Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 50 (2006) 3763–3783
Computer Networks
www.elsevier.com/locate/comnet

Session based access control in geographically replicated Internet services

Novella Bartolini *

Computer Science Department, University of Rome "La Sapienza", Via Salaria 113, 00198 Rome, Italy

Received 18 June 2005; received in revised form 12 March 2006; accepted 27 March 2006
Available online 17 May 2006

Responsible Editor: J. Misic

Abstract

Performance critical services over Internet often rely on geographically distributed architectures of replicated servers. Content Delivery Networks (CDN) are a typical example where service is based on a distributed architecture of replica servers to guarantee resource availability and proximity to final users. In such distributed systems, network links are not dedicated, and may be subject to external traffic. This brings up the need to develop access control policies that adapt to network load changing conditions. Further, Internet services are mainly session based, thus an access control support must take into account a proper differentiation of requests and perform session based decisions while considering the dynamic availability of resources due to external traffic.

In this paper we introduce a distributed architecture with access control capabilities at session aware access points. We consider two types of services characterized by different patterns of resource consumption and priorities. We formulate a Markov Modulated Poisson Decision Process for access control that captures the heterogeneity of multimedia services and the variable availability of resources due to external traffic. The proposed model is optimized by means of stochastic analysis, showing the impact of external traffic on service quality. The structural properties of the optimal solutions are studied and considered as the basis for the formulation of heuristics. The performance of the proposed heuristics is studied by means of simulations, showing that in some typical scenario they perform close to the optimum.

Keywords: Content Delivery Networks; QoS; Session based access control

1. Introduction

Geographical replication of resources is at the basis of several performance critical services over the Internet. Content Delivery Networks (CDN) [1] are based on a placement of server replicas and on mechanisms for request redirection that guarantee resource availability, service quality and proximity of content to the user, together with an efficient and content aware routing. A proper placement of replica servers shortens the path from servers to clients thus hedging the risk of encountering bottlenecks in the non-dedicated environment of the Internet.

* Tel.: +39 06 49918357; fax: +39 06 854 1842.
E-mail address: novella@di.uniroma1.it

1389-1286/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2006.03.007

Flash crowds and unpredictable link congestions could cause a critical performance degradation of some servers leaving few resources available to grant service availability and continuity. A simple scheme of replica placement and request redirection may not be sufficient to solve this problem. Options include increasing the capacity of servers and of network links and replicating more servers. However in many circumstances it is impossible to estimate the amount of resources required to fulfill all requests. In such a finite server capacity scenario, service-level agreements (SLA) specifying quality of service (QoS) probabilistic guarantees must be in place and penalties should be imposed when requests are not served or are served in violation of the agreements on quality.

Our study is valid for many types of service, but to make the analysis more concrete, we study the workload of two typical web services: informational web access and e-commerce requests and transactions. See [2] for a survey on types of services typically supported by CDN and more generally by geographically distributed replicated architectures. The list include informational services and e-commerce services among the others. We refer to [3] for the description of replicated architecture supporting dynamic caching for e-commerce sites.

Typical Internet services are based on the concept of *session* [4–6]. A session is a sequence of temporally and logically related requests issued by the same client. We believe that an access control mechanism that gives priority to requests belonging to an already active service session, and to critical phases of each session, could improve the client perceived performance and the profits of the service providers. The session concept must be at the basis of any access control mechanism in Web and more generally Internet services, as pointed out in [4,6]. If a service session has been started, all its composing requests should also be admitted, especially during critical phases in which more revenue could be gained or lost.

Aim of this paper is to investigate the performance of session based access control policies in a non-dedicated network environment, in presence of external traffic and possible congestions.

The proposed policies can be made available at any appliance that performs request interception and redirection over a distributed architecture. From now on, we will refer to this appliance with the name of *access point*.

Service sessions can be modelled as a sequence of service phases alternated to think phases. The sequence of phases traversed by the session during its lifetime strictly depends on the particular application. We formulate models of service sessions and underline the existence of critical phases that should not be interrupted in order not to lose profits and incur penalties.

We propose a stochastic decision model to optimize some performance parameters such as the probability of ongoing *session disruption* due to lack of resources, the probability of successful *session completion*, i.e. the probability that a session is terminated due to the client's will, and the *service refusal* probability that is the probability that a request is blocked by the admission control mechanism at the first access attempt.

The proposed decision model is based on a Markov Modulated Poisson Process (MMPP) [7] of service sessions that captures the dynamic nature of external traffic [8,9], and the phase model typical of multimedia network applications accessible through the Internet.

External traffic on the non-dedicated links is modelled by a Markov modulated vacation (or ON/OFF) process.

Through the proposed model, the optimal policy can be computed by means of common methods of operations research such as the value iteration algorithm [10,11].

The analysis of the optimal policy shows the impact of external traffic on service quality and how the access controller can adapt its decision to the external traffic scenario.

A structural analysis of the optimal solution is conducted to suggest a choice of possible heuristics to be applied by the access routers. Performance comparisons between the heuristics and the optimal policies are also given by means of simulations. Application traffic is simulated by means of a synthetic traffic generator that follows the proposed model of session lifetime. The use of a synthetic traffic generator is justified by the immense variability of application session model, that makes impossible the use of real traces to model sufficiently general situations.

The paper is organized as follows. In Section 2, we introduce the state of the art of admission control in web systems and in particular in CDN. In Section 3, we give some details on the reference architecture to enable access control to replicated servers. In Section 4, we introduce two typical CDN types of service and their related Markov modulated phase model: a purely informational web service and an e-commerce service. In Section 5, a congestion model is introduced to take into account the presence of external traffic on the non-dedicated link between routers and replica servers. In Section 6 we introduce a Markov Modulated Poisson Decision model and related revenue optimization problems. In Section 7, we conduct a structural analysis of the optimal admission control policy and propose some heuristics. In Section 8, performance comparisons among optimal policies and heuristics are given, while Section 9 concludes the paper.

2. Related work

Though much work is in progress on improving the quality of Internet replicated services through mechanisms for replica placement, server measurement and requests redirection to the best suited replica server, few works analyze the impact of the external traffic and the possibility to adopt session based admission control at the access routers.

Some works consider the session based access control problem in a single server architecture, as in [6], where a decision model is proposed to perform a non-linear optimization of the provider's revenue and the admission control policy selects which new sessions should be served based on a generalized processor sharing discipline. The authors assume that requests belonging to the same session can be clustered into stages and propose an example of session lifetime for a commercial site. In [12] a single server architecture is also considered in which high priority requests are admitted based on the service time estimation and on the evaluation of the available server capacity, while low priority requests are restricted during high load periods to avoid service block or interruption to high priority requests. In [4] a session based admission control is proposed to prevent a single web server from becoming overloaded showing that overloaded servers compromise the performance of longer sessions that, in commercial web sites, are the ones that most likely result in purchases. A periodical evaluation of the server utilization is performed for predictive purposes to implement the proposed admission policy. When the predicted utilization exceeds a given threshold, only requests belonging to already admitted sessions are accepted, while new session requests are discarded. Once the predicted utilization drops below the given threshold, the server admits new sessions again.

All the works cited above consider the server capacity and utilization while performing the admission decisions but, differently from our model, do not take into account the presence of external traffic on the non-dedicated links of the CDN.

In [13] a single server architecture is considered, where the congestion on the links between the client and the server is also taken into account by the admission control policy. The authors show that servers may be slowed down due to resource limitations or to congestions on the response network path. An application level mechanism is introduced to mitigate flash crowds. The access control is performed by access routers that evaluate the web request rate and response latency, and adjust the rate of accepted requests to the target server accordingly to the measured performance.

Differently from our proposals, the schemes above only consider single web servers, which means that the suggested control schemes may not be suitable for a distributed set of servers.

In [14] a distributed web site is considered where a dispatcher performs access control based on short-time prediction of request traffic and service time. The dispatcher distributes the admitted requests among the available servers. Two admission schemes are considered for a web store, one of them based on the session concept, while the other is only based on single requests. Simulations are used to compare the performance of the two policies. The proposed scheme only considers dedicated link in a locally distributed site and therefore cannot be applied to a geographically distributed environment such as the one involved in content delivery architectures, in which the unpredictable traffic on links critically affects the response probability and the response time of the servers. In [15] a prediction of the service time requirements of an incoming request is considered while taking the admission control decision with the purpose of maximizing the provider's income. Penalties are introduced for blocked requests and for requests being served with degraded performance compared with the SLA. The consideration of the estimated required service time allows a time based scheduling of requests. It is shown that the shortest remaining job first (SRJF) policy minimizes the penalties, while a modification of SRJF allows the provider's profit maximization. The problem is formulated defining a combined measure of resource availability (CPU, bandwidth) in a distributed environment without external sources of congestion such as the one also introduced in [14].

Unlike our work, the cited schemes cannot be applied to a CDN environment in which servers are replicated and geographically distributed. In such a scenario, the pool of available servers is most of the time only a subset of the server pool known by the access routers. The absence of some servers can be due to their impossibility to fulfill the SLA due to overload or to congestion on the non-dedicated response path. As in [6,4] we introduce a session model of the lifetime of an accepted request. Our work, starting with [16], differs from the previously cited works because our admission policy is based on the evaluation of the status of the ongoing sessions and of the impact of congestion on the service time of requests.

3. Reference architectures: geographically distributed services and Content Delivery Networks

The reference scenario of our work is one with distributed and replicated servers such as a CDN.

The proposed model can be adopted by any appliance that intercepts requests and is responsible to redirect them to the best suited replicated server.

A discussion on the possible architectural choices to perform server selection and request redirection is out of the scope of this paper. We refer to [2] for a short survey on this topic. We limit our discussion to the consideration that while in several commercial implementations of CDN, request redirection is performed at the DNS level, DNS authoritative servers are not the best place to enable access control policies and performance based request redirection, due to caching of address resolution. Many DNS implementation in fact do not honor the DNS TTL field and cache some name-to-address resolution for an unpredictably long time, impeding a fast reaction of the system to sudden changes of workload and network traffic. Further DNS request-routing does not take into account the IP address of the clients. Only the Internet location of the client DNS server is known: this limits the ability of the request-router and access controller to determine a client's proximity to the replica server and to predict the client perceived performance.

We consider a CDN architecture in which admission control operations are performed by access routers (see Fig. 1) or by application level dispatchers. This CDN architecture was inspired by [17] and similar schemes are those described in [18–20] just to give some examples.

The access points (routers or dispatchers) may collect statistics about replica servers by means of active and passive measurements, or receive status update messages from the available servers, create user and session profiles, perform replica server selection mechanisms and access control to guarantee the required performance to all types of services, with prioritization of sessions and of requests belonging to critical session phases.

In this paper we focus on the access control capabilities of these systems.

4. Session models of CDN services

We assume a Markov modulation of the session lifetime through service dependent phases. As in [5,6] we introduce a phase classification and the

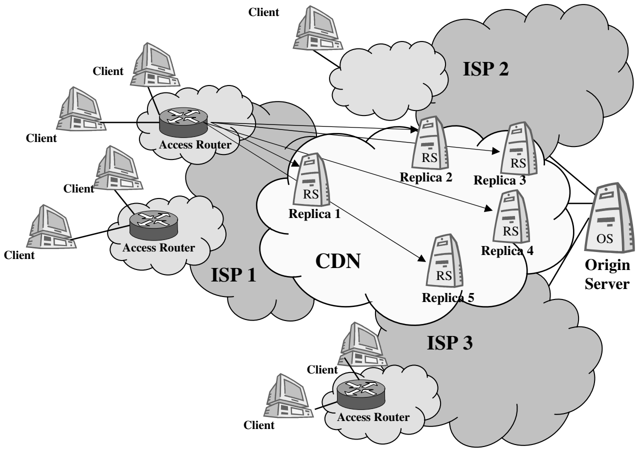


Fig. 1. Content Delivery Network architecture enhanced with access control capability.

probabilities of having phase transitions during a service session. We differentiate between two classes of clients *premium* and *basic*, such that the premium clients receive better service than basic clients in case of high workload. As an example of premium class service we introduce an e-commerce service based on a web site where pages are generated dynamically, on the basis of request parameters and on the basis of currently available service information. On the contrary, a service of the basic class can be any typical informational service based on static web pages. More types of service can be considered such as streaming or interactive games and others, provided that a proper session model is formulated, given a site trace.

We assume a full replication of content among the replica servers. The access routers periodically collect statistics regarding the servers performance. Under the assumption of full replication any request redirection is only based on quality of service considerations. In [5] the concept of session is introduced, differentiating among *compulsory sessions*, where the sequence of requests is automatically issued by the browser itself such as in the case of a download of a static page with embedded objects, and *voluntary sessions* where the requests belonging to the same session are generated by the client. In the present work we focus on the second type of session, while the first can also be modelled using the same methodology. We introduce a session model for each considered type of service. The rationale behind the study of the session life cycle is in the necessity to focus on the critical phase transitions that more likely produce incomes for the providers. The session life cycle model is also useful for tuning the parameters of the admission control mechanism to minimize the probability of blocking sessions at critical phases.

4.1. Session model for informational web service

An *informational web service* is generally realized through a web site that is only constituted by static pages. A session of this type of service will consist of few phases possibly traversed many times. The session starts with a browsing phase A that most of the time is followed by a compulsory request of embedded objects, alternated with a *think phase* B as shown in Fig. 2. During think phases the user spends time thinking before deciding which next request to issue and does not consume resources with the exception of session identifiers. The two phases related to the html static page request (browsing) and to the download of its embedded objects are aggregated since we want to focus only on voluntary phase transitions.

The index 1 associated to the transition probabilities in Fig. 2 is introduced to differentiate the informational web type of service from the second one, e-commerce, that will be introduced in Section 4.2. In our model we assume exponential arrivals with average rate λ_A^1. When a request arrives, it enters the phase A in which an http request is issued to one of the servers, selected by the access router. Phase A requests are characterized by a resource consumption b_A^1 of 1 unit, $b_A^1 = 1$, and the residence time is exponentially distributed with average μ_A^1. When the client gets the response, the request enters phase B in which the user thinks about the next request to issue. This phase is introduced to keep track of ongoing sessions although inactive (think times), and is characterized by null resource utilization: $b_B^1 = 0$, with the exception of a session identifier, while the average residence time is $\mu_B^1 = \mu_{\mathtt{think}}$. Both in phase A and in phase B there is a probability that the client voluntary terminates the session. To represent phase transitions and voluntary terminations, we introduce the transition probabilities between phases: π_{AB}^1, that is the probability of having a transition from phase A to phase B, and the opposite tran- sition π_{BA}^1. The definition of these transition probabilities introduces Poissonian events of phase transitions and session completions. A phase A request enters phase B with exponential distribution and average rate $\mu_A^1 \cdot \pi_{AB}^1$. A phase A

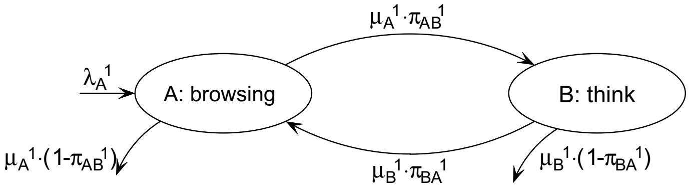


Fig. 2. Session model of the informational web type of service.

request is willingly terminated by the client with exponential distribution and average rate $\mu_A^1 \cdot (1-\pi_{AB}^1)$. New session arrivals only occur in phase A, meaning that a new informational web session is only started when a related http request is issued. Real traces of an informational site can be used to tune the values of the involved parameters. Since the particular choice of the site trace is not particularly meaningful for our purposes, we consider average values of different traces:

$$\lambda_A^1 = 50\ \mathrm{s}^{-1}, \quad \mu_A^1 = 100\ \mathrm{s}^{-1}, \quad \mu_B^1 = \mu_{\mathtt{think}} = 0.05\ \mathrm{s}^{-1},$$
$$\pi_{AB}^1 = 0.95 \quad \text{and} \quad \pi_{BA}^1 = 0.6.$$

4.2. A session model for e-commerce service

In [21] the authors point out that e-commerce traffic often exhibits short-time fluctuations and presents a very high variability. This type of service is very critical in most of its phases. Users get frustrated by low performances and they may decide to abandon a site if a high latency is encountered, causing revenue to be lost. It is very important for an e-commerce site to be able to self-configure the admission control policy to cope with short-term fluctuations in the workload and to meet the desired QoS levels. During an e-commerce session, many phases can be considered in which dynamic http requests are issued. The fulfillment of an e-commerce request usually requires the execution of database queries, the production of dynamic html pages, together with the creation of an SSL-secured process for economic transactions. E-commerce services are typically more resource-intensive than informational web service. This is modelled by means of longer phase times. Each request belonging to an e-commerce site will require only one server unit, but its processing time are typically much longer than with informational web services.

Fig. 3 shows the e-commerce session phase model.

As seen in [5,6] the e-commerce session starts in a browsing phase A in which the client issues an http request to enter the site. If the client is interested in a particular product, after some thinking time in phase B, it enters the search request phase C, in which a query is submitted to the e-commerce database and a dynamic html page is produced with the related results. After another phase of thinking, represented by phase D, the client may decide to issue a new query to the database, going back to phase C, or to put some products in the trolley, entering phase E. Once entered phase E the session is considered very critical because its interruption before getting to phase F, due to congestion or overload, potentially causes a profit loss. In all phases the client may willingly terminate the session. As seen in Section 4.1, the lifetime of the session is Markov modulated among the described phases. Phases B and D represent think phases with null resource consumption, that is $b_B^2 = b_D^2 = 0$, where the session in the other phases consumes a single resource unit, that is $b_A^2 = b_C^2 = b_E^2 = b_F^2 = 1$. Realistic values of the parameters can be considered as follows: $\lambda_A^2 = 10\ \mathrm{s}^{-1}$, $\mu_A^2 = 100\ \mathrm{s}^{-1}$, $\mu_B^2 = \mu_D^2 = \mu_{\mathtt{think}} = 0.05\ \mathrm{s}^{-1}$, $\mu_C^2 = 0.333\ \mathrm{s}^{-1}$, $\mu_E^2 = 1\ \mathrm{s}^{-1}$ and $\mu_F^2 = 0.2\ \mathrm{s}^{-1}$. Table 1 shows the transition probabilities between phases. As for the model introduced in Section 4.1, traces taken from real sites can also be used.

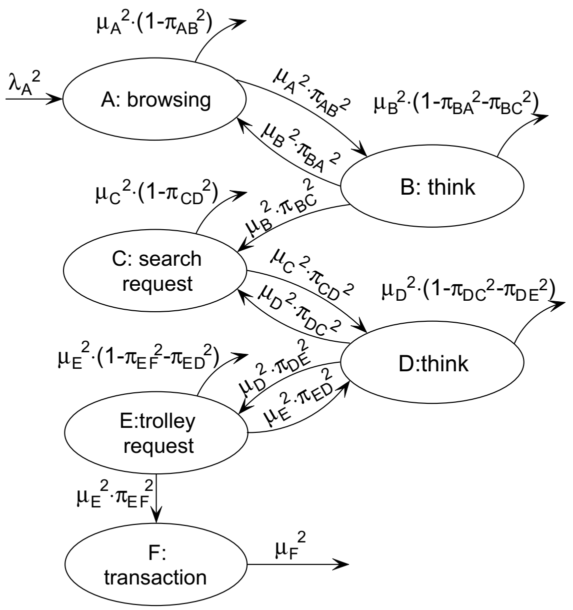

$\mu_A{}^2 \cdot (1-\pi_{AB}{}^2)$
$\lambda_A{}^2$
A: browsing
$\mu_A{}^2 \cdot \pi_{AB}{}^2$
$\mu_B{}^2 \cdot (1-\pi_{BA}{}^2-\pi_{BC}{}^2)$
$\mu_B{}^2 \cdot \pi_{BA}{}^2$
B: think
$\mu_C{}^2 \cdot (1-\pi_{CD}{}^2)$
$\mu_B{}^2 \cdot \pi_{BC}{}^2$
C: search request
$\mu_C{}^2 \cdot \pi_{CD}{}^2$
$\mu_D{}^2 \cdot \pi_{DC}{}^2$
$\mu_D{}^2 \cdot (1-\pi_{DC}{}^2-\pi_{DE}{}^2)$
D:think
$\mu_E{}^2 \cdot (1-\pi_{EF}{}^2-\pi_{ED}{}^2)$
$\mu_D{}^2 \cdot \pi_{DE}{}^2$
$\mu_E{}^2 \cdot \pi_{ED}{}^2$
E:trolley request
$\mu_E{}^2 \cdot \pi_{EF}{}^2$
F: transaction
$\mu_F{}^2$

Fig. 3. Session model of the e-commerce type of service.

Table 1
Phase transition probabilities for an e-commerce site

π_{**}	A	B	C	D	E	F	Exit
A	0	0.8	0	0	0	0	0.2
B	0.3	0	0.6	0	0	0	0.1
C	0	0	0	0.5	0	0	0.5
D	0	0	0.2	0	0.5	0	0.3
E	0	0	0	0.5	0	0.3	0.2
F	0	0	0	0	0	0	1

5. External traffic model

An active/passive measurement support is at the basis of performance based request redirection, to enable the access routers to select the best suited replica among the set of replica servers they have knowledge about [22,17,23]. External traffic on the non-dedicated links between access routers and replica servers has an impact on the server availability, making sometimes impossible even the use of uncongested servers. Though an access router has knowledge of a set of fully replicated servers for a total of C available servers, every time a request arrives, it selects the best suited replica from a restricted *available pool* that is the set of servers that have enough free capacity to fulfill the request within the agreed QoS service levels.

In our model we neglect the effects of external traffic on the links between the clients and the access routers. We only consider external traffic on the non-dedicated links between the access routers and the replica servers.

The available pool is formed by the replica servers that are not loaded at their maximum capacity and can be reached through a link that guarantees the fulfillment of the agreements on service quality. A link is reputed to be under congestion effect if its expected round trip time is increased by an intolerable latency. The response time of a server at the end of a congested link $T_{\mathtt{congested}}$ is thus greater than the response time measured in non-congested situations $T_{\mathtt{non\_congested}}$. Replica servers that, whatever route is considered, cannot be reached without an intolerable level of congestion are dynamically removed from the available pool.

We repute that congestion level is intolerable when the observed response time is increased by more than a fixed percentage value $\alpha_{\mathtt{cong}}$, that is when the observed response time is $T_{\mathtt{congested}}$, where

$$T_{\mathtt{congested}} \geqslant T_{\mathtt{non\_congested}} \cdot (1 + \alpha_{\mathtt{cong}}), \tag{1}$$

where the value of $\alpha_{\mathtt{cong}}$ is selected according to the required QoS levels.

Congested servers will be kept out of the available pool until the measured state of the link goes back to normal conditions. The active sessions that are being processed by congested servers are prosecuted by a non-congested server if available, otherwise they are interrupted. Newly congested replicas that are in the middle of processing a session are removed from the available pool but are allowed to complete the elaboration of the current phase even though with an increased service time.

If no more free and non-congested servers are available at the epoch of the next phase transition, the session will be abruptly terminated.

We model congestion on the non-dedicated links by means of the random variable $x_{\mathtt{cong}}$ ($0 \leqslant x_{\mathtt{cong}} \leqslant C$). The value of this variable is governed by the following events:

1. Congestion arrival on a server (a server departs from the available pool): this event can only occur when there is at least one non-congested server, that is when the available pool is not empty and thus $x_{\mathtt{cong}} < C$. It happens with exponential distribution, with average rate $\mu_{\mathtt{AP}} \cdot (C - x_{\mathtt{cong}})$.
2. Congestion termination on a server (a server goes back to the available pool): this event can only occur when there is at least one congested server, that is when $x_{\mathtt{cong}} > 0$. It happens with exponential distribution with average rate $x_{\mathtt{cong}} \cdot \lambda_{\mathtt{AP}}$.

We now introduce the state definition of our model, differentiating into types of service and phases as we described in Section 4. Without loss of generality we consider only two types of service. We model the state of the process by using a random variable for each phase of each type of service, plus another random variable to represent the congestion level.

The state is thus described through an $(N+1)$ dimensional vector $\mathbf{x} = (\mathbf{x}^N, x_{\mathtt{cong}})$, where the vector $\mathbf{x}^N$ represents the phase occupancy of both the types of service: two phases for the web informational service and six phases for the e-commerce service for a total of $N = 8$ phases.

The vector $\mathbf{x}^N$ is made up as follows: x_1 and x_2 represent the number of ongoing informational web sessions in phase A and B, respectively, and $x_3, \ldots, x_8$ will instead represent the number of ongoing e-commerce sessions in phases A, ..., F, respectively. The transition rates will also be defined according to the same convention used in the enumeration of the state variable. Therefore $\lambda_1 \triangleq \lambda^1_{\mathtt{A}}$ and $\lambda_3 \triangleq \lambda^2_{\mathtt{A}}$. Analogous enumeration will be used to define the outgoing rates $\mu_1, \ldots, \mu_8$, in place of $\mu^1_{\mathtt{A}}, \ldots, \mu^2_{\mathtt{F}}$ and for the capacity requirements $b_1, \ldots, b_8$, in place of $b^1_{\mathtt{A}}, \ldots, b^2_{\mathtt{F}}$.

Since a single dispatcher or access router has to keep track of ongoing sessions in the geographically distributed servers, there is a limit $C^{\mathtt{ID}}$ on the number of sessions that can be tracked at the same time. This limit is typically the maximum number of process identifiers that can be managed by the dispatcher operating system.

Since the number of available server is limited to C and the number of session identifiers is limited to $C^{\mathtt{ID}}$, the following inequalities hold: $0 \leqslant \sum_{i=1}^{N} b_i x_i \leqslant C$ and $0 \leqslant \sum_{i=1}^{N} x_i \leqslant C^{\mathtt{ID}}$, while $0 \leqslant x_{\mathtt{cong}} \leqslant C$.

We denote with $\beta_{\mathtt{congested}}(\mathbf{x})$ the percentage of congested servers in use in state $\mathbf{x}$, that is the ratio between the number of congested busy servers and the total number of servers in use

$$\beta_{\mathtt{congested}}(\mathbf{x}) \triangleq \frac{\max\{0; \sum_{i=1}^{N} b_i x_i + x_{\mathtt{cong}} - C\}}{\sum_{i=1}^{N} b_i x_i}. \tag{2}$$

Notice that this is the percentage of congested servers actually in use by some active session with respect to the total number of servers in use. The normalization is performed on the total number of busy server that is on the number of servers where the congestion is actually perceived by the clients of the considered Internet or CDN service.

We assume an average impact of congestion on the servers actively used. Therefore, when there is at least one congested server actively used by an ongoing session, we assume that the congestion affects all ongoing services proportionally to the percentage of congested servers in use among the busy servers, $\beta_{\mathtt{congested}}(\mathbf{x})$.

Notice that if phase i is a think phase, no congestion latency must be taken into account since there is no resource consumption with the only exception of session descriptors.

Under the given assumption of having an average impact of congestion on the ongoing sessions, the average phase completion rate of phase i is

$$\begin{aligned}\bar{\mu}_i(\mathbf{x}) \triangleq{} & (1 - b_i) \cdot \mu_{i\,\mathtt{non\_congested}} \\ & + b_i \cdot (\beta_{\mathtt{non\_congested}}(\mathbf{x}) \cdot \mu_{i\,\mathtt{non\_congested}} \\ & + \beta_{\mathtt{congested}}(\mathbf{x}) \cdot \mu_{i\,\mathtt{congested}}).\end{aligned} \tag{3}$$

where $\mu_{i\,\mathtt{non\_congested}}$ is the phase i completion rate μ_i, as described by the session lifetime models seen in Section 4, $\mu_{i\mathtt{congested}} = \mu_i/(1 + \alpha_{\mathtt{cong}})$ and $\beta_{\mathtt{non\_congested}}(\mathbf{x}) \triangleq (1 - \beta_{\mathtt{congested}}(\mathbf{x}))$.

6. A Markov modulated decision process for session based access control

We considered two types of service, where each ongoing session is modulated in a Poissonian way, among different phases of resource consumption and think times, thus creating a Markov Modulated Poisson Process (MMPP) of services. The controlled dynamic of the described system constitutes a Semi Markov Decision Process (SMDP). To formally define this process we introduce a state and a decision space, a decision dependent transition probability matrix and a state and decision dependent reward/cost function.

6.1. State and action space

As introduced in Section 5, the state of the process can be defined by means of an $(N + 1)$ dimensional vector $\mathbf{x}$, where N is the total number of phases in the different types of service.

The $(N + 1)$th component of the state vector represents the level of congestion. The state space of the process is

$$\Lambda = \left\{ \mathbf{x} = (x_1, x_2, \ldots, x_N, x_{N+1}) : \sum_{i=1}^{N} b_i x_i \leqslant C; \right. \\ \left. \sum_{i=1}^{N} x_i \leqslant C^{\mathtt{ID}};\ x_{N+1} \leqslant C;\ x_i \geqslant 0 \right\}. \tag{4}$$

We summarize the events that cause the dynamic of the process with the related rates. Arrivals in session initiating phases i occur with rate λ_i, where $\lambda_i = 0$ if $i \neq 1,3$ (phases 1 and 3 are the initial phases of sessions for the informational web and e-commerce types of service, respectively).

Phase terminations happen at average rate $x_i \cdot \bar{\mu}_i$, where $\bar{\mu}_i$ is given by Eq. (3).

Servers abandon the available pool with rate $(C - x_{N+1}) \cdot \mu_{\mathrm{AP}}$ while congestion terminates and the servers go back to the available pool with rate $x_{N+1} \cdot \lambda_{\mathrm{AP}}$.

We define a decision as an N dimensional vector $\mathbf{a}$. Each component of this vector could represent a decision regarding the acceptance of requests belonging to the correspondent phase in the state vector. To simplify the action space, we do not consider mid-session interruption, unless there are no more resources available, we only consider accept/reject decisions at the beginning of a new session. This assumption makes the state vector $\mathbf{a}$ actually only a couple, but we keep the notation as an N-tuple for reasons of readability of formulas.

After being accepted a session is kept alive as long as there are available and non-congested servers. For this reason a_i is null if the phase i is not the initial phase of any type of service (in our model $a_i = 0$ for $i \neq 1, 3$). The indicator a_i denotes the admission, with value 1, or the denial of service, with value 0, of class-i new session requests. The decision space is

$$\mathscr{A} = \{\mathbf{a} = (a_1, a_2, \ldots, a_N) : a_i \in \{0,1\}\}. \tag{5}$$

More precisely, the decision space is actually a state-dependent subset of $\mathscr{A}$ denoted by

$$\mathscr{A}_{\mathbf{x}} = \{\mathbf{a} \in \mathscr{A} : a_i = 0 \quad \text{if } \mathbf{x} + \mathbf{e}_i \notin \Lambda, \\ i = 1, \ldots, N\}, \tag{6}$$

where $\mathbf{e}_i$ is an identity vector, that is a vector of zeroes, except for a one in the ith position. The formulation (6) of the decision space, guarantees that the only feasible actions are the ones that can be satisfied with the free resources.

We refer to $\mathscr{S}$ as to the set of all feasible couples of vectors (*state, decision*).

6.2. Transition probabilities

The SMDP process we are describing is not uniform and the dwell time in each state $\tau(\mathbf{x}, \mathbf{a})$ for each couple $(\mathbf{x}, \mathbf{a}) \in \mathscr{S}$ is state and decision dependent

$$\tau(\mathbf{x}, \mathbf{a}) = \left\{ \sum_{i=1}^{N} [\lambda_i a_i + x_i \cdot \bar{\mu}_i] + (C - x_{N+1}) \cdot \mu_{\mathrm{AP}} + x_{N+1} \cdot \lambda_{\mathrm{AP}} \right\}^{-1}. \tag{7}$$

The set of rates which characterizes the process is bounded by above by the maximum outgoing rate from a state, therefore the process can be uniformized at any rate Γ that exceeds the maximum outgoing rate from any state. Since $\mu_i/(1 + \alpha_{\mathtt{cong}}) \leqslant \mu_i$, a formulation of Γ is the following:

$$\Gamma = \sum_{i=1}^{N} [\lambda_i + b_i \mu_i \cdot C + (1 - b_i)\mu_i \cdot C^{\mathrm{ID}}] + (\mu_{\mathrm{AP}} + \lambda_{\mathrm{AP}}) \cdot C. \tag{8}$$

The uniformization technique transforms the original SMDP with non-identical transition times into an equivalent continuous-time Markov Decision process in which the transition epochs are generated by a Poisson process at uniform rate (see [10,11]).

Let $\tilde{p}^{\mathbf{a}}_{\mathbf{xy}}$ denote the uniformized transition probability from state $\mathbf{x} = (\mathbf{x}^N, x_{\mathtt{cong}})$ to state $\mathbf{y} = (\mathbf{y}^N, y_{\mathtt{cong}})$ if the decision $\mathbf{a}$ is taken and $(\mathbf{x}, \mathbf{a}) \in \mathscr{S}$. The values of $\tilde{p}^{\mathbf{a}}_{\mathbf{xy}}$ are described below.

1. New session request in starting session phase i, for any starting session phase i (in our case $i = 1, 3$): $y_{\mathtt{cong}} = x_{\mathtt{cong}}$ and $\mathbf{y}^N = \mathbf{x}^N + \mathbf{e}_i$

$$\tilde{p}^{\mathbf{a}}_{\mathbf{xy}} = \lambda_i \cdot a_i / \Gamma, \quad \text{where } a_i = 0 \\ \text{if } \sum_{k=1}^{N} b_k x_k + x_{\mathtt{cong}} \geqslant C. \tag{9}$$

2. Transition from phase i to phase j: $y_{\mathtt{cong}} = x_{\mathtt{cong}}$ and $\mathbf{y}^N = \mathbf{x}^N - \mathbf{e}_i + \mathbf{e}_j$. While a transition towards a think phase is always allowed, an active processing phase j can be reached by a session coming from phase i only if there is no congestion or if the congestion does not affect the servers required to fulfill the phase transition request that is when $\sum_{k=1}^{N} b_k \cdot x_k + (b_j - b_i) + x_{\mathtt{cong}} \leqslant C$. We denote with $\mathscr{I}_i(\mathbf{x})$ the set of phases j that cannot be reached by a session coming from phase i due to congestion. The set $\mathscr{I}_i(\mathbf{x})$ is empty if there is no congestion and phase transitions are always possible

$$\mathscr{I}_i(\mathbf{x}) = \left\{ j : b_j > C + b_i - \left(\sum_{k=1}^{N} b_k x_k + x_{\mathtt{cong}} \right) \right\}, \tag{10}$$

$$\tilde{p}^{\mathbf{a}}_{\mathbf{xy}} = x_i \cdot \frac{\bar{\mu}_i}{\Gamma} \cdot \pi_{ij} \quad \text{if } j \notin \mathscr{I}_i(\mathbf{x}). \tag{11}$$

3. Session termination in phase i: $y_{\mathtt{cong}} = x_{\mathtt{cong}}$ and $\mathbf{y}^N = \mathbf{x}^N - \mathbf{e}_i$.
The event of a session termination may occur for two reasons. A session can be terminated by the system due to congestion while attempting a phase transition towards an active phase j, that is when $j \in \mathscr{I}_i(\mathbf{x})$, with rate $x_i \bar{\mu}_i \pi_{ij} / \Gamma$. A session can also be voluntary terminated by the user at the end of phase i with rate $x_i \bar{\mu}_i (1 - \sum_{j=1}^{N} \pi_{ij}) / \Gamma$. Therefore the overall probability of having a phase-i termination is

$$\tilde{p}^{\mathbf{a}}_{\mathbf{xy}} = \frac{x_i \bar{\mu}_i}{\Gamma} \left(1 - \sum_{j \notin \mathscr{I}_i(\mathbf{x})} \pi_{ij} \right). \tag{12}$$

4. *Congestion arrival*: $\mathbf{y}^N = \mathbf{x}^N$ and $y_{\mathtt{cong}} = x_{\mathtt{cong}} + 1$. Non-congested servers may exit from the available pool due to congestion on the non-dedicated links with rate $\mu_{\mathrm{AP}}(C - x_{\mathtt{cong}}) / \Gamma$. Thence

$$\tilde{p}^{\mathbf{a}}_{\mathbf{xy}} = \frac{\mu_{\mathrm{AP}} \ (C - x_{\mathtt{cong}} \ )}{\Gamma} \quad \text{if } x_{\mathtt{cong}} < C. \tag{13}$$

5. *Congestion termination*: $\mathbf{y}^N = \mathbf{x}^N$ and $y_{\texttt{cong}} = x_{\texttt{cong}} - 1$. Congested servers may return to normal conditions and become available again to serve requests with rate $\lambda_{\texttt{AP}} \cdot x_{\texttt{cong}}/\Gamma$. Thence

$$\tilde{p}^{\mathbf{a}}_{\mathbf{xy}} = \frac{\lambda_{\texttt{AP}} \cdot x_{\texttt{cong}}}{\Gamma} \quad \text{if } x_{\texttt{cong}} > 0. \tag{14}$$

6. Dummy transitions from each state to itself: $\mathbf{y} = \mathbf{x}$. This transitions are added to the chain of the original, non-uniform process, in agreement with the uniformization procedure

$$\tilde{p}^{\mathbf{a}}_{\mathbf{xy}} = \frac{1}{\Gamma} \cdot \left\{ \Gamma - \left[\sum_{i=1}^{N} (\lambda_i a_i + x_i \bar{\mu}_i) + (C - x_{\texttt{cong}})\mu_{\texttt{AP}} + x_{\texttt{cong}} \lambda_{\texttt{AP}} \right] \right\}. \tag{15}$$

The transitions between other states that are not considered in this list have null probability.

6.3. Profits and losses during session lifetime

Aim of this section is to give a formulation of a cost/profit function that associates penalties and incomes to states, events and decisions. To complete the description of the decision process and to be capable of formulating an objective function, we introduce costs and rewards for each couple $(\textit{state}, \textit{decision}) \in \mathcal{S}$. The objective function we formulated is the average expected gain per unit of time.

The admission control will decide whether a new session request should be admitted or not. If a new session request is rejected, a rejection penalty will be paid. Phase transitions are not subject to the admission control, and all subsequent phases of an admitted session are admitted provided that enough non-congested servers are available. If there are no available servers to complete a session the system will incur an interruption penalty usually higher than the rejection penalty. On the other hand, if a session is successfully completed, that is the user willingly terminates the session, more luckily with a purchase, a profit is gained.

The decision control system must be capable to accept or reject new sessions only when the whole session is likely to be completed. On the other hand the system should be able to reject a new session if there is a high probability that it will be interrupted in a critical phase. The interruption of a session in a critical phase would cost a much higher penalty than the rejection of a session at its beginning.

As seen before, a differentiation between types of service must be made, because one type is considered premium while the other is considered basic, and we want the system to pay a higher penalty for the block of a premium session than for the block of a basic request. We also want to differentiate among different phases belonging to the same session, because some of them are much more critical than others.

We introduce the penalty $H_{\texttt{EC}}$ to be paid by the system for the denial of service to an e-commerce request. An analogous penalty $H_{\texttt{IW}}$ is incurred by the system when an informational web request is refused. Since the e-commerce type of service is considered premium, the penalty for the denial of this type of service will be higher than for the other, therefore $H_{\texttt{EC}} > H_{\texttt{IW}}$. These costs may be incurred because the system is unable to assign resources to a new session due to congestion or overload, or because there is an explicit rejection decision of the access controller.

A second type of penalty relates to the interruption of an ongoing session. This is not related to a decision, but in most of the cases comes from an underestimation of the congestion problem or of the workload situation. Since some phases are more critical than others, we differentiate the interruption penalties from phase to phase. In the case of the informational web type of service, none of the phases is particularly critical. The transition from phase A to phase B is always admitted since it brings the system from an active processing phase to a think phase. The opposite transition from phase B to phase A is instead permitted only if there are available and non-congested servers, otherwise a penalty $H_{\texttt{TA\_IW}}$ is incurred.

In the case of the e-commerce type of service, the phase transition A–B, C–D and E–D, is always admitted since it is directed towards a think phase. Transitions B–C and D–E are not considered very critical and if the session is interrupted during these transition the system will incur a penalty $H_{\texttt{TA\_EC}}$ that is less than the penalty $H_{\texttt{AA\_EC}}$ that the system will pay in case of interruption of the session during the very critical transition from phase E to phase F.

In order to make the system accept a new session only if it is likely to guarantee continuity of service until the end of the session, the penalties for the denial of service will be lower than any phase interruption penalty, no matter if the considered phase is critical or not. Further, the penalties introduced for the premium class are higher than for the basic

Table 2
Uniformized cost function

State $\mathbf{x}$	Decision $\mathbf{a}$	Motivation	Uniformized cost $r_{\mathtt{cost}}(\mathbf{x},\mathbf{a})$
*	$a_1 = 0$	IW reject	$H_{\mathtt{IW}} \cdot \lambda_1/\Gamma$
*	$a_3 = 0$	EC reject	$H_{\mathtt{EC}} \cdot \lambda_3/\Gamma$
$\sum_{k=1}^{N} b_k x_k + x_{\mathtt{cong}} \geqslant C$	*	EC interruption in B–A, B–C, D–C, D–E	$[(\pi_{43} + \pi_{45})x_4\bar{\mu}_4 + (\pi_{65} + \pi_{67})x_6\bar{\mu}_6]H_{\mathtt{TA\_EC}}/\Gamma$
$\sum_{k=1}^{N} b_k x_k + x_{\mathtt{cong}} > C$	*	EC interruption in E–F	$\pi_{78}x_7\bar{\mu}_7\ H_{\mathtt{AA\_EC}}/\Gamma$
$\sum_{k=1}^{N} b_k x_k + x_{\mathtt{cong}} \geqslant C$	*	IW interruption in B-A	$\pi_{21}x_2\bar{\mu}_2 H_{\mathtt{TA\_IW}}/\Gamma$
$\sum_{k=1}^{N} b_k x_k + x_{\mathtt{cong}} > C$	*	Degraded service due to congestion	$\frac{\left(\sum_{k=1}^{N} b_k x_k + x_{\mathtt{cong}}\ - C\right) H_{\mathtt{BC}}}{\Gamma}$

class, to ensure a proper prioritization of one type of service over the other. These kind of costs are transition related and are paid whenever a request occurs and the decision of rejection is taken.

Further we consider a state related cost. State related costs are not paid when a particular event occurs, but are paid as long as the system dwells in the considered state. We think that a state related penalty should be paid by the system in case a session receives a degraded service due to congestion. This state-related cost, to which we refer as $H_{\mathtt{BC}}$ (the index "BC" stands for "busy and congested"), is not transition related and is paid per unit of time.

The uniformization technique [10] must be applied also to the cost function. Table 2 gives a summary of the values of the uniformized cost function $r_{\mathtt{cost}}(\mathbf{x},\mathbf{a})$.

We now introduce the profits the system will gain in case of successful completion of a session. The user may willingly terminate the session in any phase of its lifetime. If the session is terminated by the user and does not encounter congestion or overload problems, it is considered successfully terminated, and a profit is associated to the related transitions. We introduce the profit $V_{\mathtt{IW}}$ that is gained when the user terminates an informational web session. For what concerns the e-commerce type of service we instead differentiate the profits the system will gain in case the user terminates the session with a purchase or not. If the e-commerce session is terminated without a purchase the income will be $V_{\mathtt{EC}}$, while if it is terminated with a purchase the income will be $W_{\mathtt{EC}}$, where $V_{\mathtt{IW}} < V_{\mathtt{EC}} < W_{\mathtt{EC}}$.

As before, we apply the uniformization technique to obtain the reward function $r_{\mathtt{rew}}(\mathbf{x},\mathbf{a})$ of Table 3.

The unichain property, together with the finiteness of $\mathscr{S}$ implies the existence of a unique stationary state probability distribution which is independent of the initial state of the process. The existence of a stationary policy allows us to conclude that an optimal solution can be expressed through a decision variable $x_{\mathbf{sa}}$ that represents the probability for the system to be in state $\mathbf{s}$ and contemporaneously to take the decision $\mathbf{a}$.

The Linear Programming (LP) formulation associated with our SMDP for the minimization of the average cost is

$$\begin{aligned}
&\text{Minimize}\\
&\quad \sum_{(\mathbf{s},\mathbf{a})\in\mathscr{S}} [r_{\mathtt{cost}}\ (\mathbf{s},\mathbf{a}) - r_{\mathtt{rew}}\ (\mathbf{s},\mathbf{a})] \cdot x_{\mathbf{sa}}\\
&\text{constrained to}\\
&\quad x_{\mathbf{sa}} \geqslant 0 \quad (\mathbf{s},\mathbf{a}) \in \mathscr{S},\\
&\quad \sum_{(\mathbf{s},\mathbf{a})\in\mathscr{S}} x_{\mathbf{sa}} = 1,\\
&\quad \sum_{\mathbf{a}\in\mathscr{B}_{\mathbf{j}}} x_{\mathbf{ja}} = \sum_{(\mathbf{s},\mathbf{a})\in\mathscr{S}} \tilde{p}^{\mathbf{a}}_{\mathbf{sj}} x_{\mathbf{sa}} \quad \mathbf{j} \in \Lambda.
\end{aligned} \tag{16}$$

The problem (16) has $\|\Lambda\|$ constraints and $|\mathscr{S}| = |\Lambda| \cdot |\mathscr{A}|$ variables. It can be solved with polynomial time complexity by means of common LP methods and, from now on, the corresponding optimal solution will been named OPT.

Table 3
Uniformized reward function

State $\mathbf{x}$	Decision $\mathbf{a}$	Motivation	Uniformized reward $r_{\mathtt{rew}}(\mathbf{x},\mathbf{a})$
*	*	IW termination	$\sum_{i=1}^{2} x_i\bar{\mu}_i(1 - \sum_{j=1}^{2}\pi_{ij})V_{\mathtt{IW}}/\Gamma$
*	*	EC termination no purchase	$\sum_{i=3}^{7} x_i\bar{\mu}_i(1 - \sum_{j=3}^{8}\pi_{ij})V_{\mathtt{EC}}/\Gamma$
*	*	EC termination with purchase	$x_8\bar{\mu}_8(1 - \sum_{j=3}^{8}\pi_{8j})W_{\mathtt{EC}}/\Gamma$

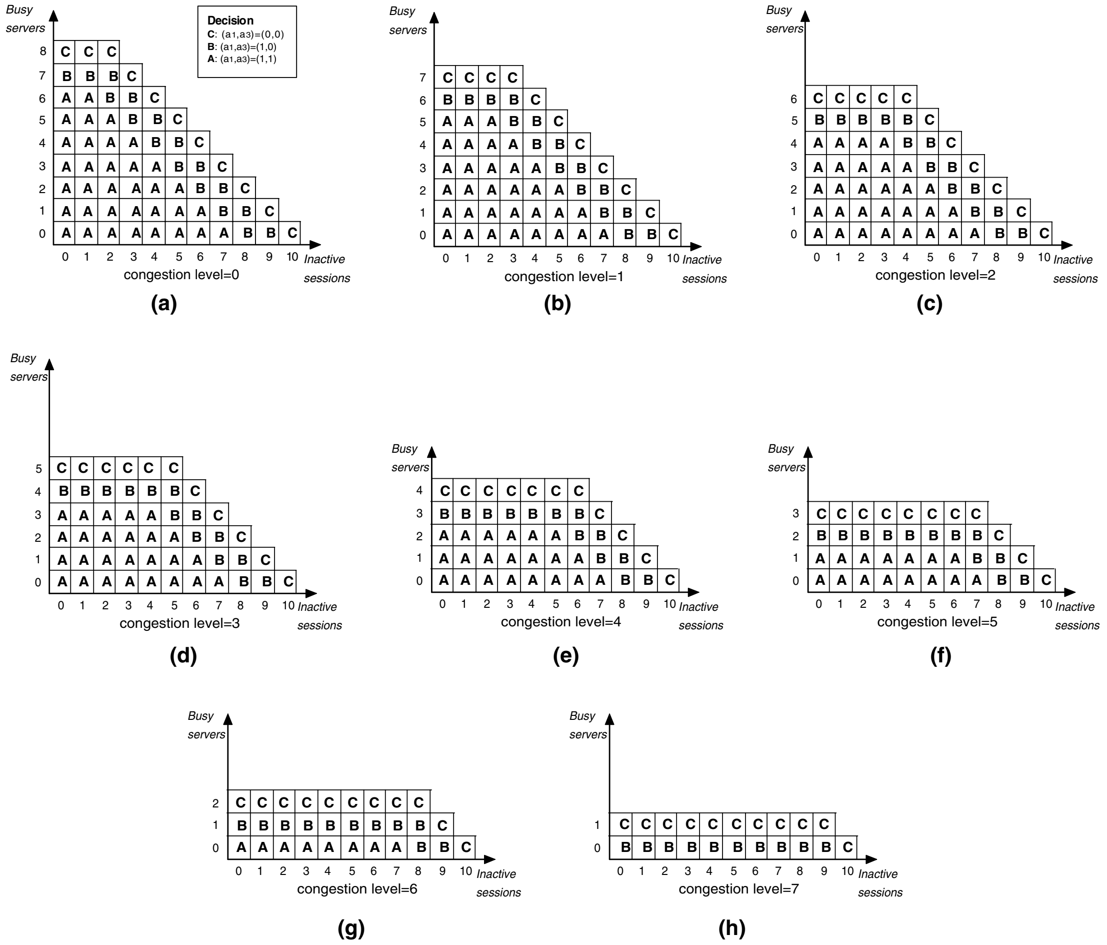
7. Structural analysis of the optimal admission policy and heuristics

The decision process formulated in Section 6 allows the search for the optimal policy in a wide general class that also includes non-stationary *randomized* policies. The existence of a stationary and deterministic optimal solution can be proven by means of the analysis of the set of constraints associated to the optimization problem (16) [24].

The high dimensionality of the Markovian process makes it inappropriate for use in realistic scenarios in which the topology of the network can be very complex with potentially hundreds of servers such as in CDNs. Nevertheless iterative methods can be adopted to obtain the optimal policy in some significative small cases to have clues for the formulation of possible heuristics to be adopted in more general large scale scenarios.

By solving iteratively problem (16) we saw that the optimal policy does not always show regular properties or intuitive behaviors. Nevertheless apart from some particular scenarios, we found that in the most typical cases the optimal policy shows a double threshold behavior when deciding which request to accept: a first threshold is in terms of available servers, and the second is in terms of process identifiers available at the dispatcher level. The implementation of such a regular policy is inexpensive and can be easily added to access controllers.

Just to give an example, before we formalize this property, in Fig. 4 we show the behavior of the optimal policy in a scenario where the traffic parameters are those described in Section 4 for each class of service, the number of available servers is $C = 8$ and $C^{\mathrm{ID}} = 10$, the costs are $H_{\mathtt{EC}} = 10{,}000$, $H_{\mathtt{IW}} = 5000$, $H_{\mathtt{TA\_EC}} = 11{,}000$, $H_{\mathtt{AA\_EC}} = 100{,}000$, $H_{\mathtt{TA\_IW}} = 6000$,



	0	1	2	3	4	5	6	7	8	9	10
8	C	C	C								
7	B	B	B	C							
6	A	A	B	B	C						
5	A	A	A	B	B	C					
4	A	A	A	A	B	B	C				
3	A	A	A	A	A	B	B	C			
2	A	A	A	A	A	A	B	B	C		
1	A	A	A	A	A	A	A	B	B	C	
0	A	A	A	A	A	A	A	A	B	B	C

congestion level=0

(a)

	0	1	2	3	4	5	6	7	8	9	10
7	C	C	C	C							
6	B	B	B	B	C						
5	A	A	A	B	B	C					
4	A	A	A	A	B	B	C				
3	A	A	A	A	A	B	B	C			
2	A	A	A	A	A	A	B	B	C		
1	A	A	A	A	A	A	A	B	B	C	
0	A	A	A	A	A	A	A	A	B	B	C

congestion level=1

(b)

	0	1	2	3	4	5	6	7	8	9	10
6	C	C	C	C	C						
5	B	B	B	B	B	C					
4	A	A	A	A	B	B	C				
3	A	A	A	A	A	B	B	C			
2	A	A	A	A	A	A	B	B	C		
1	A	A	A	A	A	A	A	B	B	C	
0	A	A	A	A	A	A	A	A	B	B	C

congestion level=2

(c)

	0	1	2	3	4	5	6	7	8	9	10
5	C	C	C	C	C	C					
4	B	B	B	B	B	B	C				
3	A	A	A	A	A	B	B	C			
2	A	A	A	A	A	A	B	B	C		
1	A	A	A	A	A	A	A	B	B	C	
0	A	A	A	A	A	A	A	A	B	B	C

congestion level=3

(d)

	0	1	2	3	4	5	6	7	8	9	10
4	C	C	C	C	C	C	C				
3	B	B	B	B	B	B	B	C			
2	A	A	A	A	A	A	B	B	C		
1	A	A	A	A	A	A	A	B	B	C	
0	A	A	A	A	A	A	A	A	B	B	C

congestion level=4

(e)

	0	1	2	3	4	5	6	7	8	9	10
3	C	C	C	C	C	C	C	C			
2	B	B	B	B	B	B	B	B	C		
1	A	A	A	A	A	A	A	B	B	C	
0	A	A	A	A	A	A	A	A	B	B	C

congestion level=5

(f)

	0	1	2	3	4	5	6	7	8	9	10
2	C	C	C	C	C	C	C	C	C		
1	B	B	B	B	B	B	B	B	B	C	
0	A	A	A	A	A	A	A	A	B	B	C

congestion level=6

(g)

	0	1	2	3	4	5	6	7	8	9	10
1	C	C	C	C	C	C	C	C	C	C	
0	B	B	B	B	B	B	B	B	B	B	C

congestion level=7

(h)

Fig. 4. Optimal policy ($C = 10$, $C^{\mathrm{ID}} = 8$).

$H_{\mathtt{BC}} = 100$ and the rewards are $V_{\mathtt{IW}} = 5000$, $V_{\mathtt{EC}} = 9000$ and $W_{\mathtt{EC}} = 110{,}000$, thus giving high priority to e-commerce services that could end in a commercial transaction.

The number of inactive sessions that is indicated on the x-axis of these figures, represents the number of sessions in think phase during observation, that is the number of sessions that are actually using an identifier but are not consuming the computational capacity of the servers. On the y-axis we report the number of servers currently in use, while the sequence of diagrams from (a) to (h) varies with the number of congested servers. The optimal decision depends both on the number of ongoing sessions in the system ($x_{\mathtt{busy}} + x_{\mathtt{think}}$) and on the number of servers that are unavailable, due to external traffic ($x_{\mathtt{cong}}$). Each diagram of Fig. 4 shows the decision value as a function of $x_{\mathtt{busy}}$ and $x_{\mathtt{think}}$, given the value of $x_{\mathtt{cong}}$. For a given number of congested server, the decision depends on a double threshold policy on the number of busy or congested servers and on the number of running sessions (session identifiers in use by the dispatcher). As the number of congested servers grows (diagram (a)–(h)), these thresholds decrease.

To clarify the meaning of the diagrams (a)–(h), consider the states $\mathbf{x}_1$ with $x_{\mathtt{cong}} = 3$, $x_{\mathtt{think}} = 4$ and $x_{\mathtt{busy}} = 3$. In $\mathbf{x}_1$ the optimal decision is A: to accept all types of requests. If the next state transition leads the system to state $\mathbf{x}_2$, with a higher number of busy servers $x_{\mathtt{busy}} = 4$ and $x_{\mathtt{cong}} = 3$, $x_{\mathtt{think}} = 4$ the system will then take decision B: to accept only requests of the high priority type. If at the next state transition there is one more congested server, the state will be $\mathbf{x}_3$ with $x_{\mathtt{cong}} = 4$ and $x_{\mathtt{busy}} = 4$, $x_{\mathtt{think}} = 4$ and the decision will be C: to deny service to all requests.

Though not generalizable, Fig. 4 shows a structure of the optimal policy that holds in most of the analyzed scenarios. In most cases the optimal policies consists in reserving resources (computational capacity and identifiers) to the high priority customers (e-commerce stream of requests). The amount of reserved resources strictly depends on the presence of external traffic.

For this reason we considered the following heuristic (HEU) that mimics the behavior of the optimal policy (OPT). The HEU policy reserves $K_{\mathtt{reserved\_servers}}$ units of server capacity and $K_{\mathtt{reserved\_id}}$ session identifiers to the high priority stream of session activation requests.

We refer to $x_{\mathtt{think}}$ as to the number of inactive sessions, that is the number of ongoing sessions in the think phase, $x_{\mathtt{think}} = \sum_{i=1}^{N}(1 - b_i) \cdot x_i$. As seen previously in Section 5, $x_{\mathtt{busy}} = \sum_{i=1}^{N} b_i \cdot x_i$, while $x_{\mathtt{cong}}$ is the number of congested units of server capacity. We define the following threshold values: $T^{\mathtt{servers}} \triangleq C - K_{\mathtt{reserved\_servers}}$ and $T^{\mathtt{id}} \triangleq C - K_{\mathtt{reserved\_id}}$.

The HEU policy can be formulated as follows:

- If $x_{\mathtt{busy}} < \min\{T^{\mathtt{ID}} - x_{\mathtt{think}};\ T^{\mathtt{servers}} - x_{\mathtt{cong}}\}$ take decision $(a_1, a_3) = (1, 1)$, i.e. give service to both streams of requests.
- If $\min\{T^{\mathtt{ID}} - x_{\mathtt{think}};\ T^{\mathtt{servers}} - x_{\mathtt{cong}}\} \leqslant x_{\mathtt{busy}} < \min\{C^{\mathtt{ID}} - x_{\mathtt{think}};\ C - x_{\mathtt{cong}}\}$ take decision $(a_1, a_3) = (0, 1)$, i.e. give service only to e-commerce requests.
- If $x_{\mathtt{busy}} \geqslant \min\{C^{\mathtt{ID}} - x_{\mathtt{think}};\ C - x_{\mathtt{cong}}\}$ take decision $(a_1, a_3) = (0, 0)$, i.e. no new session can be admitted, neither from the informational web stream, nor from the e-commerce stream, due to the lack of available resources.

As we explained, the formalization of the problem in terms of LP and its optimization was very useful in giving clues about possible heuristics and a benchmark for comparisons. Further, the solution of the optimal problem gives insight on threshold tuning for the heuristic HEU even when the optimal policy does not have this regular behavior.

Experimentations showed that the best threshold choice depends on many factors, among which the most important are: costs and rewards (that should be discussed with the service provider), session arrival rates (that can be measured at the dispatcher level) and average lifetime of successfully completed sessions (sessions terminated due to user's will, that can be measured at the dispatcher level). The solution of problem (16) showed in fact that when there is a high rate of short lived informational web sessions, provided that the termination of informational web implies a non-null reward, the optimal access controller tends to privilege the low priority type of service in spite of high priority requests. In this case the system will more likely gain profits from small frequent rewards coming from the satisfaction of low priority request than with long lived high priority sessions that could potentially lead to high interruption costs in case of high load. The benefits of such a policy strictly depend on the particular business model that is behind the considered types of service.

8. Experimental results

Knowledge of traffic parameters is a key issue when selecting the access control policy. Some traffic parameters can easily be obtained by performing a measurement activity at the dispatcher, while some other parameters must come from an accurate economical analysis of costs and profits, and of the agreements on quality. The values of these parameters are very variable from application to application, but the general guidelines proposed in our study remain valid.

In this section we describe the experimental results obtained by means of a simulator based on OPNET [25] using synthetic traffic generators that follow the session models introduced in Section 4.

Although the simulation technique enables to remove many of the assumptions made in the analytical model, the use of the optimal admission control policy OPT as a benchmark for comparisons in only possible in the same scenario for which OPT is the optimal policy.

For this reason we begin the analysis of the simulator results with a very simple scenario, named scenario 1, with few resources, to ensure the tractability of problem (16) and be able to compute the optimal solution.

We analyze the effects of the policies introduced in the previous Section 7 and provide performance comparisons among the optimal policy OPT and the heuristics HEU with different choices of the threshold parameters. A trivial policy, consisting in doing nothing to improve performance, will be named noAC, and will be used as a benchmark for comparisons. With the noAC policy both streams of session activation requests are treated alike and no discrimination is done between service classes.

Scenario 1 is characterized by the following parameter setting: $C = 8$, $C^{\mathrm{ID}} = 10$. The traffic parameters of the informational web class of requests assume the following values $\lambda_A^1 = 0.1\ \mathrm{s}^{-1}$, $\mu_A^1 = 10\ \mathrm{s}^{-1}$, $\mu_B^1 = \mu_{\mathtt{think}} = 0.05\ \mathrm{s}^{-1}$, $\pi_{AB}^1 = 0.95$ and $\pi_{BA}^1 = 0.6$, and for the e-commerce class of requests $\lambda_A^2 = 0,1\ \mathrm{s}^{-1}$, $\mu_A^2 = 10\ \mathrm{s}^{-1}$, $\mu_B^2 = \mu_D^2 = \mu_{\mathtt{think}} = 0.05\ \mathrm{s}^{-1}$, $\mu_C^2 = 0.0333\ \mathrm{s}^{-1}$, $\mu_E^2 = 0.1\ \mathrm{s}^{-1}$, $\mu_F^2 = 0.1\ \mathrm{s}^{-1}$, $\pi_{AB}^2 = 0.8$, $\pi_{BA}^2 = 0.3$, $\pi_{BC}^2 = 0.6$, $\pi_{CD}^2 = 0.5$, $\pi_{DC}^2 = 0.2$, $\pi_{DE}^2 = 0.7$, $\pi_{ED}^2 = 0.05$ and $\pi_{EF}^2 = 0.9$. The values of congestion parameters are $\alpha_{\mathtt{cong}} = 0.5$, $\lambda_{\mathrm{AP}} = \mu_{\mathrm{AP}} = 10^{-4}$ while costs and rewards are set to $H_{\mathtt{BC}} = 100$, $H_{\mathtt{IW}} = 20$, $H_{\mathtt{TA\_IW}} = 20$, $H_{\mathtt{EC}} = 20{,}000$, $H_{\mathtt{TA\_EC}} = 21{,}000$, $H_{\mathtt{AA\_EC}} = 100{,}000$, $V_{\mathtt{IW}} = 30$, $V_{\mathtt{EC}} = 10{,}000$ and $W_{\mathtt{EC}} = 110{,}000$.

Fig. 5 points out the effect of the OPT policy in scenario 1. While OPT obviously guarantees a higher acceptance probability of e-commerce requests, it causes a significant increase in the blocking probability of new session activation requests.

In this scenario, with the application of OPT, the blocking probability of informational web requests is very high (approximately 96%). Such a high blocking probability can be considered acceptable in some cases, for example if the low priority type of service is only introduced to differentiate users that are exploring a non-profit area of the same site. This value should probably considered too high if the two types of service involve separate classes of users. On the contrary, the high priority stream, that is the e-commerce stream of requests, experiences an increased performance both in terms of reduced blocking probability and in terms of increased successful termination probability when the optimal admission control policy is applied.

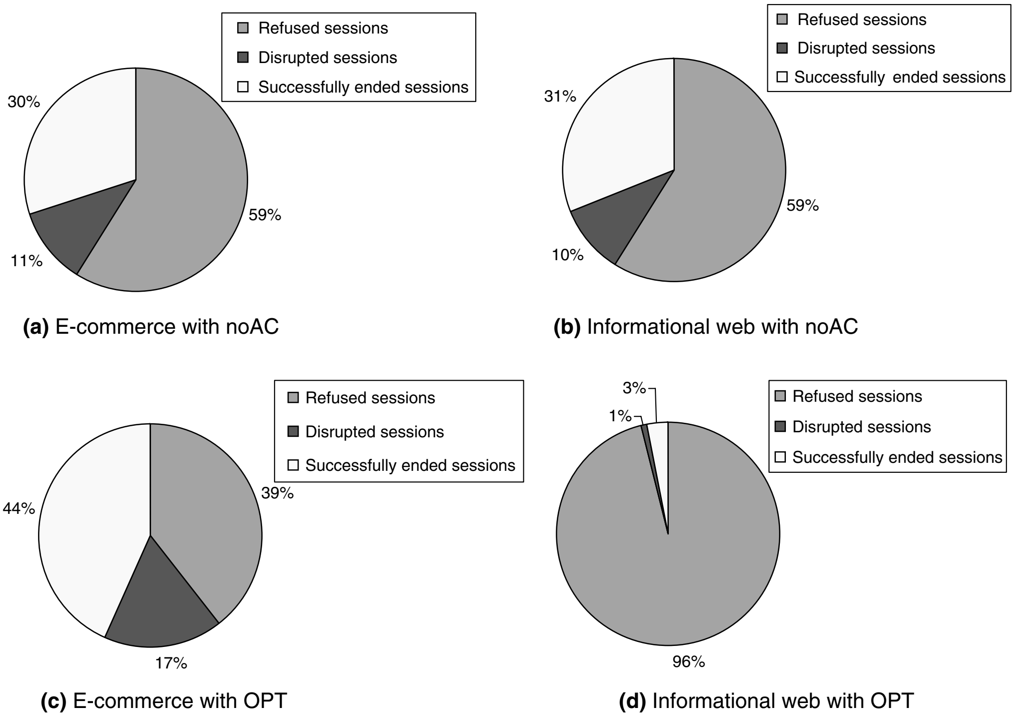
The performance of the admission control policies can be measured from the service provider's point of view in terms of revenues. One possible metric is the average value of the reward function

$$W = \sum_{(\mathbf{s},\mathbf{a})\in\mathscr{S}} [r_{\mathtt{rew}}\ (\mathbf{s},\mathbf{a}) - r_{\mathtt{cost}}\ (\mathbf{s},\mathbf{a})] \cdot x_{\mathbf{sa}}. \tag{17}$$

The average reward function may be positive or negative depending on the considered workload scenario and on the costs associated to actions such as disrupting an ongoing session in a given phase, refusing a connection, completing a session with or without purchase, etc.

Problem (16) optimizes the average reward, as expressed by Eq. (17). In the given scenario, the optimal solution OPT has the general behavior of HEU with few exceptions in some states, and suggests the use of very low threshold ($T^{\mathtt{servers}} \leqslant 2$) on the number of busy or congested servers. In order to evaluate the performance of HEU and the correctness of the threshold choice we evaluated this heuristics with several different choices of the threshold values $T^{\mathtt{server}}$ and $T^{\mathtt{id}}$.

Fig. 6 points out the different performance of the heuristics HEU with different threshold values. The performance is measured in terms of provider's revenue according to Eq. (17). It can be seen that, as suggested by the structural analysis conducted on the optimal policy, the best parameter choice is with $T^{\mathtt{server}} = 2$ and $T^{\mathtt{id}} = 8$. We will refer to the



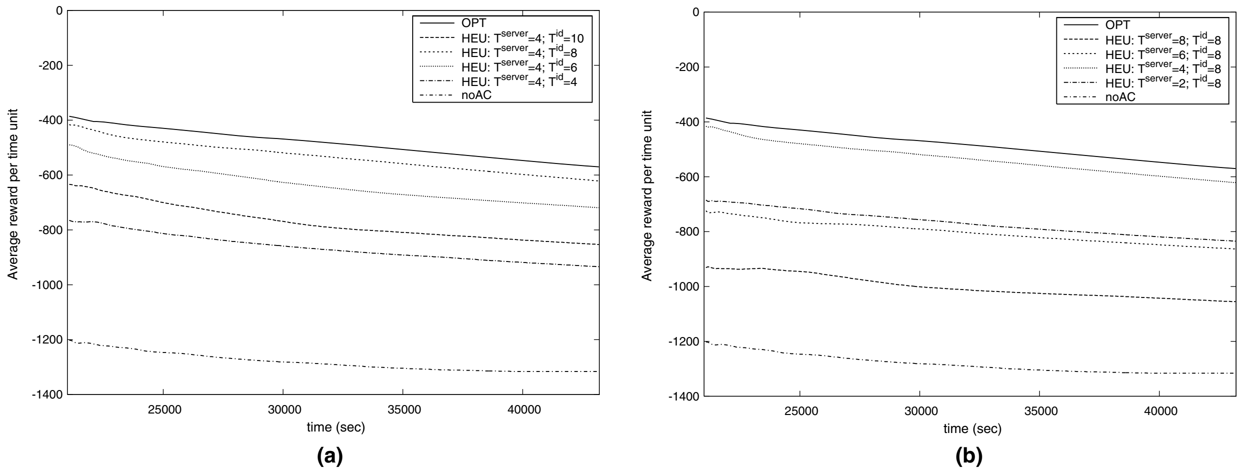
(a) E-commerce with noAC

(b) Informational web with noAC

(c) E-commerce with OPT

(d) Informational web with OPT

Fig. 5. Probability of successful termination of e-commerce and informational web requests (scenario 1).



(a)

(b)

Fig. 6. Average reward function (scenario 1).

heuristics with this parameter setting with the name of HEU28.

In the experiments we tried to see the difference among several settings of the heuristic parameters, and we varied the two threshold independently. The simulations of Fig. 6(a) were conducted fixing the parameter $T^{\mathtt{id}}$ to its best value (as suggested by the structural analysis of the optimal policy) and varying $T^{\mathtt{server}}$. The simulations of Fig. 6(b) were conducted fixing the parameter $T^{\mathtt{server}}$ and varying $T^{\mathtt{id}}$. HEU28 is the heuristic that mostly mimics the optimal policy obtained by solving problem (16) with the parameter setting of the analyzed scenario.

The rate of successful termination of e-commerce requests at the transaction phase is also a

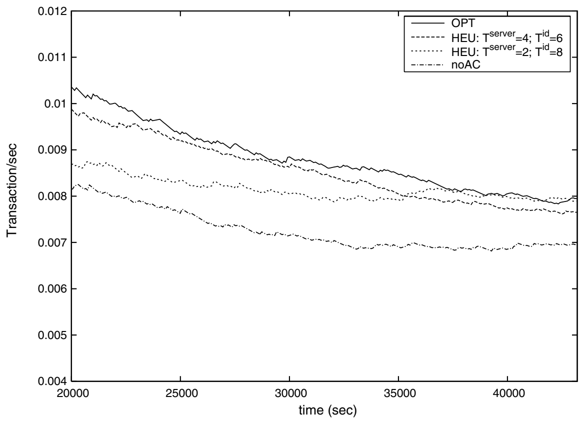


Fig. 7. Rate (requests/seconds) of e-commerce session successful termination with a transaction (scenario 1).

meaningful parameter to evaluate the performance of the system from the provider's point of view. The trend of this measure is shown in Fig. 7 where HEU28 shows a good approximation of the behavior of the OPT policy. It comes out from this experiment that the heuristic HEU with $T^{\mathtt{server}} = 4$ and $T^{\mathtt{id}} = 6$ (HEU46, that is characterized by less reserved servers but more reserved identifiers than HEU28), performs better than HEU28 in terms of rate of e-commerce sessions terminated with a commercial transaction. This is mostly due to the effect of the threshold on the number of session identifiers $T^{\mathtt{id}} = 6$, that is lower than the number of available servers $C = 8$ in the experiment. By setting $T^{\mathtt{id}} < C$ we obtain a policy that can reject every type of service if the server load exceed a given level even if there are available, non-congested servers. According to this policy several requests, even belonging to the high priority type, can be rejected in the hope to make place for already ongoing sessions, possibly leading to a purchase.

Fig. 8(a) shows that OPT has the best performance among the simulated policies in terms of e-commerce request blocking probability (59%). Unfortunately this improvement is at the expense of the informational web type of service. Fig. 8(b) shows the significant loss of OPT in terms of informational web blocking probability (96%). As discussed before, this value can be acceptable or not depending on the adopted business model. Parameter tuning of HEU is clearly a way to increase one probability at the expense of the other and there is no good value that can be suggested a priori because costs and quality of service requirements are very variable and application dependent.

We now introduce some simulations obtained in a second scenario, named *scenario 2*, characterized by a more realistic parameter setting: $C = 20$ resource units and $C^{\mathtt{ID}} = 10{,}000$ available session identifiers. The workload utilized for the previous simulations is so low for this new scenario configuration that even the noAC policy ensures a successful termination probability close to 1 with null disruption and blocking probability for both session types.

For this reason we test scenario 2 with a higher workload: arrival rates $\lambda_A^1 = \lambda_A^2 = 15\ \mathrm{s}^{-1}$, congestion parameters $\alpha_{\mathtt{cong}} = 0.5$, $\lambda_{\mathrm{AP}} = 10^{-4}$, $\mu_{\mathrm{AP}} = 10^{-5}$, phase completion rates $\mu_A^1 = \mu_A^2 = 10\ \mathrm{s}^{-1}$, $\mu_B^1 = \mu_B^2 = \mu_D^2 = \mu_{\mathtt{think}} = 0.05\ \mathrm{s}^{-1}$, $\mu_C^2 = \mu_E^2 = \mu_F^2 = 5\ \mathrm{s}^{-1}$, phase transition probabilities $\pi_{AB}^1 = \pi_{AB}^2 = 0.9$,

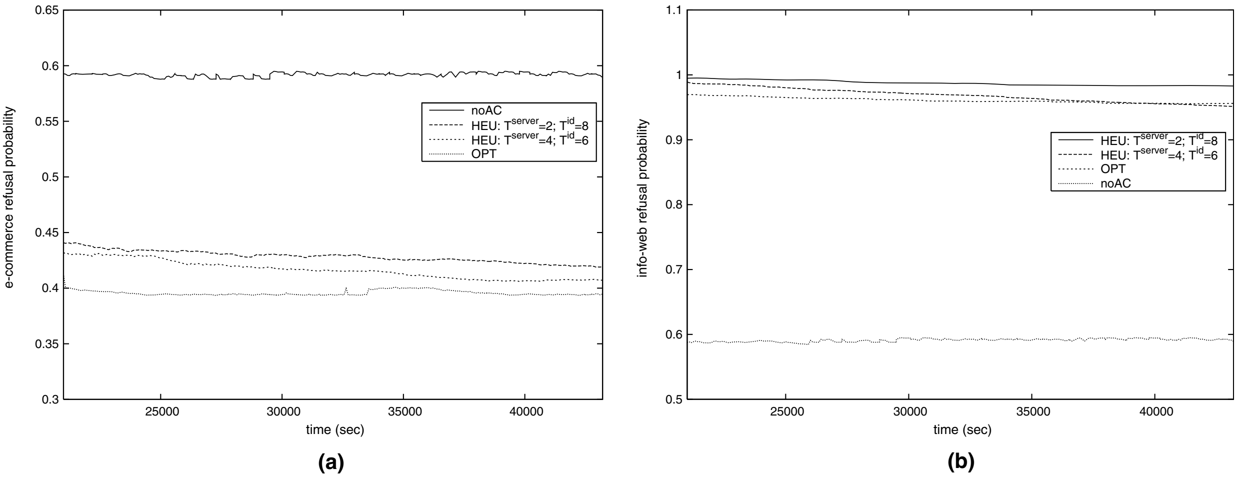
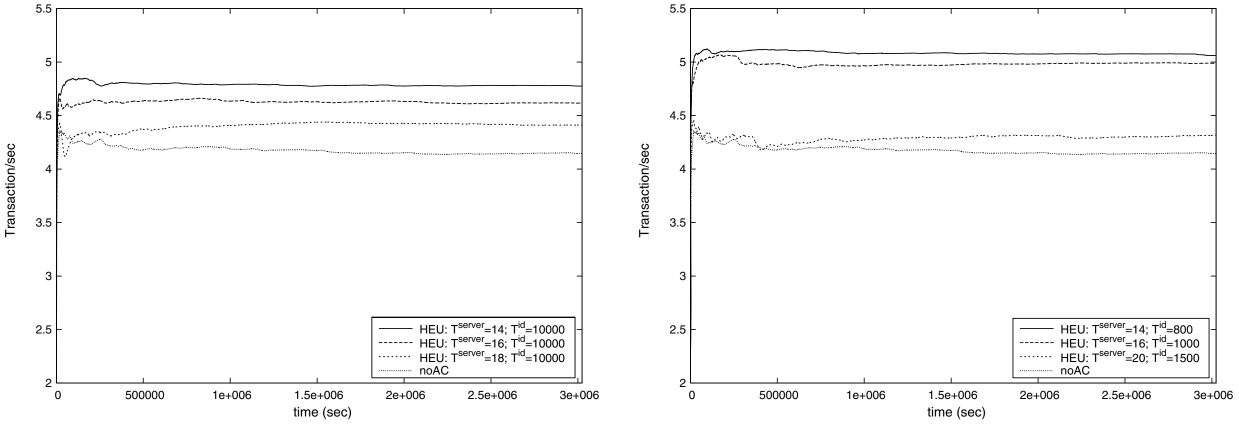
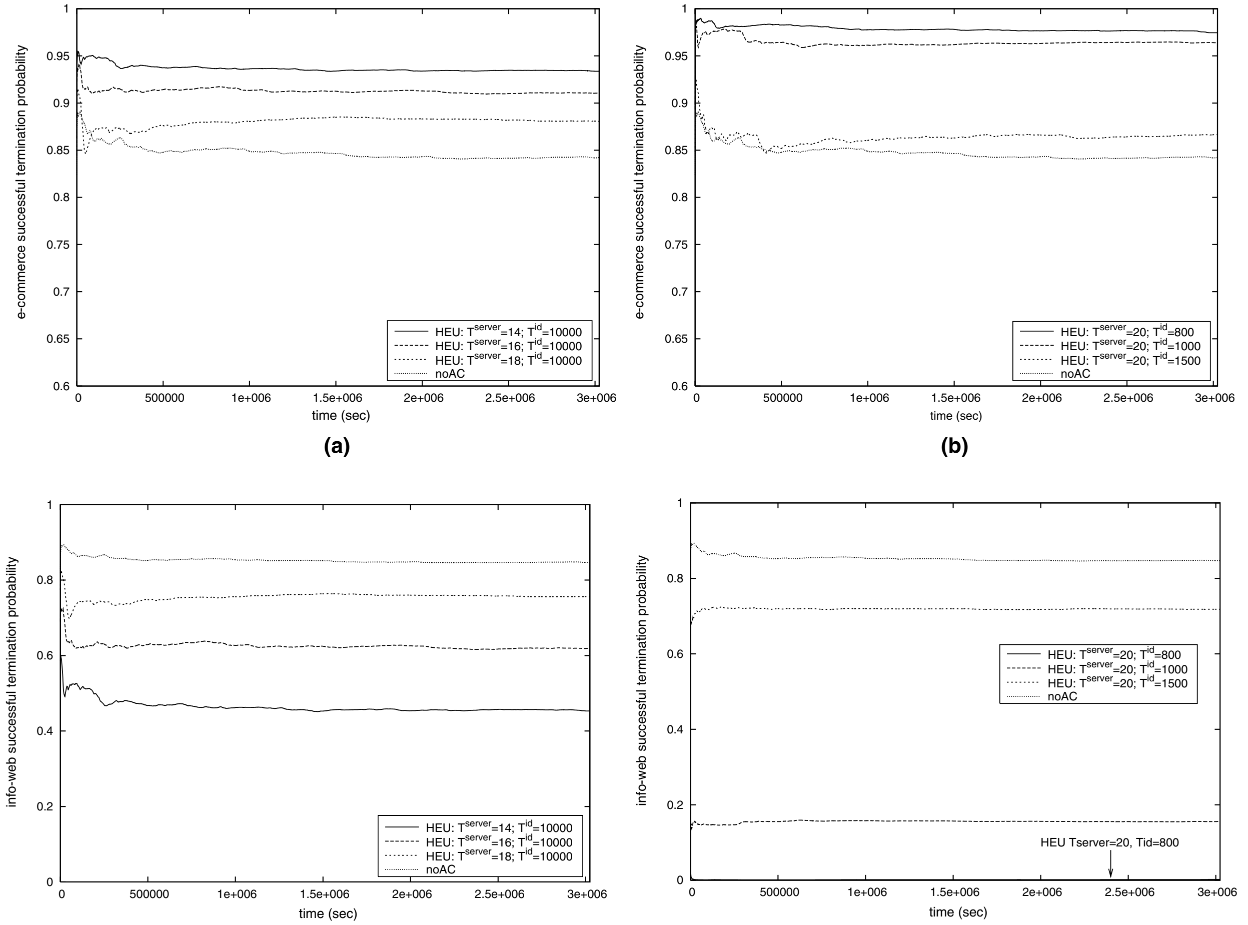


Fig. 8. Session blocking probability (scenario 1).



(a) **(b)**

Fig. 9. Rate of e-commerce session successful termination with a transaction (scenario 2).



(a) **(b)**

(c) **(d)**

Fig. 10. Successful termination probability of e-commerce and info-web sessions (scenario 2).

$\pi^1_{BA} = 0.8, \pi^2_{BA} = 0.3, \pi^2_{BC} = 0.6, \pi^2_{CD} = 0.9, \pi^2_{DC} = 0.6,$ $\pi^2_{DE} = 0.3$, $\pi^2_{ED} = 0.2$, $\pi^2_{EF} = 0.7$.

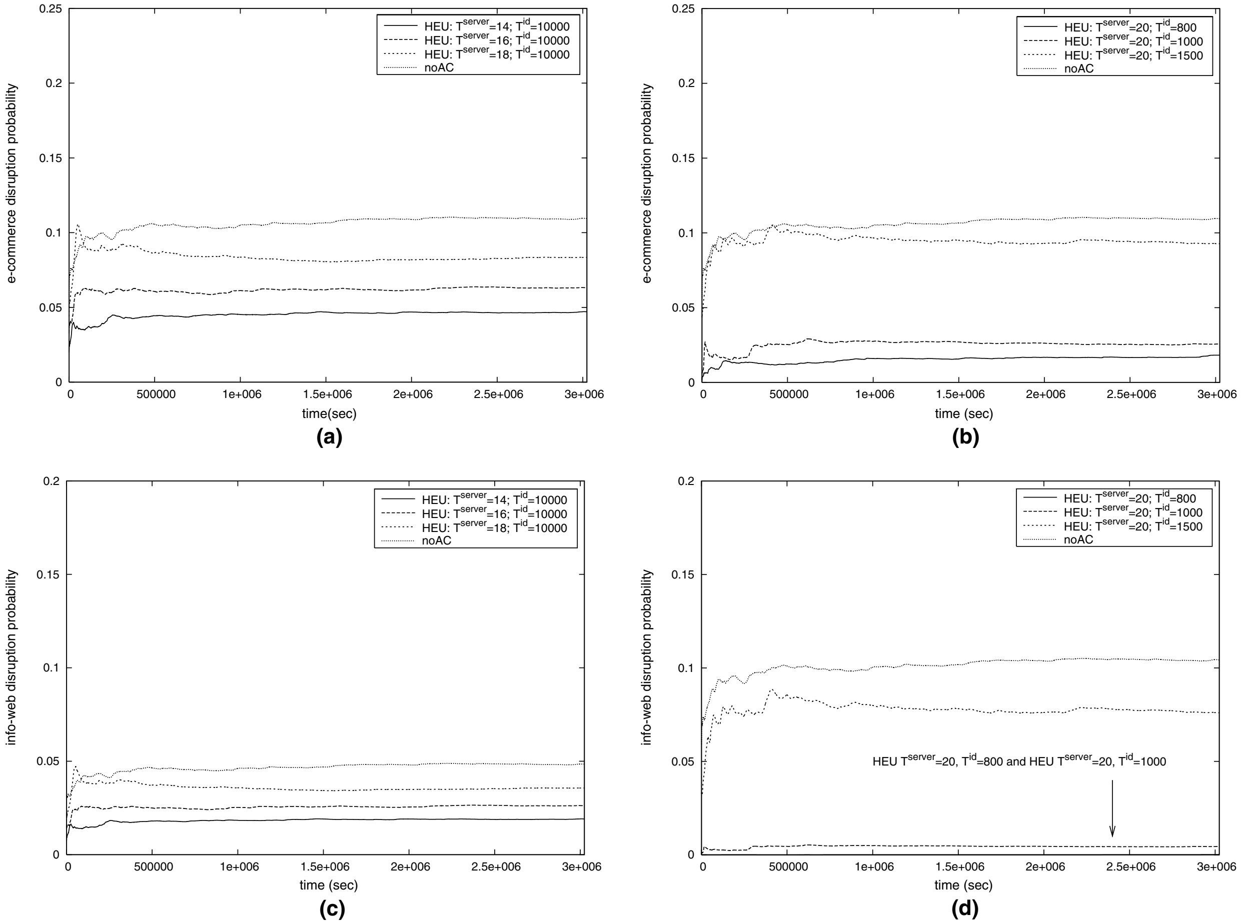
We do not show comparisons between heuristics and the optimal solution for this scenario, because the cardinality of the state space of the process is so high that we cannot compute the optimal policy within a reasonable time. We apply the intuition we obtained from the structural analysis of the optimal solution of the previous scenario, therefore we test the HEU heuristics even in the new scenario, using only noAC as a benchmark for comparisons.

In Fig. 9, we show that reserving a small amount of resources in terms of both computational resources (a) and session identifiers (b) leads to an increased rate of successful e-commerce termination with a transaction.

Of course this increase happens at the expense of other performance parameters as we can see in Fig. 10. This figure shows that an improvement in the probability of successful termination of one class, has a negative side effect on the same performance parameter of the other class. A trade-off solution between the performance of the different classes must be found and to this purpose a correct evaluation of costs and benefits is of primary importance. This evaluation is strictly application dependent. In an e-commerce site, the provider may be more interested in the probability that a successful termination (session terminated by the user) of a session also ends with a purchase, and the average purchase revenue can be an indicator of the tolerable performance loss for the low priority class.

Fig. 11 shows that the application of the policy HEU leads to an advantage for all the classes of requests in terms of disruption probability. By reserving resources to the high priority sessions, the HEU policy limits the average server utilization. For this reason, once a session has been admitted, it will more likely be conducted to its natural termination due to user's will rather than disrupted as it would be in a higher utilization scenario.

The session blocking probability is the performance metric that mostly represents the drawbacks of the HEU policy. Fig. 12 shows how reserving



(a) (b) (c) (d)

Fig. 11. Disruption probability of e-commerce and info-web sessions (scenario 2).

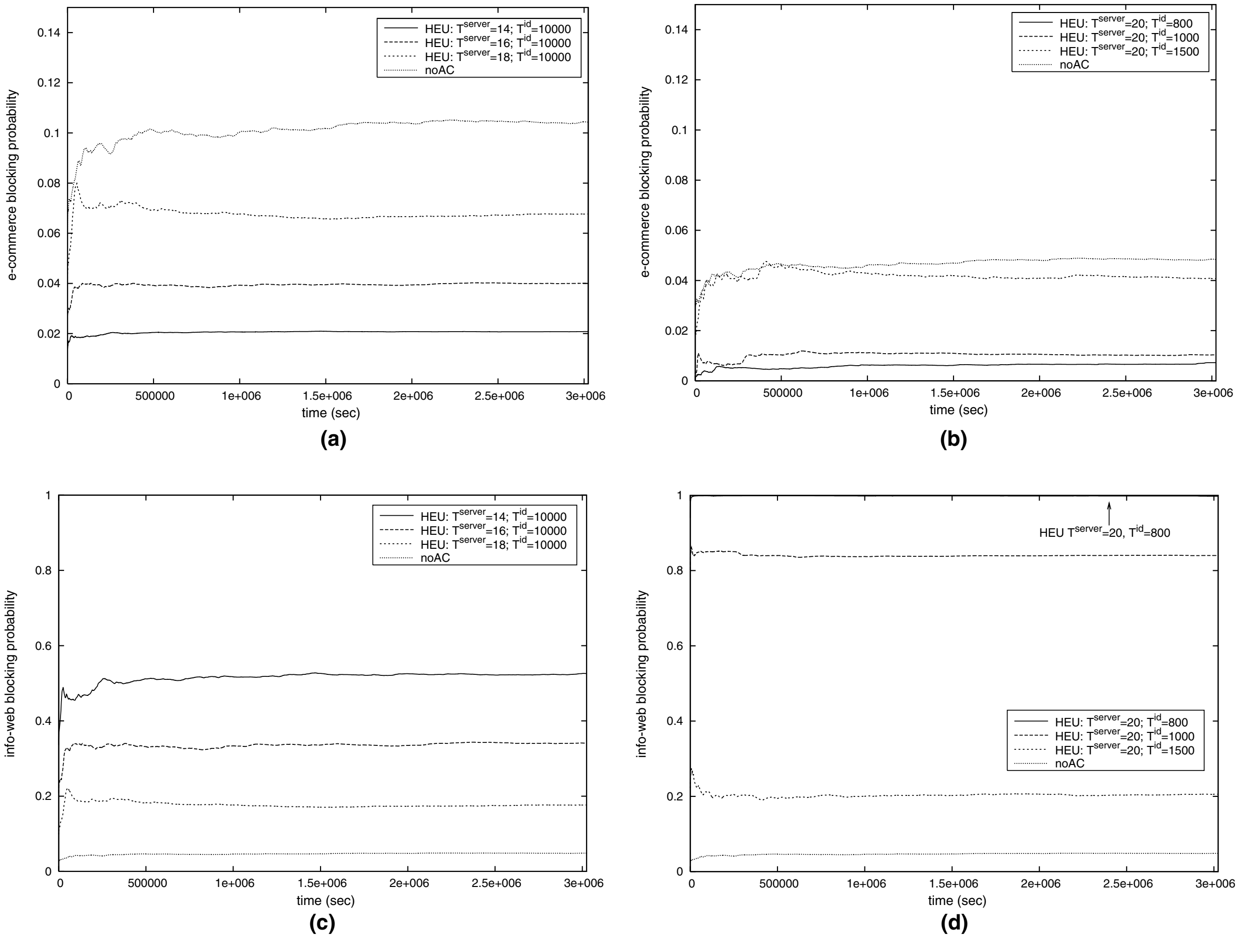


Fig. 12. Blocking probability of e-commerce and info-web sessions (scenario 2).

resources to the e-commerce class decreases its blocking probability at the expense of the blocking probability of the informational web class. It is now clear that there is no heuristic parameter tuning that can be valid for all situations; the only valid recipe is to evaluate the economic benefits and drawbacks of performance improvements and losses in the various application scenario. The benefits of the HEU policy are an improvement in the transaction rate, a decrease in the blocking probability of the high priority class and a decrease in the dropping probability of both classes. These performance improvements obviously lead to a major income from the site. The drawback of this policy is the increase of the blocking probability of the low priority stream of requests. Low priority sessions may belong to new potential users that can be discouraged from accessing the site if the experienced performance is not acceptable. The economic impact of the proposed heuristic policy must be deeply investigated to reach the best parameter tuning, that is the choice of the thresholds $T^{\mathtt{servers}}$ and $T^{\mathtt{id}}$.

9. Conclusions

This paper addresses the problem of optimizing the quality of service in geographically distributed Internet services such as those implemented by anycast based content distribution networks.

The consideration of external traffic on the non-dedicated network links is a key difference between this work and others on single server or cluster based architectures. Access control techniques, performed by session aware access points (access routers in the case of anycast based CDN, or overlay network dispatchers) are considered, on the basis of service classification and prioritization. Two session models are introduced to study typical services. A Content Delivery Network is modelled as a service center where the presence of non-dedicated links subject to external traffic is modelled as a Markov modulated vacation (or ON/OFF) process. The problem of session based access control is analyzed as a decision problem, that yields an optimal solution that shows in most cases the behavior of a congestion-dependent reserved resource policy. The structural analysis of the optimal policy shows that in most of the cases the presence of external traffic on the non-dedicated links between routers and servers has a non-negligible impact. Though computationally heavy, the study of the structure of the optimal policy gives suggestions on the formulation of possible heuristics, provides a benchmark to evaluate other policies and gives useful insight for heuristic parameter tuning. The proposed heuristics are studied by means of simulations conducted with OPNET and based on synthetic traffic generators. Simulations showed the possibility of a trade-off solution between prioritizing the e-commerce stream of requests and compromising the traffic of infor- mational web services, thus improving both the user perceived quality and the service provider revenue.

Appendix. Summary of notations

Table 4 summarizes the notations used throughout the paper.

Table 4
Summary of notations used in the paper

Notation	Description
N	Total number of session phases in the different classes
$\mathbf{x}^N$	State vector
$x_{\texttt{cong}}$	Number of congested resources
$x_{\texttt{busy}}$	Number of busy resources
$x_{\texttt{think}}$	Number of sessions in idle (think) phase
$\mathbf{x}$	Augmented state vector representation, decomposable into ($\mathbf{x}^N$,$x_{\texttt{cong}}$)
Λ	State space of the process of random variable $\mathbf{x}$
λ^{nr}_{letter}	Arrival rate of class *nr* and phase *letter* requests
λ_{nr}	Arrival rate of requests of class and phase represented by *nr*
μ^{nr}_{letter}	Completion rate of class *nr*, phase *letter* requests
$\mu_{nr} = \mu_{nr\texttt{non\_congested}}$	Completion rate of requests of class and phase represented by *nr* in absence of external traffic
$\mu_{\texttt{think}}$	Completion rate of think phases
$\bar{\mu}_{nr}$	Average figure of completion rate for phase and class denoted by *nr* in presence of external traffic
b^{nr}_{letter}	Resource consumption of class *nr*, phase *letter* requests
b_{nr}	Resource consumption of requests of class and phase represented by *nr*
$\pi^{nr}_{letter1,letter2}$	Transition probability of class *nr* sessions from phase *letter*1 to phase *letter*2
C	Number of available resources
$C^{\texttt{ID}}$	Number of available session identifiers
$\alpha_{\texttt{cong}}$	Maximum tolerable percentage delay due to external traffic according to QoS agreement
$T_{\texttt{congested}}$	Response time through a congested link
$T_{\texttt{non\_congested}}$	Response time through a non-congested link
$\mu_{\texttt{AP}}$	Congestion arrival rate on a server
$\lambda_{\texttt{AP}}$	Congestion departure rate from a server
$\beta_{\texttt{congested}}$	Percentage of congested servers in use
$\beta_{\texttt{non\_congested}}$	Percentage of non-congested servers in use
$\mathbf{a}$	Decision vector
$\mathscr{A}$	Decision space
$\mathbf{e}_i$	Identity vector
$\mathscr{S}$	Set of feasible couples of vectors (*state,decision*)
$\tau(\mathbf{x},\mathbf{a})$	Average dwell time in state $\mathbf{x}$ when decision $\mathbf{a}$ is taken
Γ	Uniformization rate
$\tilde{p}^{\mathbf{a}}_{\mathbf{xy}}$	Uniformized transition probability from state $\mathbf{x}$ to state $\mathbf{y}$ under decision $\mathbf{a}$
$\mathscr{I}_i(\mathbf{x})$	Set of arrival phases that cannot be reached from state $\mathbf{x}$ due to congestion
$r_{\texttt{cost}}(\mathbf{x},a)$	Cost function if decision $\mathbf{a}$ is taken, being in state $\mathbf{x}$
$r_{\texttt{rew}}(\mathbf{x},a)$	Reward function if decision $\mathbf{a}$ is taken, being in state $\mathbf{x}$
$x_{\mathbf{sa}}$	Probability to be in state $\mathbf{s}$ and contemporarily to take decision $\mathbf{a}$
$K_{\texttt{reserved\_servers}}$	Number of reserved servers according to the HEU policy
$K_{\texttt{reserved\_id}}$	Number of reserved session identifiers according to the HEU policy
$T^{\texttt{servers}}$	Server threshold according to the HEU policy
$T^{\texttt{ID}}$	Identifier threshold according to the HEU policy

References

[1] D.C. Verma, Content Distribution Networks, John Wiley & Sons Inc., 2002.

[2] N. Bartolini, E. Casalicchio, A walk through content delivery networks, Lecture Notes on Computer Science 2965 (2004).

[3] K.S. Candan, W.S. Li, Q. Luo, W.P. Hsiung, D. Agrawal, Enabling dynamic content caching for database driven web sites, in: Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data, Santa Barbara, CA, USA, 2001, pp. 532–543.

[4] L. Cherkasova, P. Phaal, Session based admission control: a mechanism for peak load management of commercial web sites, IEEE Transactions on Computers 51 (6) (2002).

[5] H. Chen, P. Mohapatra, Session-based overload control in qos-aware web servers, in: Proceedings of IEEE INFOCOM, 2002.

[6] J. Carlstrom, R. Rom, Application aware admission control and scheduling in web servers, in: Proceedings of IEEE INFOCOM, 2002.

[7] W. Fischer, K. Meier-Hellstern, The Markov-modulated poisson process cookbook, Performance Evaluation 18 (2) (1993).

[8] T. Yoshihara, S. Kasahara, Y. Takahashi, Practical time-scale fitting of self-similar traffic with Markov-modulated poisson process, in: Proceedings of the 6th International Conference on Telecommunications Systems, 2001.

[9] R. Morris, D. Lin, Variance of aggregated web traffic, in: Proceedings of IEEE INFOCOM, 2000.

[10] D.P. Heyman, M.J. Sobel, Stochastic Models in Operations Research, McGraw-Hill, 1984.

[11] H.C. Tijms, Stochastic Modelels. An Algorithmic Approach, John Wiley & Sons, 1994.

[12] X. Chen, P. Mohapatra, H. Chen, An admission control scheme for predictable server response times for web accesses, in: Proceedings of WWW, 2001.

[13] X. Chen, J. Heidemann, Experimental evaluation of an adaptive flash crowd protection system, ISI-TR-2003-573 July 2003, Available from: <http://www.isi.edu/xuanc/>.

[14] M. Kihl, N. Widell, Admission control schemes guaranteeing customer qos in commercial web sites, in: Proceedings of NetCon, 2002.

[15] A. Verma, S. Ghosal, On admission control for profit maximization of networked service providers, in: Proceedings of WWW, 2003.

[16] N. Bartolini, E. Casalicchio, I. Chlamtac, Session based access control in content delivery networks in presence of congestion, in: Proceedings of QShine 2004, Dallas, TX.

[17] G. Agarwal, R. Shah, J. Walrand, Content distribution architecture using network layer anycast, in: Proceedings of IEEE Workshop on Internet Applications, 2001.

[18] M. Castro, P. Druschel, A. Kermarrec, A. Rowstron, Scalable application-level anycast for higly dynamic groups, Networked Group Communications, 2003.

[19] D. Katabi, J. Wroclawski, A framework for scalable global ip-anycast (gia), in: Proceedings of SigCom, 2000.

[20] C. Partridge, T. Mendez, W. Milliken, Host anycasting service. Available from: <http://rfc.sunsite.dk/rfc/rfc1546.html>.

[21] D. Menascé, D. Barbará, R. Dodge, Preserving qos of e-commerce sites through self-tuning: a performance model approach, in: Proceedings of ACM conference on Electronic Commerce, Tampa, Florida, 2001.

[22] K. Mase, A. Tsuno, Y. Toyama, N. Karasawa, A web server selection algorithm using qos measurement, in: Proceedings of ICC, 2001.

[23] E. Zegura, M. Ammar, Z. Fei, S. Battacharjee, Application-layer anycasting: a server selection architecture and use in a replicated web service, IEEE/ACM Transaction on Networking 8 (4) (2000).

[24] C. Papadimitriou, K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Paperback, 1998.

[25] Opnet Technologies Inc., http://www.opnet.com.

Novella Bartolini graduated with honors in 1997 and received her PhD in computer engineering in 2001 from the University of Rome, Italy. She is now assistant professor at the University of Rome.

She was researcher at the Fondazione Ugo Bordoni in 1997, visiting scholar the University of Texas at Dallas in 1999–2000 and research assistant at the University of Rome ’Tor Vergata’ in 2000–2002. Her research interests lie in the area of wireless mobile networks and content delivery systems.