Understanding the effect of selfish behaviour in a series of two queues.

Jason Young, Vincent Knight

January 5, 2014

Abstract

Healthcare systems and other queueing systems are often a series of complex queueing networks. The best designed network can exhibit sub-optimal behaviour if the users of the system act rationally. This is a basic idea underlying Game Theory.

There is a vast quantity of literature understanding this inefficiency which can be measured using a game theoretical measure entitled: Price of Anarchy (PoA). However, most of the corresponding work concentrates on queues in isolation and/or in parallel.

In this paper two queues are considered in series. The decisions of customers at the entry to both queues thus has an impact on the efficiency of the entire system.

1 Introduction

What damage do the selfish choices of some afflict onto the welfare of all? The work presented in this paper answers this question in the context of queues in series.

There is a substantial quantity of literature on the subject of equilibrium behaviour of a queueing system where congestion is a factor influencing behaviour [1, 2, 3, 10, 11, 19]. This can be traced back to a set of short communications between Leeman [8, 9] and Saaty [16].

This paper builds on this literature by considering a particular network of queues. For most public services, congestion is a negative aspect of service quality. When leaving one queue; customers (which we will refer to as players) often join a second queue and this is what is referred to as a hierarchical queueing system (as shown in Figure 1).

The following is a general conclusion of a majority of the literature:

Selfish users make busier systems.

Naor cites the initial exchange between Leeman and Saaty as the motivation for his early work on queuing systems. In [11], thresholds are obtained that give the number of players

$\tilde{n}$ at which selfish users should "balk" a particular queue. Furthermore, a threshold n^* is obtained that ensures average optimal utility.

In [1], similar work is carried out for multiple $M/M/1$ queues in parallel where players chose which queue to join. Importantly, this work differs from [11] in that it considers "unobservable" queues.

In [6], results are given regarding the comparison of observable and unobservable approaches.

A good review of the mentioned works and further pieces of literature is found in [4].

The effect of selfish behaviour when compared to optimal behaviour in game theoretical contexts is measured as the "Price of Anarchy" (PoA) [7, 15]. This is defined as the ratio of the selfish ($\tilde{C}$) and optimal (C^*) costs:

$$\text{PoA} = \frac{\tilde{C}}{C^*}$$

In [5] the PoA is calculated for a healthcare system. Patient choices between hospitals are considered as $M/M/c$ queues in parallel.

The contribution of this paper is to consider two stations (to avoid confusion we will refer to a queue and service station as a station) in series allowing for two consecutive decisions: whether or not to join the station. Players who join the first station upon completion of service will potentially drop out from the entire system. A potential application of this is a healthcare system where patients must choose to see their general practitioner before obtaining care from a specialist/emergency centre. If unable to get an appointment within a short enough period of time, patients may choose to go directly to their specialist/emergency centre [12].

In Section 2, the model will be described. A novel aspect of this work is that the cost of a given policy is evaluated through the use of purpose built Agent Based Simulation (ABS). In Section 3 various heuristic approaches to obtaining optimal policies are considered. These range from random search algorithms to a more sophisticated approach that depends on the analytical formula for an $M/M/c$ queue [18]. Before concluding in Section 5 a variety of numerical experiments will be shown in Section 4 demonstrating the effect of selfish decision making in queues in series.

2 Model

Let $k \in \{1, 2\}$ be the index for each station in the system. The players arrive into the system at a rate Λ. Upon arriving at the station k players are aware of:

- The service rate μ_k;

- The number of servers c_k;
- The number of players already at station k: m_k;
- The skip cost of station k: β_k.

Upon observing a station, players must select one of two potential strategies: 'join' or 'skip'. The strategy picked at station k is denoted by $d_k \in \{0, 1\}$:

$$d_k = \begin{cases} 1, & \text{if player joins queue } k \\ 0, & \text{if player skips queue } k \end{cases} \tag{1}$$

The cost associated with joining station k is denoted by J_k and corresponds to the actual time spent in station k. The cost associated with skipping station k is denoted by β_k. This cost is measured in units of time but is a constant value to be interpreted as the "value of service" at station k. The cumulative cost of going through the entire system is given by $C = C(d) = C_1(d) + C_2(d)$ where:

$$C_k = \begin{cases} \beta_k & \text{if } d_k = 0 \\ J_k & \text{if } d_k = 1 \end{cases} \tag{2}$$

The expected value of J_k when there are m_k players present upon arrival in system k is given by:

$$E[J_k] = \frac{m_k + 1}{\mu_k c_k} \tag{3}$$

A final parameter of our model is the dropout probability p. This is the probability with which a player who joins the first station will exit the entire system (implying $C_2 = 0$).

All of this is summarised in Figure 1.

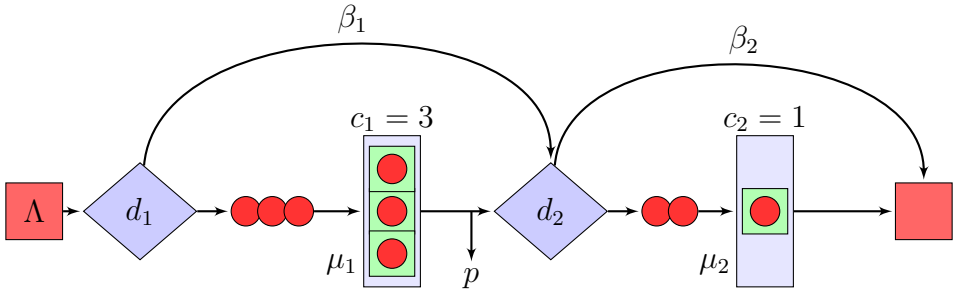


Figure 1: Diagrammatic representation of the model.

If a general population of players is considered a state dependent policy $\tau = (n_1, n_2)$ can be defined that all players follow. In general this policy will be of the form:

$$d_k = \begin{cases} 1, & \text{if } m_k \leq n_k \\ 0, & \text{otherwise} \end{cases}$$

Where m_k is the number of players present at station k upon having to make a decision d_k. In other words a policy τ is a set of two threshold policies, one for each station, simply denoting a threshold at which players skip. As $d = d(\tau)$ we have $C = C(\tau)$.

2.1 Simulation engine

To evaluate the $C(\tau)$ for given τ a bespoke simulation model has been developed (written in Python `http://www.python.org/`). The model itself is a combination of Discrete Event and Agent Based Simulation (taking advantage of the Object Orientated framework available in Python) [14]. The flow of the simulation model is shown in Figure 2.

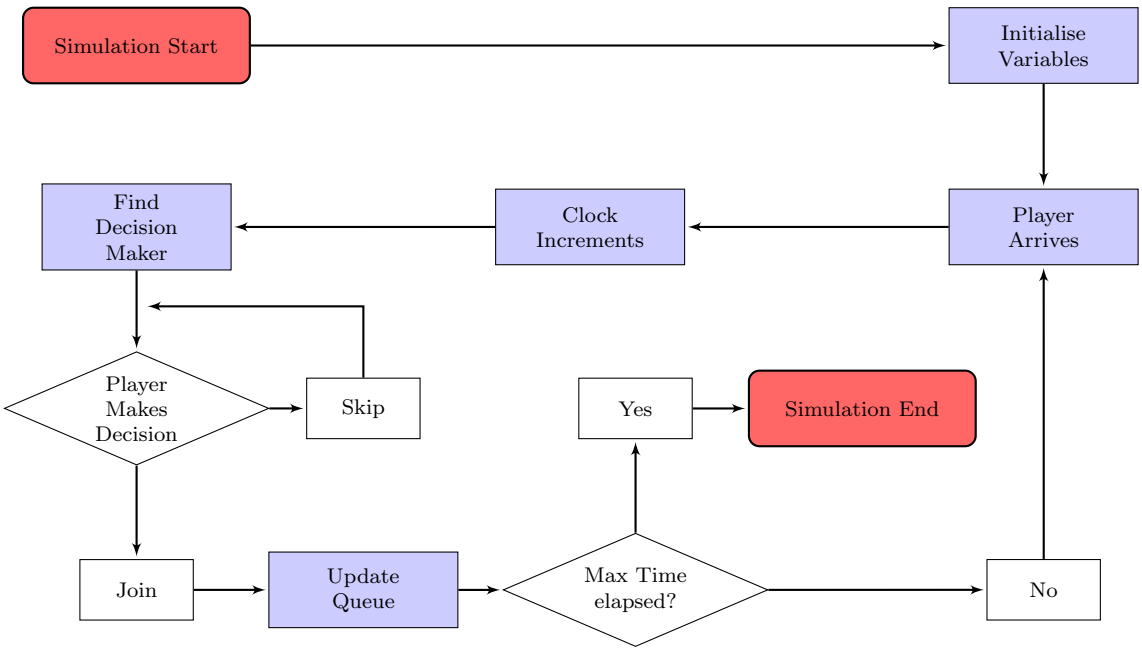
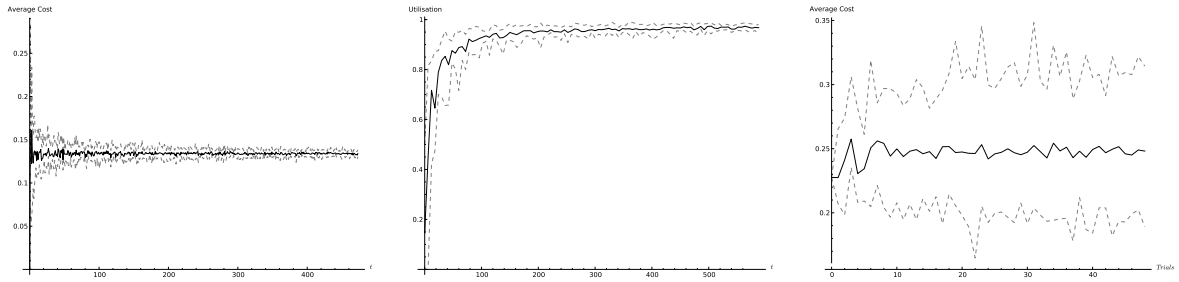


Figure 2: Flow chart describing the simulation model

The input to the model (apart from the system parameters described in Section 2) is a policy τ and the required output is $C(\tau)$. An investigation of run time, warm up time, and number of trials was carried out to ensure a reliable value of $C(\tau)$ is output [**?**]. Graphical methods were used for system parameters discussed in Section 4 and are shown in Figure 3.



a: Run Time b: Warm up time c: Number of trials

Figure 3: Investigating run time, warm up time and number of trials required to obtain reliable values of $C(\tau)$

A run time of 250 time units was shown to reduce the variation in $C(\tau)$ (Figure 3a) whilst a warm up period of 50 time units reduced gave a stable utilisation of servers (Figure 3b). Finally experiments of 16 trials each were chosen despite the lack of variation shown in Figure 3c. This relatively large number of trials did not prove to be computationally expensive as they were all run in parallel using the Advanced Research Computing facilities at Cardiff University `http://www.cardiff.ac.uk/arcca/` (super computer).

2.2 Selfish policy: $\tilde{\tau}$

The selfish policy is the policy under which individual players reduce their expected cost.

To obtain the selfish policy an assumption is made: players are risk averse. It is assumed that players ignore the dropout probability p at station 1 so that we have (recall equation (1)):

$$\tilde{d}_k = \begin{cases} 1, & \text{if } E[J_k] \leq \beta_k \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

This in turn gives (recalling (3)):

$$\tilde{\tau} = (\lfloor \beta_1 \mu_1 c_1 - 1 \rfloor, \lfloor \beta_2 \mu_2 c_2 - 1 \rfloor) \tag{5}$$

2.3 Optimal policy: τ^*

As discussed before, in general selfish users make busier system so that it seems reasonable to assume $n_k^* \leq \tilde{n}_k^*$ for $k \in \{1, 2\}$. A general result has in fact been shown confirming this

for queues in parallel [17]. However, this is not necessarily evident for queues in series with $p > 0$. Indeed, to reduce the mean value C it is reasonable to assume that a gateway effect exists, where players are encouraged to join the first queue in order to exit the queue thanks to the dropout probability. This is in fact readily understood in practice by policy makers: [12].

Let Ω be the solution space to the optimisation problem of reducing the expected value of C^*:

$$\Omega = \left\{\tau = (n_1, n_2) \in \mathbb{Z}^2 \mid 0 \leq n_1 \leq 2\tilde{n}_1,\ 0 \leq n_2 \leq \tilde{n}_2\right\} \tag{6}$$

The optimal cost is thus defined as:

$$E[C^*] = \min_{\tau \in \Omega} E[C(\tau)] \tag{7}$$

Note that the restriction of the optimisation problem to only consider stationary policies is potentially an incorrect one. Indeed, a non homogeneous policy could in fact be optimal however considering dynamic programming techniques on the underlying Markov Decision process it can be seen that a threshold policy will in fact be optimal [13].

As discussed, the simulation model is used as an objective function for a given τ. In the next section, various heuristic approaches to obtaining τ^* (recall (7): this corresponds to the optimal policy) will be discussed.

3 Heuristic Optimal Policies

Using the simulation model as an objective function a variety of search based heuristics have been tested []. All of these use $\tilde{\tau}$ as an initial solution and make minor perturbations accepting improvements within a specific region:

- Fixed search range: randomly evaluate a fixed range around the initial policy;
- Variable search range: increase the range of the search range;
- Random descent: randomly search the entire solution space;
- Iterative random descent: JASON CHECK THESE AND ALSO INCLUDE A SENTENCE FOR THIS ONE. I DONT LIKE THE NAMES. ARE THEY THE CORRECT NAMES?

3.1 Heuristic 1

A comparison of the performance of the various heuristics above is shown in Figure 4.

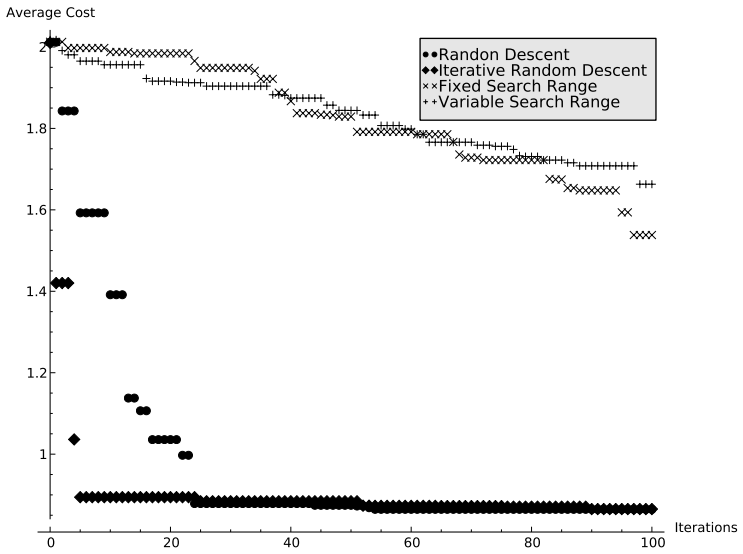


Figure 4: Comparing local search heuristics

As shown, the most efficient of the chosen heuristics is the 'Random Descent'. This algorithm is nonetheless computationally expensive. Some scenarios of Section 4 taking various hours to run (despite the use of [**?**]). The aim of the work presented here is to gain a global understanding of PoA behaviour and as such a large number of optimisations are needed. The next sections will demonstrate further heuristics.

3.2 Heuristic 2

This heuristic is based on the assumption that the arrival rate λ_2 at the second station is in fact Markovian. For $p = 0$ this is in fact not an assumption **??** however for $p > 0$ the arrival rates at the second station no longer exhibit Markovian behaviour. Nevertheless as shown in Figure 5 this heuristic performs very favourably and in fact the non Markovian behaviour seems to not affect the performance of our heuristic.

The arrival rate at the second station is given by:

$$\lambda_2 = \Lambda \times (1 - p \times (1 - \pi^{(1)}_{n_1})) \tag{8}$$

Where $\tau = (n_1, n_2)$ and $\pi^{(1)}_i$ is the probability that the first station is in state $0 \leq i \leq n_1$. Note that the first station corresponds to an $M/M/c/n_1$ queue and so we have:

$$\pi_i^{(1)} = \frac{1}{i!}\left(\frac{\Lambda}{\mu_1}\right)^i \pi_0 \qquad 0 \leq i \leq c_i \tag{9}$$

$$\pi_i^{(1)} = \frac{1}{c_i^{i-c_i} c_i!}\left(\frac{\Lambda}{\mu_1}\right)^i \pi_0 \qquad c_i < i \leq K \tag{10}$$

where

$$\pi_0^{(1)} = \left[\sum_{n=0}^{c_i-1} \frac{1}{n!}\left(\frac{\Lambda}{\mu_1}\right)^n + \sum_{n=c_i}^{K} \frac{1}{c_i^{n-c_i} c_i!}\left(\frac{\Lambda}{\mu_1}\right)^n\right]^{-1} \tag{11}$$

Using the Markovian assumption as to the distribution of λ_2 a similar formula for $\pi_i^{(2)}$ can be obtained, replacing the relevant parameters.

Using this and recalling (**??**):

$$E[C_i] \approx \pi_{n_i}^{(i)} \beta_i + \sum_{j=0}^{c_i} \frac{\pi_j^{(i)}}{\mu_i} + \sum_{j=c_i}^{n_i-1} \frac{\pi_j^{(i)}(j+1)}{\mu_i c_i} \tag{12}$$

and so:

$$E[C] \approx E[C_1] + \frac{\lambda_2 E[C_2]}{\Lambda} \tag{13}$$

Note that the approximation of (12) is in fact an equality for $i = 1$. Using (13) the cost of any policy τ can be approximated immediately and so τ^* can be heuristically obtained by an exhaustive search of the solution space. To obtain an actual mean cost of τ^* the simulation model needs to only be run once. A comparison of heuristics 1 and 2 is shown in Figure 5, it is evident that the assumption made with regards to the Markovian nature of λ_2 do not seem to have a large effect. The fact that results for $\Lambda > 130$ are not shown for heuristic 1 are due to the fact that they were not obtained as they required more than 80 hours of computations on [**?**].

Before considering various results in Section 4, one final heuristic is proposed that only applies if $c_1 = c_2 = 1$.

3.3 Heuristic 3

In [**?**] thresholds n^* are obtained for single server queues that optimise the expected cost. n^* is shown to be the solution to the following inequality:

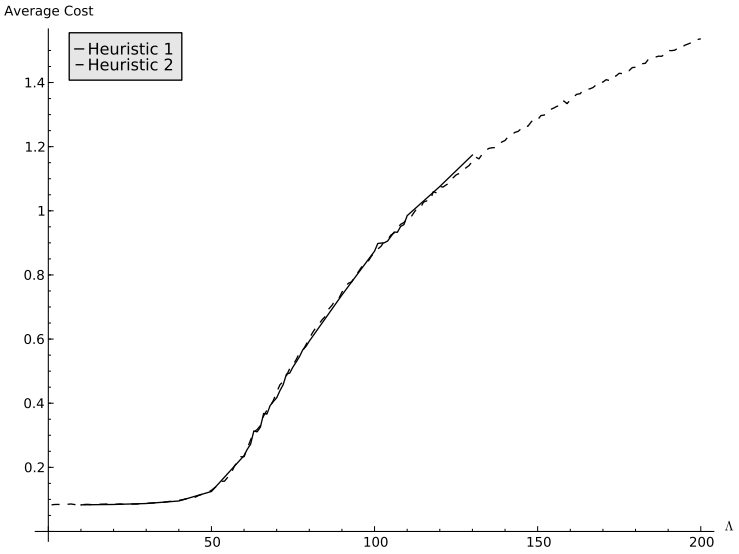


Figure 5: Comparing Heuristic 1 and 2

$$\leq n^* \leq \tag{14}$$

Using (14) another potential policy is $\tau = (n_1^*, n_2^*)$. Figure 6 shows a comparison of these three heuristic approaches for a series of single server queues.

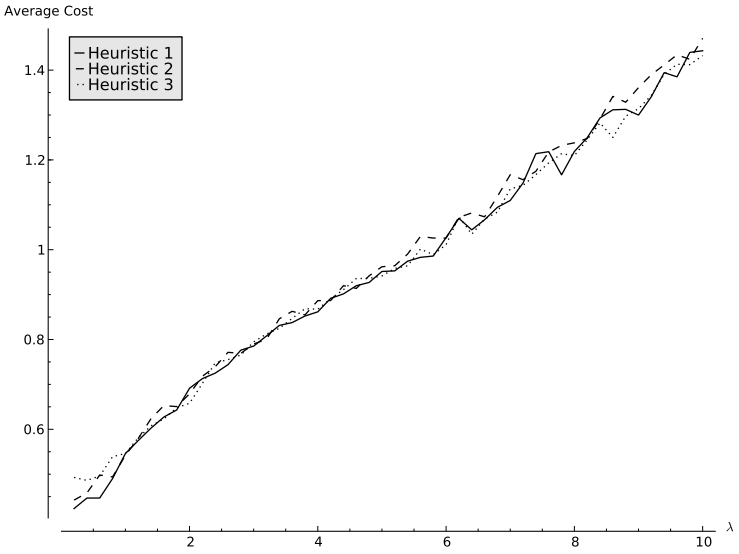


Figure 6: Comparing all heuristics

All the heuristics seem to behave adequately (this has been confirmed through a large quantity of experimentation). For the purposes of this paper heuristic 2 will be used from now

on given the very low computational cost.

4 Results

As discussed in Section an immediate application of the models developed is healthcare. Using data from [] we consider two stations in series:

- A General practitioner with 5 servers each able to see 12.5 patients a day. Furthermore patients consider a wait of 2 days as acceptable to see their GP:

$$c_1 = 5, \mu_1 = 12.5, \beta_1 = 2$$

- After seeing a GP or after deciding the wait to see a GP is too long [**?**] patients may choose to proceed to the Emergency Department (ED). It is assumed that the ED has a capacity of 5 servers each able to 2 patients an hour. An acceptable wait is now assumed to be 8 hours which is in line with national targets [].

$$c_2 = 5, \mu_1 = 48, \beta_1 = 1/3$$

It is assumed that 80% of patients who see their GP do not need to attend the ED:

$$p = .8$$

For these sets of parameters, using heuristic 2 a Price of Anarchy can be calculated. For example for $\Lambda =?$ the PoA is 5 which implies that the overall patient utility is 5 times as bad when patients act selfishly.

Figure 8 shows the effect of Λ on the PoA for different values of service (β). The profile of this curve is very similar to profiles of curves seen in [**?**]. For low demand the PoA is 1: this implies that the capacity of the system is sufficient to deal with the actions of selfish users. As the demand increases, the first station starts to get congested. Optimal behaviour will at this point encourage players to skip the first queue (before the expected wait reaches β_1. Selfish users however will wait until the system becomes very busy before they are willing to start skipping. This occurs when the demand increases further and we see an initial drop of the PoA. Recall that this simply implies that selfish users are not causing the system to behave in a different way to the optimal behaviour. This is something that has been noted in the literature: [**?**]: in congested systems selfish users don't make a difference. This is particularly relevant to healthcare where most healthcare systems are already severely over congested.

Figure **??** shows the PoA for both systems taken in isolation. BLABLABLA...

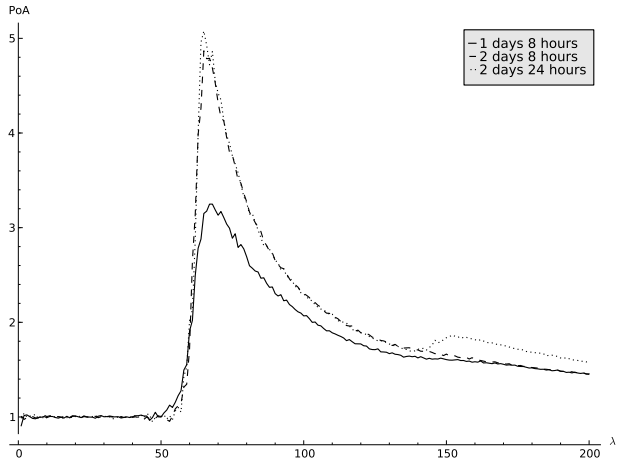


Figure 7: PoA for varying lambda

[Grab picture of PoA for both single stations]

Figure 8: PoA for varying lambda

The effect of having the two queues in series is... Note that this is shown in Figure 9 where GET PARAMETERS HERE.

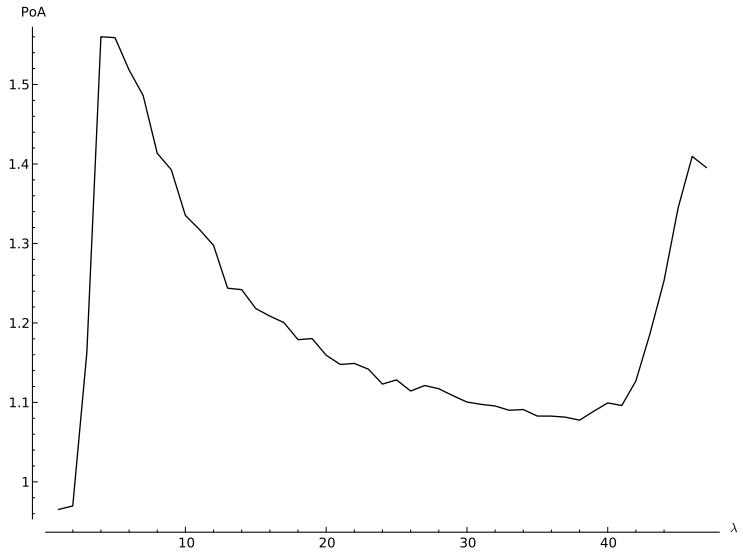


Figure 9: Another set of scenarios

5 Conclusions and Further work

In this work ...

Note that further to understanding the effect of queues in series this work can offer some indication as to best practice by GPs. In Figure 10 the PoA for the systems considered is shown for varying values of p for a value of Λ = JASON WHAT IS THIS VALUE? The effect of selfish behaviour is reduced for a low p value. This in turn would imply that GPs would have to refer 70% of patients to reduce the effect of selfish behaviour. Importantly this would imply a much busier ED and an overall inefficient system however it is worth noting that measures could be put in place to counteract selfish actions although these would be quite drastic.

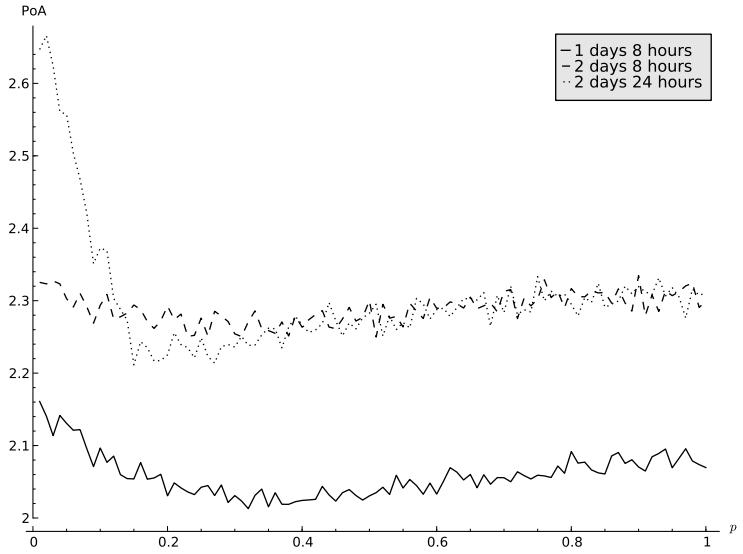


Figure 10: PoA for varying exit prob

Further work:

- MDP
- Single decision
- Mixture of player types

References

[1] Colin E Bell and Shaler Stidham. Individual versus social optimization in the allocation of customers to alternative servers. Management Science, 29(7):831–839, 1983.

[2] Niels Chr. Individual and social optimization in a multiserver queue with a general cost-benefit structure. Econometrica: Journal of the Econometric Society, pages 515–528, 1972.

[3] Noel M Edelson. Congestion tolls under monopoly. The American Economic Review, 61(5):873–882, 1971.

[4] Refael J Hassin and Moshe Haviv. To Queue or Not to Queue: Equilibrium Behaviour in Queueing Systems, volume 59. Kluwer Academic Pub, 2003.

[5] Vincent Knight and Paul Harper. Selfish routing in public services. European Journal of Operational Research, 2013.

[6] Vincent Knight, Janet Williams, and Rob Shone. Comparisons between observable and unobservable m/m/1 queues with respect to optimal customer behavior. European Journal of Operational Research, 2012.

[7] Elias Koutsoupias and Christos Papadimitriou. Worst-case equilibria. Computer science review, 3(2):65–69, 2009.

[8] Wayne A Leeman. The reduction of queues through the use of price. Operations Research, 12(5):783–785, 1964.

[9] Wayne A Leeman. Letter to the editorcomments on preceding note. Operations Research, 13(4):680–681, 1965.

[10] Israel Luski. On partial equilibrium in a queuing system with two servers. The Review of Economic Studies, 43(3):519–525, 1976.

[11] Pinhas Naor. The regulation of queue size by levying tolls. Econometrica: journal of the Econometric Society, pages 15–24, 1969.

[12] Mail Online. Millions of A&E patients 'should just see their GP' to avoid crippling emergency services. `http://www.dailymail.co.uk/news/article-2414539/Millions-A-amp-E-patients-just-GP-avoid-crippling-emergency-services.html`. Accessed: 2014-01-04.

[13] Martin L Puterman. Markov decision processes: discrete stochastic dynamic programming, volume 414. Wiley. com, 2009.

[14] Stewart Robinson. Simulation: the practice of model development and use. Wiley. com, 2004.

[15] Tim Roughgarden and Éva Tardos. How bad is selfish routing? Journal of the ACM (JACM), 49(2):236–259, 2002.

[16] Thomas L Saaty and Wayne A Leeman. The burdens of queuing charges-comments on a letter by leeman. Operations Research, 13(4):679–681, 1965.

[17] Rob Shone, Vincent Knight, Paul Harper, Janet Williams, and John Minty. Containment of socially optimal policies in multiple-facility markovian queueing systems. Under Review, 2014.

[18] William J Stewart. Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling. Princeton University Press, 2009.

[19] Uri Yechiali. Customers' optimal joining rules for the gi/m/s queue. Management Science, 18(7):434–443, 1972.