exponential congestion model, Eq. 8, was considered. Unitary per-unit costs f_i were assumed but their value does not change the solutions since the threshold blocking must be satisfied anyway. A threshold of 0.1% (0.001) was used. Several topologies were under study, namely series, merges, and splits (see Figure 2). Experiments were done for 3 and 5-node networks with arrival rates of 1, 2, and 4 ped/s.

A well-known fact, since the works of Yuhaski and MacGregor Smith [3] and Mitchell and MacGregor Smith [9], is that the width has a more significant effect than the length over the performance measures of $M/G/c/c$ queueing systems. Thus, only homogeneous (same length) networks were considered and it was assumed that the widths were under design control. All nodes were assumed 8.0 meters long.

Additionally, for each network considered, in order to confirm the accuracy of the optimal solutions, simulations were performed for the optimal and around the optimal solutions, with slight perturbations on the capacity of the queues. A discrete-event digital simulation model [12] was used for this purpose. All simulations were run for 22,000 seconds, with a *burn-in* period of 2,000 seconds. In order to compute 95% confidence intervals, 30 replications were performed. The cpu times reported are only for the simulations. The optimization algorithm usually takes less than a minute to generate a solution.

Tables 1, 2, and 3 show the results. Notice in Table 1 that the nodes tend to be wider at the end of the topology. The effect of blocking at the end nodes is amplified back at the upstream nodes so that some extra space must be allocated in order to avoid the effect and to meet the requested performance. Additionally, it is remarkable that this progressive increasing allocation pattern repeats over longer networks. However, the pattern was not always observed. It seems that it depends on the length of the corridors since, in other experimental set with tandem networks, assuming 8.5 meters long nodes (not shown), some of the optimal capacities found were equal for all nodes.

Table 2 and 3 shows the results obtained for split and merge topologies. For split topologies, unbalanced splitting probabilities of 0.6 and 0.4 were considered, and, as a consequence, unbalanced allocations were obtained for the nodes following the splitting node. For merging topologies, the arrival rates at the two front nodes were balanced and, as expected, balanced capacities were allocated there. In all 5-node networks, a similar effect as in the series topologies (*i.e.*, a progressively increasing

Table 2: Split topology results.

			$\max p_{c_i}$			
				simulation		
λ	**c** (ped)	**w** (m)	GEM	average	95% CI	cpu
1.0	41 → 29, 21	1.04 → 0.74, 0.54	0.0025	0.0018	[0.0011 ;0.0024]	1m 4s
	42 → 30, 22	1.06 → 0.76, 0.56	0.0010*	0.0008	[0.0004 ;0.0012]	1m 3s
	43 → 31, 23	1.09 → 0.79, 0.59	0.0004	0.0004	[0.0001 ;0.0006]	1m 2s
	41 → 29 → 31, 21 → 23	1.04 → 0.74 → 0.79, 0.54 → 0.59	0.0025	0.0018	[0.0011 ;0.0024]	1m 37s
	42 → 30 → 32, 22 → 24	1.06 → 0.76 → 0.81, 0.56 → 0.61	0.0010*	0.0008	[0.0004 ;0.0012]	1m 36s
	43 → 31 → 33, 23 → 25	1.09 → 0.79 → 0.84, 0.59 → 0.64	0.0004	0.0004	[0.0001 ;0.0006]	1m 35s
2.0	78 → 49, 35	1.96 → 1.24, 0.89	0.0020	0.0046	[0.0028 ;0.0065]	2m 58s
	79 → 50, 36	1.99 → 1.26, 0.91	0.0009*	0.0031	[0.0010 ;0.0051]	2m 54s
	80 → 51, 37	2.01 → 1.29, 0.94	0.0004	0.0007	[-0.0001;0.0015]	2m 50s
	78 → 49 → 50, 35 → 37	1.96 → 1.24 → 1.26, 0.89 → 0.94	0.0023	0.0107	[0.0060 ;0.0154]	4m 50s
	79 → 50 → 51, 36 → 38	1.99 → 1.26 → 1.29, 0.91 → 0.96	0.0010*	0.0037	[0.0012 ;0.0063]	4m 38s
	80 → 51 → 52, 37 → 39	2.01 → 1.29 → 1.31, 0.94 → 0.99	0.0004	0.0009	[-0.0004;0.0021]	4m 31s
4.0	150 → 93, 63	3.76 → 2.34, 1.59	0.0025	0.0346	[0.0117 ;0.0575]	9m 52s
	151 → 94, 64	3.79 → 2.36, 1.61	0.0011*	0.0175	[-0.0005;0.0354]	9m 10s
	152 → 95, 65	3.81 → 2.39, 1.64	0.0005	0.0004	[0.0000 ;0.0009]	8m 39s
	150 → 93 → 127, 63 → 63	3.76 → 2.34 → 3.19, 1.59 → 1.59	0.0031	0.0822	[0.0509 ;0.1135]	18m 31s
	151 → 94 → 128, 64 → 64	3.79 → 2.36 → 3.21, 1.61 → 1.61	0.0014*	0.0340	[0.0122 ;0.0558]	16m 23s
	152 → 95 → 129, 65 → 65	3.81 → 2.39 → 3.24, 1.64 → 1.64	0.0006	0.0090	[-0.0015;0.0194]	15m 30s)

* optimization algorithm best solution

capacity allocation) was observed in the tandem links. Finally, one can surprisingly see an economy-of-scale effect since neither the capacity at the node after merging equals the sum of capacities of nodes just before merging in the merge topologies, nor the sum of capacities of nodes that follow a split equals the capacity of the node before splitting.

6 Evacuation Networks

In this section, a simple and yet interesting and relevant application for the SCA problem is illustrated. The objective is to design an optimal evacuation network for a ten-story building, see Figure 6. Stairwells 8.5 meters long are assumed to interconnect each floor and each of them is accessible by a corridor 8.5 meters in length, forming a 20 node series-merge network, as seen in Figure 6. Each corridor and stairwell is modeled as an $M/G/c/c$ state dependent queue and it is assumed that the widths are under design control. An arrival rate λ equally assigned for each of the ten floors (queues) is considered but other configurations could be treated as well. The arrivals at the middle of the network complicate the analysis since significant blocking can occur if sufficient capacity is not provided.

Table 4 shows the optimal allocations, for a threshold value of 0.1% and three different arrival rates of 0.25, 0.50, and 1.0 ped/s. In order to check the accuracy of this solution, additional simulations were performed. For each arrival rate, 30 replications were run, for 42,000 seconds with a *burn-in* of 2,000 seconds. The simulation times are rather long and are shown in the bottom of the table. It is possible to confirm within 95% of confidence that the threshold is respected for all nodes and that all

Table 3: Merge topology results.

			$\max p_{c_i}$			
				simulation		
λ	**c** (ped)	**w** (m)	GEM	average	95% CI	cpu
1.0	23 → 43 ← 23	0.59 → 1.09 ← 0.59	0.0023	0.1067	[0.0401 ;0.1733]	1m 39s
	24 → 44 ← 24	0.61 → 1.11 ← 0.61	0.0008*	0.0481	[-0.0019;0.0981]	1m 21s
	25 → 45 ← 25	0.64 → 1.14 ← 0.64	0.0003	0.0289	[-0.0109;0.0687]	1m 15s
	23 → 24 → 44 ← 24 ← 23	0.59 → 0.61 → 1.11 ← 0.61 ← 0.59	0.0028	0.0342	[0.0037 ;0.0647]	2m 5s
	24 → 25 → 45 ← 25 ← 24	0.61 → 0.64 → 1.14 ← 0.64 ← 0.61	0.0010*	0.0394	[-0.0028;0.0816]	2m 20s
	25 → 26 → 46 ← 26 ← 25	0.64 → 0.66 → 1.16 ← 0.66 ← 0.64	0.0003	0.0042	[-0.0006;0.0091]	1m 39s
2.0	41 → 80 ← 41	1.04 → 2.01 ← 1.04	0.0024	0.0284	[-0.0108;0.0676]	3m 25s
	42 → 81 ← 42	1.06 → 2.04 ← 1.06	0.0010*	0.0093	[-0.0089;0.0275]	3m 4s
	43 → 82 ← 43	1.09 → 2.06 ← 1.09	0.0004	0.0005	[0.0002 ;0.0007]	2m 51s
	41 → 44 → 82 ← 44 ← 41	1.04 → 1.11 → 2.06 ← 1.11 ← 1.04	0.0024	0.0088	[-0.0084;0.0259]	5m 23s
	42 → 45 → 83 ← 45 ← 42	1.06 → 1.14 → 2.09 ← 1.14 ← 1.06	0.0010*	0.0009	[0.0005 ;0.0013]	4m 32s
	43 → 46 → 84 ← 46 ← 43	1.09 → 1.16 → 2.11 ← 1.16 ← 1.09	0.0004	0.0005	[0.0002 ;0.0007]	4m 27s
4.0	78 → 151 ← 78	1.96 → 3.79 ← 1.96	0.0017	0.0015	[0.0000 ;0.0029]	8m 50s
	79 → 152 ← 79	1.99 → 3.81 ← 1.99	0.0008*	0.0008	[0.0000 ;0.0017]	8m 36s
	80 → 153 ← 80	2.01 → 3.84 ← 2.01	0.0003	0.0002	[0.0000 ;0.0004]	8m 40s
	78 → 78 → 153 ← 78 ← 78	1.96 → 1.96 → 3.84 ← 1.96 ← 1.96	0.0011	0.0294	[0.0088 ;0.0500]	16m 32s
	79 → 79 → 154 ← 79 ← 79	1.99 → 1.99 → 3.86 ← 1.99 ← 1.99	0.0010*	0.0031	[-0.0001;0.0063]	15m 13s
	80 → 80 → 155 ← 80 ← 80	2.01 → 2.01 → 3.89 ← 2.01 ← 2.01	0.0004	0.0022	[-0.0013;0.0057]	14m 48s

* optimization algorithm best solution

solutions are feasible.

In order to show the optimality of all solutions, perturbed configurations were also considered in which 10% was reduced for each optimal width. The results may be observed in Table 5. The simulation times are even longer and blocking increases dramatically. Thus, within 10%, which may represent as little as 4 centimeters in the upper stories stairwell widths, optimality is reached in this cases. However, concerning the lower levels, for which the flows are considerably higher, it is worthwhile mentioning that, even reduced by 10%, these nodes were able to accommodate the flow without blocking.

The immediate explanation one might think of is that the width (capacity) reductions at the upper stories certainly will lead to increases in the blocking at these levels as well as to reductions in the service rates. As a consequence, downstream (lower level) nodes will not be congested after all, in spite of their own 10% width reductions. However, simulation results (not shown) seem to indicate that probably the explanation is not universal since the GEM actually may be overestimating the actual blocking probabilities.

Figure 7 shows some additional performance measures for the optimized 10-story building. Here, one can see a very close agreement between the analytical and simulation methods. Additionally, the throughput corresponds exactly to the arrival rate at the node, and the expected number of customers in the system is maximum. Thus, minimizing the widths subject to a threshold is roughly equivalent to an optimization problem as (i) maximizing the throughput subjected to using the minimum space, and also to (ii) minimizing the number of users in the systems subjected to a blocking probability threshold. Concerning the expected service times (expected delay),

Table 4: Optimal results.

	$\lambda = 0.25$					$\lambda = 0.50$					$\lambda = 1.00$				
			p_{c_i}					p_{c_i}					p_{c_i}		
				simulation*					simulation*					simulation*	
Node	**w**	**c**	GEM	average	[95% IC]	**w**	**c**	GEM	average	[95% IC]	**w**	**c**	GEM	average	[95% IC]
20	0.37	15	0.0010	0.0008	[0.0005 ;0.0011]	0.60	25	0.0010	0.0021	[-0.0006;0.0048]	1.07	45	0.0010	0.0162	[0.0072 ;0.0253]
19	0.37	15	0.0010	0.0009	[0.0005 ;0.0014]	0.60	25	0.0010	0.0011	[0.0007 ;0.0014]	1.07	45	0.0010	0.0008	[0.0000 ;0.0017]
18	0.37	15	0.0009	0.0009	[0.0006 ;0.0011]	0.60	25	0.0008	0.0009	[0.0006 ;0.0012]	1.07	45	0.0005	0.0005	[0.0003 ;0.0007]
17	0.37	15	0.0009	0.0008	[0.0005 ;0.0010]	0.60	25	0.0008	0.0009	[0.0007 ;0.0012]	1.07	45	0.0005	0.0006	[0.0003 ;0.0008]
16	0.37	15	0.0009	0.0007	[0.0005 ;0.0009]	0.60	25	0.0008	0.0009	[0.0007 ;0.0012]	1.07	45	0.0005	0.0002	[0.0001 ;0.0003]
15	0.37	15	0.0009	0.0009	[0.0006 ;0.0011]	0.60	25	0.0008	0.0008	[0.0005 ;0.0010]	1.07	45	0.0005	0.0005	[0.0003 ;0.0007]
14	0.37	15	0.0009	0.0008	[0.0005 ;0.0010]	0.60	25	0.0008	0.0009	[0.0006 ;0.0011]	1.07	45	0.0005	0.0002	[0.0001 ;0.0004]
13	0.37	15	0.0009	0.0010	[0.0008 ;0.0013]	0.60	25	0.0008	0.0009	[0.0007 ;0.0010]	1.07	45	0.0005	0.0006	[0.0004 ;0.0009]
12	0.37	15	0.0009	0.0008	[0.0006 ;0.0010]	0.60	25	0.0008	0.0007	[0.0005 ;0.0008]	1.07	45	0.0005	0.0003	[0.0001 ;0.0004]
11	0.37	15	0.0009	0.0009	[0.0007 ;0.0011]	0.60	25	0.0008	0.0008	[0.0007 ;0.0010]	1.07	45	0.0005	0.0006	[0.0004 ;0.0009]
10	0.41	17	0.0001	0.0013	[-0.0010;0.0035]	0.65	27	0.0002	0.0054	[-0.0018;0.0125]	1.07	45	0.0005	0.0466	[0.0235 ;0.0696]
9	0.65	27	0.0001	0.0042	[-0.0018;0.0103]	1.12	47	0.0001	0.0051	[-0.0029;0.0131]	2.01	85	0.0003	0.0083	[-0.0036;0.0202]
8	0.93	39	0.0000	0.0000	[0.0000 ;0.0001]	1.59	67	0.0000	0.0000	[0.0000 ;0.0000]	2.93	124	0.0001	0.0000	[0.0000 ;0.0000]
7	1.21	51	0.0000	0.0000	[0.0000 ;0.0000]	2.13	90	0.0000	0.0000	[0.0000 ;0.0000]	3.89	165	0.0000	0.0000	[0.0000 ;0.0000]
6	1.45	61	0.0000	0.0000	[0.0000 ;0.0000]	2.65	112	0.0000	0.0000	[0.0000 ;0.0000]	4.84	205	0.0000	0.0000	[0.0000 ;0.0000]
5	1.78	75	0.0000	0.0000	[0.0000 ;0.0000]	3.17	134	0.0000	0.0000	[0.0000 ;0.0000]	5.75	244	0.0000	0.0000	[0.0000 ;0.0000]
4	1.97	83	0.0000	0.0000	[0.0000 ;0.0000]	3.57	151	0.0000	0.0000	[0.0000 ;0.0000]	6.72	285	0.0000	0.0000	[0.0000 ;0.0000]
3	2.20	93	0.0000	0.0000	[0.0000 ;0.0000]	4.04	171	0.0000	0.0000	[0.0000 ;0.0000]	7.66	325	0.0000	0.0000	[0.0000 ;0.0000]
2	2.44	103	0.0000	0.0000	[0.0000 ;0.0000]	4.51	191	0.0000	0.0000	[0.0000 ;0.0000]	8.62	366	0.0000	0.0000	[0.0000 ;0.0000]
1	2.72	115	0.0000	0.0000	[0.0000 ;0.0000]	5.00	212	0.0000	0.0000	[0.0000 ;0.0000]	9.59	407	0.0000	0.0000	[0.0000 ;0.0000]
				* cpu: 0h 19m 48s					* cpu: 1h 33m 40s					* cpu: 7h 01m 40s	

Table 5: Perturbed results.

	$\lambda = 0.25$					$\lambda = 0.50$					$\lambda = 1.00$				
			p_{c_i}					p_{c_i}					p_{c_i}		
				simulation*					simulation*					simulation*	
Node	0.9**w**	**c**	GEM	average	[95% IC]	0.9**w**	**c**	GEM	average	[95% IC]	0.9**w**	**c**	GEM	average	[95% IC]
20	0.33	14	0.0099	0.0411	[0.0240 ;0.0583]	0.54	22	0.1075	0.2964	[0.2820 ;0.3107]	0.96	40	0.2058	0.3664	[0.3507 ;0.3820]
19	0.33	14	0.0099	0.0337	[0.0152 ;0.0521]	0.54	22	0.1075	0.1521	[0.1242 ;0.1800]	0.96	40	0.2058	0.2137	[0.1866 ;0.2408]
18	0.33	14	0.0028	0.0155	[0.0022 ;0.0289]	0.54	22	0.0141	0.0216	[0.0172 ;0.0260]	0.96	40	0.0288	0.0494	[0.0317 ;0.0672]
17	0.33	14	0.0028	0.0105	[0.0017 ;0.0194]	0.54	22	0.0141	0.0134	[0.0124 ;0.0144]	0.96	40	0.0288	0.0303	[0.0275 ;0.0331]
16	0.33	14	0.0028	0.0072	[0.0020 ;0.0124]	0.54	22	0.0141	0.0142	[0.0133 ;0.0150]	0.96	40	0.0288	0.0287	[0.0261 ;0.0314]
15	0.33	14	0.0028	0.0047	[0.0025 ;0.0070]	0.54	22	0.0141	0.0143	[0.0133 ;0.0154]	0.96	40	0.0288	0.0278	[0.0252 ;0.0303]
14	0.33	14	0.0028	0.0030	[0.0022 ;0.0038]	0.54	22	0.0141	0.0144	[0.0130 ;0.0157]	0.96	40	0.0288	0.0306	[0.0280 ;0.0333]
13	0.33	14	0.0028	0.0034	[0.0029 ;0.0038]	0.54	22	0.0141	0.0143	[0.0133 ;0.0152]	0.96	40	0.0288	0.0270	[0.0244 ;0.0296]
12	0.33	14	0.0028	0.0028	[0.0023 ;0.0033]	0.54	22	0.0141	0.0136	[0.0125 ;0.0148]	0.96	40	0.0288	0.0280	[0.0256 ;0.0304]
11	0.33	14	0.0028	0.0031	[0.0028 ;0.0035]	0.54	22	0.0141	0.0135	[0.0125 ;0.0145]	0.96	40	0.0288	0.0280	[0.0253 ;0.0307]
10	0.37	15	0.0071	0.1239	[0.0929 ;0.1549]	0.59	25	0.0948	0.4386	[0.4211 ;0.4560]	0.96	40	0.1822	0.4598	[0.4461 ;0.4735]
9	0.59	24	0.0067	0.1439	[0.1106 ;0.1772]	1.01	42	0.0945	0.4552	[0.4393 ;0.4711]	1.81	76	0.1768	0.4415	[0.4182 ;0.4648]
8	0.84	35	0.0039	0.0631	[0.0343 ;0.0919]	1.43	60	0.0604	0.0879	[0.0342 ;0.1416]	2.64	112	0.1050	0.0738	[0.0149 ;0.1327]
7	1.09	46	0.0025	0.0270	[0.0002 ;0.0538]	1.92	81	0.0400	0.0010	[-0.0009;0.0029]	3.50	148	0.0663	0.0110	[-0.0105;0.0325]
6	1.31	55	0.0019	0.0256	[-0.0011;0.0524]	2.39	101	0.0312	0.0000	[0.0000 ;0.0000]	4.36	185	0.0440	0.0000	[0.0000 ;0.0000]
5	1.60	68	0.0013	0.0200	[-0.0033;0.0434]	2.85	121	0.0257	0.0000	[0.0000 ;0.0000]	5.18	220	0.0330	0.0000	[0.0000 ;0.0000]
4	1.77	75	0.0011	0.0201	[-0.0034;0.0437]	3.21	136	0.0218	0.0000	[0.0000 ;0.0000]	6.05	257	0.0161	0.0000	[0.0000 ;0.0000]
3	1.98	84	0.0009	0.0038	[-0.0014;0.0090]	3.64	154	0.0157	0.0000	[0.0000 ;0.0000]	6.89	292	0.0124	0.0000	[0.0000 ;0.0000]
2	2.20	93	0.0005	0.0002	[-0.0002;0.0005]	4.06	172	0.0092	0.0000	[0.0000 ;0.0000]	7.76	329	0.0038	0.0000	[0.0000 ;0.0000]
1	2.45	104	0.0001	0.0000	[0.0000 ;0.0000]	4.50	191	0.0027	0.0000	[0.0000 ;0.0000]	8.63	366	0.0017	0.0000	[0.0000 ;0.0000]
				* cpu: 0h 53m 58s					* cpu: 2h 33m 56s					* cpu:11h 16m 41s	

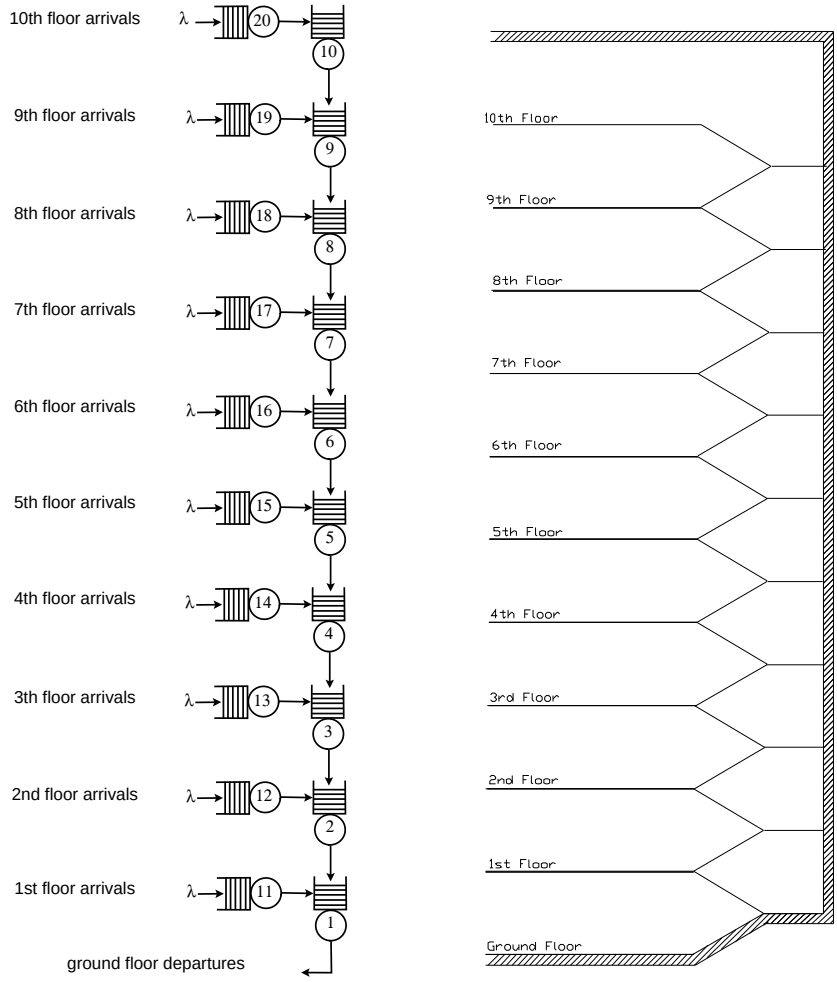


Figure 6: A ten-story building.

one can see that they are roughly constant along the stair-well. The mean waiting (service) time W could be lower, since it is limited from below by the mean waiting time for a lone occupant, $8.5/1.5 \approx 5.7$ seconds.

Finally, Figure 8 shows the evacuation times for the optimized configuration. Although we recognize the importance of low service (evacuation) times in buildings, we remark that W would not be a good objective to be optimized since, without any space constraint, that would lead us to an unbounded optimization problem.

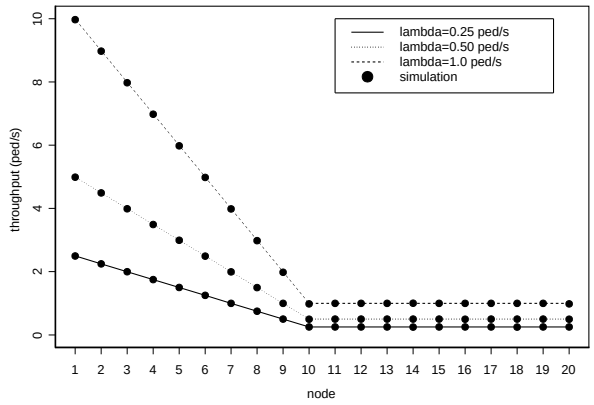
7 Concluding Remarks

A methodology based on $M/G/c/c$ state-dependent queueing systems, suitable for analysis and synthesis of systems subject to congestion effects, in particular, pedestrian networks, was presented. The importance of the model was stressed and a short review of recent results on the area was presented. In detail, the application of the model to pedestrian network planning was discussed. Computational results were provided to demonstrate the effectiveness of the approach.

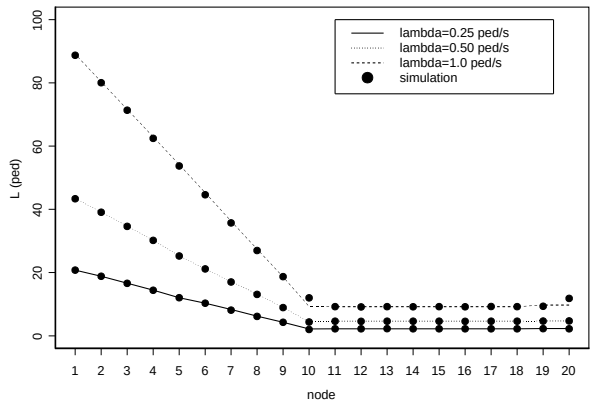
However, many research questions remain open. Further tests must to be done under different topologies and blocking probabilities, as well as under heavier and lighter arrival rates. In the evacuation area, different building heights, different distributions of occupants on the floors, and different floor plan designs could be evaluated. Another possibility is to extend the congestion model to modeling vehicular networks. These are only some possible directions for future research in the area.

Acknowledgments

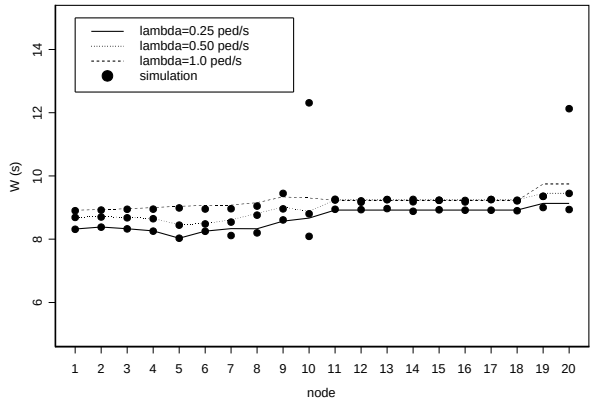
The research of Frederico R. B. Cruz has been partially funded by the CNPq (*Conselho Nacional de Desenvolvimento Científico e Tecnológico*) of the Ministry for Science and Technology of Brazil, grants 301809/96-8 and 201046/94-6, the FAPEMIG (*Fundação de Amparo à Pesquisa do Estado de Minas Gerais*), grants CEX-289/98 and CEX-855/98, and PRPq-UFMG, grant 4081-UFMG/RTR/FUNDO/PRPq/99.



a) throughput



b) expected number of customers



c) expected service time

Figure 7: Performance measures for the optimized ten-story building.

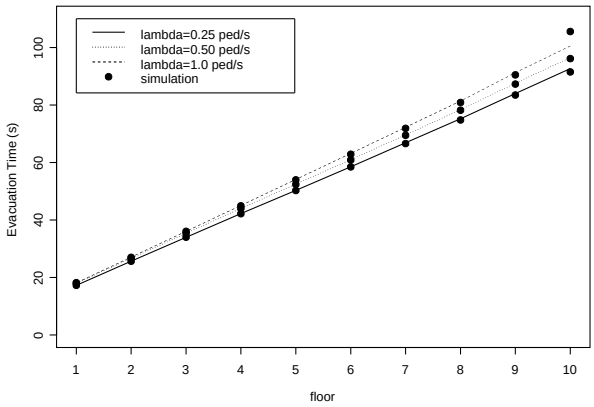


Figure 8: Evacuation times for the optimized ten-story building.

References

[1] Newell GF. Approximation methods for queues with application to the fixed-cycle traffic light. SIAM Reviews 1965; 7(2):223–40.

[2] May AD, Keller HEM. A deterministic queueing model. Transportation Research 1967; 1(2):117–28.

[3] Yuhaski SJ, MacGregor Smith J. Modeling circulation systems in buildings using state dependent models. Queueing Systems 1989; 4:319–38.

[4] Cheah J, MacGregor Smith J. Generalized $M/G/C/C$ state dependent queueing models and pedestrian traffic flows. Queueing Systems 1994; 15:365–86.

[5] Jain R, MacGregor Smith J. Modeling vehicular traffic flow using $M/G/C/C$ state dependent queueing models. Transportation Science 1997; 31(4):324–36.

[6] MacGregor Smith J. State dependent queueing models in emergency evacuation networks. Transportation Research Part B 1991; 25:373–89.

[7] MacGregor Smith J. Application of state-dependent queues to pedestrian/vehicular network design. Operations Research 1994; 42:414–27.

[8] MacGregor Smith J. Topological network design of state-dependent queueing networks. Networks 1996; 28:55–68.

[9] Mitchell DH, MacGregor Smith J. Topological network design of pedestrian networks. Transportation Research Part B 2001; 35:107–35.

[10] Thumsi V, MacGregor Smith J. $M/G/C/C$ state-dependent service rates in material handling systems. Manuscript, 1998.

[11] Tregenza PR. The Design of Interior Circulation. New York: Van Nostrand Reinhold Company, 1976.

[12] Cruz FRB, MacGregor Smith J, Medeiros RO. An $M/G/C/C$ state dependent network simulation

model. Computers & Operations Research 2003 (accepted).

[13] Kerbache L, MacGregor Smith J. The generalized expansion method for open finite queueing networks. European Journal of Operational Research 1987; 32:448–61.

[14] Kerbache L, MacGregor Smith J. Asymptotic behavior of the expansion method for open finite queueing networks. Computers & Operations Research 1988; 15(2):157–169.

[15] MacGregor Smith J, Gershwin SB, and Papadopoulos CT (Eds.). Performance Evaluation and Optimization of Production Lines. Annals of Operations Research, volume 93. Kluwer Academic Publishers, 2000.

[16] Dijkstra EW. A note on two problems in connection with graphs. Numerical Mathematics 1959; 1:269–71.