Adaptive Status Arrivals Policy (ASAP) Delivering Fresh Information (Minimise Peak Age) in Real World Scenarios

Basel Barakat[1], Simeon Keates[2], Ian Wassell[3], and Kamran Arshad[4]

[1] Faculty of Engineering and Science, University of Greenwich, Kent, ME4 4TB, UK.
bb141@gre.ac.uk
[2] School of Engineering and the Built Environment, Edinburgh Napier University, Edinburgh, EH10 5DT, UK.
[3] Computer Laboratory, University of Cambridge, Cambridge, CB2 1TN, UK.
[4] Ajman University of Science and Technology, Ajman, Ajman, UAE.

Abstract. Real-time systems make their decisions based on information communicated from sensors. Consequently, delivering information in a timely manner is critical to such systems. In this paper, a policy for delivering fresh information (or minimising the Peak Age of the information) is proposed. The proposed policy, i.e., the Adaptive Status Arrivals Policy (ASAP), adaptively controls the timing between updates to enhance the Peak Age (PA) performance of real-time systems. Firstly, an optimal value for the inter-arrival rate is derived. Afterwards, we implemented the policy in three scenarios and measured the ASAP PA performance. The experiments showed that ASAP is able to approach the theoretical optimal PA performance. Moreover, it can deliver fresh information in scenarios where the server is located in the cloud.

Keywords

Real Time Systems, Cloud, Adaptive Systems, Information Freshness, Peak Age of Information.

1 Introduction

Recently, several applications rely on real-time communications had been investigated such as autonomous cars, tactile internet and telehealth. Delivering the information in a timely manner is critical for these applications. For instance, in a robotic surgery, excessive delays might be life-threatening. Hence, several researchers proposed polices for delivering information with as low latency as possible. However, most of the work done is based on assumptions that might be oversimplifying. These assumptions and the whole latency current paradigm should be questioned.

Consider a vegetable market as an analogy, an insightful question would be what do we mostly care about: (i) the speed of the vegetable carrier, (ii)

the time the vegetables took to arrive at the market or (iii) the freshness of the vegetables? Usually, one only care about the freshness of the vegetables. It is obvious that freshness is affected by both the speed of transporting the vegetables and the time it took for them to arrive at the market. Moreover, the freshness is also affected by when the vegetables were grown. This analogy can represent several systems where the information has a window of time in which it is useful and after that window, the information loses its usefulness, for example, an adaptive control system or systems with online machine learning algorithms. In queuing theory/data networks terms, the speed of transmitting information in a network is called the data throughput, the time taken to communicate a piece of information is the delay or latency and the time to grow the vegetables is the time to generate the package/information. However, until recently no metric has focused on information freshness.

To evaluate the freshness of an update, let us assume that we have a stopwatch that starts counting as soon as an update was generated. The stopwatch stops immediately after receiving the next update at the destination. The time elapsed until the stopwatch is stopped is called Age of the Information (AoI) [1]. The final value, the stopwatch shows is the Peak Age of the Information (PA) [2]. The PA is defined as the maximum value of AoI [3]. It can be noticed that the PA consists of the inter-arrival time and the delay time. Unlike the conventional queuing theory delay metric, PA evaluates the freshness from the destination's perspective. In other words, PA reflects the information's freshness, not the time it took to communicate it.

Several policies have been proposed to minimise PA in the literature and can be classified into two main categories [1]. The first category controls the buffer size, which regulates the maximum number of updates waiting in the buffer, e.g., in [3]. The second category attempts to minimise the updates' waiting time and hence reduce the PA. For instance, the Zero-Wait (ZW) policy minimises the waiting time by only permitting the updates' source to communicate a new update after it has ensured that the destination is idle, by waiting for an Acknowledgement (ACK) from the destination after every transmitted update [4]. The state-of-art policies reduce PA, however, their performance does not always approach the minimum PA, as shown in section 2.

In this paper, we propose a policy that is able to reach a near-optimal PA performance. The proposed policy, i.e., Adaptive Status Arrivals Policy (ASAP), adapts the status updating inter-arrivals rate to reach the optimal PA performance. The ASAP was tested in three scenarios, as detailed in section 2. The first scenario is a single queue with single service rate (μ). The second scenario also has a single queue, however, the mean service time can take four values, that emulates a system where the load on the server changes between four service durations or a wireless channel with adaptive coding and modulation [6, 7]. In the third scenario, the updates were transmitted through the internet to a destination located in the cloud. The ASAP continuously changes the inter-arrivals rate (λ) to reach the optimal value, which is derived in section 3. The PA performance of the ASAP presented in section 4 shows that it can deliver the

information with a freshness that is close to the theoretical optimal freshness. This paper is concluded in section 5.

2 Problem Statement and System Model

2.1 Peak Age Metric

The PA metric was proposed as a policy for calculating the peaks of a typical AoI sawtooth profile [1]. The AoI is defined as the time elapsed from the generation of the last successfully received message [1]. In Fig. 1, an illustration of AoI evolution is presented. When an update is generated, the AoI is zero and as time passes, the AoI grows linearly until the receipt of the next update. The highest value of AoI, which forms the peak of the AoI pattern, is PA, which represents the highest value of AoI.

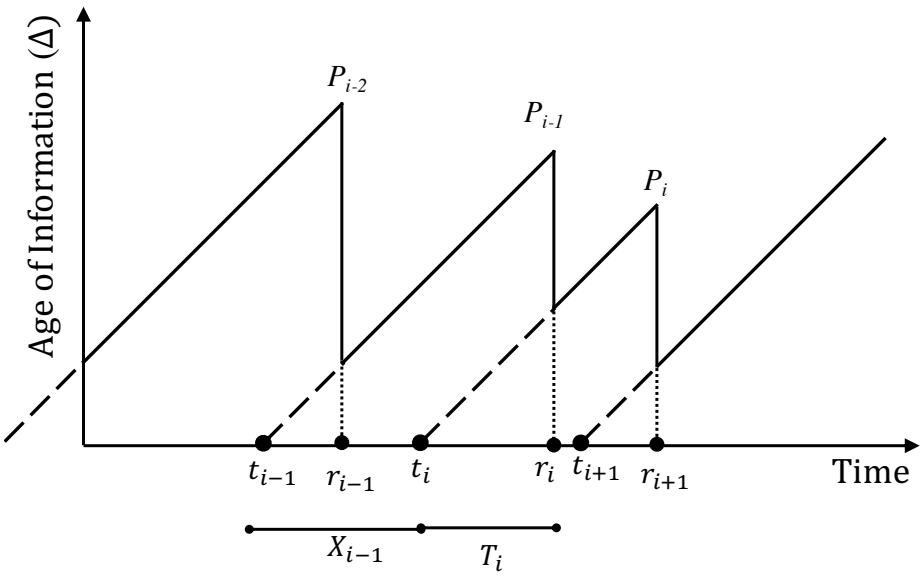

Fig. 1. Age of Information (AoI.pdf) illustration, the AoI for update (i) starts when an update is generated (t_i) and keeps counting until the server receives the next update (R_{i+1}). The maximum value of AoI is called Peak Age (P_i), which is equal to inter-arrival time (X) plus the system time (T).

As shown in Fig. 1, the PA for a general queue with a single server (G/G/1) can be obtained as follows [3],

$$P \triangleq \mathbb{E}[X] + \mathbb{E}[T] = \mathbb{E}[X] + \mathbb{E}[S] + \mathbb{E}[W]. \tag{1}$$

where $\mathbb{E}[.]$ is the expectation operator, X is the inter-arrival time, i.e., $1/\lambda$ and T is the delay time, i.e., the the service time (S) plus the queuing time (W).

The service time and the waiting time affect the delay time and hence the PA. The service time is either deterministic or follows a random distribution, depending on the time the server takes to process the information. The waiting time depending on the service time, inter-arrival time, queue service discipline, number of servers and the maximum queue length.

2.2 Tested Scenarios

In this paper, the proposed policy is tested on three main scenarios. The first scenario is a single First in First Out (FIFO) queue and the service time follows a single distribution. The second scenario is similar to the first scenario, however, the mean service time follows several values. The final scenario is that the server is located in the cloud.

Single queue and mean service time scenario In this scenario, the proposed policy was tested on an M/D/1 queue, i.e., a queue where inter-arrival time distribution followed exponential distributions and the service time follows a deterministic one. This scenario emulates a system where the sensors transmit the information to a decision maker that is located in the same chip, such as a prosthetic limp that has a finger tips sensors and a micro-controller controls the power of the grip. Also, it is usually used to model a communication system where the distance between the source of the information and the destination is short such as Internet of things Machine to Machine communication (M2M) [10, 11].

For an M/D/1 queue, the delay time ($T^{M/D/1}$) is calculated using the Pollaczek-Khinchine (P-K) formula [5] as follows,

$$T^{M/D/1} = \frac{1}{2\mu}\left(\frac{2-\rho^2}{1-\rho}\right). \tag{2}$$

From (1) and (2), the PA of the M/D/1 queue ($P^{m/D/1}$) is,

$$P^{M/D/1} = \frac{1}{\mu}\left(1+\frac{1}{\rho}+\frac{\rho}{2(1-\rho)}\right). \tag{3}$$

Single Queue with several mean service time values In this scenario, the proposed policy was tested on a server with a mean service rate that can take several values. This scenario emulates a server that has several probable data processing speeds e.g. a data communication channel with Adaptive Modulation and Coding (AMC) such as Long Term Evolution (LTE) which has 15 Channel Quality Indicator (CQI) ranges [6, 7]. In this scenario, we have tested the ASAP for a server has four mean service rates as shown in Fig. 2.

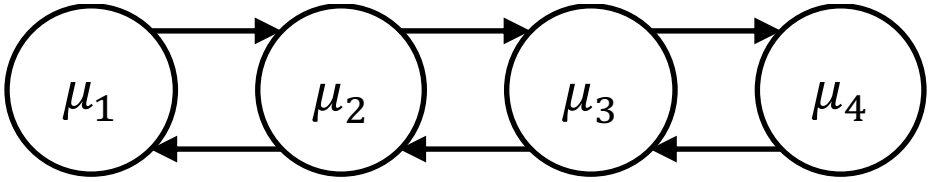

Fig. 2. Second Tested Scenario. In this scenario, the server mean service can take one of four mean values.

Status updating through the Internet Several major cloud services offer Infrastructure as a service for Internet of Things (IoT) applications, e.g., Amazon and Google. However, delivering information that is as fresh as possible to the cloud might be challenging. Hence, the models proposed in the literature, only consider a single queue. In this scenario, the source located at University of Greenwich, Medway Campus, will be transmitting information through the internet to a server located in a cloud service is shown in Fig. 3. The inter-arrival time follows an exponential distribution with mean $1/\lambda$, where λ is the inter-arrival rate. The sever service time mean is deterministic and is equal to $1/\mu$, where μ is the service rate.

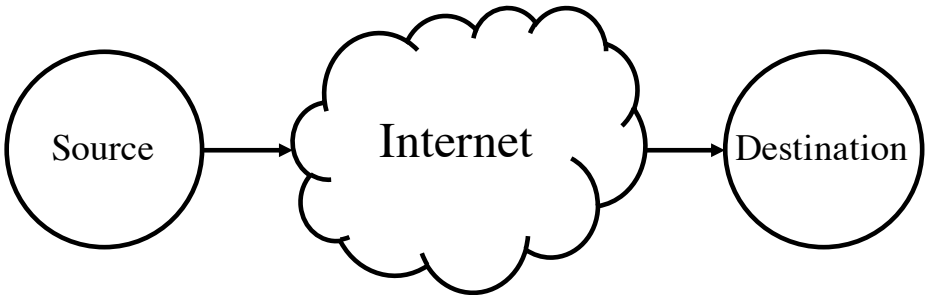

Fig. 3. Third tested scenario. In this scenario, the server is located in a cloud services provider.

Delivering fresh information in the third scenario, is more challenging than the previous two scenarios. On the other hand, it might be critical for the next generations of communication networks. Since several new applications and systems rely on the remote control of other machines that might be very distant from the controller. For example, the control of the *Cambridge Minicar*, which the control of a fleet of *Minicars* [8]. Also, the control of Base Stations antenna tilting in a cooperative self organisation network [9].

3 Adaptive Status Arrivals Policy (ASAP)

In the proposed policy, the client adapts its status inter-arrivals rate λ to reach the optimal PA value, as shown in Fig. 4. The optimal inter-arrival rate (λ^{opt}) relies on the μ value, hence it can be calculated using [5],

$$\lambda^{\text{opt}} = \rho^{\text{opt}} \times \mu, \tag{4}$$

where, ρ^{opt} is the optimal server utilisation value. Now ρ^{opt} depends on the scenario, hence in the next section, ρ^{opt} is derived for the tested scenarios.

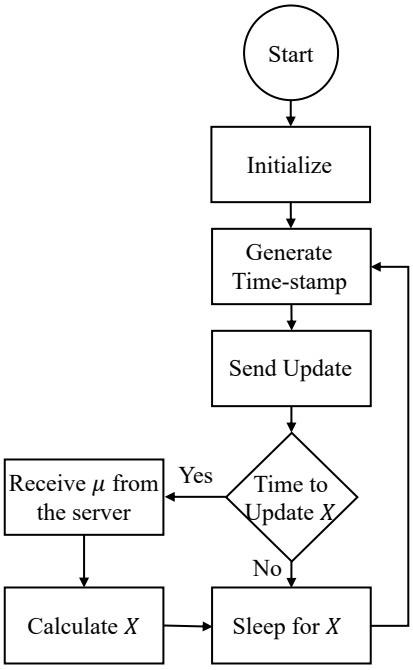

Fig. 4. ASAP client flow chart. The Client generates a time-stamp, sends the update and sleeps for the duration of the inter-arrival time (X). When its time to update the (X) it would receive the server service time μ and adapt its X accordingly.

3.1 Optimal server utilisation for minimising the Peak Age

An optimisation problem with its objective to minimise the PA with respect to ρ was formulated, as follows:

$$\begin{aligned} P^{\text{opt}} \triangleq \min_{\rho} \quad & P(\rho) \\ \text{subject to:} \quad & \rho < 1 \\ & \lambda \leq \lambda^{\max}. \end{aligned} \tag{5}$$

The P^{opt} refers to the optimal PA value, the first constraint is to ensure that the queue is stable, since if $\rho \to 1$ then the delay time $T \to \infty$ [5]. The second constraint ensures that λ does not exceed the λ^{max}, which is determined by the device capabilities (for instance the sensor clock cycle).

For an M/D/1 queue, the ρ^{opt} can be obtained as follows,

$$\frac{d}{d\rho} P^{M/D/1}(\rho) = 0. \tag{6}$$

From (3) and (6), ρ^{opt} can be derived by,

$$\frac{d}{d\rho}\left[\frac{1}{\lambda} + \frac{1}{2\mu}\left(\frac{2-\rho^2}{1-\rho}\right)\right] = 0 \tag{7}$$

$$\frac{d}{d\rho} P^{M/D/1}(\rho) = \frac{1}{2\rho - 2} - \frac{2(\rho-2)}{(2\rho-2)^2} - \frac{1}{\rho^2} = 0. \tag{8}$$

The optimal server utilisation for the M/D/1 queue is,

$$\rho^{opt} \approx 0.5858. \tag{9}$$

The PA value for M/M/1 and M/D/1 queues are plotted in Fig. 5. It can observed that optimal ρ derived in (9) achieve the minimum PA value.

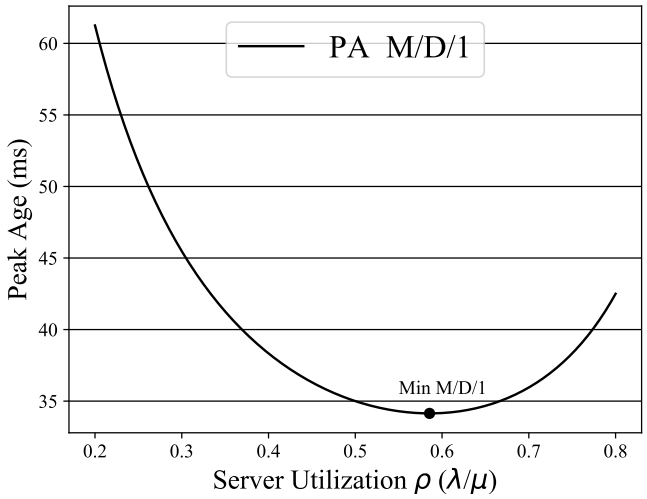

Fig. 5. Peak Age versus server utilisation, showing minimum value for M/D/1 queue with $\mu = 100$. The optimal Server Utilisation value for M/D/1 queues is equal to 0.5858.

The derived optimal values for ρ, are only applicable to a single queue. However, the PA in the third scenario (where the server is located on the cloud) the

internet load would affect the inter-arrival time, hence it must be considered. The Inter-arrival time in this scenario (X), as observed from the server, is

$$X(n) = X^* + \epsilon(n). \tag{10}$$

Where X^* refers to the inter-arrival time and $\epsilon(n)$ is a random value representing the time it takes the information (n) to be transmitted through the internet.

From (10), the optimal inter-arrival rate in the third scenario is,

$$\lambda(n)^* = \frac{1}{X^*} = \frac{1}{X(n) - \epsilon(n)}. \tag{11}$$

Hence,

$$\rho(n)^* = \frac{1}{(X(n) - \epsilon(n)) \times \mu}. \tag{12}$$

3.2 Experimental System Model

To evaluate the PA performance, we implemented an experimental system, consisting of a Client and Server as shown in Fig. 6. The Client sends statutes updates, to the server, then it would sleep for the duration of the inter-arrival time, as shown in Fig. 7. In the experiment, each update consists of the instantaneous time-stamp (t_n). The server records time it received the updates (r_n). The updates were sent using TCP/IP protocol.

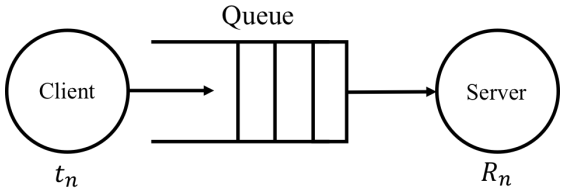

Fig. 6. Client-Server network model. The updates are generated in the Client and sent to the Server. The time of generating the update is t_n and the time of receiving the update is r_n.

The PA in the experiment was calculated by using the logged time-stamps. As shown in Fig. 8, the PA of update n can be calculated as follows,

$$P_n = r_{n+1} - t_n. \tag{13}$$

The experimental PA was obtained by taken the median value of the PA of all the updates sent,

$$P = \tilde{P}_{(1,2,\ldots,N)} \tag{14}$$

where, $\tilde{P}$ refers to the median value and N refers to the total number of transmitted updates.

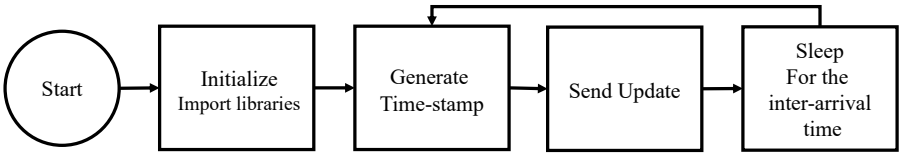

Fig. 7. Client flow chart. The Client initially import the socket libraries, then generate the instant time-stamp using the the Time module, after sending the update to the Server it would sleep for the inter-arrival duration.

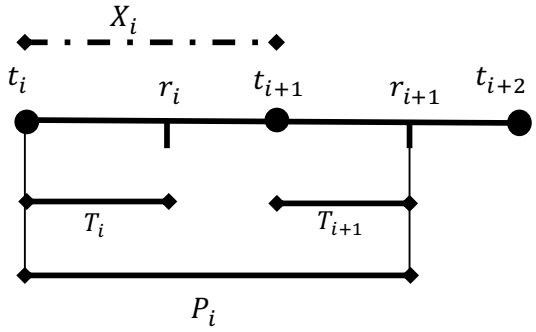

Fig. 8. Peak Age illustration. For the experiment the Peak Age value (P) is equal to the inter-arrival time (X) plus the update system time (T). Consequently, in the experiment, the Peak Age value of an update (i) can be obtained by subtracting the value of generating an update t_i from the time of receiving the next update (r_{i+1}).

4 ASAP Peak Age performance

The ASAP PA performance is presented in the three scenarios. In Fig. (9) the time series PA performance for an M/D/1 for the first scenario is shown. It can be observed that the ASAP PA performance varies with time as it keeps changing its inter-arrival time, i.e., λ, according to the server mean service time, μ. In our experiments, the server sends its service rate after receiving 100 updates and the sent value is the median service rate ($\tilde{\mu}$). Fig. 10, presents the mean PA value of the updates shown in Fig. 9. It is observed that the PA approaches the theoretical optimal PA value.

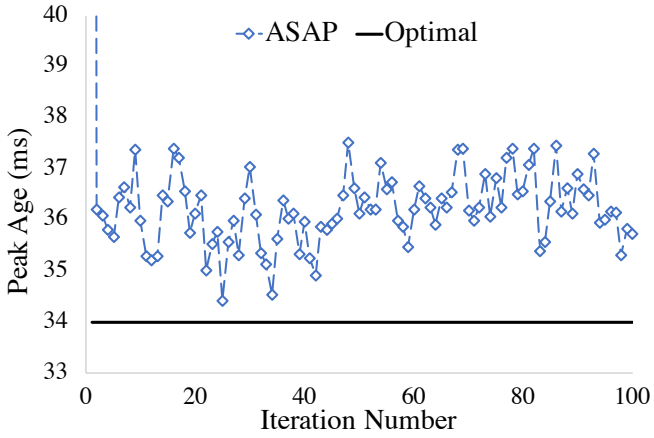

Fig. 9. ASAP Peak Age time-series performance. Each point in the ASAP represents the median value of 100 values. The presented values for the Optimal, represent the theoretical value for PA at the used service time. The Peak Age performance of ASAP policy continually changes, hence the service time is random.

The ASAP PA performance for the second scenario is presented in Fig. 11, where the service rate can take four possible mean values, the ASAP managed to deliver the updates with almost optimal freshness. It can be noted that the achieved performance is very close to the theoretical optimal performance. It is worth mentioning the achieved instantaneous PA performance might outperform the theoretical optimal PA, as the optimal represent the mean (average) performance.

In the third scenario, the ASAP managed to handle the internet load fluctuations as shown in Fig. 12. It is worth mentioning that the presented optimal performance in Fig. 12 represents the optimal PA for a single queue with the service time equal to the service time of the server plus the approximated value internet delay, hence the internet delays are random and hard to predicate.

Changing the inter-arrival rate makes ASAP a dynamic policy that is able to change its sampling rate to best fit the server. This feature can be critical in

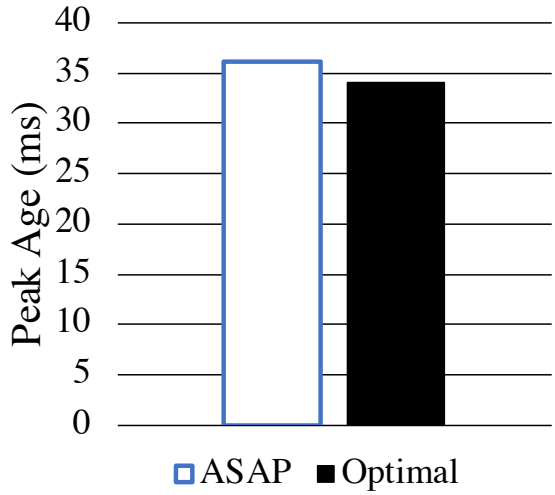

Fig. 10. ASAP Mean Peak Age performance. The values presented are the mean value of the results presented in Fig. 9.

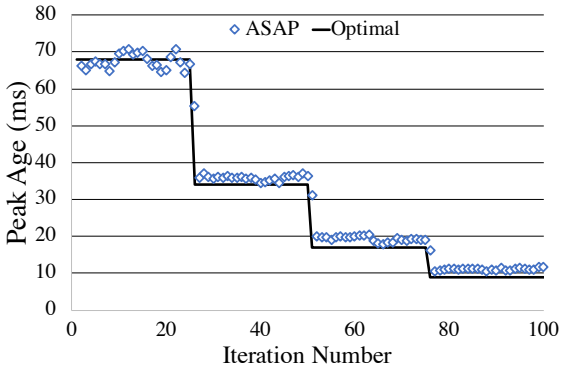

Fig. 11. ASAP Peak Age performance in the second scenario. Each point in the ASAP represents the median value of 1000 values. The presented values for the Optimal, represent the theoretical value for PA at the used service time. The Peak Age performance of ASAP policy continually changes, hence the service time is random. ASAP can outperform the Optimal value if the majority of the updates service time is less than the mean service time. Consequently, in the experiment the ASAP can outperform the Optimal value.

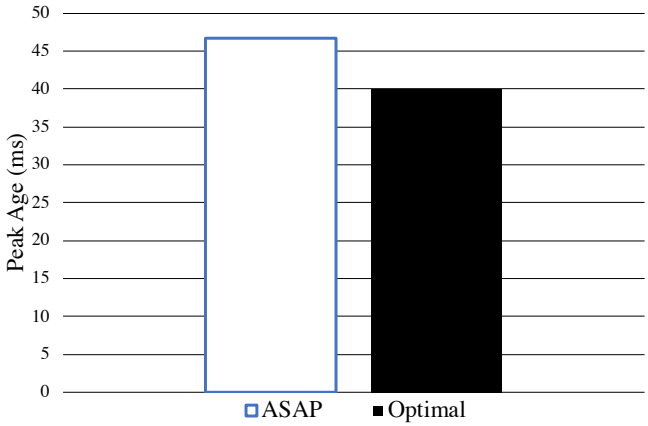

Fig. 12. ASAP Peak Age performance in the third scenario. Each bar in the ASAP represents the median value of 1000 values. The presented values for the Optimal, represent the theoretical value for PA at the used service time for a single queue.

real-world applications where the server might have several background processes running on it. Using ASAP, instead of the client imparing an extra load on the server, it can reduce its transmissions but maintain a near optimal freshness performance.

5 Conclusions

In this paper, the ASAP policy was proposed for minimising the Peak Age of Information. The policy regulates the inter-arrival time of status updates. The performance was measured by conducting experiments on three scenarios. The ASAP PA performance in the tested scenarios approaches the optimal value. Moreover, it can adapt to the server load and the varying load on the internet. The next step is to use the ASAP in real life applications to enhance its PA performance.

References

1. S. Kaul, R. Yates, M. Gruteser, "Real-time status: How often should one update?", Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM), pp. 2731-2735, Mar. 2012.
2. L. Huang and E. Modiano, "Optimizing age-of-information in a multi-class queueing system," 2015 IEEE International Symposium on Information Theory (ISIT), Hong Kong, 2015, pp. 1681-1685.
3. M. Costa, M. Codreanu and A. Ephremides, "On the Age of Information in Status Update Systems With Packet Management," in IEEE Transactions on Information Theory, vol. 62, no. 4, pp. 1897-1910, April 2016.doi: 10.1109/TIT.2016.2533395
4. Y. Sun, E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal and N. B. Shroff, "Update or Wait: How to Keep Your Data Fresh," in IEEE Transactions on Information Theory, vol. 63, no. 11, pp. 7492-7508, Nov. 2017.

5. D. Bertsekas and R. Gallager, Data Networks. Englewood Cliffs, NJ: Prentice-Hall, 1987
6. B. Barakat and K. Arshad, "An adaptive hybrid scheduling algorithm for LTE-Advanced," 2015 22nd International Conference on Telecommunications (ICT), Sydney, NSW, 2015, pp. 91-95.
7. S. O. Aramide, B. Barakat, Y. Wang, S. Keates and K. Arshad, "Generalized proportional fair (GPF) scheduler for LTE-A," 2017 9th Computer Science and Electronic Engineering (CEEC), Colchester, 2017, pp. 128-132.
8. Prorok, A., Hyldmar, N., & He, Y. A Fleet of Miniature Cars for Experiments in Cooperative Driving. IEEE International Conference on Robotics and Automation.
9. M. Sharsheer, B. Barakat and K. Arshad, "Coverage and capacity self-optimisation in LTE-Advanced using active antenna systems," 2016 IEEE Wireless Communications and Networking Conference, Doha, 2016, pp. 1-5.
10. B. Barakat, S. Keates, K. Arshad and I. J. Wassell, "Deriving Machine to Machine (M2M) Traffic Model from Communication Model," 2018 Fifth International Symposium on Innovation in Information and Communication Technology (ISIICT), Amman, 2018, pp. 1-5.
11. B. Barakat and K. Arshad, "Energy efficient scheduling in LTE-advanced for Machine Type Communication," 2015 International Conference and Workshop on Computing and Communication (IEMCON), Vancouver, BC, 2015, pp. 1-5.