Egyetemi doktori (PhD) értekezés

Proxy Cache szerverek hatékonyságának vizsgálata

Performance Modeling of Proxy Cache Servers

Bérczes Tamás

Témavezető: Prof. Dr. Sztrik János

Debreceni Egyetem
Informatikai Tudományok Doktori Iskolája
Debrecen, 2011

Tartalomjegyzék

1 Bevezetés

Napjainkban az egyik leginkább közkedvelt információszerzési lehetőség az internet használata. Az internet gyors és egyszerű lehetőséget biztosít több ezer Web szerver adatainak a megismerésére, letöltésére. Az internet használata az elmúlt években rohamosan növekedett. A felhasználók száma a 2001-es 474 millióról 2002-re 590 millióra növekedett. 2006-ra az internetet használók száma elérte a 948 milliót. Figyelembe véve, hogy 1996-ban mindösszesen 40 millióan használták az internetet, a növekedés üteme igen jelentős. A felhasználók számának növekedésével párhuzamosan növekedett az internet forgalma is. Ennek hatására egyre nagyobb igény mutatkozik a színvonalas és gyors internet elérésre és kiszolgálásra. Az információ keresés és letöltés közben a válasz a távoli Web szervertől a kliens gépéig gyakran igen sok időt vesz igénybe. A probléma egyik oka, hogy ugyanabban az időben ugyanazt a fájlt más felhasználó is le akarja tölteni. Ebből adódóan ugyanazon fájlok másolatai mennek keresztül a hálózaton. Ez tulajdonképpen a kiszolgálási idő növekedését eredményezi. Természetes megoldásnak mutatkozik az információk tárolása. Ennek egyik megoldási lehetősége a böngésző szoftverbe való implementálás ([1]). Ebben az esetben a tárolt adatokhoz azonban csak egy személy férhet hozzá. Egy másik lehetőség Proxy Cache szerver használata.

A felhasználó szemszögéből nézve lényegtelen, hogy az általa keresett fájl fizikailag hol található: egy Proxy Cache szerveren valahol a munkahelyének belső hálózatán vagy a világ túlsó felén egy távoli Web szerveren. A keresett dokumentum érkezhet a távoli Web szervertől vagy a Proxy Cache szervertől. Kliens oldalról nézve a Proxy Cache szerver funkciója ugyanaz mint egy Web szervernek, valamint a Web szerver felől nézve a Proxy Cache szerver ugyanúgy viselkedik, mint egy kliens.

Felmerül a kérdés, hogy a jelneleg elérhető technológiával, vajon melyik megoldás biztosítja a legjobb eredményt. Feltételezhető, hogy egy Proxy Cache szerver beüzemelése egy cég belső hálózata és az internet közé, kisebb sávszélesség igényt valamint kisebb válszidöket eredményezhet [10]. Így a vállalatok több felhasználót kapcsolhatnak ugyanakkora sávszélességre, mivel a Proxy Cache szerver redundánsan tárolja az adatokat, több felhasználó számára.

Korábbi kutatások elsősorban a különböző "Cachelési" algoritmusok vizsgálatával illetve fejlesztésével foglalkoztak [1], [2], [14], [17].

Jelen disszertáció keretében a Proxy Cache szerverek hatékonyságát vizsgáljuk meg. Kutatásunk nem a különböző algoritmusok közötti különbségekre irányúl, hanem kifejezetten azt vizsgálja, hogy milyen környezeti feltételek mellett éri meg egy Proxy Cache szerver üzemeltetése [8], [6], [4], [7], [5].

2 Kutatási módszertan

A Proxy Cache szerverek hatékonyságának vizsgálatához különböző módszereket alkalmaztam a matematikai modellezés és a szimuláció területeiről. A Proxy Cache szerver rendszer matematikai modelljének megalkotásához a [18] és [8] modelljét vettem alapul.

Az első téziscsoportban felállítottam egy általánosított Proxy Cache szerver modellt, majd megvizsgáltam, hogy milyen paraméterértékek mellett érheti meg egy Proxy Cache szerver üzemeltetése, azaz milyen esetekben lesz egy igény teljes válaszideje alacsonyabb Proxy Cache szerver használatával, mint nélküle.

A második téziscsoportban további általánosításként feltételeztem, hogy mind a Web szerver mind pedig a Proxy Cache szerver elromolhat, azaz nem megbízhatóak. Az így kapott modell bonyolultsága miatt a válaszidők kiszámításához a MOSEL (Modeling, Specification and Evaluation Language) [3] programcsomagot használtam. A MOSEL egy leíró nyelv, mely segítségével különböző programcsomagokat használhatunk, mint például az SPNP-t (Stochastik Petri Net Package) vagy a TimeNet programcsomagot. A MOSEL által szolgáltatott eredményeket grafikusan is ábrázolni tudjuk az IGL

(Intermediate Graphical Language) segítségével, mely része a MOSEL-nek.

A harmadik téziscsoportban az érkezési folyamat egy úgynevezett "GI - General inter-arrival" folyamat, melyet az érkezési időközök várható értékével és a reletív szórásnégyzetével (c^2) jellemzünk, valamint a kiszolgálási idő bármilyen általános eloszlású lehet. Az így kapott modellben a rendszerparaméterek kiszámításához a GI/G/1 approximációt használtam, mely egy példa a Paraméter dekompozíciós eljárás használatára [9]. Az aproximáció validálásához egy szimulációs programot készítettem.

A negyedik téziscsoportban megvizsgáltam, milyen hatással van a heterogén forgalom a Proxy Cache szerver hatékonyságára. Ebben az esetben a keresett fájlokat a méretük alapján két osztályba soroljuk.

3 Új eredmények

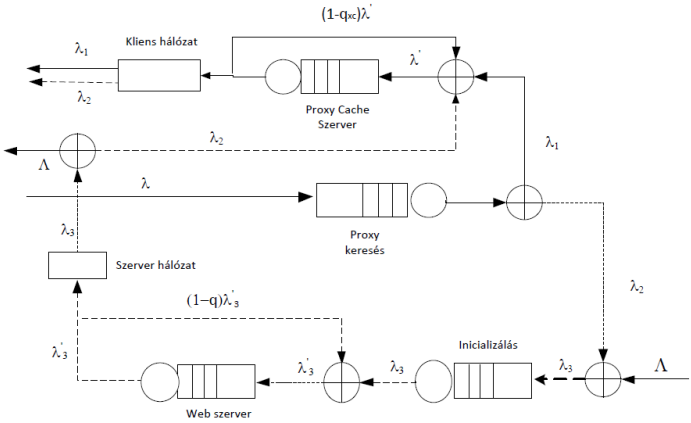
3.1. Proxy Cache szerver modell

Proxy Cache szervert használva, ha egy fájlt le akarunk tölteni egy távoli Web szerverről, először meg kell vizsgálni, hogy a keresett fájl megtalálható-e a Proxy Cache szerveren. Ennek a valószínűségét p-vel jelöljük. Amennyiben a keresett dokumentum megtalálható a Proxy Cache szerveren, egy másolat a fájlról azonnal továbbítódik a felhasználónak. Amennyiben a dokumentum nem található meg a Proxy Cache szerveren, az igény továbbítódik a távoli Web szerverhez. A dokumentum a Web szerverről először a Proxy Cache szerverre érkezik vissza, ahonnan egy másolat a fájlról azonnal a felhasználóhoz kerül. Az eredeti példány tárolódik a Proxy Cache szerveren, így a későbbiekben elérhető lesz a felhasználók számára.

1. Tézis (J2, J3, C1, O2). [1] *(4. Fejezet)*[2] *Megvizsgáltam a 3.1 grafikonon szereplő Proxy Cache szerver modellt, mely egy tetszőleges igény útját ábrázolja a felhasználótól kiindulva egészen a visszaérkezésig. Feltételeztem, hogy az igények a*

[1] A folyóiratcikkeket "J", a konferenciacikkeket "C", míg az egyéb közleményeket "O" kezdőbetűk jelölik

[2] A hivatkozások a disszertáció aktuális fejezetére vonatkoznak



3.1. ábra. Proxy Cache szerver modellje

Proxy Cache szerverhez λ paraméterű Poisson-folyamat szerint érkeznek, és a Web szerverhez kívülről érkező igények Λ paraméterű Poisson-folyamat alapján érkeznek, valamint mind a Proxy Cache szerver mind pedig a Web szerver kiszolgálási ideje exponenciális eloszlású valószínűségi változó. Megmutattam, hogy a fenti modellben Proxy Cache szervert használva

egy tetszőleges belső igény teljes válszideje:

$$\begin{aligned} T_{xc} &= \frac{1}{\frac{1}{I_{xc}}-\lambda} + p\left\{\frac{\frac{F}{B_{xc}}}{\frac{1}{\left(Y_{xc}+\frac{B_{xc}}{R_{xc}}\right)}-\frac{\lambda}{q_{xc}}} + \frac{F}{N_c}\right\} \\ &+ (1-p)\left\{\frac{1}{\frac{1}{I_s}-\lambda_3} + \frac{\frac{F}{B_s}}{\frac{1}{\left(Y_s+\frac{B_s}{R_s}\right)}-\frac{\lambda_3}{q}}\right. \\ &\left. + \frac{F}{N_s} + \frac{\frac{F}{B_{xc}}}{\frac{1}{\left(Y_{xc}+\frac{B_{xc}}{R_{xc}}\right)}-\frac{\lambda}{q_{xc}}} + \frac{F}{N_c}\right\} \end{aligned} \tag{3.1}$$

valamint egy belső igény válaszideje Proxy Cache szerver használata nélkül:

$$T = \frac{1}{\frac{1}{I_s}-(\lambda+\Lambda)} + \frac{\frac{F}{B_s}}{\frac{1}{\left(Y_s+\frac{B_s}{R_s}\right)}-(\lambda+\Lambda)/q} + \frac{F}{N_s} + \frac{F}{N_c} \tag{3.2}$$

ahol a használt jelölések a 3.1 táblázatban szerepelnek.

Megvizsgáltam a Proxy Cache szerver hatékonyságát a belső valamint külső érkezési intenzitások függvényében, valamint vizsgáltam a keresett fájl méretének és a Proxy Cache szerver találati valószínűségének hatását a válaszidőre. Megállapítottam, hogy mind a belső mind pedig a külső érkezési intenzitás növelésével a válaszidők nőnek, függetlenül a Proxy cache szerver jelenlététől. Abban az esetben ha a belső igények érkezési intenzitása magas, kisebb találati valószínűség esetén is alacsonyabb válaszidőket kapunk Proxy Cache szerver használatával.

3.1. táblázat. **A Proxy Cache szerver modell jelölései**

λ:	A belső igények érkezési intenzitása
Λ:	A külső igények érkezési intenzitása
F:	Az átlagos fájl méret (byte-ban)
p:	A találati valószínűség
B_{xc}:	A Proxy Cache szerver kimenő puffere (byte-ban)
I_{xc}:	A Proxy Cache szerver keresési ideje (másodpercben)
Y_{xc}:	A PCS statikus szerver ideje (másodpercben)
R_{xc}:	A dinamikus szerver arány a Proxy Cache szerveren
N_c:	Kliens oldali sávszélesség (bit/másodperc)
B_s:	Web szerver kimenő puffere (byte-ban)
I_s:	Inicializálási idő (másodpercben)
Y_s:	A Web szerver statikus szerver ideje (másodperc)
R_s:	A Web szerver dinamikus szerver aránya
N_s:	Szerver oldali sávszélesség (bit/másodperc)

Míg amennyiben a találati valószínűség nagy, a válaszidők alacsony érkezési intenzitás mellett is alacsonyabbak Proxy Cache szerver használatával.

3.2. A Web szerver és a Proxy Cache szerver meghibásodásának hatása a válszidőre

A korábban ismertetett Proxy Cache szerver modellt általánosítottam úgy, hogy egy méginkább valósághű modellt kapjunk. A korábbiakban mind a Proxy Cache szerver, mind pe-

dig a távoli Web szerver megbízhatóak voltak, most pedig feltesszük, hogy egyik sem megbízható, azaz bármelyikük elromolhat. Az általánosítással az a célunk, hogy megvizsgáljuk a szerverek meghibásodásának hatását a rendszerparaméterekre. A továbbiakban vizsgálni fogjuk a teljesítménybeli különbségeket, amennyiben "blokkolt" valamint "intelligens" forrásokat feltételezünk.

2. Tézis (J6, J5, O2). *(5. Fejezet) A vizsgált Markov lánc állapottere, mely a módosított modellt leírja túl nagy, az egyensúlyi egyenlet felírása és ezek megoldása túl bonyolult lenne. Ezért a MOSEL (Modeling, Specification and Evaluation Language) [3] programcsomagot használtam a modell leírására valamint a rendszerjellemzők kiszámítására. Feltételeztem, hogy a Proxy Cache szerver és a Web szerver meghibásodhat a* $(t, t+dt)$ *intervallumban* $\delta_{pcs}dt+o(dt)$ *valamint* $\delta_{web}dt+o(dt)$ *valószínűséggel ha szabadok, valamint* $\gamma_{pcs}dt+o(dt)$ *és* $\gamma_{web}dt+o(dt)$ *valószínűséggel ha foglaltak. Ha a Proxy Cache szerver vagy a Web szerver foglalt állapotban romlanak el, akkor a megszakadt igény feldolgozása a javítás befejezése után folytatódik. A javítási idő exponenciális eloszlású* $1/\nu_{pcs}$ *és* $1/\nu_{web}$ *átlaggal. Ha a szerverek közül valamelyik elromlik két különböző esetet különböztettem meg:*

- ***Blokkolt eset**: a szerver meghibásodása alatt nem érkezik új igény a szerverhez.*
- ***Nem blokkolt eset**: a szerver meghibásodása alatt is érkezhetnek újabb igények a szerverhez.*

A rendszer állapotát a t időpillanatban a

$$X_{PCS}(t) = (Y_{PCS}(t), C_{PCS}(t), Q_{PCS}(t)) \tag{3.3}$$

valamint a

$$X_{Web}(t) = (Y_{Web}(t), C_{Web}(t), Q_{Web}(t)) \tag{3.4}$$

folyamat írja le, ahol $Y_{PCS}(t) = Y_{Web}(t) = 0$ *ha a szerver működik, és* $Y_{PCS}(t) = Y_{Web}(t) = 1$ *ha a szerver hibás, valamint* $C_{PCS}(t) = C_{Web}(t) = 0$ *ha a szerver nem foglalt, valamint* $C_{PCS}(t) = C_{Web}(t) = 1$ *ha a szerver foglalt. Legyen* $Q_{PCS}(t)$ *és* $Q_{Web}(t)$ *a pufferben lévő igények átlagos száma a Proxy Cache szerver valamint a Web szerver esetén.*

Legyenek a stacionárius valószínűségek:

$$P_{PCS}(q, r, j) = \lim_{t\to\infty} P(Y_{PCS}(t), C_{PCS}(t), Q_{PCS}(t)), \tag{3.5}$$

$$q = 0, 1, r = 0, 1, j = 0, \cdots, K_{PCS},$$

és

$$P_{Web}(q, r, j) = \lim_{t\to\infty} P(Y_{Web}(t), C_{Web}(t), Q_{Web}(t)), \tag{3.6}$$

$$q = 0, 1, r = 0, 1, j = 0, \cdots, K_{Web},$$

ahol K_{PCS} *és* K_{Web} *a szerverek puffer mérete.*

A MOSEL programcsomagot használva meghatároztam a válaszidőket. Megvizsgáltam a Proxy Cache szerver és a Web

szerver különböző meghibásodási és javítási paramétereinek hatását a válszidőre mind foglalt mind pedig szabad szerverek esetében.

3.3. A Proxy Cache szerver GI/G/1 modellje

Ebben a modellben az érkezési folyamat egy úgynevezett "GI - General inter-arrival" folyamat, melyet az érkezési időközök várható értékével és a relatív szórásnégyzetével (c^2) jellemzünk, valamint a kiszolgálási idő bármilyen általános eloszlású lehet. Az approximáció használatához a következő feltételeknek kell tejesülniük:

- Az érkezési folyamat úgynevezett "felújítási" folyamat, azaz az érkezési időközök független, azonos eloszlású valószínűségi változók.
- A kiszolgálási idők valószínűségi változója bármilyen általános eloszlású lehet.
- Adott az érkezési folyamat inetnzitása λ_A, valamint az érkezási folyamat relatív szórásnégyzete (c_A^2).
- Adott a kiszolgálási idő várható értéke τ_S, valamint a kiszolgálási idő relatív szórásnégyzete (c_S^2).
- Az azonnali visszacsatolást amikor egy sor távozó folyamata vissza van irányítva egyből ugyanahoz a sorhoz, külön kell vizsgálni.

Ez az aproximáció olyan algoritmusokat szolgáltat, melyekkel modellezhetjük az általános hálózati folyamatokat, mint például a forgalom egyesítést, a sortól való távozást, valamint a forgalom szétválását.

Minden esetben, a részletes számítások előtt a modellt módosítanunk kell, hogy eliminálhassuk az azonnali visszacsatolásokat. Ezt a módosítást az érintett sor kiszolgálási idejének megváltoztatásával végezzük.

A szükséges kalkulációk eredményeképp az aproximáció segítségével megkapjuk a szükséges rendszer jellemzőket (átlagos sorhossz, átlagos várakozási idő, stb.), mind a sorokra, mind pedig az egész hálózatra vonatkozóan.

Az approximációs eljárás részletes bemutatása megtalálható a disszertációban valamint a [9] könyvben.

3. Tézis (J4). *(6. Fejezet) Módosítottam a 3.1 grafikonon szereplő Proxy Cache szerver modellt, hogy alkalmazni lehessen rá a GI/G/1 approximációt. Újrakalkuláltam a módosított modellben a szükséges rendszerjellemzőket, hogy megkapjam a keresett válaszidőket. Az így kapott eredmények validálására egy szimulációs programot készítettem, mely segítségével ellenőrizhető az approximáció helyessége. Megvizsgáltam két különböző esetet. Abban az esetben amikor a használt relatív szórásnégyzetek egynél kisebbek illetve azt az esetet amikor a használt relatív szórásnégyzetek egynél nagyobbak. Megállapítottam, hogy a relatív szórásnégyzet növelésével a használt approximáció pontossága romlik.*

3.4. A heterogén forgalom hatása a Proxy Cache szerverek hatékonyságára

Módosítottam az eredeti Proxy cache szerver modellt, hoggy vizsgálni lehessen a heterogén forgalom hatását a Proxy Cache szerverek hatékonyságára. A kliensek által keresett fájlokat méretük alapján két osztolyba soroltam. Az átlagosnál nagyobb méretű fájlok az a, míg a kis méretű fájlok a b osztályba kerülnek. Mindkét osztályba tartozó igény esetén először megvizsgáljuk, hogy a fájl megtalálható-e a Proxy Cache szerveren vagy sem. Ezt a találati valószínűséget p_a illetve p_b -vel jelöljük az a illetve b osztályba tartozó fájlok esetén. Amennyiben a keresett fájl megtalálható a Proxy Cache szerveren, akkor mindkét osztály esetén a fájl egy másolata azonnal továbbítódik a klienshez. Ellenkező esetben, amikor is a fájl nem található meg a Proxy Cache szerveren az igény továbbítódik a távoli Web szerverhez függetlenül az osztályától. Miután az igényelt fájl visszaérkezik a Proxy Cache szerverhez egy másolat továbbítódik a klienshez.

4. Tézis (J1). *(7. Fejezet) A 3.2 ábra mutatja egy belső fájl lehetséges útját az igény indulásától egészen a fájl klienshez való megérkezéséig. Az ábrán az a illetve b index jelzi, hogy az adott igény melyik osztályba tartozik. A használt jelölések megtalálhatóak a 3.1 és 3.2 táblázatban. Feltételezzük, hogy mindkét osztályhoz tartozó igények a Proxy Cache szerverhez Poisson-folyamat szerint érkeznek, és a Web szerverhez kívülről érkező igények szintén Poisson-folyamat alapján érkeznek, valamint mind a Proxy Cache szerver mind pedig a Web szerver kiszolgálási ideje független exponenciális eloszlású*

valószínűségi változó. Az a illetve a b osztályhoz tartozó belső igények válaszideje T_a^{xc}, illetve T_b^{xc} ahol:

$$
\begin{aligned}
T_a^{xc} = & \frac{1}{\frac{1}{I_{xc}} - (\lambda_a + \lambda_b)} \\
& + p_a * \left\{ \frac{\frac{1}{q_{a,xc}} * (Y_{xc} + \frac{B_{xc}}{R_{xc}})}{1 - \sum_{j=a}^{b} \frac{\lambda_j}{q_j} (Y_{xc} + \frac{B_{xc}}{R_{xc}})} + \frac{F_a}{N_c} \right\} \\
& + (1 - p_a) * \left\{ \frac{1}{\frac{1}{I_s} - (\lambda_{a,3} + \lambda_{b,3})} + \frac{\frac{1}{q_a} * (Y_s + \frac{B_s}{R_s})}{1 - \sum_{j=a}^{b} \frac{\lambda_{j,3}}{q_j} (Y_s + \frac{B_s}{R_s})} \right. \\
& \left. + \frac{F_a}{N_s} + \frac{\frac{1}{q_{a,xc}} * (Y_{xc} + \frac{B_{xc}}{R_{xc}})}{1 - \sum_{j=a}^{b} \frac{\lambda_j}{q_{j,xc}} (Y_{xc} + \frac{B_{xc}}{R_{xc}})} + \frac{F_a}{N_c} \right\},
\end{aligned}
\tag{3.7}
$$

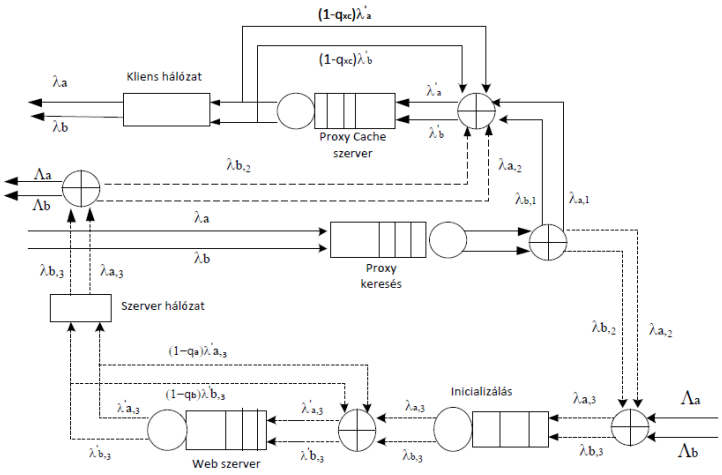
és

$$
\begin{aligned}
T_b^{xc} = & \frac{1}{\frac{1}{I_{xc}} - (\lambda_a + \lambda_b)} \\
& + p_b * \left\{ \frac{\frac{1}{q_{b,xc}} * (Y_{xc} + \frac{B_{xc}}{R_{xc}})}{1 - \sum_{j=a}^{b} \frac{\lambda_b}{q_{b,xc}} (Y_{xc} + \frac{B_{xc}}{R_{xc}})} + \frac{F_b}{N_c} \right\} \\
& + (1 - p_b) * \left\{ \frac{1}{\frac{1}{I_s} - (\lambda_{a,3} + \lambda_{b,3})} + \frac{\frac{1}{q_b} * (Y_s + \frac{B_s}{R_s})}{1 - \sum_{j=a}^{b} \frac{\lambda_{j,3}}{q_j} (Y_s + \frac{B_s}{R_s})} \right. \\
& \left. + \frac{F_b}{N_s} + \frac{\frac{1}{q_{b,xc}} * (Y_{xc} + \frac{B_{xc}}{R_{xc}})}{1 - \sum_{j=a}^{b} \frac{\lambda_b}{q_{b,xc}} (Y_{xc} + \frac{B_{xc}}{R_{xc}})} + \frac{F_b}{N_c} \right\},
\end{aligned}
\tag{3.8}
$$

Így a teljes válaszidő:

$$T_{xc} = \frac{\lambda_a}{\lambda_a + \lambda_b} * T_a^{xc} + \frac{\lambda_b}{\lambda_a + \lambda_b} * T_b^{xc} \tag{3.9}$$

Megvizsgáltam a Proxy Cache szerver hatékonyságát a belső valamint külső érkezési intenzitások függvényében, valamint vizsgáltam a multimédia és normál fájl méretének és a multimédia tartalom arányának hatását a válaszidőre. Megmutattam, hogy mind a belső mind pedig a külső érkezési intenzitás növelésével a válaszidők nőnek, függetlenül a Proxy Cache szerver jelenlététől. Amennyiben az a osztályos kérések arányát növeljük a válaszidők szintén nőnek, valamint magas a osztály arányt használva már alacsonyabb érkezési intenzitás esetén is megéri a Proxy Cache szerver használata. Alacsony a osztály arány, alacsony érkezési intenzitás és alacsony találati valószínűség használatával Proxy Cache szerverrel magasabb válaszidőket kapunk mint nélküle.



3.2. ábra. Proxy Cache szerver heterogén modellje

3.2. táblázat. **Heterogén forgalmi modell paraméterei**

λ_a:	belső a osztályos igények érkezési intenzitása
λ_b:	belső b osztályos igények érkezési intenzitása
Λ_a:	külső a osztályos igények érkezési intenzitása
Λ_b:	külső b osztályos igények érkezési intenzitása
F_a:	az a osztályhoz tartozó fájlok mérete (byte-ban)
F_b:	a b osztályhoz tartozó fájlok mérete (byte-ban)
p_a:	találati valószínűség az a osztály esetén
p_b:	találati valószínűség a b osztály esetén

4 Introduction

The World Wide Web (WWW) can give a quick and easy access to a large number of web servers where users can find all kind of information, documents and multimedia files. From the user's point of view it does not matter whether the requested files are on the firm's computer or on the other side of the world. The usage of the web has been growing very fast. The number of internet users increased from 474 million in 2001 to 590 million in 2002, and in 2006 was 948 million users. According to the fact, that in 1996 the number of users was only 627000, the growth is rapid and we can justify an exponential grow in the traffic, too. The users want to get a high quality service and modest response time. The answer from the remote web server to the client often takes a long time. One of the problems is that the same copy of the file can be claimed by other users at the same time. Because of this situation, identical copies of many files pass through the same network links, resulting in an increased response time. A natural solution to avoid this situation is to store this information close to the clients. In general caching can be implemented at browser software and the boundary between the local area network and the Internet. Browser cache are inefficient since they cache for only one user. It has been suggested that the greatest improvement in response time for corporations will

come from installing a Proxy Cache server at the boundary between the local area network and the Internet. Requested documents can be delivered directly from the web server or through a Proxy Cache server. A Proxy Cache server has the same functionality as a web server when looked at from the client and the same functionality as a client when looked at from a web server. The primary function of a Proxy Cache server is to store documents close to the users to avoid retrieving the same document several times over the same connection.

It has been suggested that, given the current state of technology, the greatest improvement in response time for corporations will come from installing a Proxy Cache server at the boundary between the corporate LAN and the Internet. The primary benefits include lower bandwidth requirements and faster response times. Corporations can accommodate more users with a given Internet connection capacity since the Proxy Cache server can satisfy redundant requests from different users. Delivering duplicate requests directly from the Proxy Cache server at LAN speed also improves the response time. This type of server is the primary focus of this dissertation.

5 Methodology

To analyze the performance of Proxy Cache servers, several tools are applied from the field of mathematical modeling and simulation. To create the mathematical model of the Proxy Cache servers I have used the models described in [18] and [8].

In the first thesis I described the mathematical model of Proxy Cache servers and I studied the conditions under which installing a Proxy Cache server becomes beneficial. I also analyze how various factors affect the performance of a Proxy Cache server.

The purpose of the second thesis is to generalize the performance model of the Proxy Cache server using a more realistic case when the Proxy Cache server and the remote Web server are unreliable. Because of the fact, that the state space of the describing Markov chain is very large, it is difficult to calculate the system measures in the traditional way of writing down and solving the underlying steady-state equations. To simplify this procedure I used the software tool MOSEL (Modeling, Specification and Evaluation Language), see [3], to formulate the model and to obtain the performance measures. By the help of MOSEL we can use various performance tools

(like SPNP Stochastic Petri Net Package) to get these characteristics. The results of the tool can graphically be displayed using IGL (Intermediate Graphical Language) which belongs to MOSEL.

In the third thesis the arrival process is a general (GI) arrival process characterised by a mean arrival rate and a squared coefficient of variation (SQV) of the inter-arrival time and the service time may have any general distribution. To obtain the response times I used the GI/G/1 approximation which is an example of a method using Parametric Decomposition. To validate the approximation I created simulation program.

The focus of the fourth thesis is to examine the performance behavior of a Proxy Cache server when we use heterogeneous traffic. In this thesis we describe the modified multi-class queuing network model of the Proxy Cache server. In this case we separate the requests in two classes by virtue of their size.

6 New results

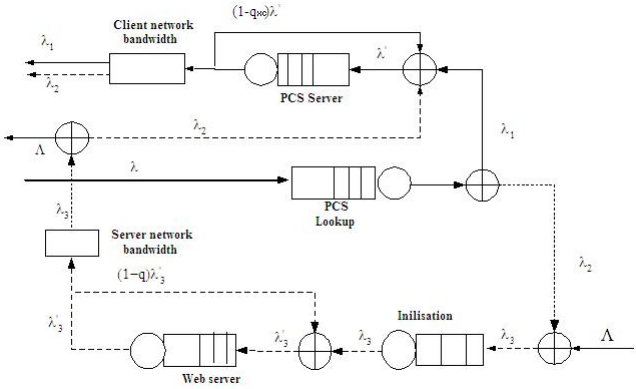
6.1. The Proxy Cache server model

Using Proxy Cache server, if any information or file is requested to be downloaded, first it is checked whether the document exists on the Proxy Cache server or not. (We denote the probability of this existence by p). If the document can be found on the Proxy Cache server then its copy is immediately transferred to the user. In the opposite case the request will be sent to the remote Web server. After the requested document arrived back to the Proxy Cache server then a copy of it is delivered to the user.

1. Thesis (J2, J3, C1, O2). [1] *(Chapter 4.)*[2] *I described the model shown on the figure 6.1, which illustrates the path of a request starting from the user and ending with the return of the answer to the user. We assume that the requests of the Proxy Cache server users arrive according to a Poisson process with rate λ, and the external requests arrive to the Web server according to a Poisson process with rate Λ, respectively,*

[1] References for conference papers start with letter "C", journals with "J" and others with "O"

[2] Here, the regarding sections of the Dissertation are referred.



6.1. Figure. Proxy Cache server model

and the Proxy Cache server and Web server have an exponentially distributed service time distribution. I have shown, that the overall response time in presence of Proxy Cache server is given by:

$$
\begin{aligned}
T_{xc} \;=\; & \frac{1}{\frac{1}{I_{xc}}-\lambda} + p\left\{ \frac{\frac{F}{B_{xc}}}{\frac{1}{\left(Y_{xc}+\frac{B_{xc}}{R_{xc}}\right)}-\frac{\lambda}{q_{xc}}} + \frac{F}{N_c} \right\} \\
& + (1-p)\left\{ \frac{1}{\frac{1}{I_s}-\lambda_3} + \frac{\frac{F}{B_s}}{\frac{1}{\left(Y_s+\frac{B_s}{R_s}\right)}-\frac{\lambda_3}{q}} \right. \\
& \left. + \frac{F}{N_s} + \frac{\frac{F}{B_{xc}}}{\frac{1}{\left(Y_{xc}+\frac{B_{xc}}{R_{xc}}\right)}-\frac{\lambda}{q_{xc}}} + \frac{F}{N_c} \right\}
\end{aligned}
\tag{6.1}
$$

and the overall response time without using Proxy Cache server:

$$T = \frac{1}{\frac{1}{I_s} - (\lambda + \Lambda)} + \frac{\frac{F}{B_s}}{\frac{1}{\left(Y_s + \frac{B_s}{R_s}\right)} - (\lambda + \Lambda)/q} + \frac{F}{N_s} + \frac{F}{N_c}, \quad (6.2)$$

where the notations of the most important basic parameters used are collected in Table 6.1

I analyzed how various factors affect the performance of a Proxy Cache server. These factors include the arrival rates of requests, the "cache hit rate" probability and the external arrival rates. I also examine the effect of the file size. I noticed that, when the arrival rate of requests increases, then the response times increase as well, regardless of the existence of a Proxy Cache server. When we used a high visit rate with a high cache hit rate probability, then we get smaller response time using Proxy Cache server. Increasing the visit rate for the external users, the difference between response time with and without a Proxy Cache server was smaller and smaller until this difference vanished and the existence of a Proxy Cache server resulted lower response times.

6.2. The impact of servers breakdown on the performance of Proxy Cache servers

In this chapter we generalize the performance model of the Proxy Cache server using a more realistic case when the Proxy

6.1. Table. **Notations**

λ:	arrival rate
Λ:	external arrival rate
F:	average file size (byte)
p:	cache-hit rate probability
B_{xc}:	the size of the Proxy server output buffer (in bytes)
I_{xc}:	the lookup time of the Proxy server (in seconds)
Y_{xc}:	the static server time of the Proxy server (in sec.)
R_{xc}:	dynamic server rate of the Proxy server (in byte/sec.)
N_c:	client network bandwidth
B_s:	the size of the Web server output buffer (in bytes)
I_s:	one-time initialization time (in seconds)
Y_s:	the static server time of the Web server (in seconds)
R_s:	dynamic server rate of the Web server (in byte/sec.)
N_s:	server network bandwidth

Cache server and the remote Web server are unreliable. Our aim is to illustrate graphically the effect of the non-reliability of both Proxy Cache servers and Web servers on the steady state system measures. Furthermore, we examine the difference in the performance using blocked and intelligent sources.

2. Thesis (J6, J5, O2). *(Chapter 5.) Since the state space of the describing Markov chain is very large, it is dificult to calculate the system measures in the traditional way of writing down and solving the underlying steady-state equations. To simplify this procedure we used the software tool MOSEL (Modeling, Specification and Evaluation Language), see [3], to formulate the model and to obtain the performance measures. The*

Proxy Cache server and the Web server can fail during the interval $(t, t+dt)$ *with probability* $\delta_{pcs}dt+o(dt)$ *and* $\delta_{web}dt+o(dt)$ *if they are idle, and with probability* $\gamma_{pcs}dt+o(dt)$ *and* $\gamma_{web}dt+o(dt)$ *if they are busy, respectively. If the Proxy Cache server or the Web server fails in busy state, it continues servicing the interrupted request after it has been repaired. The repair time is exponentially distributed with a finite mean* $1/\nu_{pcs}$ *and* $1/\nu_{web}$*. If one of the servers fails two different cases can be treated:*

- ***Blocked case****: during the CPU is down, no new requests may come to the server buffer.*
- ***Unblocked case****: the new requests can fill the server buffer during the breakdown, until it is full.*

All the times involved in the model are assumed to be mutually independent of each other.
The system state at time t *can be described by the processes*

$$X_{PCS}(t) = (Y_{PCS}(t), C_{PCS}(t), Q_{PCS}(t)),$$

and

$$X_{Web}(t) = (Y_{Web}(t), C_{Web}(t), Q_{Web}(t)),$$

where $Y_{PCS}(t) = Y_{Web}(t) = 0$ *if the server is up,* $Y_{PCS}(t) = Y_{Web}(t) = 1$ *if the server is failed,* $C_{PCS}(t) = C_{Web}(t) = 0$ *if the server is idle and* $C_{PCS}(t) = C_{Web}(t) = 1$ *if the server is busy, respectively. Let* $Q_{PCS}(t)$ *and* $Q_{Web}(t)$ *denote the number of requests in the buffer of the Proxy Cache server and Web server, respectively. Let us define the stationary probabilities by:*

$$P_{PCS}(q,r,j) = \lim_{t\to\infty} P(Y_{PCS}(t), C_{PCS}(t), Q_{PCS}(t)),$$
$$q = 0,1, r = 0,1, j = 0,\cdots, K_{PCS}, \tag{6.3}$$

and

$$P_{Web}(q,r,j) = \lim_{t\to\infty} P(Y_{Web}(t), C_{Web}(t), Q_{Web}(t)), \tag{6.4}$$
$$q = 0,1, r = 0,1, j = 0,\cdots, K_{Web},$$

where K_{PCS} and K_{Web} denote the buffer size of the servers.

Using Mosel I had to calculate the overall response time. I examined the effect of the non-reliability of both Proxy Cache servers and Web servers on the steady state system measures and the difference in the performance using blocked and intelligent sources.

6.3. The GI/G/1 model of a Proxy Cache server

In this model, the arrival process is a general (GI) arrival process characterised by a mean arrival rate and a squared coefficient of variation (SQV) of the inter-arrival time, and the service time may have any general distribution. In order to apply this method we assume the following:

- The arrival process to a network node is renewal, so the arrival intervals are independent, identically distributed random variables.

- The service time may have any general distribution.
- We know the parameters of the arrival process: λ_A - the mean arrival rate and c_A^2 - the SQV of the inter-arrival time.
- We know the parameters of the service time τ_S - the mean service time, and c_S^2 - the SQV of the service time.
- Immediate feedback, where a fraction of the output of a particular queue enters the queue once again, needs special treatment.

This approximation contains procedures required for modeling of the basic network operations of merging, departure and splitting, arising due to the common sharing of the resources and routing decisions in the network. Before the detailed analysis of the queueing network is done, the method first removes immediate feedback in a queue by suitably modifying its service time. The approximation provide performance measures (i.e. mean queue lengths, mean waiting times, etc.) for both per-queue and per-network.

3. Thesis (J4). *(6. Chapter) I modified the original (M/M/1) performance model of Proxy Cache server drown on figure 3.1. I modified the performance model of Proxy Cache servers to get a more powerful variant when the inter-arrival times and the service times are generally distributed. In this case we can use the GI/G/1 approximation to obtain the overall response time. I recalculated the basic performance parameters of the modified performance model using the approximation method.*

The accuracy of the new model is validated by means of a simulation study over an extended range of test cases. I studied two different cases. When the $SQV < 1$ the overall response time obtained by approximation is very close to the response time obtained by simulation; they are the same at least up to the 3rd-4th decimal digit. In case when the $SQV > 1$ the response times are the same only to 2nd-3rd decimal digit. So, using greater SQV the approximation error is greater.

6.4. The impact of heterogeneous traffic on the performance of Proxy Cache servers

The focus of the fourth thesis is to examine the performance behavior of Proxy Cache servers when we use heterogeneous traffic. In this thesis we describe the multi-class queuing network model of the Proxy Cache server, where we separate the requests in two classes by virtue of their size. If the size of the requested document is greater than average we put it into class a. In the opposite case, when the size of the requested file is small we put it into class b. In booth cases first it is checked whether the document (class a or class b) exists on the Proxy Cache server or not. We denote the probability of this existence by p_a in case of class a and by p_b in case of class b. If the document can be found on the Proxy Cache server then its copy is immediately transferred to the user. In the opposite case the request will be sent to the remote Web server. After the requested document arrived back to the Proxy Cache server then a copy of it is delivered to the user.

4. Thesis (J1). *(Chapter 7.) Figure 6.2 illustrates the path of a request in the modified model starting from the user and ending with the return of the answer to the user. In this figure the subscript a denotes the class a and the subscript b denote the class b. The notations used for the most important basic parameters are collected in Tables 6.1 and 6.2. We assume that the requests of the Proxy Cache server users for both classes arrive according to a Poisson process with rate λ_a and λ_b, and the external requests for booth classes arrive to the Web server according to a Poisson process with rate Λ_a and Λ_b, respectively, and the Proxy Cache server and Web server have an exponentially distributed service time distribution. I have shown, that the overall response time in presence of a Proxy Cache server of class a (T_a^{xc}) and class b (T_b^{xc}) is given by:*

$$
\begin{aligned}
T_a^{xc} = & \frac{1}{\frac{1}{I_{xc}} - (\lambda_a + \lambda_b)} \\
& + p_a * \left\{ \frac{\frac{1}{q_{a,xc}} * (Y_{xc} + \frac{B_{xc}}{R_{xc}})}{1 - \sum_{j=a}^{b} \frac{\lambda_j}{q_j} (Y_{xc} + \frac{B_{xc}}{R_{xc}})} + \frac{F_a}{N_c} \right\} \\
& + (1 - p_a) * \left\{ \frac{1}{\frac{1}{I_s} - (\lambda_{a,3} + \lambda_{b,3})} + \frac{\frac{1}{q_a} * (Y_s + \frac{B_s}{R_s})}{1 - \sum_{j=a}^{b} \frac{\lambda_{j,3}}{q_j} (Y_s + \frac{B_s}{R_s})} \right. \\
& \left. + \frac{F_a}{N_s} + \frac{\frac{1}{q_{a,xc}} * (Y_{xc} + \frac{B_{xc}}{R_{xc}})}{1 - \sum_{j=a}^{b} \frac{\lambda_j}{q_{j,xc}} (Y_{xc} + \frac{B_{xc}}{R_{xc}})} + \frac{F_a}{N_c} \right\},
\end{aligned}
\tag{6.5}
$$

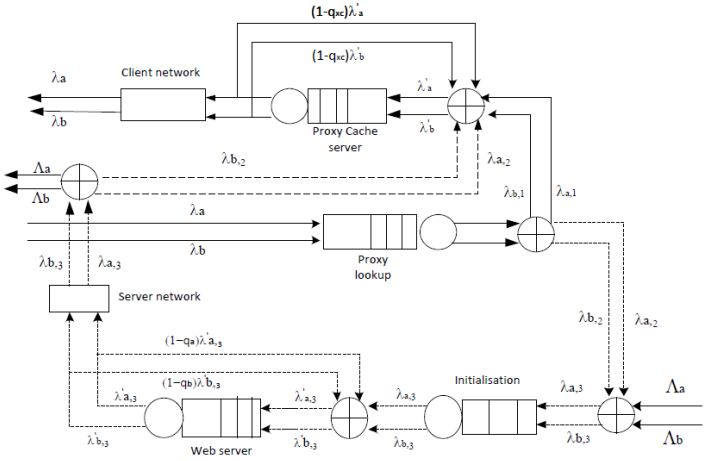
and

$$
\begin{aligned}
T_b^{xc} = & \frac{1}{\frac{1}{I_{xc}} - (\lambda_a + \lambda_b)} \\
& + p_b * \left\{ \frac{\frac{1}{q_{b,xc}} * (Y_{xc} + \frac{B_{xc}}{R_{xc}})}{1 - \sum_{j=a}^{b} \frac{\lambda_b}{q_{b,xc}} (Y_{xc} + \frac{B_{xc}}{R_{xc}})} + \frac{F_b}{N_c} \right\} \\
& + (1 - p_b) * \left\{ \frac{1}{\frac{1}{I_s} - (\lambda_{a,3} + \lambda_{b,3})} + \frac{\frac{1}{q_b} * (Y_s + \frac{B_s}{R_s})}{1 - \sum_{j=a}^{b} \frac{\lambda_{j,3}}{q_j} (Y_s + \frac{B_s}{R_s})} \right. \\
& \left. + \frac{F_b}{N_s} + \frac{\frac{1}{q_{b,xc}} * (Y_{xc} + \frac{B_{xc}}{R_{xc}})}{1 - \sum_{j=a}^{b} \frac{\lambda_b}{q_{b,xc}} (Y_{xc} + \frac{B_{xc}}{R_{xc}})} + \frac{F_b}{N_c} \right\},
\end{aligned}
\tag{6.6}
$$

We get the overall response time T_{xy} by:

$$
T_{xc} = \frac{\lambda_a}{\lambda_a + \lambda_b} * T_a^{xc} + \frac{\lambda_b}{\lambda_a + \lambda_b} * T_b^{xc} \tag{6.7}
$$

I analyzed how various factors affect the performance of a Proxy Cache server when we use heterogeneous traffic. In general when the arrival rate of requests increases, then the response time increases as well regardless of the existence of a Proxy Cache server. When we use a higher percentage of the class a and we use a high arrival rate, then the response time gap is more significant between the cases with and without a Proxy Cache server. Using a low percentage of class a files, a low arrival rate and low cache hit rate probability we get higher response time in presence of a Proxy Cache server.



6.2. Figure. Heterogeneous traffic model of Proxy Cache server

6.2. Table. **Notations of heterogeneous traffic model**

λ_a:	arrival rate for requests of class a
λ_b:	arrival rate for requests of class b
Λ_a:	external arrival rate for requests of class a
Λ_b:	external arrival rate for requests of class a
F_a:	average file size of files of class a
F_b:	average file size of files of class b
p_a:	cache hit rate probability for files of class a
p_b:	cache hit rate probability for files of class b

Irodalomjegyzék\Bibliography

[1] C. Aggarwal, J.L. Wolf, and P.S. Yu. Caching on the world wide web. *IEEE Transactions on Knowledge and Data Engineering*, 11:94–107, 1999.

[2] M.A. Arlitt and C.L. Williamson. Internet web servers: workload characterization and performance implications. *IEEErACM Transactions on Networking*, 5:631–645, 1997.

[3] K. Begain, G. Bolch, and H. Herold. *Practical performance modeling, application of the MOSEL language*. Kluwer Academic Publisher, Boston, 2001.

[4] T. Berczes. Approximation approach to performance evaluation of proxy cache server systems. *Annales Mathematicae et Informaticae*, 36:15–28, 2009.

[5] T. Berczes, G. Guta, G. Kusper, W. Schreiner, and J. Sztrik. Analyzing web server performance models with the probabilistic model checker prism. Technical report no. 08-17 in RISC Report Series, 2008.

[6] T. Berczes and J. Sztrik. Performance modeling of proxy cache servers. *Journal of Universal Computer Science*, 12:1139–1153, 2006.

[7] T. Berczes, J. Sztrik, and C.S. Kim. The impact of multimedia traffic on the performance of a proxy cache server. *Annales Univ. Sci. Budapest, Sect. Comp.*, 25:153–169, 2005.

[8] I. Bose and H.K. Cheng. Performance models of a firms proxy cache server. *Decision Support Systems and Electronic Commerce*, 29:45–57, 2000.

[9] S.K. Bose. *An introduction to queueing systems*. Kluwer Academic/Plenum Publishers, New York, 2002.

[10] S.J. Caughey, D.B. Ingham, and M.C. Little. Flexible open caching for the web. *Computer Networks and ISDN Systems*, 29:1007–1017, 1997.

[11] T. V. Do, U. R. Krieger, and R. Chakka. Performance modeling of an apache web server with a dynamic pool of service processes. *Telecommunication Systems*, 39(2):117–129, 2008.

[12] V.T Do, R. Chakka, T. Le Nhat, and O. Gemikonakli. A new performance model for web servers. *Federation of European Simulation Societies*, pages 19–21, 2004.

[13] V.T Do, R. Chakka, T. Le Nhat, and U. Krieger. Performance modelling of a web server with a dynamic pool of service processes. *Center for Mathematics and Computer Science*, pages 19–21, 2006.

[14] M. Kurcewicz, W. Sylwestrzak, and A. Wierzbicki. A filtering algorithm for web caches. *Computer Networks and ISDN Systems*, pages 2203–2209, 1998.

[15] E.D. Lazowska, J. Zahorjan, G.S. Graham, and K.C. Sevcik. *Quantitative System Performance.* Prentice Hall, 1984.

[16] D.A. Menasce and V.A.F. Almeida. *Capacity Planning for Web Performance: Metric, Models, and Methods.* Prentice Hall, 1998.

[17] P. Scheuermann, J. Shim, and R. Vingralek. A case for delayconscious caching of web documents. *Computer Networks and ISDN Systems*, 29:997–1005, 1997.

[18] L.P. Slothouber. A model of web server performance. *5th International World Wide Web Conference*, 1996.

Publikációim\Publications:

- **[J] Folyóiratcikkek\Journal papers**:

J1 T. Berczes, J. Sztrik, Kim, C.S., The impact of multimedia traffic on the performance of a proxy cache server. *Annales Univ. Sci. Budapest, Sect. Comp.*, **25** (2005), 153-169

J2 T. Berczes and J. Sztrik, Performance Modeling of Proxy Cache Servers. *Journal of Universal Computer Science.*, **12** (2006), 1139–1153

J3 T. Berczes and J. Sztrik, Performance evaluation of proxy cache servers. *Híradástechnika.*, Selected Papers **LXI** (2006/1), 2-5

J4 T. Berczes, Approximation approach to performance evaluation of Proxy Cache Server systems. *Annales Mathematicae et Informaticae.*, **36** (2009), 15-28

J5 T. Berczes, G. Guta, G. Kusper, W. Schreiner and J. Sztrik, Comparing the Performance Modeling Environment MOSEL and the Probabilistic Model Checker PRISM for Modeling and Analysing Retrial Queueing Systemszes, *Annales Mathematicae et Informaticae.*, **37** (2010), 51-75

J6 T. Berczes, A. Hazy, and J. Sztrik, The impact of servers breakdown on the performance of proxy cache servers Mathematical and Computer Modelling (Submitted)

- **[C] Konferencia kiakdványok\Conference papers:**

C1 T. Berczes and J. Sztrik, A queueing network model to study Proxy Cache Servers. *Proceedings of 7th International Conference on Applied Informatics.*, Eger, Hungary Vol. **1** (2007), 203-211.

C2 T. Berczes, G. Guta, G. Kusper, W. Schreiner and J. Sztrik, Analyzing a Proxy Cache Server Performance Model with the Probabilistic Model Checker PRISM. *WWV 2009 Automated Specification and Verification of Web Systems 5th International Workshop.*, Linz (2009).

- Angel Vassilev Nikolov, Effects of the coherency on the Performance of the Web Cache Proxy Server. *International Journal of Computer Science and Network Security.*, **9** (2009), 158-162.

- LIU Xu-jun, MA Yue and YU Dong Analysis and Evaluation of Real-time Performance of Publish/Subscribe Communication Mode. *Computer Engineering.*, Vol. **9**, No. **20** (2010), 229-231.

- **[O] Egyéb\Others**:

O1 T. Berczes, G. Guta, G. Kusper, W. Schreiner and J. Sztrik, Analyzing Web Server Performance Models with the Probabilistic Model Checker PRISM. *Technical report no. 08-17 in RISC Report Series* , University of Linz, Austria. November 2008

O2 T. Berczes, G. Guta, G. Kusper, W. Schreiner and J. Sztrik, Comparing the Performance Modeling Environment MOSEL and the Probabilistic Model Checker PRISM for Modeling and Analysing Retrial Queueing Systems. *Technical report no. 07-17 in RISC Report Series* , University of Linz, Austria. November 2007