Management and Production Engineering Review
Volume 4 • Number 1 • March 2013 • pp. 65–73
DOI: 10.2478/mper-2013-0008

SENSITIVITY ANALYSIS OF KEY CUSTOMERS AND REVENUE-ORIENTED ADMISSION CONTROL AND SCHEDULING ALGORITHM

Grażyna Suchacka[1], Leszek Borzemski[2]

[1] *Opole University, Institute of Mathematics and Informatics, Poland*
[2] *Wrocław University of Technology, Institute of Informatics, Poland*

Corresponding author:
Grażyna Suchacka
Opole University
Institute of Mathematics and Informatics
Oleska 48, 45-052 Opole, Poland
phone: +48 77 4527227
e-mail: gsuchacka@uni.opole.pl

Received: 15 January 2013
Accepted: 2 February 2013

Abstract
The paper deals with the problem of Quality of Web Service (QoWS) in e-commerce Web servers, i.e. in retail Web stores. It concerns the admission control and scheduling algorithm for a Web server system, which aims at preventing the system from overload to provide high QoWS level and ultimately, to increase Web site's conversion rate, i.e. to turn more visitors into customers. The sensitivity of the algorithm to changes in its basic parameter values was analyzed by using a simulation-based approach. Special attention was paid to evaluation of the parameter impact on conventional and business-related system performance metrics.

Keywords
Quality of Web Service, QoWS, Web server, e-commerce, simulation, conversion rate

Introduction

Due to the still increasing number of Internet users and intensive development of Web-based technologies, contemporary Web servers have to face with highly variable and unpredictable service demand. Server systems are designed to be capable of handling peak loads, usually through overprovision of their resources. However, it is unavoidable that during extremely high traffic bursts server resources are overutilized and the server is overloaded. Consequently, request response times increase excessively and users experience unacceptably high delays.

Such situation is especially undesirable for e-commerce applications, including B2C (*Business-to-Consumer*) applications, i.e. retail Web stores. As a result of poor QoWS (*Quality of Web Service*), online retailers loose potential revenue and let their loyal customers down. Even if a company applies a customer-focused strategy and uses advanced CRM (*Customer Relationship Management*) techniques, the company's effort may be thus thwarted.

One of possible ways of coping with this problem is introducing a method for differentiated request service in the Web server system. A number of such methods have been dedicated to e-commerce servers, e.g. in [1–7]. What distinguishes these approaches from others is the goal of revenue maximization, as well as taking into consideration characteristics of user sessions in a Web store, such as session states corresponding to types of business operations performed in the store, probabilities of transitions between the states, session lengths, and the contents of shopping carts.

This paper refers to KARO (*Key customers And Revenue Oriented admission and scheduling*) method and KARO-Rev (*KARO – Revenue-oriented*) algorithm, which were presented in [4] in detail. The next section briefly outlines the idea of

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM

the method and presents the algorithm, for which results of simulation experiments will be discussed further.

Key customers and revenue-oriented QoWS mechanism

We consider a B2C Web site which is equipped with a Web server allowing Internet users – potential customers to browse and search for products, add them to the shopping carts, confirm purchase transactions, and so on, like in a traditional, brick-and-mortar store. A single user's visit to the Web site is called a *user session*.

Every time a user opens a new page of the Web site, his or her browser generates a sequence of HTTP requests which are sent to the Web server (the first request in the sequence is for an HTML file and subsequent requests are for objects embedded in the page, such as graphic files). Many users may visit the store at the same time.

HTTP requests come to the Web server and are queued and processed in the system usually according to FIFO (*First In First Out*) policy. Here we assume that the whole system is organized as a multi-tier system, in which requests are processed at three logical layers: Web server, application server and database server. Some requests are processed only by the Web server whereas other have to be additionally processed by the back-end server as dynamic requests (Fig. 1).

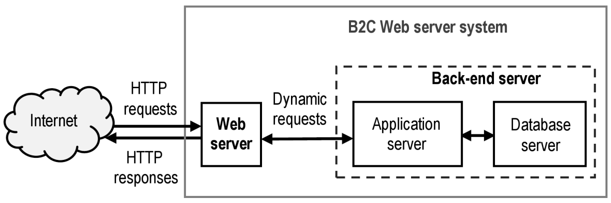

Fig. 1. Multi-layer configuration of a B2C Web site [4].

Idea of KARO method

The basic idea of KARO method is the periodical computation of customer values, storing them in a customer database on the Web server and using them in an automated process of HTTP request service in a Web store [4]. The customer value is determined by using RFM analysis and is calculated according to the following formula:

$$CVS = w_R \times R + w_F \times F + w_M \times M, \qquad (1)$$

where R, F, M are codes of behavioral variables for the customer: recency, frequency and the monetary value, respectively, and w_R, w_F, w_M are weights assigned to the corresponding behavioral variables according to the business strategy of the company.

KARO method extends the Web server system with the ability to give the precedence of service to requests from users who are likely to confirm a purchase and to requests from users who are the most valuable customers. Requests are handled taking into consideration sessions they belong to. Each i-th HTTP request which belongs to the p-th page in user session s is denoted by x^s_{ip} and a corresponding dynamic request is denoted by y^s_{jp}.

Each active user session at the n-th moment is described with a number of attributes, which together determine the importance of the session from the e-business perspective. The attributes of session s at the n-th moment include:

- the session class, $c^s(n)$, which may mean a key customer (KC) or an ordinary customer (OC);
- the session rank, $RFM^s(n)$, reflecting the customer value, which corresponds to key customer's CVS read from database or is zero for an ordinary customer;
- the session state, $e^s(n)$, corresponding to one of the following business operations: entry to the home page (H), logging on (L), browsing (B), searching for products (S), product details (D), adding a product to the shopping cart (A), registration (R), or purchase confirmation (P);
- the financial value of products in the shopping cart, $v^s(n)$;
- the session length, $l^s(n)$, i.e. the number of pages visited in the session until the n-th moment.

The attributes of session s determine the session priority at the n-th moment:

$$P^s(n) = \begin{cases} 4 \text{ for } (c^s(n) = KC) \vee [(c^s(n) = OC) \\ \quad \wedge (v^s(n) > 0) \wedge (e^s(n) = P)], \\ 3 \text{ for } [(c^s(n) = OC) \wedge (v^s(n) > 0) \\ \quad \wedge (e^s(n) \neq P)] \vee [(c^s(n) = OC) \\ \quad \wedge (v^s(n) = 0) \wedge (l^s(n) < T_{Med})], \\ 2 \text{ for } (c^s(n) = OC) \wedge (v^s(n) = 0) \\ \quad \wedge (T_{Med} \leq l^s(n) < T_{Low}), \\ 1 \text{ for } (c^s(n) = OC) \wedge (v^s(n) = 0) \\ \quad \wedge (l^s(n) \geq T_{Low}), \end{cases} \qquad (2)$$

where T_{Med} and T_{Low} ($T_{Med} < T_{Low}$) are predefined thresholds which determine three ranges for the session length. The priority 4 is the highest possible priority value, assigned to most important sessions, and the priority 1 is the lowest possible priority.

To support differentiated QoWS in the system, admission control (AC) under the system overload together with priority-based scheduling in front of the Web server (in queue Q_1) and in front of the

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM

back-end server (in queue Q_2) have been proposed (Fig. 2).

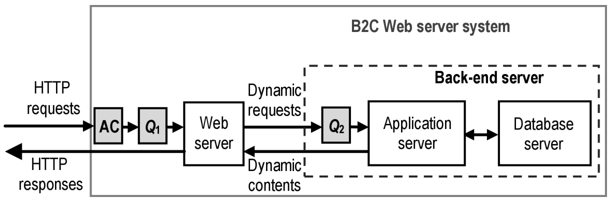

Fig. 2. Multi-layer configuration of a B2C Web site according to KARO method.

The indicator of system load intensity is the length of queue Q_2 at the n-th moment, $L(n)$. Admission control is based on two thresholds for system load intensity, I_1 and I_2 ($I_1 < I_2$). Depending on the current system load and the session priority, some requests may be rejected under overload at the Web server input. On the other hand, priority scheduling in queues Q_1 and Q_2 makes it possible to change the order of request execution at the Web server and the back-end server, respectively. For each queue a strict priority scheduling is applied between four priorities, i.e. all higher-priority requests are queued before the lower-priority ones.

KARO-Rev algorithm

The session priorities and attributes are used in the request admission control and scheduling algorithm KARO-Rev [4].

For each new admitted HTTP request and each new dynamic request, the request position in the corresponding queue (Q_1 for an HTTP request and Q_2 for a dynamic request) is determined. Let a^{s_a} be a request a belonging to session s_a, waiting in a queue Q_k, $k \in \{1, 2\}$. We define the following sets and subsets of requests waiting in the queue Q_k at the n-th moment, when a new request belonging to session s appears:

$$Q_k(n) = \{a^{s_a} \in Q_k\}, \tag{3}$$

$$Q_{k\_2}(n) = \{a^{s_a} \in Q_k : P^{s_a}(n) \in \{2,3,4\}\}, \tag{4}$$

$$\begin{aligned} Q_{k\_3}(n) = \{a^{s_a} \in Q_k : (P^{s_a}(n) = 4) \\ \vee[(P^{s_a}(n) = 3) \wedge (v^{s_a}(n) \geq v^s(n))]\}, \end{aligned} \tag{5}$$

$$\begin{aligned} Q_{k\_4}(n) = \{a^{s_a} \in Q_k : (P^{s_a}(n) = 4)\wedge \\ [(v^{s_a}(n) > v^s(n)) \vee [(v^{s_a}(n) = v^s(n)) \\ \wedge(RFM^{s_a}(n) \geq RFM^s(n))]]\}. \end{aligned} \tag{6}$$

The position number $z_k(n)$ in a queue Q_k, determined for a new request having arrived at the n-th moment, is calculated according to the following formula:

$$z_k(n) = \begin{cases} |Q_k(n)| + 1, & \text{if } P^s(n) = 1, \\ |Q_{k\_2}(n)| + 1, & \text{if } P^s(n) = 2, \\ |Q_{k\_3}(n)| + 1, & \text{if } P^s(n) = 3, \\ |Q_{k\_4}(n)| + 1, & \text{if } P^s(n) = 4, \end{cases} \tag{7}$$

where $k = 1$ for an admitted HTTP request and $k = 2$ for a dynamic request, and $|\bullet|$ means cardinality of queue Q_k at the n-th moment. A pseudocode of KARO-Rev algorithm is presented in Fig. 3.

```
Algorithm KARO-Rev
if (a new HTTP request $x^s_{ip}$ )
  if (the request concerns an HTML file)
    update session attributes: $c^s(n)$, $RFM^s(n)$, $e^s(n)$, $l^s(n)$, $v^s(n)$;
    update session priority $P^s(n)$;
    //admission control
    if [(($I_1$ ≤ system_load < $I_2$)
          and ($P^s(n)$ == 1))
       or ((system_load ≥ $I_2$)
          andD (($P^s(n)$ == 1) or ($P^s(n)$ == 2))]
      reject the request;
    else
      accept the request;
    end if
  else //the request concerns an embedded object
    accept the request;
  end if
  //scheduling in queue $Q_1$
  if (the request has been accepted)
    put the request into the queue $Q_1$ at the position determined
    according to (7);
  end if
else if (a new dynamic request $y^s_{jp}$ )
  //scheduling in queue $Q_2$
  put the request into the queue $Q_2$ at the position determined
  according to (7);
end if
```

Fig. 3. A pseudocode of KARO-Rev algorithm [4].

Simulation environment

The efficiency of a Web server system under KAR-Rev algorithm was verified by using a simulation-based approach. We designed and developed a simulation environment for B2C Web server system and used it to carry out experiments for different values of KARO-Rev parameters.

Our simulation tool consists of a session-based workload generator and a B2C Web server system simulator. The tool, system performance metrics, and a methodology for carrying out the experiments were presented in [8] in detail. The model of a Web server system and the way of request processing at the system resources were discussed in [9]. A detailed performance study of a Web server system under FIFO scheduling was discussed in [10].

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM

In experiments presented in this paper *Typical* workload scenario was applied, emulating the most common workload conditions, when the user page latency limit is equal to 8 seconds and the percentage of key customers in the Web store is equal to 10.

We address the server system performance both in terms of "conventional" system performance metrics and the business-related ones. Performance metrics under consideration in the sensitivity analysis include the 90-percentile of page response time, the number of successfully completed sessions per minute, percentage of aborted *KC* sessions, revenue throughput, and potential revenue lost per minute (in dollars per second).

Simulated time of a single experiment included a 10-hour preliminary phase and the following 3-hour observation phase, during which the system performance was monitored.

Each simulation experiment was run for a constant number of new user sessions arriving at the site per minute (i.e. for the constant session arrival rate). The system performance was then presented as a curve on a graph for the session arrival rate ranging from 20 to 300 with a step of 20.

Parameters of KARO-Rev algorithm

Parameters of KARO-Rev algorithm may be divided into two groups: parameters related to e-business specificity and general parameters of the algorithm.

Business-related parameters

Business-related parameters are connected with RFM analysis and they have no direct impact on the computer system performance. When KARO method is used with a Web server system, each key customer of the Web store is characterized by three values of recency, frequency and monetary codes. Depending on the specificity of business conducted through the Web site, each of these components may be of different importance. It is the marketing strategy of the company which determines the choice of the corresponding weights w_R, w_F, w_M in (1), defining a *CVS* of a key customer. Due to unavailability of appropriate business data from real online retailers, we have assumed that all three weights are equal to 3, which gives *CVS* values ranging from 9 to 45. Such a range is large enough to differentiate between key customers themselves and allows evaluating relative levels of their QoWS without going into business-specific details.

In simulation experiments verifying the efficiency of KARO-Rev algorithm, at the beginning of each experiment a database with a given number of key customers is generated. For each key customer a date of the last purchase, the total number of purchases and the appropriate sum of money spent in a Web store are generated. A five-year history of the customers' purchases in the Web store has been simulated. Numbers of purchases in the Web store vary from 1 to 75 and are generated regardless of the customers' last purchase dates. For each customer record in database, the sum of money spent by a customer is generated depending on the customer's number of purchases, where a product price ranges from $5 to $100.

After creating the customer database, a segmentation of key customer records according to RFM analysis is performed for a given date. In each simulation experiment the customer database and the resulting key customers' RFM scores are exactly the same. Table 1 summarized the business-related parameters.

Table 1
Business-related parameters of KARO-Rev algorithm.

Parameter	Values
Number of key customers in database	5000
Frequency code weight w_F	3
Recency code weight w_R	3
Monetary code weight w_M	3
Range of last customers' purchases [years]	5
Number of customer's purchases	1–75

General parameters

As opposed to the parameters related to RFM analysis, general parameters of KARO-Rev algorithm may directly affect its efficiency and in reality, they should be determined based on the analysis of Web traffic at a given e-commerce Web site. We chose their values based on our simulation model and results of preliminary simulation experiments (Table 2). A sensitivity of KARO-Rev to changes of AC thresholds and session length thresholds is analyzed in the next Section.

Table 2
General parameters of KARO-Rev algorithm.

Category	Symbol	Best value
AC thresholds [number of requests in Q_2]	I_1	30
	I_2	80
Session length thresholds [number of pages visited in the session]	T_{Med}	2
	T_{Low}	20
Queue timeouts [seconds]	T_1	8
	T_2	8

The indicator of system load intensity, $L(n)$, is the length of queue Q_2 at the n-th moment. The admission control thresholds, i.e. parameters I_1 and I_2,

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM

determine two thresholds for the actual system load, used to take decision on the request acceptance or rejection (cf. Fig. 3). The best values of I_1 and I_2, determined based on results of simulation experiments discussed in the next Section, are 30 and 80 dynamic requests in Q_2, respectively. I_1 equal to 30 allows one to admit the adequate number of HTTP requests to protect the system from overload and prevent significant underutilization of its resources. I_2 is set to $I_1 + 50$ to clearly differentiate between QoWS levels for sessions with priorities 1 and 2.

The best values of session length thresholds, T_{Med} and T_{Low}, are equal to 2 and 20, respectively. They have been determined based on a user session model applied in simulation experiments and taking into consideration the way of determining the session priority according to (2). T_{Med} equal to 2 ensures a relatively small number of requests with priority 3 in the queues, and thereby it enables the site to give a relatively fast service to requests with priority 3 from unlogged key customers. Since the average length of an *OC* session is about 18 pages, a good value of T_{Low} is 20. As a result, priorities of *OC* sessions with empty shopping carts will be lowered from 2 to 1 only when the session length achieves 20, i.e. for really long *OC* sessions (probably generated by Web bots).

Values of queue timeouts, T_1 and T_2, after which a request will be dropped from queues Q_1 and Q_2, respectively, are both equal to 8 seconds. According to 8-second rule [11] this is the maximum latency tolerated by most Internet users.

Analysis of KARO-Rev parameters

In this section, the sensitivity of KARO-Rev algorithm to changes in its basic parameters is analyzed. In particular, the impact of different values of AC thresholds, I_1 and I_2, as well as session length thresholds, T_{Med} and T_{Low}, on business-oriented system performance metrics was analyzed.

Parameters I_1 and I_2 determine two thresholds for the length of the queue in front of the back-end server, $L(n)$, which is an indicator of the Web server system load intensity. Thus, their values affect a decision on HTTP request acceptance or rejection. Depending on these two parameters, the system load in a simulation experiment will be of different intensity. If these values are small, relatively many requests will be rejected at the Web server input. Consequently, the system will be well protected against overload, but its resources may be underutilized. On the other hand, if these values are high, much more requests will be accepted and then will compete for the use of system resources. This may lead to longer page response times and a bigger number of aborted sessions due to exceeding the user page latency limit or the timeout defined for the queue in front of the back-end server, T_2. In order to study the sensitivity of KARO-Rev to AC thresholds, the influence of their various values on system performance is analyzed.

In all experiments in this group, values of the session length thresholds, T_{Med} and T_{Low} have been equal to 2 and 20, respectively. The experiments have been run for workload scenario *Typical*, for various combinations of the AC thresholds. After running preliminary experiments evaluating a relation between the mean number of dynamic requests in the back-end server and the mean number of aborted sessions in one second intervals, potential values of I_1 have been chosen. They varied from 20 (when some number of aborted sessions has just been observed) to 50, with a step of 10. Results for I_1 equal to 100 are presented as well, in order to evaluate the KARO-Rev system performance for extremely high AC thresholds and thereby under a relatively high system load. In all cases, I_2 has been equal to I_1 +50, to clearly differentiate QoWS levels for sessions with priorities 1 and 2. Values of parameters I_1 and I_2 in each experiment are denoted as a pair (I_1, I_2).

Figure 4 presents the 90-percentile of page response time as a function of the session arrival rate. As it can be seen, values of AC thresholds do have influence on QoWS measures connected with page response times provided by the system. The lower the AC thresholds, the lower the 90-percentile of page response time, because the overall system load is lower, mean lengths of the system queues are lower, and request waiting times are better.

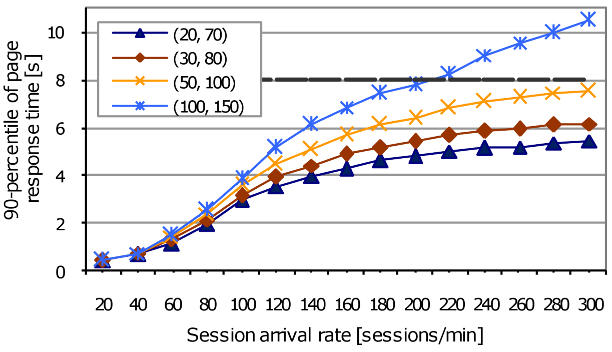

Fig. 4. 90-percentile of page response time for different AC thresholds.

The difference in the 90-percentile of page response time starts at the session arrival rate of 40 sessions/min and evidently increases with the increase in the system load. For the maximum load, 90% of Web pages have been processed within 5.4 seconds

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM

for the lowest AC thresholds and within 10.4 seconds for the highest ones. The relation between the AC thresholds and page response times has been observed for the whole range of load intensity. However, it does not translate itself directly to other system performance measures, such as the number of successfully completed sessions per minute. As it can be seen in Fig. 5, lower AC thresholds lead to higher system throughput for very heavy loads (above the session arrival rate of 100 sessions/min), and to lower throughput for medium loads (for the session arrival rates of 40–80 sessions/min). It means that unless the system is heavily loaded, for low AC thresholds too many requests have been rejected at the system input, while there had been enough resources to effectively process some of them.

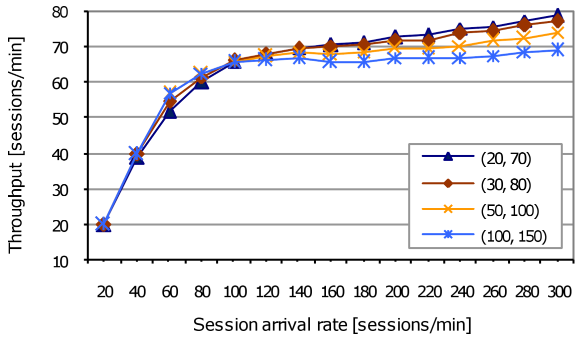

Fig. 5. Number of completed sessions per minute for different AC thresholds.

It is confirmed in Fig. 6, presenting a percentage session breakdown for different AC thresholds in two cases: for the session arrival rate of 80 and 300 sessions/min. Both for the moderately and heavily loaded system, the smaller number of sessions aborted due to admission control implies the bigger number of sessions aborted due to two other reasons: if request waiting times in the back-end server queue has exceeded timeout T_2 or if users had been waiting too long for responses and finally gave up. However, in both cases the impact of the AC thresholds on the number of successfully completed sessions is different.

Because the focus of our research is on system performance in terms of business-oriented metrics, we have examined which sessions had been aborted in the experiments for different AC thresholds, apart from why or where they had been aborted. The experiments demonstrated that values of the AC thresholds have no impact on the percentage of successfully completed *KC* sessions, which has ranged from 99.93 to 100% in all cases. However, they have impact on the amount of achieved revenue (Fig. 7), affecting the revenue throughput in a similar way as the system throughput in completed sessions but to a significantly lower degree. As one can observe in Fig. 7, under very heavy load smaller values of the AC thresholds generally provide higher revenue throughput. However, under medium load, the least satisfactory results have been achieved for the case (20, 70), although for the case (30, 80) they have been very similar to the case (100, 150). In all cases, the KARO-Rev algorithm has been able to effectively cope with losses of potential revenue (Fig. 8) and to provide 99.3–100% of achieved potential revenue.

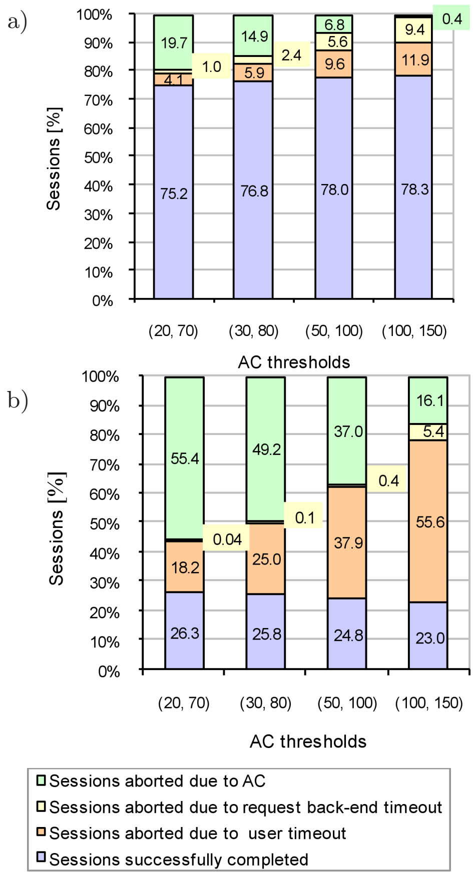

Fig. 6. Percentage session breakdown for different AC thresholds a) session arrival rate of 80 sessions/min, b) session arrival rate of 300 sessions/min.

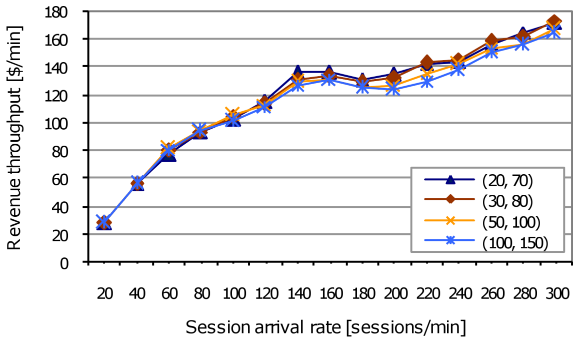

Fig. 7. Revenue throughput for different AC thresholds.

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM

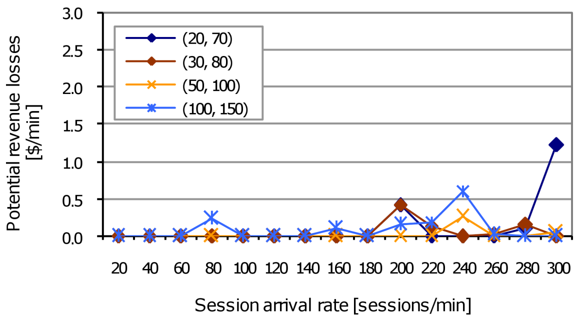

Fig. 8. Potential revenue lost per minute for different AC thresholds.

To sum up this part of analysis, the results have shown that KARO-Rev is sensitive to AC thresholds values with regard to both conventional and business-oriented performance metrics. The results suggest it would be worth proposing a dynamic way of adapting values of I_1 and I_2 to current system load. We leave this issue to future work. The results show that the values I_1 of 30 and I_2 of 80 are most advantageous for KARO-Rev algorithm.

Session length thresholds

Parameters T_{Med} and T_{Low} determine two thresholds for the session length (i.e. the number of Web pages visited in a session), at which the priority of an OC session with an empty shopping cart will be lowered from 3 to 2 and from 2 to 1, respectively, according to (2). Values of these thresholds influence especially the number of OC requests with 3 and 2 priorities in the system. Thus, they also affect response times of requests from unlogged key customers, and may affect the amount of achieved revenue as well. That is why the sensitivity of KARO-Rev to the session length thresholds is evaluated.

Experiments in this group were performed for the Web server system under control of KARO-Rev algorithm for *Typical* workload scenario and for various combinations of the session length thresholds. In all the experiments, I_1 and I_2 have been equal to 30 and 80, respectively. Values of the session length thresholds, T_{Med} and T_{Low} in each experiment are denoted as a pair (T_{Med}, T_{Low}).

First, the impact of the first session length threshold, T_{Med}, on business-oriented system performance metrics has been analyzed. Corresponding experiments have been run for different values of T_{Med} and a fixed value of T_{Low} equal to 20, which is a little bit more than the average length of an OC session.

It has to be noticed, that all new sessions receive priority 3 – when users enter the home page of the site – and at that time key customers are not logged on yet. That is why requests with priority 3 from unlogged key customers and from ordinary customers with empty shopping carts are treated exactly in the same way – they are queued after requests with priority 3 from ordinary customers with not empty shopping carts. For that reason, all requests with priority 3 should be processed relatively quickly, to enable key customers to get service in reasonable time and log into the site. The smaller the value of T_{Med} is, the less OC sessions with priority 3 there are, and thereby better service is experienced by all requests with priority 3.

In the applied workload model, all key customers log on just after entering the site, i.e. when the session length is equal to two. That is why we set the initial value of T_{Med} of 2 in the experiments. In this case, OC sessions will be moved to priority 2 very quickly, and the number of requests with priority 3 in both queues, Q_1 and Q_2, will be relatively small. In successive experiments, the value of T_{Med} has been augmented by 2, up to 8.

Figures 9 and 10 plot the amount of revenue achieved and lost per minute, respectively, for different values of T_{Med} as a function of the session arrival rate. As it can be seen, for the extremely heavy loaded Web server system (corresponding to the session arrival rate of above 200 sessions/min), bigger values of T_{Med} imply lower revenue throughput, while potential revenue lost per minute is not clearly dependent on T_{Med}. For the maximum offered load, the difference in the revenue throughput for two extreme cases has been about \$15/min.

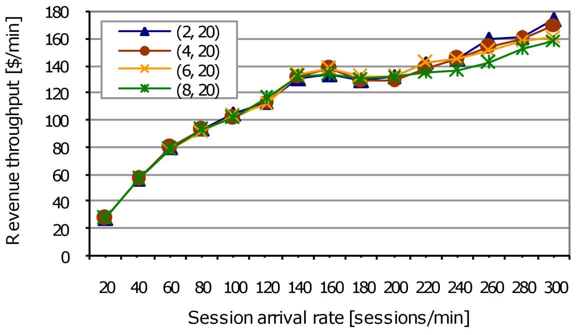

Fig. 9. Revenue throughput for different values of T_{Med}.

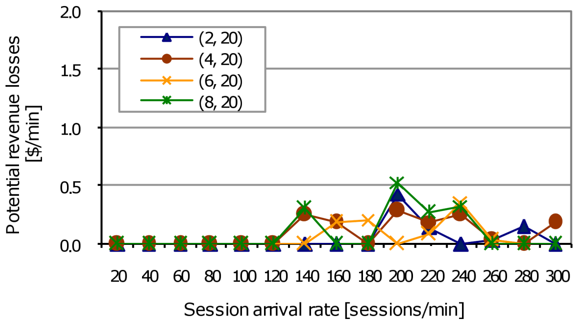

Fig. 10. Potential revenue lost per minute for different values of T_{Med}.

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM

As it can be seen in Fig. 11 and Fig. 12, lower revenue throughput in the case of higher T_{Med} has been caused mainly by the significantly more KC sessions aborted due to long page response times, either due to the back-end server queue timeout or due to the user patience timeout. The bigger T_{Med}, the more requests with priority 3 from OC sessions and thus, the higher page response times for sessions with priority 3 (including KC sessions at the earliest stage). Consequently, more KC sessions were aborted and less key customers had a chance of logging into the site and being promoted to priority 4. As a result, adequately less products could be added to KC shopping carts, and smaller amount of potential revenue could be turned into the achieved revenue. As it can be seen in Fig. 11, for the maximum offered load, the percentage of aborted KC sessions has varied from 0.02 for $T_{Med} = 2$ to 13.35 for $T_{Med} = 8$. On the contrary, the percentage of aborted OC sessions has appeared to be independent of the first session length threshold and in all cases has been about 84% (not shown).

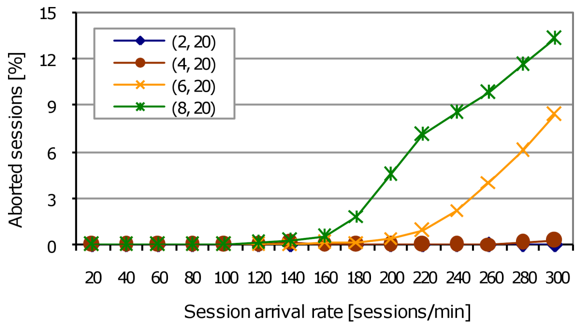

Fig. 11. Percentage of aborted KC sessions for different values of T_{Med}.

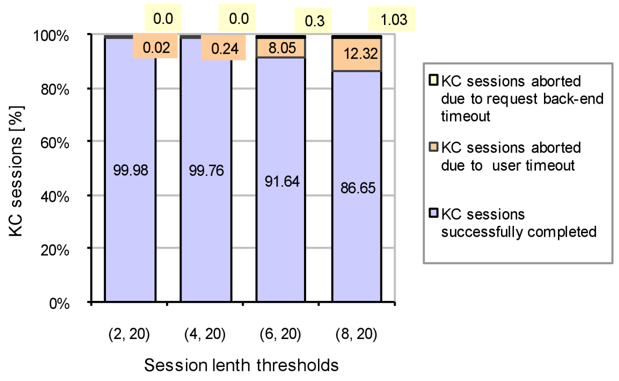

Fig. 12. Percentage KC session breakdown for different values of T_{Med}, for the session arrival rate of 300 sessions/min.

After evaluating the impact of the first session length threshold on the system performance in terms of business-oriented performance metrics, similar experiments for the second session length threshold have been performed. The experiments have been run for different values of T_{Low} and a fixed value of T_{Med} equal to 2.

Figures 13 and 14 show that the amount of revenue achieved and lost per minute do not depend clearly on the value of T_{Low}. There have been small differences between the corresponding metrics for individual session arrival rates, but one cannot observe their dependence on T_{Low}. Similarly, there has been no relation between T_{Low} and the percentage of successfully completed KC sessions, which has been rather stable and has varied from 99.95 to 100. The value of T_{Low} has only a little impact on the percentage of successfully completed OC sessions (not shown). For the session arrival rates of 60–120 sessions/min, smaller values of T_{Low} imply a bit higher values of this percentage, but the difference does not exceed 3%.

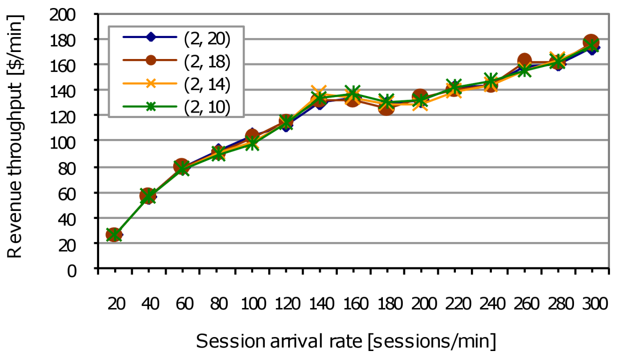

Fig. 13. Revenue throughput for different values of T_{Low}.

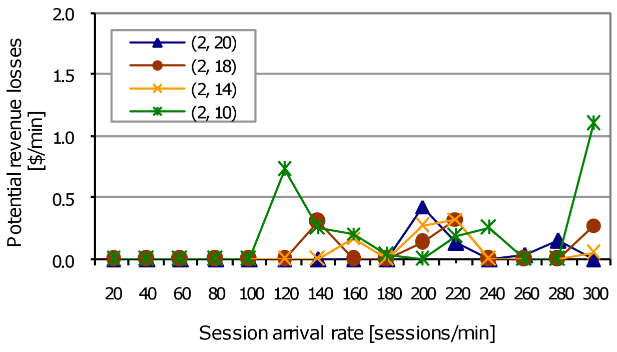

Fig. 14. Potential revenue lost per minute for different values of T_{Low}.

To sum up this part of analysis, KARO-Rev has turned out to be insensitive to the value of the second session length threshold, T_{Low}, in terms of business-oriented system performance metrics. However, the algorithm is sensitive to the value of the first session length threshold, T_{Med}, especially under system overload. For the applied model of a KC session at the B2C site, T_{Med} equal to 2 has led the system

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM

to achieve the highest revenue and to offer the best QoWS for key customers. Therefore, we can say that values T_{Med} of 2 and T_{Low} of 20 are most advantageous for KARO-Rev.

In reality, the choice of T_{Med} should be made based on the analysis of key customers' navigational patterns at a given B2C Web site. If most of key customers log on at the very beginning of their sessions (like in our workload model), a very low value of T_{Med} will be a good choice. However, if most of them log on later, higher values of this threshold would probably be better and would prevent the system from lowering the priorities of unlogged key customers' sessions too early.

Conclusions

Simulation results discussed in the paper have shown that the proper adjustment of KARO-Rev parameters may be significant with regard to business-oriented and conventional performance metrics. The results suggest that applying the algorithm in a real-life B2C Web site should be preceded by the analysis of key customers' navigational patterns at the B2C Web site to tune the session length thresholds, T_{Med} and T_{Low}, to turn more visitors into customers. Moreover, it seems to be worth working out a dynamic way of adapting values of the AC thresholds, I_1 and I_2, to the current load of the Web site.

References

[1] Ataullah A., *MyQoS: a profit oriented framework for exploiting customer behavior in online e-commerce environments*, Proc. of WISE'07, Nancy, France, pp. 533–542, 2007.

[2] Poggi N., Moreno T., Berral J.L., Gavald R., Torres J., *Web customer modeling for automated session prioritization on high traffic sites*, Proc. of UM'07, pp. 450–454, 2007.

[3] Shaaban Y.A., Hillston J., *Cost-based admission control for Internet commerce QoS enhancement*, Electron. Commerce. Res. Appl., 8, 142–159, 2009.

[4] Suchacka G., Borzemski L., *Integrating a key customer-oriented strategy into the B2C e-commerce service*, Management and Production Engineering Review, 1, 4, 67–76, 2010.

[5] Totok A., Karamcheti V., *RDRP: reward-driven request prioritization for e-commerce Web sites*, Electron. Commerce. Res. Appl., 9, 549–561, 2010.

[6] Yue C., Wang H., *Profit-aware overload protection in e-commerce Web sites*, Journal of Network and Computer Applications, 32, 2, 347–356, 2009.

[7] Zhou X., Wei J., Xu C.-Z., *Resource allocation for session-based two-dimensional service differentiation on e-commerce servers*, IEEE Trans. Parallel Distrib. Syst., 17(8), 838–850, 2006.

[8] Suchacka G., Borzemski L., *Simulation-based performance study of e-commerce Web server system – methodology and metrics*, Information Systems Architecture and Technology – Web Information Systems Engineering, Knowledge Discovery and Hybrid Computing, ISBN 978-83-7493-630-9, Wrocław, pp. 25–35, 2011.

[9] Borzemski L., Suchacka G., *Business-oriented admission control and request scheduling for e-commerce websites*, Cybernet. Syst., Taylor & Francis, 41, 8, 592–609, 2010.

[10] Suchacka G., Borzemski L., *Simulation-based performance study of e-commerce Web server system – results for FIFO scheduling*, Multimedia and Internet Systems: Theory and Practice, AISC, chapter 24, vol. 183, Springer, Berlin Heidelberg, pp. 249–259, 2013.

[11] *8 Seconds to Capture Attention*, Silverpop's Landing Page Report, 2007, http://www.silverpop.com/practices/studies/landing_page (access date: March 15, 2008).

Unauthenticated | 89.67.242.59
Download Date | 6/2/13 7:38 PM