Controlling Robots of Web Search Engines

J. Talim – Z. Liu – P. Nain
INRIA, B.P. 93, 06902, Sophia Antipolis Cedex, France
{jtalim, liu, nain}@sophia.inria.fr

E. G. Coffman, Jr.
Bell Labs, Lucent Technologies
Murray Hill, NJ 07974, USA
egc@bell-labs.com

May 20, 1999

Abstract

Robots are deployed by a Web search engine for collecting information from different Web servers in order to maintain the currency of its data base of Web pages. In this paper, we investigate the number of robots to be used by a search engine so as to maximize the currency of the data base without putting an unnecessary load on the network. We use a queueing model to represent the system. The arrivals to the queueing system are Web pages brought by the robots; service corresponds to the indexing of these pages. The objective is to find the number of robots, and thus the arrival rate of the queueing system, such that the indexing queue is neither starved nor saturated. For this, we consider a finite-buffer queueing system and define the cost function to be minimized as a weighted sum of the loss rate and the fraction of time that the system is empty. Both static and dynamic policies are considered. In the static setting the number of robots is held fixed; in the dynamic setting robots may be reactivated/desactivated at some particular points in time. Under the assumption that arrivals form a Poisson process, and that service times are independent and identically distributed random variables with an exponential distribution, we obtain a closed-form expression for the optimal number of robots to deploy in the static setting, and we compute the optimal policy in the dynamic setting by using tools from Markov decision process theory.

Keywords: Web search engines; Web robots; Queues; Markov decision process.

1 Introduction

To be modified

The World Wide Web has become a major information publishing and retrieving mechanism on the Internet. The amount of information as well as the number of Web servers has been growing exponentially fast in recent years. In order to help users find useful information on the Web, search engines such as Alta Vista, HotBot, Yahoo, Infoseek, Magellan, Excite and Lycos, etc. are available. These systems consist of four main components: a database that contains web pages (full text or summary), a user interface that deals with queries, an indexing engine that updates the database, and robots that traverse the Web servers and bring Web pages to the indexing engine. Thus, the quality of a search engine depends on many factors, e.g., query response time, completeness, indexing speed, currency, and efficient robot scheduling.

Our interest here focuses on the function served by robots: establishing currency by bringing new pages to be indexed and bringing changed/updated pages for re-indexing. We investigate the problem of choosing the number of robots to meet the conflicting demands of low network traffic and an up-to-date data base. The specific model, illustrated in Figure 1, centers on the indexing engine, which is represented by a finite, single-server queue/buffer, and multiple robots acting as sources of arriving pages. The times between successive page accesses are independent and identically distributed for each robot; the robots themselves are identical and function independently. The indexing (service) times are independent, identically distributed, and independent of the arrival processes.

When a robot arriving with a page for the indexing buffer finds the buffer full, the page being delivered is lost, at least temporarily. In this situation, a potential update or new page has been lost, and network congestion has been created to no benefit. On the other hand, if the buffer is ever empty, and hence the indexing engine is idle, data base updating is at a standstill waiting for the robots to bring more pages. To reduce the probability of the first of these two events, we want to keep the number of robots suitably small, but to reduce the probability of the second, we want to keep the number of robots suitably large. To make the objective concrete, we will formulate a cost function as a weighted sum of the probabilities of an empty buffer and a full buffer. We will then study the problem of finding the number of robots that minimizes this cost function.

There is a large literature on search engines and their components. The search engines themselves may well be their own best source of references; we recommend this entree to the research on any aspect of the subject. In particular, much can be found on the design and control (including distributed control) of robots. However, we have found very little on the modeling and analysis of robot scheduling and the indexing queue. The work in [2] is the only such effort we know about. In [2], the authors propose a natural model of Web-page obsolescence, and study the problem of scheduling a single search engine robot so as to minimize the extent to which the search engine's data base is out-of-date.

Section ?? introduces the probability model, sets notation, and formalizes the optimization problem. Section ?? solves the optimization problem for exponentially distributed service times and presents an explicit computation of the optimal number of robots. The sensitivity of the results to various model parameters is also addressed.

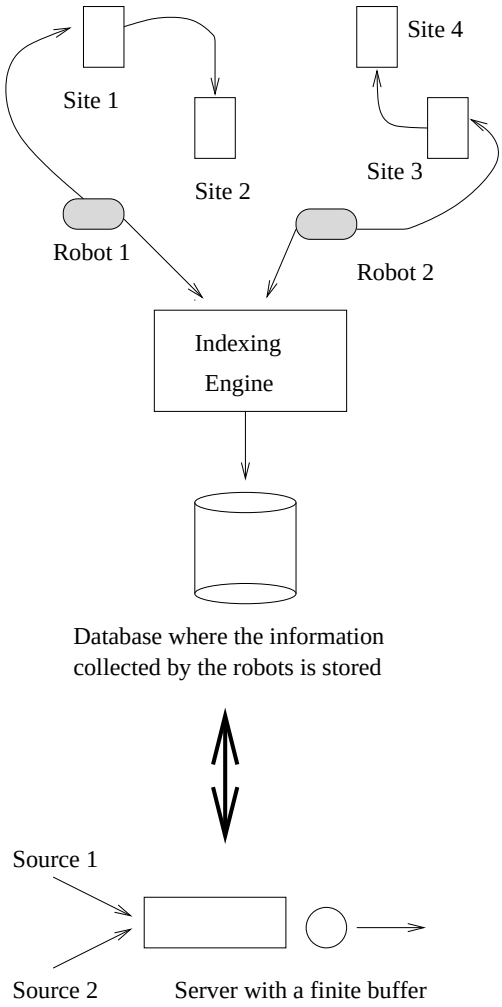


Figure 1: Model of search engine with 2 robots

Section 6 concludes with a brief discussion of ongoing research and open issues.

2 Static approach

The search engine is modeled as a single server finite capacity queue. The system capacity is $K \geq 2$ (including the position in the server). There are $N \geq 1$ robots: each robot brings new pages to the queue according to a Poisson process with rate $\lambda > 0$. These N Poisson processes are assumed to be mutually independent and independent of the indexing (service) times. Hence, new pages are generated according to a Poisson process with intensity λN. An incoming page finding a full queue is lost. Indexing times are assumed to be independent and identically random variables with negative exponential distribution with mean $\sigma > 0$. Define $\mu = 1/\bar{\sigma}$.

The search engine is therefore modeled as the well known M/M/1/K queue. In this notation we define the cost function as the weighted sum of two terms:

- the fraction of time that the system is empty, hereafter referred to as the *starvation probability*;
- the expected number of times when an arriving robot finds a full system per unit time, hereafter referred to as the *loss rate*.

Let X (resp. X^*) be the stationary queue-length at arbitrary epochs (resp. stationary queue-length at arrival epochs) in a M/M/1/K queue with arrival rate λN and service rate μ.

With $\rho := N\lambda/\mu > 0$ and for $\gamma > 0$ the cost function is then defined as

$$C(\rho, \gamma, K) := \gamma \, \mathbf{P}\left(X = 0\right) + \lambda N \, \mathbf{P}\left(X^* = K\right) \tag{1}$$

with $\mathbf{P}\left(X = 0\right)$ and $\lambda N \mathbf{P}\left(X^* = K\right)$ the starvation probability and the loss rate, respectively. Since $\mathbf{P}\left(X^* = i\right) = \mathbf{P}\left(X = i\right)$ for $i = 0, 1, \ldots, K$ from the PASTA property [6], (1) rewrites as

$$C(\rho, \gamma, K) = \gamma \, \mathbf{P}\left(X = 0\right) + \lambda N \mathbf{P}\left(X = K\right). \tag{2}$$

In next sectionn we will resort to queueing theory to compute $C(\rho, \gamma, K)$ and to optimize this quantity as a function of ρ, or equivalently of N, the number of robots.

2.1 Optimizing the number of robots

In the M/M/1/K with trafic intensity ρ the stationary queue-length probabilities at arbitrary epochs are given by [3]:

$$\mathbf{P}\left(X = i\right) = \frac{1 - \rho}{1 - \rho^{K+1}} \rho^i \tag{3}$$

for $i = 0, 1, \ldots, K$. Therefore,

$$C(\rho, \gamma, K) = \frac{(1 - \rho)(\gamma + \mu\rho^{K+1})}{1 - \rho^{K+1}}. \tag{4}$$

In particular, $C(\rho, \gamma, K) = (\gamma + \mu)/(K + 1)$ when $\rho = 1$.

Lemma 1 shows the existence of a unique minimum for $C(\rho, \gamma, K)$ considered as a function of ρ.

Lemma 1 *For any $\gamma > 0$, $K \geq 2$, the mapping $\rho \to C(\rho, \gamma, K)$ has a unique minimum in $[0, \infty)$, to be denoted $\rho(\gamma, K)$. Furthermore, $0 < \rho(\gamma, K) < 1$ if $\gamma < \gamma(K)$, $\rho(\gamma, K) = 1$ if $\gamma = \gamma(K)$ and $\rho(\gamma, K) > 1$ if $\gamma > \gamma(K)$, with $\gamma(K) := \mu(K + 2)/K$.* ◇

Proof. Throughout the proof $f_{x_i}(x_1, \ldots, x_n)$ denotes the partial derivative of $f(x_1, \ldots, x_n)$ w.r.t. x_i, $i = 1, 2, \ldots, n$.

Differentiating (4) with respect to ρ gives

$$C_\rho(\rho, \gamma, K) = \frac{R(\rho, \gamma, K)}{\left(1 - \rho^{K+1}\right)^2} \tag{5}$$

with

$$R(\rho,\gamma,K) := \mu\rho^{2(K+1)} - (\gamma K + \mu(K+2))\,\rho^{K+1} + (K+1)(\mu+\gamma)\,\rho^{K} - \gamma. \tag{6}$$

Note that $\rho = 1$ is a zero of order two of $R(\rho,\gamma,K))$ $(R(1,\gamma,K) = R_\rho(1,\gamma,K) = 0)$ so that the r.h.s. of (5) is well-defined for all $\rho \geq 0$.

Differentiating now $R(\rho,\gamma,K)$ with respect to ρ gives

$$R_\rho(\rho,\gamma,K) = (K+1)\rho^{K-1}\,S(\rho,\gamma,K)$$

with

$$S(\rho,\gamma,K) := 2\mu\rho^{K+2} - (\gamma K + \mu(K+2))\,\rho + K(\mu+\gamma).$$

Let

$$\rho_0(\gamma,K) := \left(\frac{\gamma K + \mu(K+2)}{2\mu(K+2)}\right)^{1/(K+1)}$$

be the unique zero of the mapping $\rho \to S_\rho(\rho,\gamma,K)$ in $[0,\infty)$. We see that $S_\rho(\rho,\gamma,K) < 0$ for $\rho < \rho_0(\gamma,K)$ and $S_\rho(\rho,\gamma,K) > 0$ for $\rho > \rho_0(\gamma,K)$.

Let us determine the sign of $R_\rho(\rho_0(\gamma,K),\gamma,K)$, or equivalently the sign of $S(\rho_0(\gamma,K),\gamma,K)$. To this end, consider the mapping $T : \gamma \to S(\rho_0(\gamma,K),\gamma,K)$. We have

$$T_\gamma(\gamma) = K(1-\rho_0(\gamma,K)) \quad \forall\gamma > 0. \tag{7}$$

Define $\gamma(K) := \mu(K+2)/K$. Since $\rho_0(\gamma,K) < 1$ if $\gamma < \gamma(K)$, $\rho_0(\gamma,K) = 1$ if $\gamma = \gamma(K)$ and $\rho_0(\gamma,K) > 1$ if $\gamma > \gamma(K)$, we conclude from (7) that $T_\gamma(\gamma) > 0$ if $\gamma < \gamma(K)$, $T_\gamma(\gamma) = 0$ if $\gamma = \gamma(K)$, and $T_\gamma(\gamma) < 0$ if $\gamma > \gamma(K)$.

Therefore, the mapping $\gamma \to T(\gamma)$ is strictly increasing in $(0,\gamma(K))$ and strictly decreasing in $(\gamma(K),\infty)$. Since $T(\gamma(K)) = 0$, we deduce that $T(\gamma) \leq 0$ for $\gamma \geq 0$ with $T(\gamma) = 0$ iff. $\gamma = \gamma(K)$, or equivalently from the definition of $T(\gamma)$, that $S(\rho_0(\gamma,K)) \leq 0$ for $\gamma > 0$ with $S(\rho_0(\gamma,K)) = 0$ iff. $\gamma = \gamma(K)$ or equivalently $\rho_0(\gamma,K) = 1$.

The results obtained so far and their consequences are collected in Tables 3-5 in Appendix B, from which the lemma follows.

■

We now return to the original problem, namely the computation of the number N of robots that minimizes the cost function $C(\rho,\gamma,K)$ with $\rho = \lambda N/\mu$. The answer is found in the next result which is a direct corollary of Lemma 1.

Proposition 1 *For any $\gamma > 0$, $K \geq 2$, let $N(\gamma,K)$ be the optimal number of robots to use.*

Then,

$$N(\gamma,K) = \arg\min_n C(n\lambda/\mu,\gamma,K) \tag{8}$$

with $n \in \{\lfloor \rho(\gamma,K)\mu/\lambda \rfloor, \lceil \rho(\gamma,K)\mu/\lambda \rceil\}$*, where for any real number* x*,* $\lfloor x \rfloor$ *(respectively* $\lceil x \rceil$*) denotes the largest (respectively smallest) integer less (respectively greater) than or equal to* x*.* $\diamond$

In the next section we investigate the impact of the parameter γ on the optimal number of robots.

2.2 Impact of γ on the optimal number of robots

Recall that the parameter γ is a positive constant that allows us to stress either the probability of starvation or the loss rate. Part of the impact of γ on $\rho(\gamma,K)$, and therefore on $N(\gamma,K)$, the optimal number of robots, is captured in the following result.

Proposition 2 *For any* $K \geq 2$*, the mapping* $\gamma \to \rho(\gamma,K)$ *is nondecreasing in* $(0,\infty)$*, with* $\lim_{\gamma\to\infty} \rho(\gamma,K) = \infty$*.* $\diamond$

Proof. Pick two constants $0 < \gamma_1 < \gamma_2$ and define

$$\begin{aligned} \Delta(\rho,\gamma_1,\gamma_2,K) &:= C(\rho,\gamma_2,K) - C(\rho,\gamma_1,K) \\ &= \frac{1-\rho}{1-\rho^{K+1}}\,(\gamma_2-\gamma_1). \end{aligned}$$

We assume that $\rho(\gamma_2,K) < \rho(\gamma_1,K)$ and show that this yields a contradiction.

Under the condition $\gamma_1 < \gamma_2$ the mapping $\rho \to \Delta(\rho,\gamma_1,\gamma_2,K)$ is strictly decreasing in $[0,\infty)$. Therefore,

$$\begin{aligned} 0 &< \Delta(\rho(\gamma_2,K),\gamma_1,\gamma_2,K) \\ &\quad -\Delta(\rho(\gamma_1,K),\gamma_1,\gamma_2,K) \\ &= [C(\rho(\gamma_2,K),\gamma_2,K) - C(\rho(\gamma_1,K),\gamma_2,K)] \\ &\quad + [C(\rho(\gamma_1,K),\gamma_1,K) - C(\rho(\gamma_2,K),\gamma_1,K)] \leq 0 \end{aligned} \tag{9}$$

where the last inequality follows from the definition of $\rho(\gamma,K)$. Since the mapping $\rho \to \Delta(\rho,\gamma_1,\gamma_2,K)$ is strictly decreasing on $[0,\infty)$ we deduce from (9) that $\rho(\gamma_2,K) \geq \rho(\gamma_1,K)$ must hold, which contradicts the assumption that $\rho(\gamma_2,K) < \rho(\gamma_1,K)$. Therefore $\rho(\gamma_2,K) \geq \rho(\gamma_1,K)$ and the mapping $\gamma \to \rho(\gamma,K)$ is nondecreasing in $[0,\infty)$.

Assume now that $\liminf_{\gamma\to\infty} \rho(\gamma,K) := L$. There exists a sequence $(\gamma_n)_n$ with $\lim_{n\to\infty} \gamma_n = \infty$ such that $\lim_{n\to\infty} \rho(\gamma_n,K) = L$. Observe first that $L > 1$ since we have shown in Lemma 1 that $\rho(\gamma,K) > 1$ for $\gamma > \mu(K+1)/K$. Substituting ρ for $\rho(\gamma_n,K)$ and γ for γ_n in (6) and using the identity $R(\rho(\gamma_n,K)) = 0$ that holds by definition of $\rho(\gamma,K)$, gives

$$0 = \mu\rho(\gamma_n,K)^{2(K+1)} - (\gamma_n K + \mu(K+2))\,\rho(\gamma_n,K)^{K+1} + (K+1)(\mu+\gamma_n)\,\rho(\gamma_n,K)^K - \gamma_n. \tag{10}$$

Letting $n \to \infty$ in (10) yields

$$\left(KL^{K+1} - (K+1)L^K + 1\right) \lim_{n\to\infty} \gamma_n = \mu L^K \left(L^{K+2} - (K+2))\,L + (K+1)\right). \tag{11}$$

Since $L > 1$ it is easily seen that $KL^{K+1} - (K+1)L^K + 1 > 0$ which implies that the l.h.s. of (14) is infinite. Therefore, (14) cannot holds if $L < \infty$ and $\lim_{\gamma \to \infty} \rho(\gamma, K) = \infty$. This concludes the proof.

■

Proposition 2 has a simple physical interpretation. As the parameter γ increases the probability of starvation becomes the main quantity to minimize. The minimization is done by increasing the arrival rate or, equivalently, by increasing the number of robots. Figure 2 provides two numerical examples, illustrating the monotonicity of the optimal number of robots.

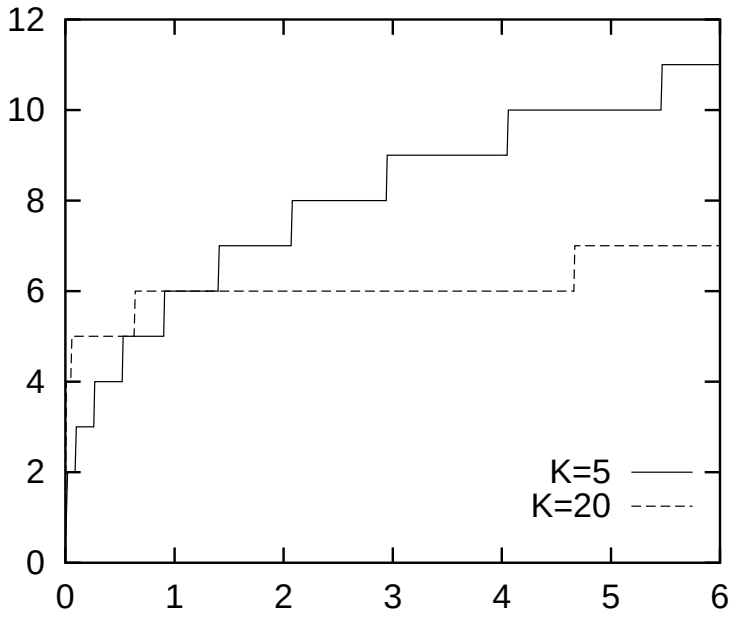


Figure 2: The mapping $\gamma \to N(\gamma, K)$ $(\mu/\lambda = 5.7)$

THIS FIGURE MUST BE MODIFED WITH THE NEW COST

The next section focuses on the impact of the buffer size K on the optimal number of robots, as $K \to \infty$.

2.3 Impact of K on the optimal number of robots

In this section, we examine the behavior of $\rho(\gamma, K)$ as a function of K.

Lemma 2 *The mapping $K \to \rho(\gamma, K)$ is*

- *nondecreasing in $[2, \infty)$ if $0 < \gamma \leq \mu$;*
- *nondecreasing in $[2, K_0)$ and nonincreasing in $[K_0, \infty)$ if if $\gamma > \mu$, where K_0 is given by ....*

Proof. From (6) we find

$$\begin{aligned} R(\rho(\gamma,K),\gamma,K+1) &= \rho(\gamma,K)\, R(\rho(\gamma,K),\gamma,K) \\ &\quad +(1-\rho(\gamma,K))\left(-\mu\rho^{2K+3}(\gamma,K)+(\mu+\gamma)\rho(\gamma,K)^{K+1}-\gamma\right) \\ &= (1-\rho(\gamma,K))\left(-\mu\rho^{2K+3}(\gamma,K)+(\mu+\gamma)\rho(\gamma,K)^{K+1}-\gamma\right) \qquad (12) \end{aligned}$$

since $R(\rho(\gamma, K), \gamma, K) = 0$ by definition of $\rho(\gamma, K)$.

Assume first that $0 < \gamma \leq \mu$. Then, the results is proved if one can show that

$$R(\rho(\gamma, K), \gamma, K + 1) < 0$$

Any ideas?

■

The next lemma examines the limiting behavior of $\rho(\gamma, K)$.

Lemma 3 *For any $\gamma > 0$, $K \geq 2$*

$$\lim_{K\to\infty} \rho(\gamma, K) = 1. \qquad (13)$$

◇

Proof. Assume first that $\liminf_{K\to\infty} \rho(\gamma, K) = L \neq 1$. There exists a sequence $(K_n)_n$ with $\lim_{n\to\infty} K_n = \infty$ such that $\lim_{n\to\infty} \rho(\gamma, K_n) = L$.

Substituting ρ for $\rho(\gamma, K_n)$ and K for K_n in (6) and using the identity $R(\rho(\gamma, K_n)) = 0$ that holds by definition of $\rho(\gamma, K)$, gives

$$0 = \mu\rho(\gamma,K_n)^{2(K_n+1)} - (\gamma K_n + \mu(K_n+2))\,\rho(\gamma,K_n)^{K_n+1} + (K_n+1)(\mu+\gamma)\,\rho(\gamma,K_n)^{K} - \gamma. \quad (14)$$

Letting $n \to \infty$ in (14) we see that the r.h.s. of (14) converges to $-\gamma$ if $L < 1$ and converges to infinity if $L > 1$, thereby showing that necessarily $L = 1$. The same approach shows that $\limsup_{K\to\infty} \rho(\gamma, K) = 1$. Therefore, $\lim_{K\to\infty} \rho(\gamma, K) = 1$, which concludes the proof. ∎

In other words, Lemma 3 shows that the optimal arrival rate converges to the service capacity when the buffer size increases to infinity.

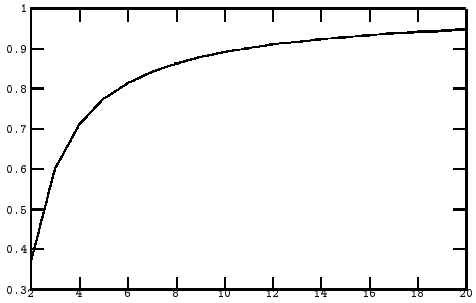
The limiting result (13) can be used to find an approximation for the optimal number of robots to be deployed when K is large. Indeed, the relation

$$\lim_{K\to\infty} N(\gamma, K) = \lim_{K\to\infty} \arg \min_{n\in\{\lfloor \mu/\lambda \rfloor, \lceil \mu/\lambda \rceil\}} C(\lambda\, n/\mu, \gamma, K), \tag{15}$$

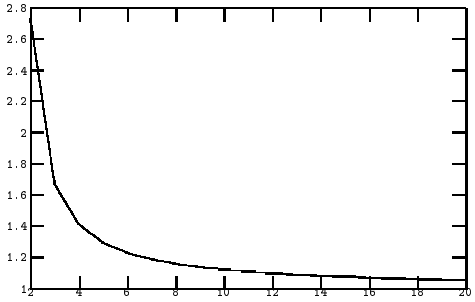
which follows from (8) and (15), suggests the following approximation, for large K:

$$N(\gamma, K) \sim \begin{cases} \lceil \mu/\lambda \rceil & \text{if } C(\rho_+, \gamma, \infty) \le C(\rho_-, \gamma, \infty) \\ \lfloor \mu/\lambda \rfloor & \text{if } C(\rho_+, \gamma, \infty) > C(\rho_-, \gamma, \infty) \end{cases} \tag{16}$$

ALL FIGURES MUST BE MODIFIED WITH THE NEW COST

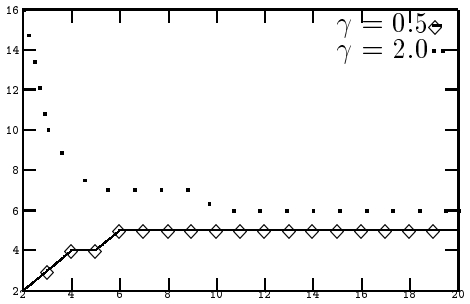


(a) $\gamma = 0.5$

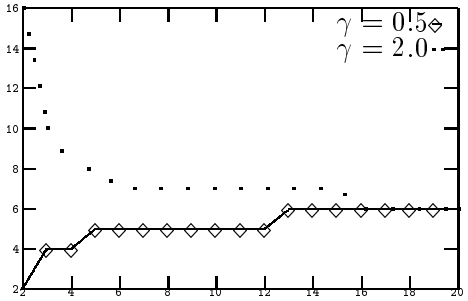


(b) $\gamma = 2$

Figure 3: The mapping $K \to \rho(\gamma, K)$



(a) $\mu/\lambda = 5.7$



(b) $\mu/\lambda = 6$

Figure 4: The mapping $K \to N(\gamma, K)$

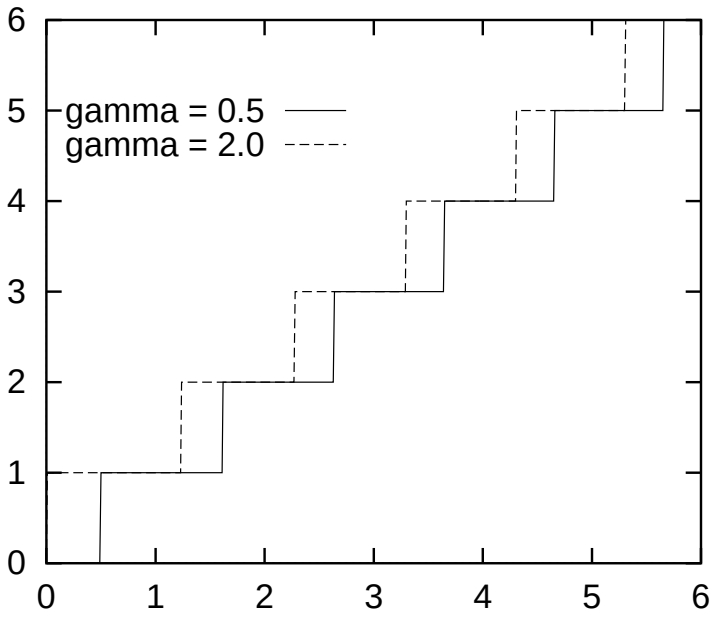


Figure 5: The mapping $\mu/\lambda \to N(\gamma, \infty)$

3 Dynamic approach

In this section we assume that the number of active robots may vary in time according to the backlog in the queue and to the number of robots already active. To address this situation we will cast our model into the Markov Decision Process (MDP) framework [1, 4, 5].

3.1 Notations

The indexing engine is again modeled as a finite-capacity single-server queue. Service times still constitute independent random variables with common negative exponential distribution (with mean $1/\mu$) and the buffer may accomodate at most $K \geq 2$ customers, including the one in service, if any. There are N available robots and each of these robots, when activated, brings pages to the server according to a Poisson process with rate λ. We assume that these N Poisson processes are mutually independent and further independent of the service time process.

The new feature in this section is that the number of active robots may be modified at any arrival and at any departure epoch. When an arrival occurs, the incoming robot is desactivated at once; the controller may then decide to keep it idle or to reactivate it. When a departure occurs the controller may either decide to activate one additional robot, if any available, or to do nothing (i.e. the number of active robots is not modified).

The objective is to find a policy (to be defined) that minimizes a weighted sum of the stationary starvation probability and the loss rate.

We now introduce the MDP setting in which we will solve this optimization problem. Since the time between transitions is variable we will use the uniformization method [1, Sec. 6.7].

At the n-th decision epoch t_n the state of the MDP is represented by the 3-uple $x_n = (q_n, r_n, s_n) \in \{0, 1, \ldots, K\} \times \{0, 1, \ldots, N\} \times \{0, 1, 2\}$, with q_n and r_n the queue-length and the number of active robots just *before* the n-th decision epoch, respectively, and s_n the type (arrival, departure, fictitious – see below) of the n-th decision epoch.

The successive decision epochs $\{t_n, n \geq 1\}$ are the jump times of a Poisson process with intensity $\nu := \lambda N + \mu$, independent of the service time process. In this setting, the n-th decision epoch t_n corresponds to an arrival in the original system with probability $\lambda r_n/\nu$ (in which case $s_n = 1$), to a departure with probability μ/ν provided that $q_n > 0$ ($s_n = 0$) and to a fictitious event with the complementary probability $((N - r_n)\lambda + \mu)/\nu$ ($s_n = 2$).

Let $a_n \in \{0, 1\}$ be the action chosen at time t_n. We assume that $a_n = 1$ if the decision is made to activate one additional robot, if any available, and $a_n = 0$ if the decision is made to keep unchanged the number of active robots. By convention we assume that $a_n = 0$ if the n-th decision epoch corresponds to a fictitious event ($s_n = 2$).

From the above definitions we see that states of the form $(\bullet, 0, 1)$ and $(0, \bullet, 0)$ are not feasible, as an arrival cannot occur if all robots are inactive and a departure cannot occur if the queue is empty,

respectively. Therefore, the state-space for this MDP is

$$\{(q,r,s), 0 \le q \le K, 0 \le r \le N, s = 0,1,2\} - \{(0,r,0), (q,0,1), 0 \le q \le K, 0 \le r \le N\}.$$

However, this set contains one absorbing state, the "fictitious" state $(0,0,2)$. To remove this undesirable state we will only consider policies (see formal definition below) that always choose action $a = 1$ when the system is in state $(1,0,0)$ so that $(0,0,2)$ can never be reached. This is not a severe restriction since a policy that never activates robots when the system is empty is of no interest. In conclusion, the state space for this MDP is

$$\begin{aligned}\mathbf{X} := &\{(q,r,s), 0 \le q \le K, 0 \le r \le N, s = 0,1,2\} \\ &-\{(0,0,2), (0,r,0), (q,0,1), 0 \le q \le K, 0 \le r \le N\}\end{aligned}$$

and the set $\mathbf{A}_x$ of allowed actions when the system is in state $x = (q,r,s) \in \mathbf{X}$ is given by

$$\mathbf{A}_x = \begin{cases} \{0\} & \text{if } s = 2 \\ \{1\} & \text{if } (q,r,s) = (1,0,0) \\ \{0,1\} & \text{otherwise.} \end{cases}$$

To complete the definition of the MDP we need to introduce the one-step cost c and the one-step transition probabilities p. Given that the process is in state $x = (q,r,s)$ and that action a is made, the one-step cost is defined as

$$c(x) = \gamma\, \mathbf{1}(q = 0) + \nu\, \mathbf{1}(q = K, s = 1), \tag{18}$$

independent of a. We will show later on in this section that this choice for the one-step cost will allow us to address, and subsequently to solve, the optimization problem at hand.

For $x \in \mathbf{X}$, the one-step transition probabilities $p_{x,x'}(a)$ are given by

$$p_{x,x'}(a) = \begin{cases} \dfrac{\mu}{\nu}\, \mathbf{1}(q > 1) & \text{if } x' = (q-1, \min\{r+a, N\{, 0) \\ \dfrac{\lambda r}{\nu} & \text{if } x' = (q-1, \min\{r+a, N\}, 1) \\ 1 - \dfrac{\mu\, \mathbf{1}(q > 1) - \lambda r}{\nu} & \text{if } x' = (q-1, \min\{r+a, N\}, 2) \end{cases} \tag{19}$$

if $s = 0$, $a = 0, 1$;

$$p_{x,x'}(a) = \begin{cases} \dfrac{\mu}{\nu} & \text{if } x' = (\min\{q+1, K\}, r+a-1, 0) \\ \dfrac{\lambda(r+a-1)}{\nu} & \text{if } x' = (\min\{q+1, K\}, r+a-1, 1) \\ 1 - \dfrac{\mu + \lambda(r+a-1)}{\nu} & \text{if } x' = (\min\{q+1, N\}, r+a-1, 2) \end{cases} \tag{20}$$

if $s = 1$, $a = 0, 1$;

$$p_{x,x'}(0) = \begin{cases} \dfrac{\mu}{\nu}\,\mathbf{1}(q > 0) & \text{if } x' = (q, r, 0) \\ \dfrac{\lambda r}{\nu} & \text{if } x' = (q, r, 1) \\ 1 - \dfrac{\mu\,\mathbf{1}(q > 0) + \lambda r}{\nu} & \text{if } x' = (q, r, 2) \end{cases} \tag{21}$$

if $s = 2$. All other transition probabilities are equal to 0.

Without loss of generality we will only consider *pure stationary* policies since it is known that nothing can be gained by considering more general policies [4, Ch. 8-9]. Recall that in the MDP setting a policy π is pure stationary if, at any decision epoch, the action chosen is a non-randomized and time-homogeneous mapping of the current state [1, 4, 5]. We define an *admissible* stationary policy as any mapping $\pi : \mathbf{X} \to \{0, 1\}$ such that $\pi(x) \in \mathbf{A}_x$.

For later use introduce $P(\pi) := [p_{x,x'}(\pi(x))]_{(x,x') \in \mathbf{X} \times \mathbf{X}}$, the transition probability matrix under the stationary policy π.

Let $\mathcal{P}$ be the class of all admissible stationary policies. For any policy $\pi \in \mathcal{P}$ introduce the long-run expected average cost per unit time

$$W_\pi(x) = \lim_{n \to \infty} \frac{1}{n}\,\mathbf{E}_\pi \left[\sum_{i=1}^{n} c(x_i) \,|\, x_1 = x \right], \qquad x \in \mathbf{X}. \tag{22}$$

The existence of the limit in (22) is a consequence of the fact that π is stationary and $\mathbf{X}$ is countable [4, Proposition 8.1.1].

We shall say that a policy $\pi^\star \in \mathcal{P}$ is average cost optimal if

$$W_{\pi^\star}(x) = \inf_{\pi \in \mathcal{P}} W_\pi(x) \qquad \forall x \in \mathbf{X}. \tag{23}$$

In order to use results from MDP theory for average cost models we first need to determine to which class (recurrent, unichain, multichain, communicating, etc.) the current MDP belongs to. Consider the following example: $N = 2$ and let π be any stationary policy that selects action 1 in states $(\bullet, r, 1)$ for $r \in \{1, 2\}$ and in state $(1, 0, 0)$, and action 0 otherwise. It is easily seen that this policy induces a MDP with two recurrent classes ($\mathbf{X} \cap \{(\bullet, 1, \bullet)\}$ and $\mathbf{X} \cap \{(\bullet, 2, \bullet)\}$ and a set of transient states ($\mathbf{X} \cap \{\bullet, 0, \bullet\}$).

We therefore conclude from this example that the MDP $\{x_n, n \geq 1\}$ is *multichain* [4, p. 348]. It is shown in Lemma 4 in Appendix B that this MDP is actually multichain *communicating* (see definition in Appendix B).

The next result follows Lemma 4 and Proposition 4 in [1, Sec. 7.1]:

Proposition 3 *There exists a scalar θ and a mapping $h : \mathbf{X} \to \mathbb{R}$ such that, for all $x \in \mathbf{X}$,*

$$\theta + h(x) = c(x) + \min_{a \in \mathbf{A}_x} \sum_{x' \in \mathbf{X}} p_{x,x'}(a)\, h(x') \tag{24}$$

with $\theta = \inf_{\pi \in \mathcal{P}} W_\pi(x)$ for all $x \in \mathbf{X}$, while if $\pi^\star(x)$ attains the minimum in (24) for each $x \in \mathbf{X}$, then the stationary policy $\pi^\star$ is optimal. ◇

The optimal average cost θ and the optimal policy $\pi^\star$ in Proposition 22 can be computed by using the following recursive scheme, known as the relative value iteration algorithm.

Proposition 4 *Let $\hat{x}$ be a fixed state in $\mathbf{X}$ and $0 < \tau < 1$ be a fixed number. For $k \geq 0$, $x \in \mathbf{X}$, define the mappings $(h_k,\ k \geq 0)$ as*

$$h_{k+1}(x) = (1 - \tau) h_k(x) + \tau \left(T(h_k)(x) - T(h_k)(\hat{x}) \right)$$

with

$$T(h_k)(x) := c(x) + \min_{a \in \mathbf{A}_x} \sum_{x' \in \mathbf{X}} p_{x,x'}(a)\, h_k(x'),$$

where $h_0(\hat{x}) = 0$ but otherwise h_0 is arbitrary.

Then, the limit $h(x) = \lim_{k \to \infty} h_k(x)$ exists for each $x \in \mathbf{X}$, $\theta = \tau T(h)(\hat{x})$, and the optimal action $\pi^\star(x)$ in state x is given by $\pi^\star(x) \in \operatorname{argmin}_{a \in \mathbf{A}_x} \sum_{x' \in \mathbf{X}} p_{x,x'}(a)\, h(x')$. ◇

Proof. Since the MDP is communicating (cf. Lemma 4) the proof follows from [4, Sec. 8.5,9.5.3] (see also [1, Prop. 4, p. 313]). ∎

We now return to our initial objective, namely, minimizing a weighted sum of the stationary starvation probability and the loss rate. To see why the solution to this problem is given by the solution to the MDP problem formulated in this section, it suffices to show that the average cost (22) is a weighted sum of the stationary starvation probability and the loss rate. It should be clear, however, that this result cannot hold for policies that induce an average cost (22) that depends on the initial state x as, by definition, the stationary starvation probability and the loss rate are independent of the initial state. We will therefore restrict ourselves to the class $\mathcal{P}_0 \subset \mathcal{P}$ of policies that generate a constant average cost, namely, $\mathcal{P}_0 = \{\pi \in \mathcal{P} : W_\pi(x) = W_\pi(x'), \forall x \in \mathbf{X}\}$.

The set $\mathcal{P}_0$ is non-empty as it is well-known that it contains, among others, all unichain policies [4, Proposition 8.2.1]. Among such policies is the *static* policy π_N that always maintain N robots active, namely, $\pi_N(x) = 1$ for all $x = (\bullet, \bullet, s) \in \mathbf{X}$ with $s = 0, 1$ and $\pi_N(x) = 0$ for all $x = (\bullet, \bullet, 2) \in \mathbf{X}$.

We may also note that reducing the search for an optimal policy to policies in $\mathcal{P}_0$ does not yield any loss of generality as it is also known that there always exits an optimal policy with constant average cost in the case of communicating MDP's [4, Proposition 8.3.2].

Fix $\pi \in \mathcal{P}_0$. Introducing (18) into (22) yields $W_\pi(x) = \gamma S_\pi(x) + L_\pi(x)$ with

$$\begin{aligned} S_\pi(x) &= \lim_{n\to\infty} \frac{1}{n} \mathbf{E}_\pi \left[\sum_{i=1}^{n} \mathbf{1}(q_i = 0) \,|\, x_1 = x \right] \\ L_\pi(x) &= \nu \lim_{n\to\infty} \frac{1}{n} \mathbf{E}_\pi \left[\sum_{i=1}^{n} \mathbf{1}(q_i = K, s_i = 1) \,|\, x_1 = x \right]. \end{aligned}$$

In the following we will drop the argument x in $S_\pi(x)$ and $L_\pi(x)$ since these quantities do not depend on x from the definition of $\mathcal{P}_0$.

Let us now interpret S_π and L_π. S_π is the stationary probability that the system is empty at decision epochs. Since the decision epochs form a Poisson process, we may conclude from the PASTA property [6] that S_π is also equal to the stationary probability that the system is empty at *arbitrary epoch* with is nothing but the stationary starvation probability.

Let us now consider L_π. Recall that $\{t_n, n \geq 1\}$, the successive decision instants, is a Poisson process with intensity ν and assume without loss of generality that $t_1 = 0$ a.s. Define $A(t)$ as the total number of customers that have arrived to the queue up to time t, including customers which have been lost, and let $Q(t)$ be the queue length at time t. We assume that the sample paths of the processes $\{A(t),\, t \geq 0\}$ and $\{Q(t),\, t \geq 0\}$ are right-continuous with left limit. With these definitions and the identity $\mathbf{E}[t_n] = n/\nu$ we may rewrite L_π as

$$L_\pi = \lim_{n\to\infty} \frac{\mathbf{E}_\pi \left[\int_0^{t_n} \mathbf{1}(Q(t-) = K)\, dA(t) \right]}{\mathbf{E}[t_n]}.$$

In other words, we have shown that L_π is the ratio, as n tends to infinity, of the expected number of losses during the first n decision epochs over the expected occurence time of the n-th decision epoch.

The interpretation of L_π as a *loss rate* now follows from the identity

$$L_\pi = \lim_{n\to\infty} \frac{\mathbf{E}_\pi \left[\int_0^{t_n} \mathbf{1}(Q(t-) = K)\, dA(t) \right]}{\mathbf{E}[t_n]} = \lim_{T\to\infty} \frac{1}{T} \mathbf{E}_\pi \left[\int_0^{T} \mathbf{1}(Q(t-) = K)\, dA(t) \right], \quad \forall \pi \in \mathcal{P}_0, \tag{25}$$

upon noticing that the latter quantity represents the mean number of losses per unit time or the loss rate. The latter identity in (25) is a direct consequence of the theory of renewal reward processes [5, Theorem 7.5] and of the definition of the set $\mathcal{P}_0$.

The optimal policy has been computed for different values of the model parameters. Figures 6-8 display the optimal policy for $N = 16$, $K = 5$, $\lambda = 0.1$, $\mu = 1.0$ and for different values of γ ($\gamma < \gamma(K) = 1.4$, $\gamma = \gamma(K)$ and $\gamma > \gamma(K)$). The results were obtained by running the value iteration algorithm given in Proposition 4 with the stopping criterion $\max_{x\in\mathbf{X}} |(h_{k+1}(x) - h_k(x))/h_k(x)| < 10^{-5}$ was used (254, 255 and 256 iterations were needed to compute the optimal

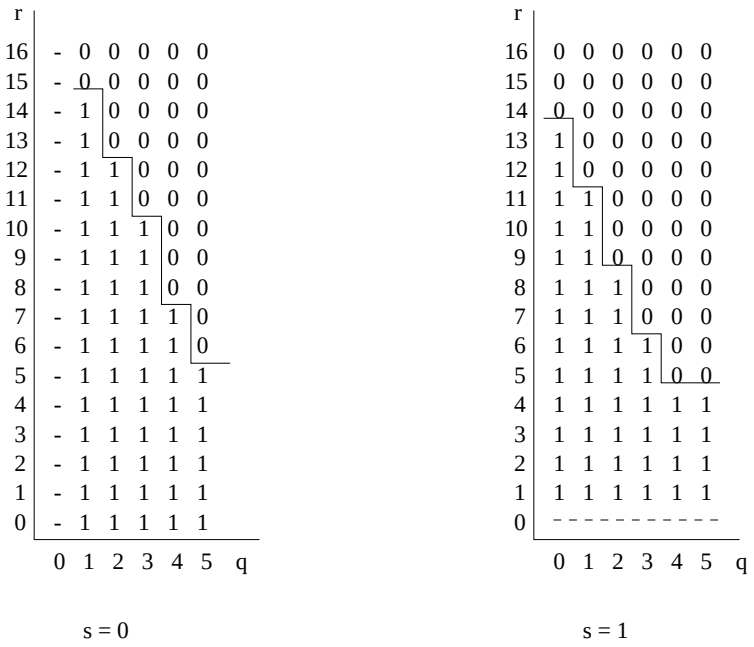


Figure 6: Optimal policy (gamma =1.0, Cost = 0.20907, Iterations number = 254)

policy displayed in Figures 6, 7 and 8, respectively). We see from these figures that the optimal policy is a *monotone switching curve*, namely, there exist two monotone (decreasing here) integer mappings $f_s : \{0, 1, \dots, N\} \to \{0, 1, 2, \dots\}$, $s \in \{0, 1\}$, such that $\pi^\star(x) = \mathbf{1}(f_s(r) \geq q)$ for all $x = (q, r, s) \in \mathbf{X}$ with $s = 0, 1$ (we must also have $f_0(0) \geq 1$ so that $\pi^\star(1, 0, 0) = 1$ as required). We suspect that the optimal policy always exhibits such a structure but we have not able been to prove it.

4 Static versus dynamic policies

In this section we compare the performance of static policies to that obtained under dynamic policies.

The results are reported in Table 1. Throughout the experiments μ was fixed and set to 0.5. For different sets of parameters (λ, K, γ) we first computed the optimal number of robots N_s (given by Proposition 1) to be used in the *static* case and computed the associated average cost C_s (given in (4)).

In a second step, for each set of parameters (λ, K, γ), we set the value of N to N_s and determined, through the relative value iteration algorithm given in Proposition 4 (with $\tau = 0.99999$ – the closer τ is from 1 the faster the algorithm converges), the optimal average cost (23) as well as the minimun (resp. expected) number of robots activated by the optimal policy. We stopped the procedure when the relative error between two consecutive iterates was (uniformly) less than 10^{-5}. The number of iterations is reported in the last column of Table 1.

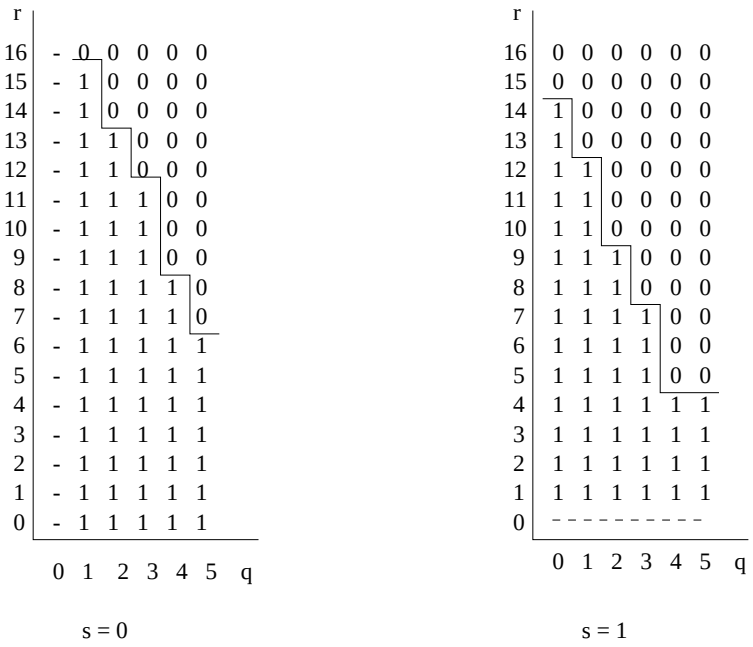


Figure 7: Optimal policy (gamma =1.4, Cost = 0.25924, Iterations number = 255)

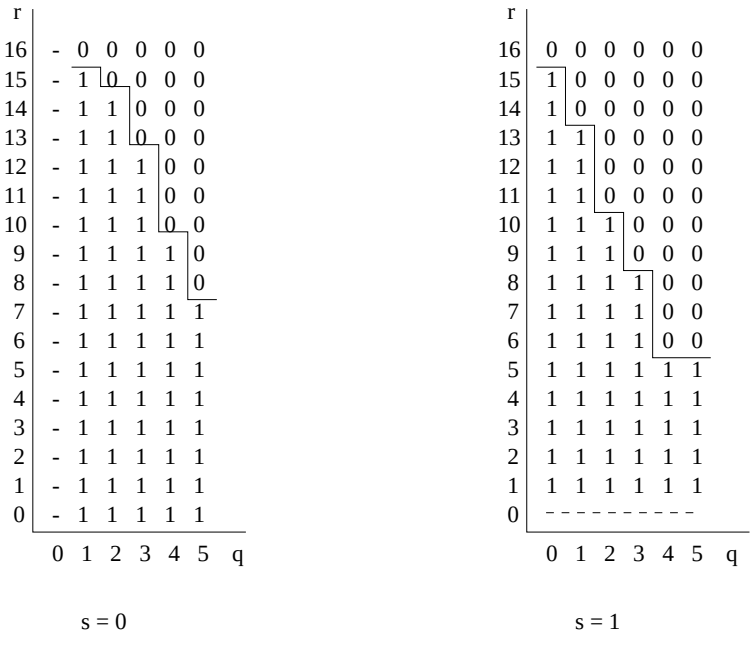


Figure 8: Optimal policy (gamma =2.0, Cost = 0.32211, Iterations number = 256)

			Static Approach		Dynamic Approach			
λ	K	γ	C_s	N_s	C_d	$N_{\min}$	$\bar{N}$	N_{iter}
0.01	5	0.4	0.17541	73	0.16804	57	70.3	1634
-	-	1.4	0.40000	100	0.38336	86	95.1	1911
-	-	2.4	0.53834	114	0.51746	101	108.4	2051
0.01	10	0.4	0.10207	86	0.09062	60	82.6	1794
-	-	1.2	0.20000	100	0.17534	77	94.1	1939
-	-	2.4	0.28347	110	0.24798	88	102.3	2039
0.01	15	0.4	0.07177	91	0.05891	58	87.7	1860
-	-	1.13	0.13333	100	0.10720	70	94.5	1953
-	-	2.4	0.19192	107	0.15342	78	99.7	2024
0.05	5	0.4	0.17578	15	0.15127	7	13.8	338
-	-	1.4	0.40000	20	0.34733	12	17.7	391
-	-	2.4	0.53841	23	0.46583	15	20.2	422
0.05	10	0.4	0.10220	17	0.08308	5	16.2	369
-	-	1.2	0.20000	20	0.14955	8	18.2	402
-	-	2.4	0.28347	22	0.20541	10	19.4	423
0.05	15	0.4	0.07184	18	0.05514	4	17.4	401
-	-	1.13	0.13333	20	0.09117	6	18.7	426
-	-	2.4	0.19372	21	0.13895	8	19.3	438
0.1	5	0.4	0.17600	7	0.15239	1	6.5	167
-	-	1.4	0.40000	10	0.32198	4	8.6	200
-	-	2.4	0.54067	11	0.44989	5	9.3	211
0.1	10	0.4	0.10403	9	0.06838	0	8.4	204
-	-	1.2	0.20000	10	0.13854	2	9.0	218
-	-	2.4	0.28347	11	0.18585	3	9.6	227
0.1	15	0.4	0.07184	9	0.05326	0	8.7	312
-	-	1.13	0.13333	10	0.08538	1	9.3	359
-	-	2.4	0.19458	11	0.09606	1	9.7	376

Table 1: Static vs. dynamic policies (with $\mu = 1.0$ and $\tau = 0.99999$)

5 Final Remarks

In this paper, we have applied the queueing model $M/M/1/K$ to our problem, in a static and a dynamic approach. The comparison between both approaches conform the natural intuition.

Useful generalizations are obtained by replacing exponential distributions by general ones. We have done this in part by solving our optimization problem for a broad class of indexing-time distributions. This analysis will appear in an expanded version of the paper.

Finally, a realistic model may require that robots not all be considered identical. They may operate in different geographical neighborhoods, for example, in which case our problem could become part of a larger problem in which the optimal location of robots is also included.

			Static Approach		Dynamic Approach			
λ	K	γ	C_s	N_s	C_d	$N_{\min}$	$\bar{N}$	$N_{\max}$
0.05	5	0.4	0.17578	15	0.13770	7	15.9	20
-	-	1.4	0.40000	20	0.31712	13	19.8	27
-	-	2.4	0.53841	23	0.43292	16	21.9	32
0.05	10	0.4	0.10220	17	0.04128	5	19.0	29
-	-	1.2	0.20000	20	0.12020	8	19.9	33
-	-	2.4	0.28347	22	0.20541	10	20.6	36
0.05	15	0.4	0.07184	18	0.00969	2	19.8	35
-	-	1.13	0.13333	20	0.01818	4	20.0	38
-	-	2.4	0.19372	21	0.02782	6	20.1	41
0.1	5	0.4	0.17600	7	0.11097	2	8.2	12
-	-	1.4	0.40000	10	0.25924	4	9.8	16
-	-	2.4	0.54067	11	0.35805	6	10.7	18
0.1	10	0.4	0.10403	9	0.01937	0	9.7	18
-	-	1.2	0.20000	10	0.03887	1	9.9	20
-	-	2.4	0.28347	11	0.05894	2	10.1	22
0.1	15	0.4	0.07184	9	0.00188	0	10.0	24
-	-	1.13	0.13333	10	0.00368	0	10.0	25
-	-	2.4	0.19458	11	0.00585	0	10.0	27

Table 2: Static vs. dynamic policies (with $\mu = 1.0$ and $\tau = 0.99999$)

6 Concluding remarks

Simple queueing models (the M/M/1/K and M/G/1/K queues) of search engines have been proposed, analyzed, and optimized in order to find the optimal number of robots to use. The cost function is a weighted sum of the loss probability and the starvation probability.

Extensions of these models to dynamic models where the number of active robots may change over time as a function of the workload in the queue have been proposed in a companion paper [**?**].

Several interesting, open issues remain, including the situation where the robots are not homogeneous and/or are allocated to different parts of the network. For instance, one may wish to determine the optimal number of robots to be allocated to a given area.

A Appendix

A MDP is *communicating* [4, p. 348] if, for every pair of states $(x, x') \in \mathbf{X} \times \mathbf{X}$, there exists a stationary policy π such that x' is accessible from x (i.e., there exists $n \geq 1$ such that $P^n_{x,x'}(\pi) > 0$, where $P^n_{x,x'}(\pi)$ is the (x, x')-entry of the matrix $P^n(\pi)$).

Lemma 4 *The MDP* $(x_n, n \geq 1)$ *is communicating.* $\diamond$

Proof. A word on the notation. In the following $(\bullet, r, s)$ (resp. $(q, r, \bullet)$) will designate any state $\hat{x} = (\hat{q}, \hat{r}, \hat{s}) \in \mathbf{X}$ such that $(\hat{r}, \hat{s}) = (r, s)$ (resp. $(\hat{q}, \hat{r}) = (q, r)$). We will say that $x = (q, r, s)$ is at the same *level* as $x' = (q', r', s')$ if $r = r'$.

There are three steps in the proof depending whether state $x' = (q', r', s')$ to be reached from $x = (q, r, s)$ is at the same level as x (step (1)), at a higher level (step (2)) or at a lower level (step (3)).

(1) $r = r'$. Assume first that $r > 0$. Then select any policy $\pi \in \mathcal{P}$ such that

$$\pi(\bullet, r, 0) = 0 \quad \text{and} \quad \pi(\bullet, r, 1) = 1.$$

Under that policy once the process enters level $r > 0$ it cannot leave that level and moreover all states of that level are recurrent.

Assume now that $r = 0$. If $q \geq q'$ then choose $\pi(i, 0, 0) = 0$ for $i = q, q + 1, \ldots, q' + 1$. If $q < q'$ then x' cannot be reached from x without jumping at level 1 since no arrival may occur at level 0. In this case, select a policy that goes from x to state $(0, 1, \bullet)$ ($\pi(i, 0, 0) = 0$ for $i = q, q + 1, \ldots, 2$ and $\pi(1, 0, 0) = 1$), then go to state $(q' - 1, 1, 1)$ ($\pi(i, 1, 1) = 1$ for $i = 0, 1, \ldots, q' - 2$) and then go to state x' ($\pi(q' - 1, 1, 1) = 0$).

(2) $r < r'$.

Choose any policy $\pi \in \mathcal{P}$ such that

$$\begin{aligned} &\pi(\bullet, t, 0) = 1 \quad \text{and} \quad \pi(\bullet, t, 1) = 1 \quad \text{for } t = r, r + 1 \ldots, r' - 1; \\ &\pi(\bullet, r', 0) = 0 \quad \text{and} \quad \pi(\bullet, r', 1) = 1. \end{aligned}$$

Under that policy the process will successively visit levels $r, r + 1, \ldots, r'$ and will stay forever at that last level where it will visit all states infinitely often.

(3) $r > r'$.

If $r' > 0$ choose any policy $\pi \in \mathcal{P}$ such that

$$\begin{aligned} &\pi(\bullet, t, s) = 0 \quad \text{for } t = r, r - 1 \ldots, r' - 1,\ s = 0, 1; \\ &\pi(\bullet, r', 0) = 0 \quad \text{and} \quad \pi(\bullet, r', 1) = 1. \end{aligned}$$

Under that policy the process will successively visit levels $r, r - 1, \ldots, r'$ and will stay forever at that last level where it will visit all states infinitely often.

Assume now that $r' = 0$. Under the policy $\pi(\bullet, t, \bullet) = 0$ for $t = r, r-1, \ldots, 2$ the process will go from x to level 1. Once level 1 is reached then move to state $(K, 1, 1)$ ($\pi(i, 1, 1) = 1$ for $i = 1, 2, \ldots, K-1$) then go to state $(K, 0, 0)$ ($\pi(K, 1, 1) = 0$) and finally go to state x' ($\pi(i, 0, 0) = 0$ for $i = K, K-1, \ldots, q'+1$). ∎

B Appendix

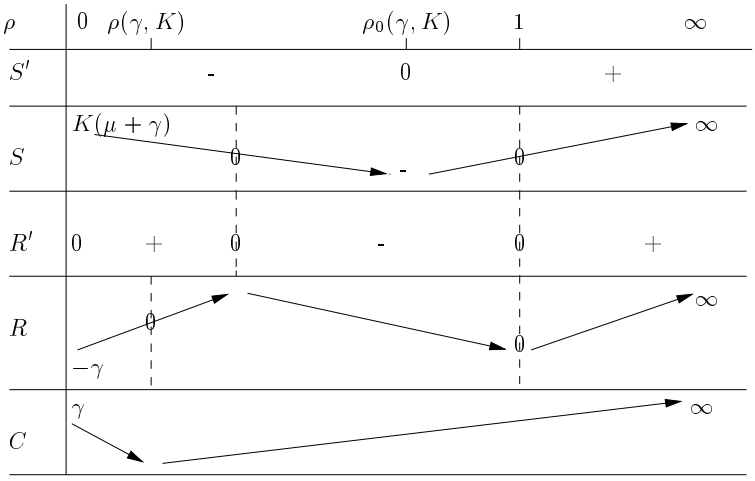


Table 3: Variation of the mapping $\rho \to C(\rho, \gamma, K)$: $\gamma < \mu(K+2)/K$

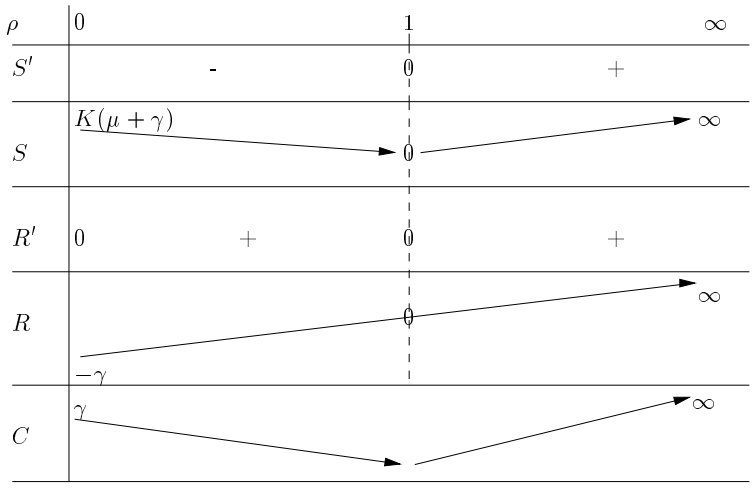


Table 4: Variation of the mapping $\rho \to C(\rho, \gamma, K)$: $\gamma = \mu(K+2)/K$

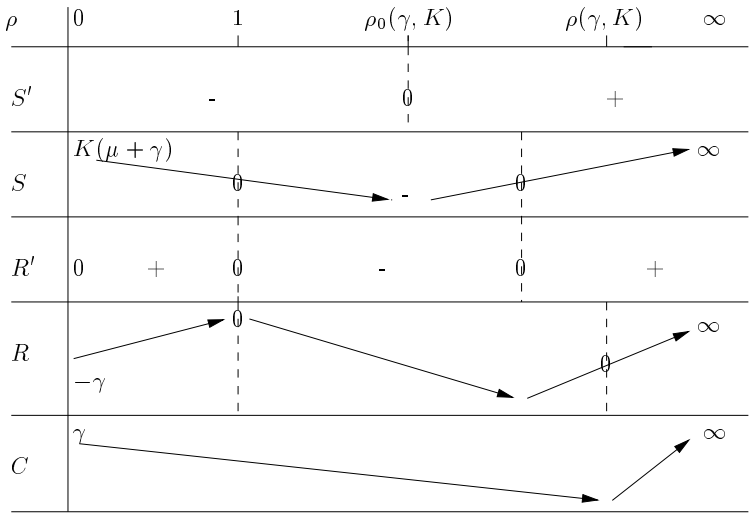


Table 5: Variation of the mapping $\rho \to C(\rho, \gamma, K)$: $\gamma > \mu(K+2)/K$

References

[1] Bertsekas, D. P., *Dynamic Programming. Deterministic and Stochastic Models*, Prentice-Hall, Inc., Englewood Cliffs, 1987.

[2] Coffman Jr., E. G., Liu, Z. and Weber, R. R., "Optimal robot scheduling for web search engines", *J. Scheduling*, **1**, pp. 14-22, 1998.

[3] Kleinrock, L., *Queueing Systems, Vol. I*, Wiley & Sons, New York, 1975.

[4] Puterman, M. L., *Markov Decision Processes*, Wiley, New York, 1994.

[5] Ross, S. M., *Introduction to Stochastic Dynamic Programming*, Academic Press, New York, 1983.

[6] Wolff, R. L., "Poisson Arrivals See Time Averages," *Operat. Res.*, vol. 30, pp. 223-231, 1982.