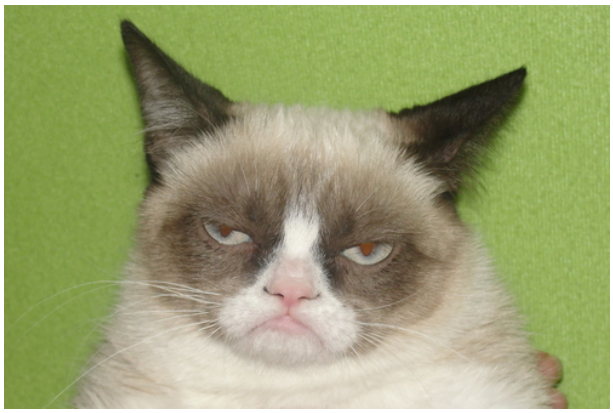
When the cat goes: the paper is ready.



Understanding the effect of selfish behaviour in hierarchical queues: an approach based on simulation.

Jason Young, Vincent Knight

December 17, 2013

Abstract

In this paper, we consider a system consisting of $M/M/C$ queues arranged in series. When customers (referred to as players) arrive at a given queue as they move through the system, they make a decision, to either join the queue or skip it at a cost. The system is considered under two types of player behaviour, one where players act selfishly and one where there is a policy dictating the decisions of the players. Players acting selfishly will attempt to reduce their own cost with no thought to the cost of other players in the system. Alternatively a optimal social policy is considered that attempts to reduce the average cost per player. We use the Price of Anarchy as a measure of the inefficiency due to choice to observe the impact that individual behavior has on the system. An Agent Based Simulation is developed that is used in conjunction with various heuristic approaches to obtain the socially optimal and selfish behavior of players. Due to the high computational cost of this model an analytical approximation is obtained and used to identify the effect of the total arrival rate on the system as well as other system parameters. In particular an optimal dropout probability rate is obtained for various scenarios which could have applications such as the optimal referral rate of general practitioners to hierarchical medical services.

1 Introduction

Consider a system where players within this system will act for their own good, attempting to reduce their cost or equivalently increase their utility. This work will quantify the impact that this choice has in hierarchical queueing systems. In this paper and without loss of generality, cost is considered to be a unit-less combination of time spent in the system and any utility lost by balking from a given queue. It is well understood that individual behaviour has a negative effect on the efficiency of queueing systems.

To illustrate this effect, consider the classic game theory problem, the Prisoners Dilemma. The game is that two prisoners are being separately questioned and must decide whether to co-operate, or defect, and the cost to each player can be seen in Fig.1 . Under selfish conditions, both players defect, giving a cost per player of 5. However, if both players co-operate, the average cost per player is reduced to 1.

	Co-operate	Defect
Co-operate	(1,1)	(7,0)
Defect	(0,7)	(5,5)

Figure 1: Payoff matrix for a prisoners dilemma game

In [3] Bell and Stidham look into a particular system where the queues were arranged in parrallel. Players knew the service time at each of these queues and could join any of them and recieve the same service. However players were not able to see the length of the queue before joining. This is an example of an unobservable queueing system, where players are not able to see the number of other players ahead of them prior to joining. Some work has been done in showing the differences in player behaviour in unobservable and observable versions of the same queue in [5]. In that paper, the effect of the amount of information available to players, including some "intentional vagueness" was studied while the system was under both selfish and social conditions. Boudali [17] studied the effect of choice in a system where there was a possibility of a catastrophe occurring and players would have to abandon the system, wasting the time spent waiting.

In this paper we will focus on a hierarchical queueing system in which players must progress through a series of queues as seen in Fig.2.

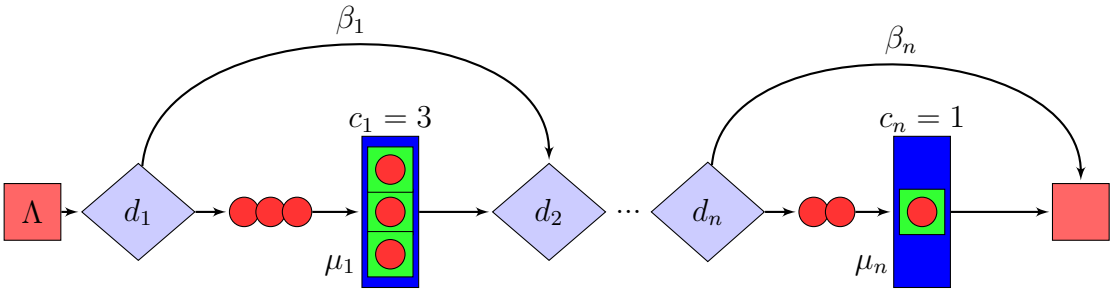


Figure 2: Diagram of n stations in series

The players arrive into the system at a rate Λ and make a decision on whether or not to join the first queue. The players know the rate of service μ_k at each queue , the number of servers at each queue c_k and also the number of players currently in each queue. Once they observe a queue, they must either join the queue or skip the queue. There is a cost associated with skipping a particular queue give by β_k . There is also a chance that after completing service at a queue, the player would leave the system completely, known as the dropout probability p_k. Players make their decision d_k by comparing the skipping cost to the expected cost of joining the queue .

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

A player arriving at queue k and observing m agents ahead of them will expect to receive a cost given by equation (2).

$$E[\sigma_k] = \frac{m+1}{\mu_k c_k} \tag{2}$$

where σ_k denotes the actual cost to a player at station k.

In addition, we let ξ denote the cumulative cost incurred by a player moving through the system. T_k in eq 3 denotes the actual time spent in station k by a player. It is the sum of the time spent waiting to be served (T_k^w) and the time spent being served (T_k^s). Thus we have the cumulative cost to a player in station

$$\xi_k = \begin{cases} 0, \text{ if } i = 0 \\ \xi_{k-1} + \begin{cases} T_k, & \text{if player joins queue k} \\ \beta_k, & \text{if player skips queue k} \end{cases} \end{cases} \tag{3}$$

Upon arriving at queue k a player will make a decision that will benefit them by comparing (2) to β_k. For example, a player arriving at a queue with the parameters listed in Fig.3 will expect to add .5 to their cost if they receive service at that station. As the balking cost for this queue is lower than the expected cost ($\beta = 0.4$), this player would skip under selfish conditions.

Parameter	Value
μ	2
c	4
m	3
β	0.4

Figure 3: Example parameters for a station

A player moving through the system with n stations can expect to receive a cost of (4) when making a choice $d_k(1)$ at queue k .

$$E[\xi_n] = \sum_{i=1}^{n} (1 - d_i) \left(\frac{m_i + 1}{\mu_i c_i} \right) + \sum_{i=1}^{n} d_i \beta_i \tag{4}$$

We want to instigate a policy which minimises the average cost per player, which is known as the optimal social policy. In the system, the policy is the length of the queue at which a player arriving at the queue will balk from it. Now consider Fig.3, and we can see that a player skips from this queue if there are 3 people in the queue.

By considering policies which benefit the system by reducing the average cost per player, we can see the effect of choice in this system, which is what is implied by a social policy (we also say that the system is under social conditions). Within this paper, we consider the Price of Anarchy (5) or PoA, the ratio of average cost per player under selfish conditions (given by $\tilde{\tau}$), to the average cost per player under social conditions (given by τ^*). In the earlier example concerning the prisoners dilemma, the Price of Anarchy is $5/1 = 5$

$$PoA = \frac{\tilde{\tau}}{\tau^*} \tag{5}$$

This paper is structured as follows: Section 2 gives an outline of how a simulation model was used to study the queueing system, Section 3 explains the heuristic methods involved in finding the optimal social policy, Section 4 describes an analytical approximation of the model used to study the system at a much lower computational cost. This approximation was validated with the simulation model. Section 5 will discuss the results we gained from the analytical approximation version of the model and then it will conclude with some further work.

2 Development of Simulation model

This paper initially proposes a bespoke simulation model [4] written in Python [15] to investigate the system. The model itself is a combination of Discrete Event Simulation[11] and an Agent Based Model [14]. The flow of the simulation model can be seen in Fig.4.

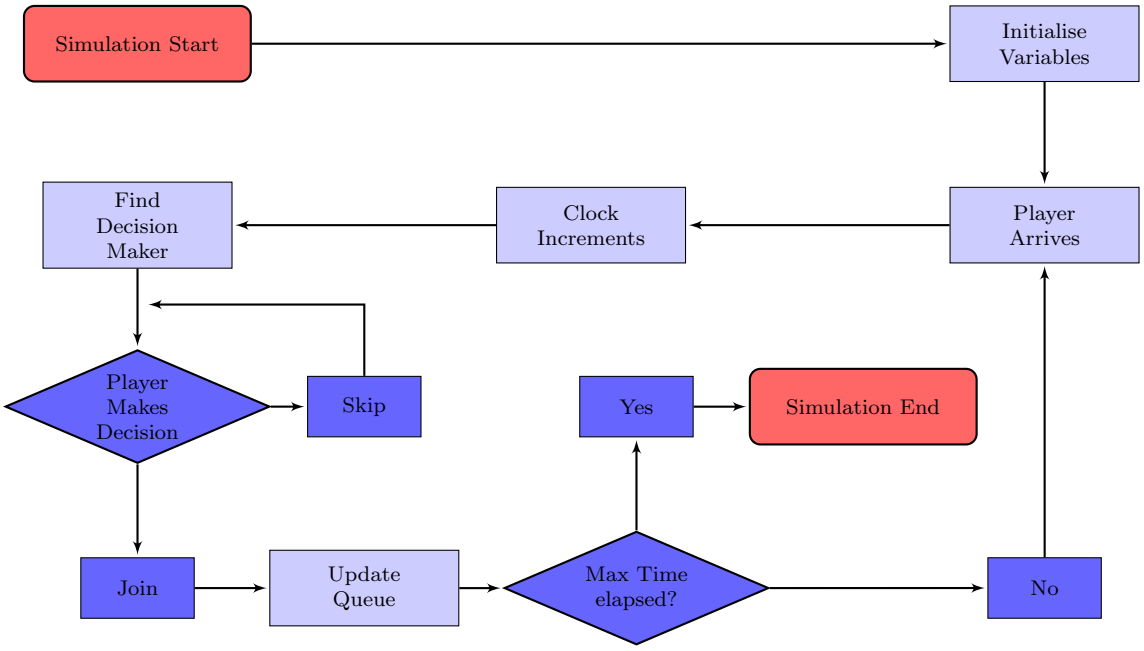
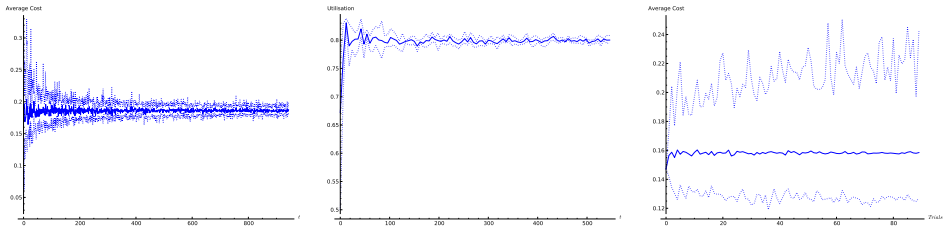


Figure 4: Flow chart describing the simulation model

Our simulation model will initialise with the system empty, however this would not represent the system it is attempting to represent. This is due to the fact that most systems do not begin with an empty queue (e.g. a GP surgery would already have appointments scheduled, which would represent players in a queue in the context of our model.) and the potential impact of this is discussed in [9]. The behaviour we are investigating occurs when the system has reached steady state. To ensure that we are getting results from the steady state of a model, we must first determine the length of time it takes before the results from the model become invariant with respect to time. In [18], a method is discussed for finding the correct length for a model to run before collecting results. The algorithm involves the simulation being run for an increasing amount of time, and the mean utilisation recorded along with the standard deviation. By plotting the mean utilisation, along with a 95% confidence interval, we can see when the model reaches steady state conditions by seeing when the graph stabilises. In Fig.5b, the mean and 95% confidence interval of the results was plotted to find the parameters necessary to ensure the accuracy of our results. The three parameters which needed to be determined were the length of the simulation, the warm-up period and the number of times each instance would be repeated. . Using the method suggested previously, we concluded the we needed to run the model for 250 time units and a warm up period of 50 time units was required. The number of trials did not cause a large variation even for small number of repeats and we decided to go with 16.



(a) Run Time (b) Warm-up time (c) Trials

Figure 5: Various graphs showing stabilisation of utilisation and cost as the simulation time increases

To be able to effectively investigate larger scenarios, we started to use the Cardiff University supercomputer, Merlin [1].The model is an example of embarrassingly parallel problem[6], due to the number of trials which enabled

us to take advantage of the large number of cpu's available on Merlin[8].

3 Optimisation

The motivation for this paper is to study the effect that choice has on these particular systems, by using the Price of Anarchy as a measure of the inefficiency. To understand the impact of choice, we do not need the optimal social policy, but merely a good approximation to the optimal social policy. In [10] a variety of heuristic methods are discussed, along with criteria for when a heuristic is applicable and what makes a good heuristic. We decided to use heuristic methods based on the conditions

- No exact method was known to find the optimal solution
- Although searching the solution space exhaustively would eventually yield the optimal solution, this proved to be far to computationally expensive

Based on these criteria, the following heuristic algorithms were tested.

We considered a variation on a local search heuristic, where after choosing our starting point as the selfish policy, we search a small area around this point exhaustively. If we find a better solution, we re-center our search space on the improved policy. One version of this algorithm did not change the search range (labeled search range 6), but there was an alternate version that if it did not find a better policy with the search space, it increased the range in which it searched (labeled variable search range 6). While this heuristic will eventually find the optimal policy (as the entire space of possible policies will eventually be searched) the method required a long time to run even small instances (even with the resources available described in chapter 3). As such the algorithm suffered a problem common to hill climbing heuristics, become locked into a local optima. By examining Fig. 6 we can see that the performance of the algorithm was significantly worse due to the limited number of iterations we could run. The use of a similar hill-climbing heuristic within the context of a probabilistic model can be found in [7], including an examination of its use to find local optima.

Another, simpler heuristic was investigated. A Random descent heuristic simply searches the solution space randomly, sometimes bound by conditions, but also completely unbound. While we experimented using a random walk

which optimised each stations policy individually (labeled iterative 6), it was found that this method was either similar, or at times worse than a completely random method. By again referring to Fig. 6 we can see that this algorithm vastly out performed the local search algorithm.

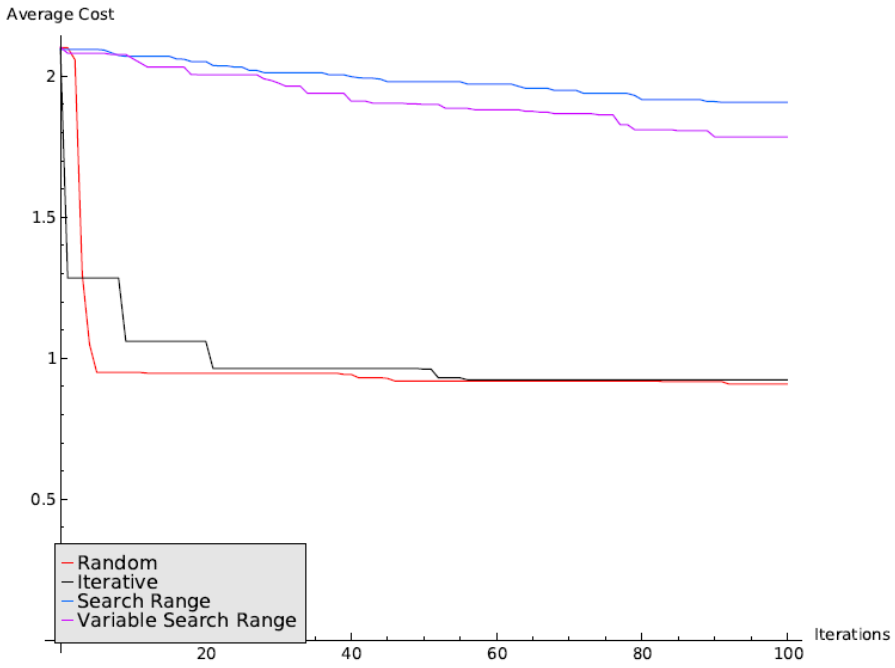


Figure 6: Comparison of the Local Search and Random Descent

4 Analytical Approximation

To reduce the computational cost of the investigations, we now propose an analytical approximation of the above model. For reasons that will become clear, we assume that despite the non-markovian nature of the decisions made by the players the actual inter arrival rate at each of the stations follows a Markovian Distribution [4]. The arrival rate into each station is given by (6). This formula is based on the dropout probability and arrival rate at the previous station.

$$\lambda_i = \lambda_{i-1} \times (1 - p_{i-1} \times (1 - \pi_N^{(i-1)})) \tag{6}$$

We also need to consider that fact that players will not dropout of the system if they balk from a particular queue. This is represented in (6) by $(1-\pi_N^{(i-1)})$ which only applies the probability of a player exiting the system to the players that do not balk. $\pi_N^{(i)}$ is the probability of there being N players in the queue, where N is the point at which players skip, either selfishly or due to a policy. We can then obtain a formula for the expected cost to a player moving through the system, which is expressed in 7. The formula is a sum of the cost at each station, multiplied by the percentage of Λ that arrives into station j

$$C = \sum_{j=1}^{n} \frac{\lambda_j C_j}{\Lambda} \tag{7}$$

We can approximate the cost to a player going through station i by calculating the probability that station i will have m players in it[12]. This probability is given by

$$\pi_n = \frac{1}{n!}\left(\frac{\lambda}{\mu}\right)^n \pi_0 \qquad 0 \leq n \leq c \tag{8}$$

$$\pi_n = \frac{1}{c^{n-c}c!}\left(\frac{\lambda}{\mu}\right)^n \pi_0 \qquad c < n \leq K \tag{9}$$

where

$$\pi_0 = \left[\sum_{n=0}^{c-1} \frac{1}{n!}\left(\frac{\lambda}{\mu}\right)^n + \sum_{n=c}^{K} \frac{1}{c^{n-c}c!}\left(\frac{\lambda}{\mu}\right)^n\right]^{-1} \tag{10}$$

Station i is then in K+1 states given that there are N players in the queue. Either the queue has less than c players(8), between c players and the maximum queue capacity K players(9) or the queue is full with K players in the system. By using these probabilities and the formula for expected cost(2) we can calculate the C given in (7)

$$C_i = \pi_N^{(i)}\beta_i + \sum_{j=0}^{c_i} \frac{\pi_j^{(i)}}{\mu_i} + \sum_{j=c_i}^{N-1} \frac{\pi_j^{(i)}(j+1)}{\mu_i c_i} \tag{11}$$

This is an analytical approximation and was verified by comparing it to the simulation model as can be seen in Fig.7. The analytical model was

a vast improvement in speed and efficiency over the simulation model. It allowed each possible policy to be evaluated and the expected cost per player to be approximated.a Searching the policy space exhaustively took less time than running a heuristic method.

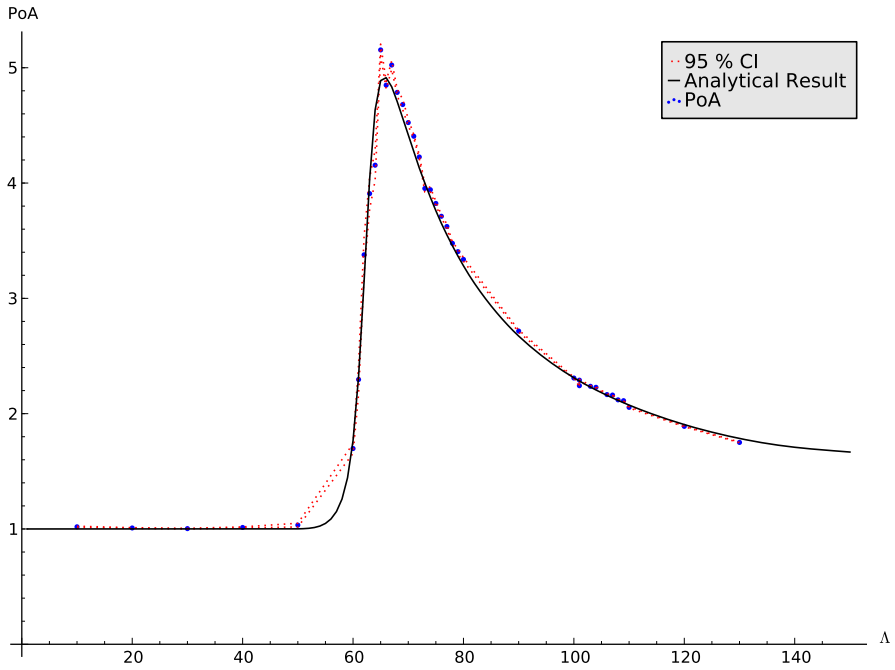


Figure 7: Comparison of the analytical model data to the simulation model for the case study

An assumption made to compute this analytical approximation was that the arrival rate into any given station was Markovian. However, due to this assumption the analytical model may not always give such a close approximation to the simulation result [8]. This is due the fact that the equations used in [12] assume an $M/M/C/K$ queue, but the dropout probability causes our system to become a series of $G/M/C/K$ queues.

Considering the case such that we have no probability of a customer leaving the system, the system as a whole exhibits Quasi-reversibility[12] which means that the arrival and departure rate is time independent. Burke's theorem [2] states that an $M/M/C$ queue with arrival rate λ which is sampled from a poison distribution has the following properties.

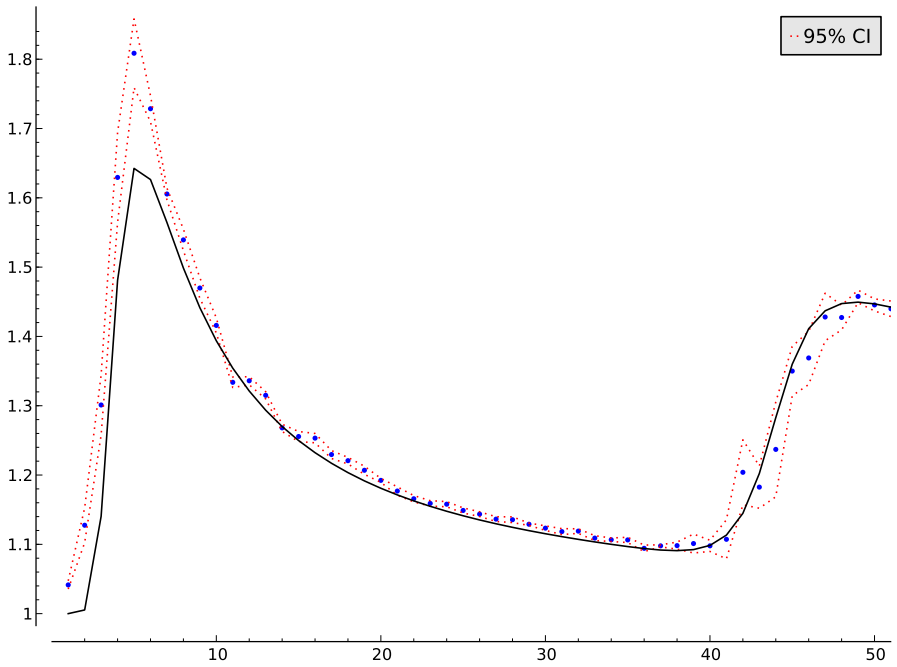


Figure 8: Comparison of the analytical model data to the simulation model with a dropout probability

- The departure process is a Poisson process with rate parameter λ.
- At time t the number of customers in the queue is independent of the departure process prior to time t.

It is possible to obtain the optimal social policy for a single $M/M/1$ queue, called the Naor threshold [13]. The result given describes the threshold at which players should balk from a single queue, and some preliminary results show that for no dropout probability and 1 server and each station, this does hold for our system.

5 Results

The results shall be discussed in two different contexts, in a mostly general framework and with respect to a case study.

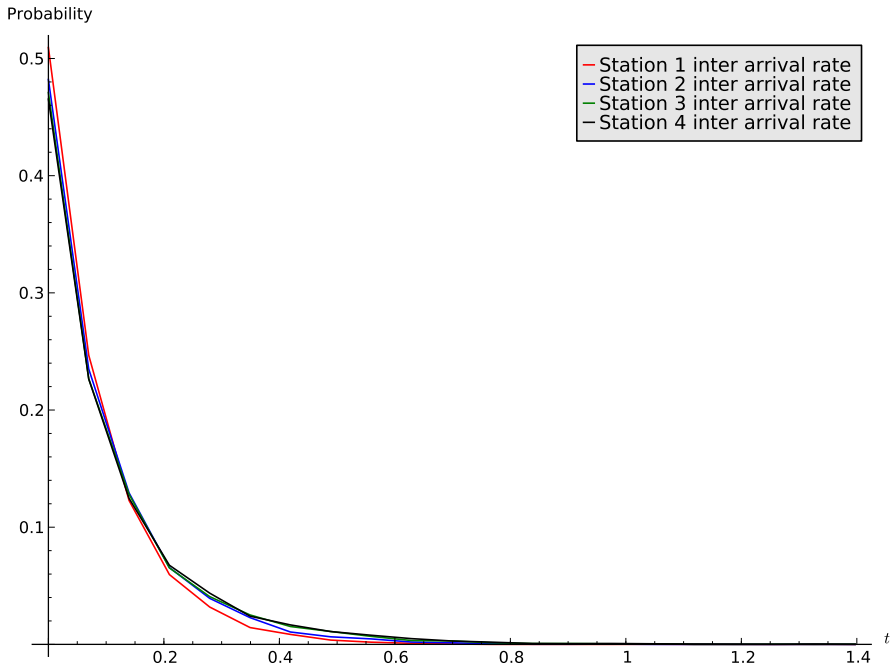


Figure 9: Showing that the probability of a player entering each station approximates a Markov distribution

The results that returned from both the analytical and simulation model showed that the exit probability had interesting effects on the Price of Anarchy. Our initial assumption was that the optimal social policy would always be less than the Nash policy, but this only held true when there was no chance for a player to leave the system. It was discovered that when we add an exit probability to a particular queue, it may improve the average cost per player if more players go through this queue. Fig.[10] compares two different systems, one with two stations in series and one with 3. The Price of Anarchy was computed for both these systems, and when the search space was expanded we can see the Price of Anarchy increase (which corresponds to a drop in the average cost per player under social conditions).

Another way in which the exit probability can affect the Price of Anarchy is to mask the presence of a second peak appearing in the demand curve, that can been seen appearing in many situations involving Price of Anarchy [17]. The peaked profile is caused by a difference in players skipping under

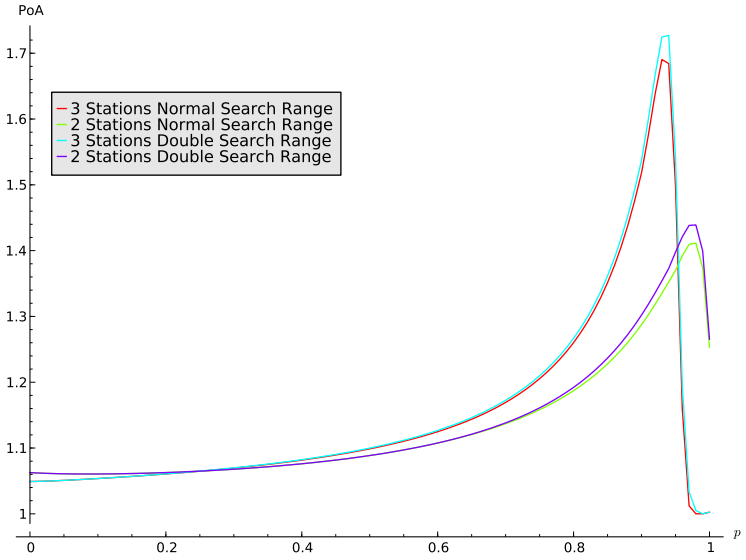


Figure 10: Increase in PoA when searching outside the selfish policy

selfish and social conditions. The Price of Anarchy increases when players are skipping selfishly but not socially. The peak then decreases when the demand increases further and players skip under both conditions. A second peak occours in systems where there are multiple stations, and the demand at which a peak appears for each station are spread out. While a single peak was common throughout the course of the investigation, there did not seem to be any effect from the hierachical structure in most cases. The reason behind this was that the exit probability was lowering demand at stations further into the system, so that the demand was too low to cause this profile. Once demand reached a threshold and all the players skipped earlier queues, the rest of the system was flooded with additional demand, and the players skipped the later queues which removed the second peak. However, we can see this second peak in Fig.[11] when there is a lower probability of players dropping out in a 2 queue system.

Our case study consisted of a General Practitioners, which leads onto an accident and emergency center. Based on nhs data, the GP surgery had 5 doctors on hand, able to see 12.5 patients a day. Patients are willing to wait up to 2 days to see the GP. We made an assumption that an A&E would have 5 doctors on hand to see patients from this particular GP surgery. The

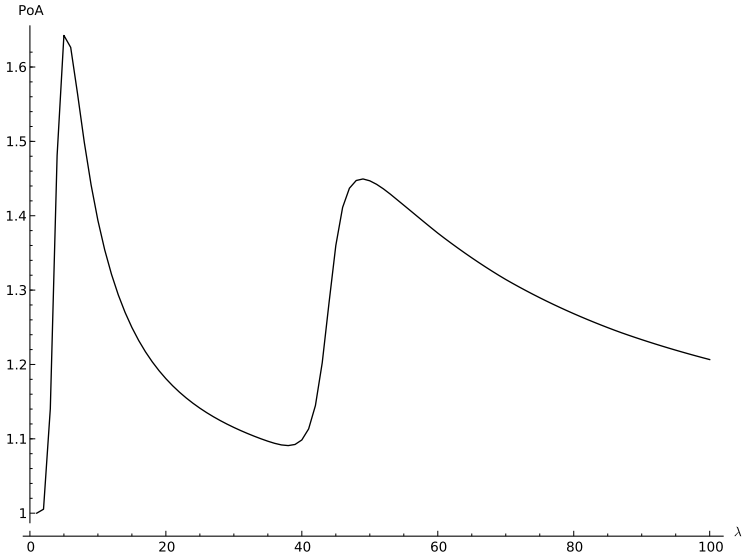


Figure 11: Second Peak appearing when 50% of players dropout of the 1st station

A&E doctors can each see 48 patients a day, patients will wait up to 8 hours to see the A&E doctor. However, if patients see a GP, 80% of them will not have to go to A&E, and simply leave the system there.

We can see in Fig.13 (where we are modeling the above system), there does exist a dropout probability the minimises the negative impact due to choice. However in the practical situation of sending 85% (Fig.13) of patients to an A&E, it may not be feasible to implement this dropout probability. As previously mention, increasing the length that patients are willing to wait causes the PoA increases which can be seen in Fig. 12. Also interesting to note is the appearance of a second peak can be seen as the value of service at the A&E increases.

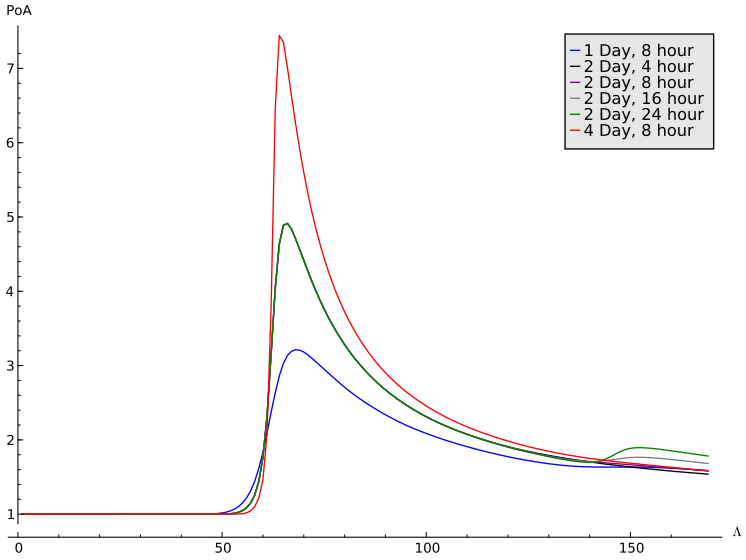


Figure 12: Varying the value of service at each station as demand increases

6 Conclusions and further work

We have considered a hierarchical system in both selfish and social conditions. By looking at the price of Anarchy in a series of $M|M|C$ queues we can conclude

- There exists a dropout probability such that the negative impact due to choice is minimised.
- The hierarchical structure contributes to the Price of Anarchy. However the dropout probability can alter and disguise this effect on the system.
- Increasing the cost to a player balking increases the Price of Anarchy.

There is much debate whether or not choice is beneficial to a given system. An example of this is within the NHS, and there are points of view that suggest that patients are happier with some element of choice [19] and some that state that it is better to have a controlling element [16]. However, from a strictly academic point of view, this paper has shown that by designing a system in certain ways we can minimises the negative impact due to individual behaviour. By only having a certain proportion of players remain within

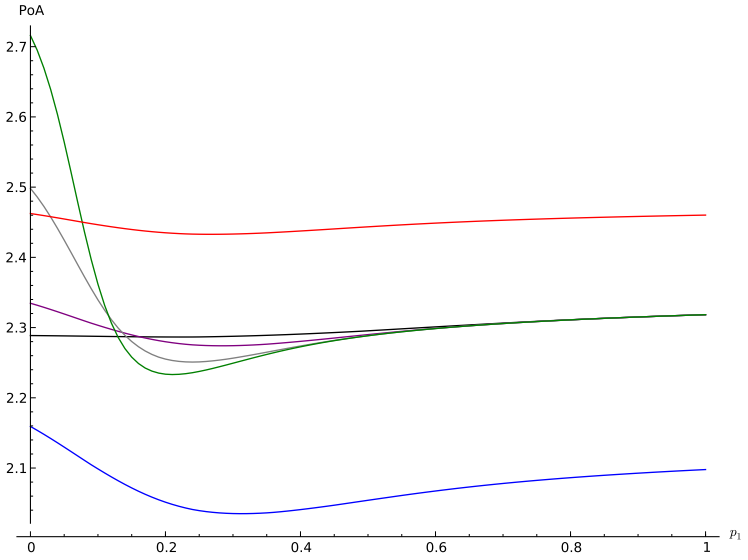


Figure 13: Varying the exit parameters for different balking costs

the system after completing service, the Price of Anarchy is minimised. The cost to a player for skipping a queue can be equated to how much service at a particular station is worth to players. By changing the worth of service at a station it is possible to increase and decrease the Price of Anarchy.It has also been shown that having a hierarchical structure within the queueing system can affect the Price of Anarchy, but is dependent on the dropout probability at previous stations. Usually it is advantageous to have a social policy that dictates players balk from queues earlier than they would under selfish conditions. However, If there is a large enough dropout probability, it can be better for players join a queue with more players ahead of them than under selfish conditions.

Further work on this project would include some technical aspects, such as improving the performence of the simulation model. In this system, the arrival and service times are sampled from a markovian distribution, and it may yield some insights into the behaviour of the system if we consider other distributions. As in [5], altering the amount of information given to players such as the exit probability at each queue would give greater insights into how that probability affects the Price of Anarchy. Giving less information to the players and making the system unobservable could be compared to the

existing system to show how this affected player choice. This model can be changed very slightly to give a completely different system. Instead of making a skip or a join decision upon entering a particular queue the player makes one decision upon entering the system, which queue to join inititally. This one change completely alters the structure of the policy and the approximation of the cost of joining at a particular point. The investigation of this model, particularly with the focus on the new structure of the policy would be the primary focus of any future work.

References

[1] Arrca. http://www.cardiff.ac.uk/arcca/.

[2] Paul J. Burke. The output of a queuing system. Operations Research, 4(6):699–704, December 1956.

[3] S. Stidham C. Bell. Individual versus social optimization in the allocation of customers to alternative server. Management Science, 29:831–839, 1983.

[4] S. R Chakravarthy. Markovian arrival processes. Wiley Encyclopedia of Operations Research and Management Science, 2011.

[5] R. Shone et al. Comparisons between observable and unobservable m/m/1 queues with respect to optimal customer behavior. European Journal of Operational Research, 2013.

[6] Ian Foster. Designing and Building Parallel Programs. Addison–Wesley, 1st edition, 1995.

[7] Russell Greiner. PALO: a probabilistic hill-climbing algorithm. Artificial Intelligence, 84(1–2):177–208, July 1996.

[8] Allan R.; et al Hoffman. Supercomputers: directions in technology and applications. National Academies, 1990.

[9] Krzysztof Pawlikowski. Steady-state simulation of queueing processes: survey of problems and solutions. ACM Comput. Surv., 22(2):123–170, June 1990.

[10] Rafael Martí and Gerhard Reinelt. Heuristic methods. In The Linear Ordering Problem, page 17–40. Springer, 2011.

[11] Norm. Matloff. Introduction to discrete-event simulation and the simpy language. 2013.

[12] R.R Muntz. Poisson departure process and queueing networks. IBM Research Report, RC 4145, 1972.

[13] P. Naor. The regulation of queue size by levying tolls. Econometrica, 37(1):15–24, January 1969. ArticleType: research-article / Full publication date: Jan., 1969 / Copyright © 1969 The Econometric Society.

[14] Amir Niazi, Muaz; Hussain. Agent-based computing from multi-agent systems to agent-based models: A visual survey. Scientometrics (Springer), 89:479 – 499, 2011.

[15] Guido Van Rossum. Python. www.python.org.

[16] B. Schwartz. The paradox of choice: Why more is less. 2005.

[17] Paul R. Harper Vincent A. Knight. Selfish routing in public services. European Journal of Operational Research, 230:122–132, 2013.

[18] John Schormans Xu, Ling. Planning simulation run length for queueing systems.

[19] V. Zigante. Subjective well-being as a measure of welfare and equity: the case of choice policies in health care. CESifo Economic Studies, 57:715–739, 2011.