UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Benjamin Ambrož

ZMOGLJIVOSTNA ANALIZA STREŽNIKA ZA NUDENJE STORITVE VIDEO NA ZAHTEVO

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

Mentor: prof. dr. Nikolaj Zimic

Ljubljana, 2010

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01631/2010
Datum: 15.01.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **BENJAMIN AMBROŽ**

Naslov: **ZMOGLJIVOSTNA ANALIZA STREŽNIKA ZA NUDENJE STORITVE VIDEO NA ZAHTEVO**

THE PERFORMANCE ANALYSES OF THE VIDEO ON DEMAND SERVER

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Ponudniki širokopasovnega dostopa naročnikom ponujajo široko paleto storitev med katerimi je tudi video na zahtevo. Pri strežniku video vsebin nastane težava, ko veliko odjemalcev istočasno želi gledati poljubne vsebine. Pri tem se je potrebno zavedati, da je število odjemalcev lahko zelo veliko, saj običajno en strežnik pokriva vse potrebe ponudnika storitev v širši regiji.

V diplomski nalogi analizirajte možne dostope do diskovnih pogonov pri strežnikih video vsebin. Za najbolj učinkovite algoritme dostopa izdelajte simulacijo. V simulaciji ustrezno opredelite vhodno breme ter ponudbo vsebin tako, da bodo rezultati simulacije čim bolj verodostojni. Simulacijski model mora biti ustrezno preverjen in potrjen. Rezultate simulacije ustrezno predstavite in jih komentirajte.

Mentor:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Franc Solina

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani Benjamin Ambrož,

z vpisno številko 63010022,

sem avtor diplomskega dela z naslovom:

Zmogljivostna analiza strežnika za nudenje storitve video na zahtevo

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom

  prof. dr. Nikolaja Zimica

  in somentorstvom

  /,

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 25. 1. 2010 Podpis avtorja:

Zahvala

Zahvaljujem se mentorju prof. dr. Nikolaju Zimicu za strokovne nasvete in usmeritve pri izdelavi diplomskega dela, korekten odnos ter za ažurnost pri pregledovanju.

Posebna zahvala gre moji družini, ki mi je študij omogočila in me ves čas spodbujala. Hvala tudi Aleksandri Zazijal za skrbno prebiranje in lektoriranje diplomskega dela in ker mi ves čas stoji ob strani.

Prav tako se zahvaljujem asistentu Domnu Šoberlu za uvodno seznanitev s simulacijskim orodjem Simprocess, Tini Krašovec za pomoč pri angleškem prevodu ter ostalim, ki so mi kakor koli pomagali z nasveti.

Posvetilo

Diplomsko delo posvečam svojima dragima staršema v spomin, mami Manici in očetu Francu.

Kazalo vsebine

Seznam uporabljenih kratic

ADSL	- Asymmetric Digital Subscriber Line
ATM OC-3	- Asynchronous Transfer Mode Optical Carrier 3
AVC	- Advanced Video Coding
BPR	- Business Process Reengineering
CBR	- Constant Bit Rate
CD	- Compact Disk
CLB	- Combination Load Balancing
CPE	- Centralno Procesna Enota
DLB	- Disk Load Balancing
DVD	- Digital Versatile Disk
EDF	- Earliest Deadline First
EIDE	- Enhanced Integrated Drive Electronics
EOD	- Everything On Demand
FTTH	- Fiber To The Home
FVOD	- Free Video On Demand
HDD	- Hard Disk Drive
HDMI	- High Definition Multimedia Interface
HDTV	- High Definition TeleVision
IEEE	- Institute of Electrical and Electronics Engineers
IKT	- Informacijsko-Komunikacijske Tehnologije
IP	- Internet Protocol
IPTV	- Internet Protocol TeleVision
IT	- Informacijska Tehnologija
LBF	- Least Busy Fit
LBI	- Load Balancing Index
LCD	- Liquid Crystal Display
LRU	- Least Recently Used
MPEG	- Motion Picture Experts Group
NPVR	- Network Personal Video Recorder
NTSC	- National Television System Committee
NVOD	- Near Video On Demand
OS	- Operacijski Sistem
P2P	- Peer-to-Peer
PAL	- Phase Alternating Line
PCM	- Pulse-Code Modulation
PPV	- Pay Per View
PVR	- Personal Video Recorder
QoS	- Quality of Service
RAID	- Redundant Array of Independent Disks
RAM	- Random Access Memory
RBP	- Request Blocking Probability

RMS	- Rate Monotonic Scheduling
RRT	- Repeated Random Trials
RSI	- Resource Sharing Index
SATA	- Serial AT Attachment
SCSI	- Small Computer System Interface
SDTV	- Standard Definition TeleVision
SECAM	- SEquentiel Couleur A Memoire
SRT	- Single Random Trial
SSD	- Solid-State Drive
STB	- Set-Top Box
STR	- Sequential Transfer Rate, Sustained Transfer Rate
SVOD	- Subscription Video On Demand
V/I	- Vhod/Izhod
VCR	- Video Cassette Recorder
VHS	- Video Home System
VOD	- Video On Demand
xDSL	- x Digital Subscriber Line

Povzetek

Storitve videa na zahtevo so se z razvojem informacijsko-komunikacijskih tehnologij močno razmahnile in postale del redne ponudbe v okviru IP televizije. Video na zahtevo (VOD) predstavlja vrhunec na področju multimedije in potrebuje za svoje delovanje zmogljive računalnike – video strežnike ter posebej prilagojene operacijske sisteme, ki podpirajo rokovanje z multimedijo, da lahko zadosti uporabniškim zahtevam po ogledu video vsebin. V teoretičnem delu diplomskega dela smo predstavili algoritme, ki jih multimedijski operacijski sistemi uporabljajo za razvrščanje procesov, predpomnjenje blokov in datotek, razvrščanje zahtev za disk ter za razmeščanje filmskih datotek po diskih. Razmeščanju datotek smo se še posebej posvetili, saj je bil cilj praktičnega dela diplomske naloge sestaviti simulacijski model bremensko uravnoteženega diskovnega podsistema VOD. Zaradi različne priljubljenosti filmov med uporabniki in posledično neenakomernega povpraševanja je problem razmestitve filmskih datotek na diske, da se doseže minimalna verjetnost zavrnitve zahtev v sistemu, netrivialen. Za hevristično rešitev NP-polnega problema smo uporabili požrešno metodo za alokacijo datotek, ki je realizacija učinkovite sheme razmeščanja z repliciranjem datotek - kombinacijske izravnave bremen (CLB). Simulacijo diskretnih dogodkov CLB z analizo zmogljivosti smo izvedli s pomočjo simulacijskega orodja Simprocess. Z izvajanjem poizkusov smo pridobili podrobnejši vpogled v delovanje diskovnega podsistema VOD pri različnih konfiguracijah ter preverili učinkovitost dveh izčrpnih algoritmov izbire resursov, ki jih shema CLB uporablja za usmerjanje zahtev na diske, in sicer RRT ter LBF, izmed katerih se je slednji izkazal za učinkovitejšega. Rezultati izvajanja poizkusov so potrdili tudi učinkovitost CLB pri enakomernem porazdeljevanju bremena in nam dali dobro oceno izkoriščenosti diskovnih resursov pri maksimalni obremenitvi, pri kateri je še izpolnjen pogoj sprejemljive kakovosti storitve. Simulacijski model se lahko uporabi za načrtovanje obsežnega, bremensko uravnoteženega diskovnega podsistema VOD ter za njegovo uglaševanje. V sklepu smo podali smernice, kako bi do rešitev prišli hitreje z analitično metodo.

Ključne besede:

Video na zahtevo, kombinacijska izravnava bremen, simulacija, razmeščanje datotek, multimedijski operacijski sistemi, Simprocess.

Abstract

With the development of information and communication technologies, video-on-demand services have become widely used, what is more, they have also become a part of regular offer of IPTV. Video-on-demand represents the peak of multimedia and it requires capable computers for its functioning – video servers and especially adjusted operating systems which support handling multimedia in the way that it can meet the user's requirements for watching video content. The theoretical part of the thesis introduces algorithms which are used by multimedia operating systems for process scheduling, block and file caching, disk scheduling and placing movie files on disks. The emphasis is on the file placement since the aim of the empirical part of the thesis has been to construct a simulation model of the load balanced disk subsystem of the VOD server. Due to the varying popularity of movies between the users and consequently the significant asymmetry in access demand, can the file assignment problem, in order to achieve minimum request blocking probability, be considered as nontrivial. For a heuristic solution of the NP-complete problem we have used a greedy file allocation method that realizes an efficient localized file placement scheme with non-uniform replication strategy – the combination load balancing scheme (CLB). The discrete event simulation study of CLB with the performance analysis has been done with the help of simulation tool Simprocess. By conducting experiments we have gained a better insight into the functioning of the disk subsystem of the VOD server at different configurations. Furthermore, we have checked the efficiency of two exhaustive resource selection schemes which are used by the CLB scheme, i.e. repeated random trials (RRT) and least busy fit (LBF). The latter has proved itself as more effective. The results of the experiments have also confirmed that a greedy file allocation method decides a good quality heuristic solution for each feasible file replication instance. The simulation model can be applied to the design of a large-scale and load balanced disk subsystem of the VOD server and for its tuning. In the conclusion we have devised how to get approximate results faster with an analytical method.

Keywords:

Video-on-demand, combination load balancing, simulation, file placement, multimedia operating systems, Simprocess.

1 Uvod

Razvoj informacijsko-komunikacijskih tehnologij (IKT) je v zadnjih letih prinesel marsikaj novega. Širokopasovni dostop do interneta je postal nekaj običajnega, cenovno ugodnega in lahko bi rekli samoumevnega. Poleg xDSL (angl. x Digital Subscriber Line) tehnologij dostopa do interneta in dostopa preko kabelskega omrežja je v zadnjih dveh letih v Sloveniji čutiti močan porast optičnih priključkov, ki ponujajo še višje hitrosti prenosov podatkov. Z napredkom tehnologije so se razmere spremenile tudi na področju televizijskih zaslonov. Tako so se veliki visokoločljivostni LCD in plazma televizorji pocenili in postali dostopnejši za širši krog ljudi, prav tako razni sistemi za domači kino, analogni način oddajanja televizijskega signala pa bo v letu 2010 v celoti zamenjal digitalni način. Pojavile so se tudi nove možnosti ogledovanja video vsebin, kot sta internetni video (npr. YouTube) in mobilna TV. Digitalni zapis filmov je popolnoma izpodrinil analognega. Videokasete (VHS) so postale stvar preteklosti, saj so jih zamenjali optični nosilci, ki lahko hranijo video vsebine visoke ločljivosti. Videoteke so povečini propadle, kljub temu da so izposojevalci videokasete zamenjali z DVD-ji. Zakaj je prišlo do tega, ni težko razumeti. Širokopasovni dostop do interneta, zmogljivejši osebni računalniki, večji diski, DVD snemalniki, razmah P2P (angl. peer-to-peer) omrežij in pojavitev BitTorrent protokola za izmenjavo datotek so omogočili, da so postali digitalni filmi na internetu prosto dostopni širši množici. Poleg t.i. spletnega piratstva pa si je mogoče filme na DVD-jih zastonj ogledati tudi tako, da si jih predhodno izposodimo v knjižnicah, kjer je izbira precej pestra. Omeniti velja še ponudbo raznih časopisnih hiš, ki prilagajo DVD filme svojim revijam po zelo ugodnih cenah. Takšen nakup filma je zelo priljubljen, saj je precej ugodnejši od nakupa v regularni prodaji. Kljub propadu videotek, ki so bile dolgo zelo priljubljene, saj so bile edina možnost, da si je uporabnik izbral film po svoji želji in ni bil vezan na televizijski spored, pa lahko ugotovimo, da se je njihova ideja »reinkarnirala« v storitvi videa na zahtevo (angl. video on demand – VOD), kateri v diplomskem delu posvečamo glavno pozornost.

Naj v uvodu nakažemo, da z izrazom video na zahtevo mislimo predvsem na storitev, ki jo operaterji ponujajo v paketu z IP televizijo (IPTV). Princip je podoben kot pri videotekah – uporabnik si izbere film po lastni želji, plača pristojbino in si film predvaja na televizijskem zaslonu – zato ni nič nenavadnega, da so operaterji storitev videa na zahtevo v svoji ponudbi poimenovali kar videoteka (primer na sliki 1.1), saj je uporabnikom izraz domač in intuitivno vedo, kaj storitev nudi. Razlika pa je očitna. Za izbiro filma jim sedaj ni potrebno zapustiti domačega naslanjača, saj si lahko film, ki se nahaja na oddaljenem strežniku, izberejo interaktivno s pomočjo daljinskega upravljalca. Iz vsega naštetega vidimo, da je razvoj različnih tehnologij pripomogel k izboljšanju uporabniške izkušnje in omogočil, da so postale storitve, kot je video na zahtevo, komercialno zanimive, saj le-te terjajo za svoje izvajanje zmogljive in posledično drage sisteme.

V diplomskem delu smo se namenili raziskati tehnološko ozadje videa na zahtevo s poudarkom na video strežnikih in multimedijskih operacijskih sistemih, ki tečejo na njih. Konkretneje so nas zanimali algoritmi, ki jih uporabljajo multimedijski operacijski sistemi za

razvrščanje procesov, predpomnjenje blokov in datotek, razvrščanje zahtev za disk ter za razmeščanje filmskih datotek po diskih. Slednjemu problemu smo se še posebej posvetili, saj je bil cilj praktičnega dela diplomske naloge realizirati simulacijski program kombinacijske izravnave bremen, ki predstavlja enega izmed učinkovitejših načinov razmeščanja datotek.

Slika 1.1 Uporabniški vmesnik videa na zahtevo kot storitve v okviru IPTV. Vir: [22].

Obravnava v teoretičnem delu diplomske naloge obsega najprej poglavje, v katerem so predstavljene značilnosti multimedije in videa na zahtevo kot posebnega področja multimedije. 2. poglavje med drugim zajema opis različnih VOD storitev z vidika poslovnih modelov, podaja infrastukturo, ki jo video na zahtevo potrebuje za svoje delovanje, ter predstavi način prenosa video vsebin v VOD sistemih. Poglavje se opira predvsem na vire [2, 3, 15, 20, 23], pri iskanju slovenskih ustreznic za angleške izraze pa smo si pomagali s [16].

Multimedija potrebuje realnočasovno razvrščanje procesov, da zadosti svojim skrajnim rokom. V 3. poglavju smo predstavili dva algoritma za prekinjevalno razvrščanje procesov v realnem času – statični RMS, ki je manj kompleksen, in dinamični EDF, ki na račun večje kompleksnosti ponuja večji izkoristek.

Multimedijski datotečni sistemi običajno uporabljajo drugačen pretočni model kot tradicionalni datotečni sistemi. V 4. poglavju smo podali razliko med potisnimi strežniki, ki jih uporablja VOD, in tradicionalnimi poteznimi strežniki. Uporaba potisnih strežnikov pri VOD sistemih je potrebna, da se zadosti zahtevam v realnem času.

V 5. poglavju so predstavljene tehnike predpomnjenja blokov in datotek pri VOD sistemih, saj običajno LRU predpomnjenje pri multimedijskih sistemih zaradi drugačnega vzorca dostopa ne deluje dobro. Poglavje opisuje, na kakšen način se da predpomnjenje pri multimediji koristno uporabiti.

Razvrščanje zahtev za disk smo obravnavali v 6. poglavju. Predvidljivost pri pretakanju multimedije omogoča, da lahko sistem uporabniške zahteve optimalno posortira in jih zatem v optimalnem vrstnem redu obdela. Predstavili smo statično in dinamično diskovno razvrščanje. Poglavja 3-6 se naslanjajo predvsem na vira [3, 4].

Največ pozornosti smo v teoretičnem delu diplomske naloge posvetili problemu razmeščanja multimedijskih datotek na diske, ki smo ga podrobno obravnavali v 7. poglavju. Namen raznih strategij razmeščanja je doseči enakomerno zasedenost resursov, kar zaradi različne priljubljenosti posameznih filmskih datotek med uporabniki ni trivialna naloga. Breme, ki ga uporabniki povzročajo s svojimi zahtevami po predvajanju filmov je potrebno enakomerno porazdeliti med diske. Filmske datoteke lahko na več diskov razmeščamo bodisi z razdeljevanjem datotek (datoteke razdelimo na več delov in te razmestimo po diskih) bodisi z repliciranjem datotek (najpriljubljenejšim filmskim datotekam določimo več kopij za razmeščanje). Pri video strežnikih, ki zagotavljajo visoko stopnjo interaktivnosti, se zaradi zmogljivostnih problemov, ki se pojavijo pri razdeljevanju, uporablja predvsem pristop razmeščanja z repliciranjem datotek. V okviru slednjega smo v podpoglavju 7.5 temeljiteje obravnavali kombinacijsko izravnavo bremen (CLB), ki je ena izmed učinkovitejših shem dodeljevanja datotek. Za učinkovito realizacijo CLB smo podali požrešno metodo za hevristično alokacijo datotek. Podpoglavja 7.1-7.4 se opirajo na vira [3, 4], podpoglavje 7.5 pa sledi ugotovitvam iz [9, 11, 12].

Praktični del diplomske naloge je zajet v 8. poglavju. Cilj je bil narediti uporaben simulacijski program kombinacijske izravnave bremen za diskovni podsistem VOD, ki bi služil načrtovanju VOD sistema s široko ponudbo filmskih vsebin. Simulacijo smo uspešno realizirali s simulacijskim orodjem Simprocess. Za zagotovitev sistematičnosti pri načrtovanju modela sistema in analizi zmogljivosti sta bila upoštevana predvsem vira [1, 5], pri potrjevanju modela pa [7, 8]. Izvajanje poizkusov nam je dalo podrobnejši vpogled v delovanje različnih variant diskovnega podsistema VOD, dobili pa smo tudi dobro oceno izkoriščenosti diskovnih resursov pri maksimalni obremenitvi, pri kateri je še izpolnjen pogoj sprejemljive kakovosti storitve. Rezultati so podrobno predstavljeni v podpoglavju 8.6.

Sklepno 9. poglavje diplomskega dela povzema glavne ugotovitve obravnavanih sklopov in podaja oceno o opravljenem delu. Predlaga tudi, kako bi do suboptimalnih rešitev prišli hitreje s pomočjo analitične metode.

2 Predstavitev multimedije in videa na zahtevo

Pred podrobnejšo obravnavo sistema, ki zagotavlja storitev video na zahtevo in algoritmov, ki jih na video strežnikih uporabljajo multimedijski operacijski sistemi za realizacijo videa na zahtevo, bomo najprej predstavili multimedijo, njeno uporabo in značilnosti ter posebno podpoglavje namenili osnovni predstavitvi storitve videa na zahtevo.

2.1 Uporaba in značilnosti multimedije

Beseda multimedija dobesedno pomeni več kot en medij. Pod oznako multimedija največkrat pomislimo na filme, čeprav bi lahko po tej definiciji kot multimedijo tretirali tudi knjige, saj so tudi iz dveh medijev – teksta in slik. V diplomskem delu bo termin multimedija uporabljen kot dokument, sestavljen iz dveh ali več sovisnih medijev, ki naj bi se ponovno izvajali na nekem časovnem intervalu.

Termin, ki je morda prav tako dvoumen, je video. Televizijski sprejemniki imajo pogosto ločena signala za video in avdio, medtem ko se izraz digitalni video ponavadi nanaša na celoten izdelek, z zvokom in s sliko. V diplomskem delu je namesto izraza video (kot celoten izdelek) uporabljen izraz film, za kratke filme in video izrezke pa izraz filmček.

Pred tehnološkimi značilnostmi multimedije velja spregovoriti še nekaj besed o njeni trenutni in prihodnji uporabi. Na enouporabniškem osebnem računalniku izraz multimedija pogosto uporabljamo v zvezi s predvajanjem vnaprej posnetega filma z DVD-ja (angl. Digital Versatile Disk). Multimedija se uporablja tudi v zvezi s presnemavanjem filmčkov preko interneta. Mnogo spletnih strani vsebuje povezave, iz katerih lahko presnamemo filmčke. Ob vse hitrejših tehnologijah za prenos podatkov, kot so npr. ADSL (angl. Asymmetric Digital Subscriber Loop), kabelska TV ter optika, je prisotnost filmčkov vse večja.

Naslednje področje, kjer mora biti multimedija podprta, je samo izdelovanje video posnetkov. Multimedijski urejevalniki morajo za svoje uspešno delovanje teči na operacijskem sistemu, ki poleg običajnega dela podpira tudi multimedijo.

Multimedija je pomembna tudi na področju računalniških iger. Igre pogosto vsebujejo kratke filmčke za uprizoritev kakšne akcije. Filmčkov je navadno veliko, zaporedje njihovega izvajanja pa je izbrano dinamično na podlagi neke akcije, ki jo je uporabnik pred tem izvedel.

Nazadnje pa je potrebno omeniti še video na zahtevo (angl. video on demand - VOD), kot vrhunec na multimedijskem področju. Pod tem izrazom mislimo na zmožnost, da si lahko

uporabniki doma z uporabo daljinskega upravljalca (ali miške) izberejo film po svoji želji in ga brez odlašanja predvajajo na svojem TV sprejemniku, oz. računalniškem monitorju.

Multimedija ima dve ključni značilnosti, ki jih je potrebno dobro razumeti:

1. Multimedija uporablja ekstremno visoke prenosne hitrosti podatkov.
2. Multimedija zahteva predvajanje v realnem času.

Visoke prenosne hitrosti podatkov sledijo iz same narave vizualne in akustične informacije. Oko in uho lahko obdelata ogromno količino informacije v sekundi in morata biti s takšno hitrostjo tudi oskrbovana s podatki, da je ustvarjen učinek sprejemljive izkušnje gledanja in poslušanja. Prenosne hitrosti podatkov nekaterih digitalnih multimedijskih virov in nekaterih običajnih strojnih naprav so naštete v tabeli 2.1. Poleg visokih prenosnih hitrosti podatkov je potrebno omeniti tudi potrebo po stiskanju podatkov, s čimer pa je povezana tudi količina potrebnega prostora za shranjevanje. Za primer lahko damo nestisnjen dveuren HDTV film, ki zasede 570 GB prostora. Video strežnik, ki hrani 1000 takšnih filmov potrebuje 570 TB diskovnega prostora, kar je netrivialna količina za današnje standarde. Brez kompresije podatkov tudi trenutna strojna oprema ne dohiteva visokih prenosnih hitrosti podatkov. IPTV sistemi, v okviru katerih operaterji ponujajo storitev videa na zahtevo, ponavadi uporabljajo en format kodiranja videa (nekateri tudi dva). Ponudniki IPTV izbirajo med formati MPEG-2, MPEG-4 ali VC-1, ponavadi pa izberejo le enega izmed njih za kodiranje vseh video vsebin, saj to močno poenostavi celoten IPTV sistem. Če video vsebina, ki pride do IPTV ponudnika, ni v pravem formatu, jo je treba preoblikovati. Ponudniki vso vsebino preoblikujejo v skupno bitno hitrost (angl. bit rate), običajno v eno za SDTV (angl. Standard-definition television) in drugo za HDTV (angl. High-definition television), kar poenostavi proces menjave video tokov in upravljanje s pasovno širino (angl. bandwidth management), saj lahko nek tok fiksne pasovne širine zamenja drug tok iste pasovne širine.

Tabela 2.1 Nekatere prenosne hitrosti podatkov za multimedijo in visoko zmogljive V/I naprave. Opozoriti velja, da je 1 Mb/s enako 10^6 bit/s, 1 GB pa 2^{30} B. Viri: [3, 14, 19].

Vir	Mb/s	GB/h	Naprava	Mb/s
Telefon (PCM)	0˙064	0˙03	Fast ethernet	100
MP3 glasba	0˙14	0˙06	EIDE disk	133
Avdio CD	1˙4	0˙62	ATM OC-3 omrežje	156
MPEG-2 film (640 x 480)	4	1˙76	IEEE 1394b (FireWire)	800
Digitalna kamera (720 x 480)	25	11	Gigabit ethernet	1000
Nestisnjena SDTV (640 x 480)	221	97	SATA disk	3000
Nestisnjena HDTV (1280 x 720)	648	288	Ultra-640 SCSI disk	5120
Nestisnjena HDTV (1920 x 1080) (format: 10-bit 4:2:2, 59.94 sličic/s)	2486	1092	HDMI 1.4	10200

Druga zahteva, ki jo multimedija narekuje sistemu, je potreba po realnočasovni dostavi podatkov. Video pošiljko digitalnega filma sestavlja neko število sličic na sekundo. NTSC

sistem (angl. National Television Standards Committee), ki ga uporabljajo v Severni in Južni Ameriki ter na Japonskem, teče pri hitrosti 29˙97 sličic/sekundo, medtem ko sistema PAL (angl. Phase Alternating Line) in SECAM (fr. SEquentiel Couleur Avec Memoire), ki sta uporabljena skoraj povsod drugod po svetu, tečeta pri hitrosti 25 sličic/sekundo. Sličice morajo biti dostavljene v točnih intervalih na 33˙3 ms, oz. na 40 ms, ali pa bo film dajal vtis zatikanja.

Uho je občutljivejše od očesa, tako da se razhajanje že za nekaj milisekund v dostavnih časih pozna. Variabilnost v hitrosti dostave imenujemo trepetanje (angl. jitter), ki mora biti za dober učinek strogo omejeno. Trepetanja ne smemo enačiti z zakasnitvijo (angl. delay). Če distribucijsko omrežje enakomerno zakasni vse bite za natančno 5000 sekund, se bo film začel prikazovati nekoliko kasneje, vendar bo izgledal dobro. Na drugi strani pa, če naključno zakasni sličice za od 100 do 200 milisekund, bo film izgledal zatikajoč.

Realnočasovne lastnosti, ki so potrebne, da je multimedija zadovoljivo predvajana, so opisane s parametri kvalitete storitve (angl. quality of service - QoS). Ti parametri vključujejo razpoložljivo povprečno pasovno širino, največjo razpoložljivo pasovno širino, minimalno in maksimalno zakasnitev (ki skupaj omejujeta trepetanje) ter verjetnost izgube bitov. Na primer, mrežni operater bi lahko ponujal storitev z zagotovljeno povprečno pasovno širino 4 Mb/s, 99% prenosnih zakasnitev v intervalu od 105 do 110 ms in stopnjo izgube bitov 10^{-10}, kar bi bilo v redu za MPEG-2 filme.

Najpogostejši način za zagotavljanje kvalitete storitve je rezervacija zmogljivosti vnaprej za vsako novo stranko. Rezervirana sredstva vključujejo del procesorja, medpomnilnike pomnilnika, diskovno prenosno kapaciteto in pasovno širino omrežja. Če se pojavi nova stranka in si želi ogledati film, video strežnik (ali omrežje) pa izračuna, da nima dovolj kapacitet za novo stranko, mora ta stranko zavrniti, da se izogne slabšanju storitve za tekoče stranke. Posledično multimedijski strežniki potrebujejo sheme rezervacije virov (angl. resource reservation schemes) in algoritem kontrole dostopa (angl. admission control algorithm) za odločitev, ali lahko sprejmejo več dela.

2.2 Video na zahtevo in njegove značilnosti

Sistemi videa na zahtevo omogočajo uporabnikom poljubno izbiro vnaprej shranjenih video vsebin (predvsem filmov in filmčkov) in predvajanje le-teh na zahtevo v realnem času. Za storitev videa na zahtevo se lahko uporabijo privatna omrežja (storitev v okviru IPTV), internet (kot internetni video, recimo YouTube) in mnoga druga omrežja. Vsebina se lahko predvaja na različnih elektronskih napravah, kot so televizijski komunikatorji (angl. set-top boxes – STB), osebni računalniki, mobilni telefoni, medijski centri (angl. media centers), določene prenosne multimedijske naprave (npr. igralne konzole) itd.

VOD je oblika predvajanja video vsebin s časovnim zamikom (angl. timeshifting). Za VOD, ki je namenjen predvajanju vsebin na TV, je značilno, da uporabniku nudi veliko funkcionalnosti videorekorderja (angl. VCR functionality), vključno s hitrim/počasnim previjanjem naprej/nazaj, ustavljanjem, skokom naprej/nazaj itd.

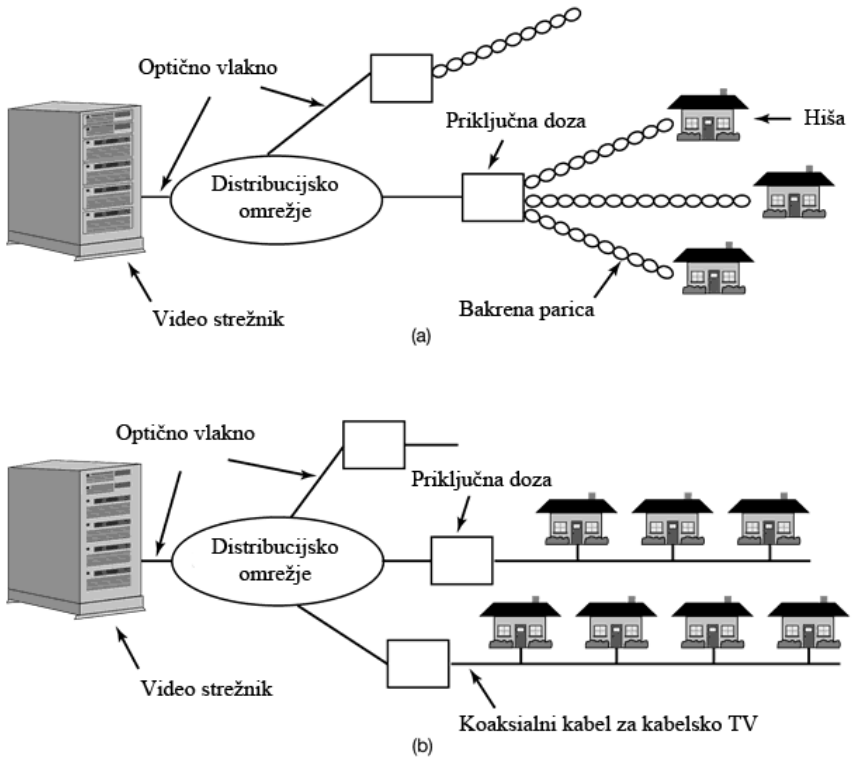
Glede na poslovne modele in tehnološke značilnosti obstaja več delitev videa na zahtevo. Ena izmed njih je predstavljena v tabeli 2.2. V diplomskem delu je pozornost namenjena predvsem prvemu tipu, torej pravemu videu na zahtevo, ki jo operaterji ponujajo v okviru IPTV. Z izrazom video na zahtevo oz. VOD je torej mišljen prvi tip v navedeni klasifikaciji.

Tabela 2.2 Tipi storitev videa na zahtevo. Vir: [2].

Tip	**Opis**
Pravi video na zahtevo (angl. True Video on Demand – **VOD**)	To je najčistejša oblika VOD, kjer vsak gledalec prejme svoj osebni video tok, nad katerim ima popoln nadzor. Gledalci lahko predvajajo, ustavljajo, previjajo vsebino … Za vsak ogled filma je ponavadi potrebno plačati pristojbino. Stroški so lahko bremenjeni s predplačniškega računa, ali pa so zajeti na mesečnem računu.
Skorajšnji video na zahtevo (angl. Near Video on Demand – **NVOD**)	Podobno kot pravi VOD, le da brez zmožnosti nadzora nad osebnim video tokom. Ena od običajnih oblik NVOD se včasih imenuje nesovpadajoče oddajanje (angl. staggercasting), pri kateri se več kopij programa/filma začne predvajati v pet minutnih razmakih. S tem je zagotovljeno, da ni nobenemu gledalcu treba čakati več kot pet minut, preden se njegov program/film začne predvajati.
Naročniški video na zahtevo (angl. Subscription Video on Demand – **SVOD**)	Enaka tehnologija dostave in gledalčev nadzor kot pri VOD, a z drugačnim plačilnim sistemom. V SVOD naročniki plačajo fiksno mesečno pristojbino za neomejen dostop do knjižnice filmskih naslovov. V mnogih sistemih je knjižnica mesečno posodobljena.
Brezplačni video na zahtevo (angl. Free Video on Demand - **FVOD**)	Varianta VOD, kjer je plačevanje odpravljeno. V večini sistemov je ta vsebina omejena na dolgo obliko oglaševanja, vodnike »kako narediti« in drugo poceni vsebino.
Vse na zahtevo (angl. Everything on Demand – **EOD**)	Za nekatere tehnološke vizionarje je to skrajna oblika video dostavnega sistema (angl. video delivery system), kjer so vsi programi/filmi dostopni vsem gledalcem ves čas.
Osebni videorekorderji (angl. Personal Video Recorders – **PVRs**)	Te naprave sprejmejo prihajajočo video vsebino, jo stisnejo in posnamejo na trdi disk, ki je tipično lociran bodisi v STB bodisi v samostojni napravi. Gledalci nato upravljajo PVR za predvajanje vsebine, vključno z zmožnostmi ustavljanja, previjanja naprej/nazaj idr. Gledalci ponavadi programirajo PVR-je, da snemajo določene programe/filme ob določenih časih.
Omrežni osebni videorekorderji (angl. Network Personal Video Recorders – **NPVRs**)	Ponujajo podobno funkcionalnost kot PVR-ji, le da je snemanje izvedeno znotraj omrežja ponudnika storitve in ne pri gledalcu. Nekateri lastniki vsebin trdijo, da je ta tehnologija v zmožnostih tako podobna pravemu VOD, da bi morala biti kot taka tudi licencirana.
Plačilo na ogled (angl. Pay Per View – **PPV**)	Ta predhodna tehnologija VOD-a je prvotno uporabljena za dostavo plačljive žive vsebine, kot so koncerti ali športni dogodki.

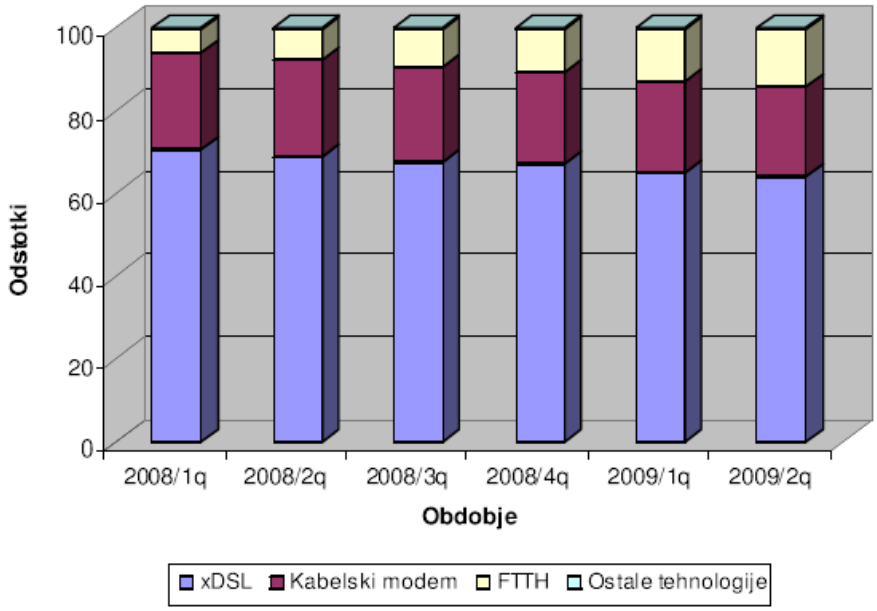
VOD storitve so postale zelo priljubljen način zabave, saj so zaradi večje dostopnosti širokopasovnih priključkov, pocenitve le-teh in porasta ponudnikov VOD storitev dostopne širšemu krogu uporabnikov. Druga raziskava VOD storitev v Evropi, ki jo je naredil Evropski avdiovizualni observatorij (angl. European Audiovisual Observatory), kaže na močan porast plačljivih VOD storitev v zadnjem času. Na začetku leta 2007 so jih zabeležili 142 v 24 evropskih državah. Do konca leta 2007 se je število teh storitev skoraj podvojilo, saj se je njihovo število povzpelo kar na 258 (več v [17]). V Sloveniji smo prvi video na zahtevo kot storitev v okviru IPTV dobili v začetku leta 2007 (v paketu trojček ga je prva ponudila družba T-2 d.o.o., malo zatem pa še SIOL d.o.o.). V letu 2009 so v Sloveniji ponujali VOD storitve v okviru IPTV štirje operaterji – poleg družb T-2 d.o.o. in Telekom Slovenije d.d. (SIOL), še družbi Tušmobil d.o.o. in Amis d.o.o.

Video na zahtevo potrebuje za svoje delovanje posebno infrastrukturo, sestavljeno iz najmanj treh komponent: enega ali več video strežnikov, distribucijskega omrežja in televizijskega komunikatorja (STB-ja), ki ga ima vsak uporabnik doma za odkodiranje signala in dekompresijo. Video strežnik je zelo zmogljiv računalnik, ki lahko shrani veliko filmov v svoj datotečni sistem in jih nato predvaja na zahtevo. Včasih se kot video strežniki uporabljajo veliki računalniki (angl. mainframes), saj je priključitev recimo 1000 velikih diskov na veliki računalnik enostavna naloga, medtem ko bi bil to za osebni računalnik velik problem. V diplomskem delu smo se osredotočili prav na video strežnike in njihove operacijske sisteme.



Slika 2.1 Video na zahtevo uporablja različne tehnologije za lokalno distribucijo video vsebin. (a) xDSL (tudi FTTH). (b) Kabelska TV. Vir: [3].

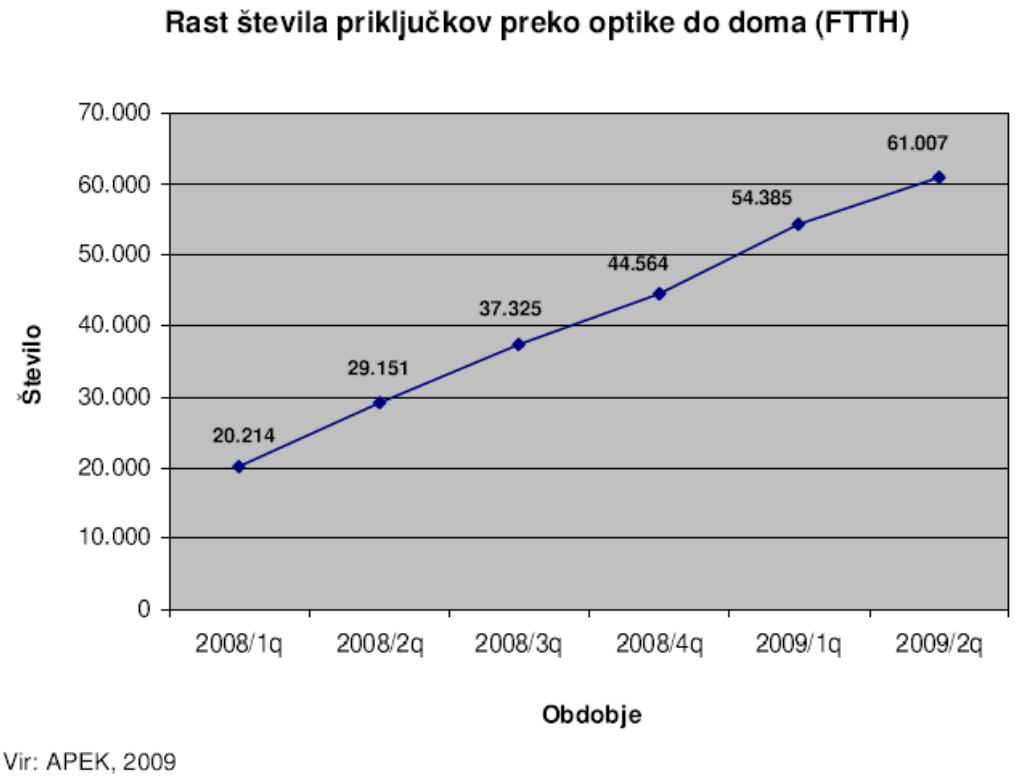
Distribucijsko omrežje med uporabnikom in video strežnikom mora biti sposobno prenašati podatke pri zelo veliki hitrosti in v realnem času. Za ta omrežja je značilno, da za omrežne povezave med video strežniki in priključnimi dozami (angl. junction-boxes) vedno uporabljajo optična vlakna (angl. optical fibers). Za lokalno distribucijo video vsebin video na zahtevo uporablja različne lokalne distribucijske tehnologije, kot sta npr. xDSL in kabelska TV (slika 2.1), ki sta v Sloveniji v letu 2009 prevladujoči (kar je razvidno z grafa 2.1). V zadnjih dveh letih se je pri nas močno pospešila gradnja optičnega omrežja, tako da nekateri operaterji za lokalno distribucijo uporabljajo tudi t.i. optiko do doma (angl. Fiber To The Home – FTTH). Porast lahko vidimo na grafu 2.2.



	2008/1q	2008/2q	2008/3q	2008/4q	2009/1q	2009/2q
xDSL	70,8	68,7	67,4	67,0	65,1	64,0
Kabelski modem	23,5	23,4	23,0	22,3	22,1	22,0
FTTH	5,5	7,6	9,3	10,4	12,3	13,6
Ostale tehnologije	0,3	0,3	0,3	0,3	0,4	0,5

Vir: APEK, 2009

Graf 2.1 V Sloveniji prevladujejo xDSL omrežja, sledijo kabelska in optična omrežja. Iz grafa je razvidno tudi gibanje njihovih deležev za leto 2008 in prvi dve četrtletji 2009. Vir: [15].



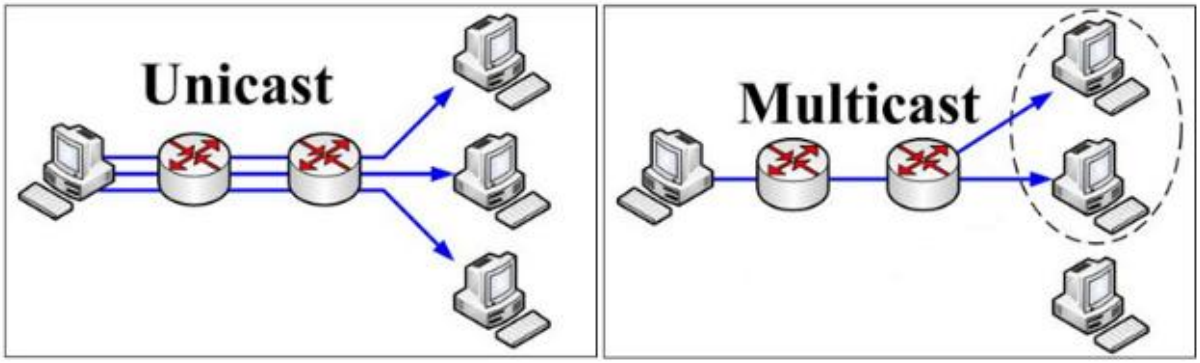
Graf 2.2 Število priključkov preko optike do doma se konstantno povečuje. Vir: [15].

Na koncu naj omenimo še način oddajanja video vsebin uporabnikom pri videu na zahtevo preko IP omrežja. VOD uporablja način oddajanja enemu prejemniku (angl. unicast). Pri unicast načinu oddajanja je vsak video tok poslan točno enemu prejemniku. Če več prejemnikov želi isti video, mora vir ustvariti ločene unicast tokove za vsakega prejemnika. Ti tokovi potem tečejo od vira do vsake destinacije preko IP omrežja.

Vsak uporabnik, ki želi gledati video vsebine, mora narediti zahtevo za vir, kjer se video nahaja. Vir mora poznati ciljni IP naslov vsakega uporabnika in ustvariti tok paketkov, naslovljenih na vsakega uporabnika. Ko se število gledalcev, ki si želi v istem času ogledati nek film, povečuje, se povečuje tudi obremenitev na vir, saj mora ta nepretrgoma ustvarjati individualne paketke za vsakega gledalca. To lahko zahteva precejšnjo količino procesorske moči in dovolj veliko omrežno povezavo za prenašanje vseh odposlanih paketkov. Na primer, če bi bil vir zmožen poslati 20 različnim uporabnikom video tok z bitno hitrostjo 2˙5 Mb/s, potem bi potreboval omrežno povezavo z vsaj 50 Mb/s pasovne širine.

Pomembna prednost oddajanja enemu prejemniku je, da lahko vsak gledalec dobi video tok po svoji meri. To omogoča video viru, da lahko nudi posebne funkcije, kot so premor, previjanje nazaj, hitro previjanje naprej ipd., kar je običajno uporabno le z vnaprej posnetimi vsebinami, a med uporabniki precej priljubljeno. Oddajanje več prejemnikom (angl. multicast) VCR funkcij ne omogoča in se tipično uporablja za prenos živih vsebin (IPTV, video konference) preko IP omrežij.

V primerjavi z unicast načinom prenosa vsebin je multicast tehnologija, ki omogoča simultan prenos enega podatkovnega toka skupini odjemalcev. Z uporabo posebnih protokolov je omrežje usmerjeno v izdelavo kopij video tokov za vsakega prejemnika. Proces kopiranja se pojavlja znotraj omrežja in ne pri viru videa. Kopije so narejene na vsaki točki v omrežju, kjer so potrebne. Slika 2.2 prikazuje razliko v načinu pretakanja podatkov pri unicast in multicast načinu oddajanja.



Slika 2.2 Primerjava unicast in multicast načina oddajanja. Vir: [20].

3 Razvrščanje procesov pri multimediji

Operacijski sistemi, ki podpirajo multimedijo, se od običajnih OS razlikujejo v treh glavnih stvareh: procesnem razvrščanju, datotečnem sistemu in razvrščanju zahtev za disk. V nadaljevanju je najprej predstavljeno procesno razvrščanje. Za razne ponazoritve delovanja sistema je v primerih uporabljen MPEG-2 način kodiranja video vsebin.

3.1 Razvrščanje homogenih procesov

Najpreprostejša vrsta video strežnikov je takšna, ki podpira prikaz fiksnega števila filmov, kjer vsi uporabljajo enako hitrost sličic (angl. frame rate), video ločljivost, hitrost prenosa podatkov (angl. data rate) in ostale parametre. Pod temi pogoji je ustrezen že preprost, a učinkovit razvrščevalni algoritem. Za vsak film imamo svoj proces (ali nit), katerega naloga je branje filma po eno sličico naenkrat in prenos te sličice do uporabnika. Ker so vsi procesi enako pomembni in opravljajo enako količino dela na sličico ter blokirajo, ko končajo procesiranje trenutne sličice, je v tem primeru krožno dodeljevanje (angl. round-robin scheduling) povsem primerno. Edini potreben dodatek za standardne razvrščevalne algoritme je nek časovni mehanizem, ki zagotavlja, da vsak proces teče pri ustrezni frekvenci.

En način za dosego ustreznega učasenja (angl. timing) je, da imamo vodilni taktovnik (angl. master clock), ki tiktaka recimo 30-krat na sekundo (za NTSC). Ob vsaki periodi se procesi zaporedno zaženejo v enakem vrstnem redu. Ko nek proces konča svoje delo, izda odložitveni sistemski klic (angl. suspend system call), ki sprosti CPE do naslednje periode taktovnika. Ob periodi se procesi znova zaženejo v enakem vrstnem redu. V kolikor je število procesov dovolj majhno, da je celotno delo še lahko opravljeno v času ene sličice, je krožno dodeljevanje ustrezno.

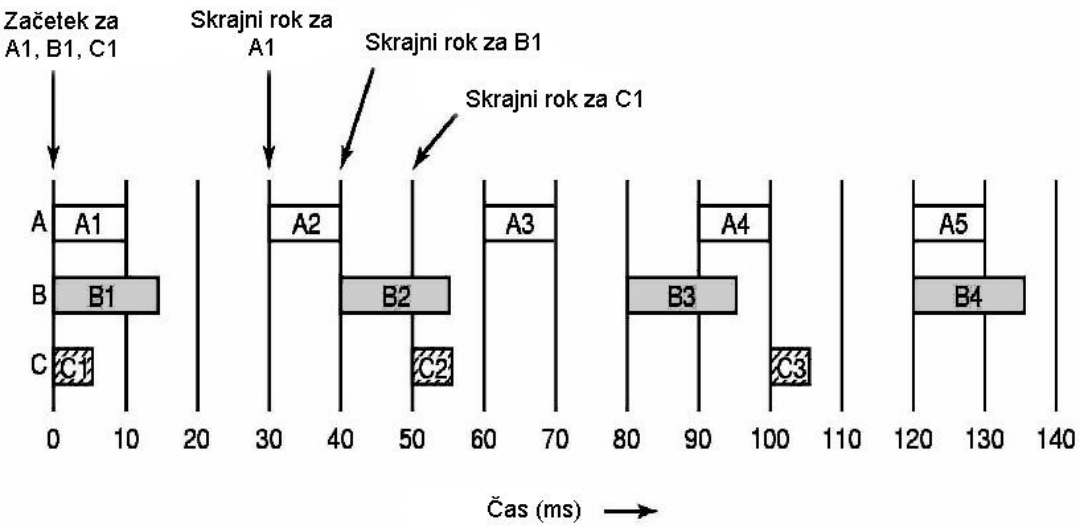
3.2 Splošno razvrščanje v realnem času

Zgornji model je v realnosti žal redko uporaben. Število uporabnikov se menja, ko gledalci prihajajo in odhajajo, velikost sličic se zaradi video kompresije (I-sličice (angl. Intracoded frames) so mnogo večje kot P-sličice (angl. Predictive frames) ali B-sličice (angl. Bidirectional frames)) zelo spreminja in različni filmi imajo lahko različno ločljivost. Posledično je verjetno, da bodo različni procesi morali teči pri različnih frekvencah, z različno količino dela in z različnimi skrajnimi roki, do katerih mora biti delo opravljeno.

Glede na te ozire je potreben drugačen model: več procesov tekmuje za CPE, vsak s svojo količino dela in skrajnim rokom. V nadaljnjih modelih smo predpostavili, da sistem pozna frekvenco, pri kateri mora teči posamezen proces, količino dela, ki jo mora opraviti, in kakšen je njegov naslednji skrajni rok. Tudi razvrščanje zahtev za disk je predmet razprave, a o tem

malo kasneje. Razvrščanje več tekmujočih procesov, izmed katerih imajo nekateri ali vsi skrajne roke, do katerih morajo biti končani, imenujemo razvrščanje v realnem času.

Primer okolja, v katerem deluje realnočasovni multimedijski razvrščevalnik, lahko ponazorimo s procesi *A*, *B* in *C* (slika 3.1). Proces *A* se zažene na vsakih 30 ms (približno NTSC hitrost). Vsaka sličica zahteva 10 ms CPE časa. Brez tekmovanja z ostalimi procesi naj bi se zaganjal v izbruhih (angl. bursts) *A1*, *A2*, *A3* itd., vsak izmed njih 30 ms za prejšnjim. Vsak CPE izbruh upravlja z eno sličico in ima svoj skrajni rok – končati se mora pred zagonon naslednjega. Proces *B* se zažene 25 krat/s (npr. PAL) in proces *C* 20 krat/s (npr. upočasnjeni NTSC ali PAL tok namenjen za uporabnika z ozkopasovno povezavo do video strežnika). Da bo problem razvrstljivosti bolj splošen, naj sličica procesa *B* zahteva 15 ms CPE časa, sličica procesa *B* pa 5 ms.



Slika 3.1 Trije periodični procesi, vsak predvaja film. Hitrost sličic in zahteve pri procesiranju na sličico se razlikujejo za vsak film. Vir: [3].

Vprašanje razvrstljivosti je, kako razvrstiti procese *A*, *B* in *C*, da se bodo zaključili do svojih skrajnih rokov. Najprej je treba preveriti, če je niz procesov sploh razvrstljiv. Če ima proces i periodo, ki traja P_i ms, in zahteva C_i ms CPE časa na sličico, potem je sistem razvrstljiv, če in samo če velja:

$$\sum_{i=1}^{m} \frac{C_i}{P_i} \leq 1,$$

kjer je m število procesov, v tem primeru 3. C_i/P_i predstavlja zasedenost CPE s strani procesa i. V zgornjem primeru zaseda proces *A* 10/30 CPE, *B* 15/40 in *C* 5/50. Skupaj procesi zasedajo 0˙808 CPE, torej je sistem procesov razvrstljiv.

Do sedaj je bila predpostavka, da imamo en proces na tok. Vendar imamo lahko tudi dva ali več procesov na tok, npr. enega za avdio in enega za video. Tečejo lahko pri različnih hitrostih in porabljajo različne količine CPE časa na izbruh. Dodajanje avdio procesov k mešanici procesov ne spremeni splošnega modela, če predpostavljamo, da imamo m procesov, ki tečejo (vsi) pri stalni frekvenci s stalno količino dela potrebnega na vsak CPE izbruh.

V nekaterih sistemih za delo v realnem času so procesi prekinljivi (angl. preemptable), v drugih ne. V multimedijskih sistemih so procesi običajno prekinljivi, kar pomeni, da lahko proces, ki je v nevarnosti, da bo zamudil svoj skrajni rok, prekine tekoči proces, preden je leta končal s svojo sličico. Ko je končan, se lahko prekinjeni proces nadaljuje. To ni nič drugega kot multiprogramiranje. V nadaljevanju smo predstavili prekinljive razvrščevalne algoritme v realnem času, saj jim v multimedijskih sistemih ni moč oporekati, so pa tudi performančno boljši od neprekinljivih. Skrbi nas lahko le v primeru, če je prenosni medpomnilnik (angl. transmission buffer) zapolnjen s kratkimi izbruhi. Medpomnilnik je v celoti zapolnjen s skrajnim rokom, tako da se lahko pošlje uporabniku v enojni operaciji. V nasprotnem primeru se lahko pojavi trepetanje (angl. jitter).

Algoritmi za razvrščanje v realnem času so lahko statični ali dinamični. Statični algoritmi vsakemu procesu vnaprej priredijo fiksno prioriteto, te prioritete pa potem uporablja prioritetno prekinjevalno razvrščanje (angl. preemptive scheduling). Dinamični algoritmi nimajo fiksnih prioritet.

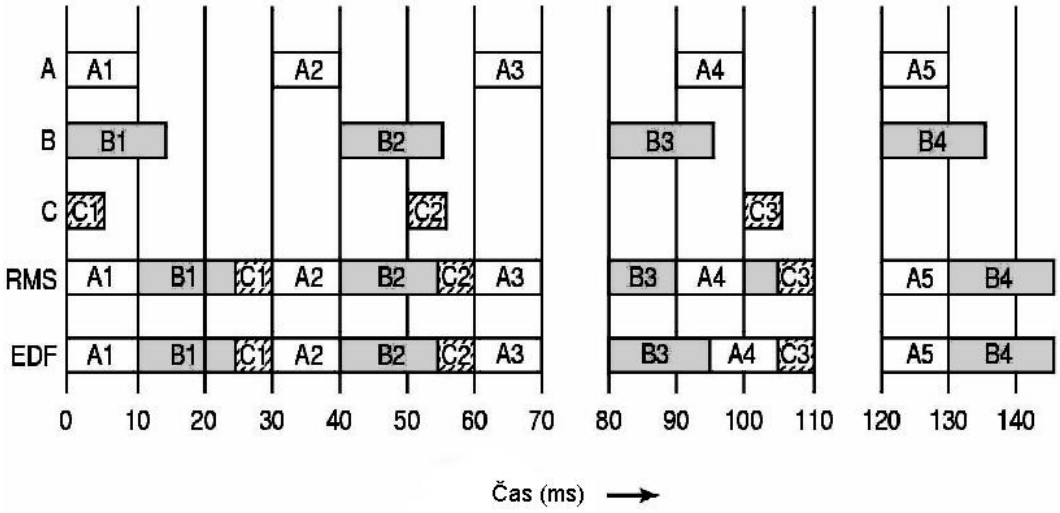
3.3 Razvrščanje po naraščajoči periodi – RMS

Klasični statični algoritem razvrščanja v realnem času za prekinljive, periodične procese je razvrščanje po naraščajoči periodi (angl. Rate Monotonic Scheduling – RMS). Uporablja se za procese, ki ustrezajo naslednjim pogojem:

1. Vsak periodični proces mora biti končan znotraj svoje periode.
2. Procesi morajo biti medsebojno neodvisni.
3. Vsak proces potrebuje enako količino CPE časa na vsak izbruh.
4. Neperiodični procesi nimajo skrajnih rokov.
5. Prekinjanje procesov je takojšnje, brez režije (angl. overhead).

Prvi štirje pogoji so logični, medtem ko zadnji ni, vendar močno poenostavlja modeliranje sistema. RMS deluje tako, da vsakemu procesu priredi fiksno prioriteto, enako frekvenci pojavitve njegovega prožilnega dogodka. Npr. proces, ki se mora zagnati vsakih 30 ms (33 krat/s), dobi prioriteto 33, tak, ki se zažene vsakih 40 ms (25 krat/s), prioriteto 25, proces, ki se zažene vsakih 50 ms (20 krat/s), pa dobi prioriteto 20. Prioritete so tako linearne glede na razmerje (število pojavitev/sekundo tekočega procesa), zato imenujemo tovrstno razvrščanje razvrščanje po naraščajoči periodi. V času izvajanja razvrščevalnik vedno zažene pripravljen proces z najvišjo prioriteto, ki lahko, če je potrebno, prekine tekoči proces. Dokazano je, da je

RMS optimalen v razredu statičnih razvrščevalnih algoritmov. Slika 3.2 prikazuje delovanje RMS algoritma za prejšnji primer, pa tudi algoritma EDF, ki smo ga predstavili v naslednjem poglavju.

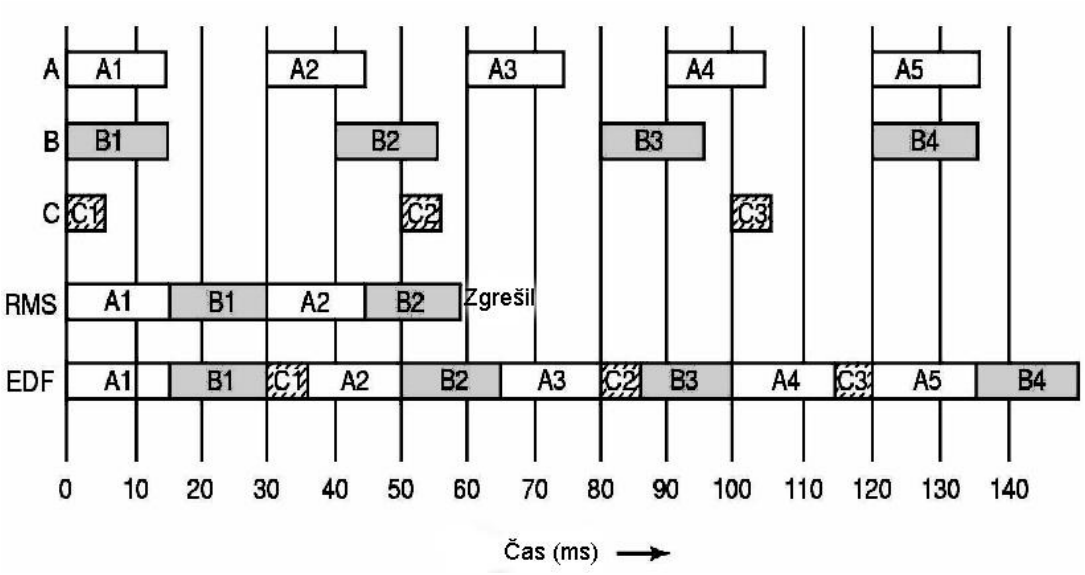


Slika 3.2 Primer RMS in EDF razvrščanja v realnem času. Vir: [3].

3.4 Razvrščanje po najbližjih skrajnih rokih – EDF

Razvrščanje po najbližjih skrajnih rokih (angl. Earliest Deadline First - EDF) je dinamični algoritem za razvrščanje v realnem času, ki ne zahteva, da so procesi periodični, niti ne zahteva enakega izvajalnega časa na CPE izbruh, tako kot to zahteva RMS. Kadarkoli potrebuje proces CPE čas, naznani svojo prisotnost in svoj skrajni rok. Razvrščevalnik hrani seznam tekočih procesov, ki so sortirani po skrajnih rokih. Algoritem zažene prvi proces na seznamu, ki ima najbližji skrajni rok. Kadarkoli je nov proces pripravljen, sistem preveri njegov skrajni rok, da ugotovi, ali se le-ta pojavi pred skrajnim rokom trenutno izvajajočega procesa. Če se, potem nov proces prekine trenutnega.

Da se znebimo občutka, da dajeta algoritma RMS in EDF vedno enake rezultate, si poglejmo naslednji primer (slika 3.3), kjer so periode procesov *A*, *B* in *C* enake kot poprej, le da sedaj potrebuje proces *A* 15 ms CPE časa na izbruh namesto 10 ms. Izračun testa razvrstljivosti nam da izkoristek CPE: 0˙500 + 0˙375 + 0˙100 = 0˙975. Le 2˙5% CPE je še neizkoriščene, vendar teoretično ni prekomerno zasedena, tako da bi naj bilo mogoče najti primeren urnik (angl. schedule).



Slika 3.3 Še en primer razvrščanja v realnem času z RMS in EDF. Vir: [3].

V tem primeru se RMS algoritem izneveri, saj proces *C* zgreši svoj skrajni rok. Zanimivo je vprašanje, zakaj se RMS izneveri. V osnovi deluje uporaba statičnih prioritet le, če ni izkoristek CPE prevelik. Dokazano je, da če za katerikoli sistem periodičnih procesov velja:

$$\sum_{i=1}^{m} \frac{C_i}{P_i} \leq m\left(2^{1/m} - 1\right),$$

potem RMS zagotovo deluje. Za 3, 4, 5, 10, 20 in 100 so maksimalni dovoljeni izkoristki 0˙780, 0˙757, 0˙743, 0˙718, 0˙705 in 0˙696. Ko raste m proti neskončnosti, se maksimalni izkoristek asimptotično približuje $ln2$. Drugače povedano, dokazano je, da za tri procese RMS vedno deluje, če je izkoristek CPE enak ali manjši 0˙780. V čisto prvem primeru je bil izkoristek CPE enak 0˙808 in je RMS vseeno deloval, vendar je bilo to le srečno naključje, kajti z drugačnimi periodami in izvajalnimi časi bi se zlahka zgodilo, da bi RMS pri izkoristku 0˙808 zatajil. V drugem primeru je bil izkoristek tako velik (0˙975), da ni bilo več upanja, da bi RMS lahko deloval.

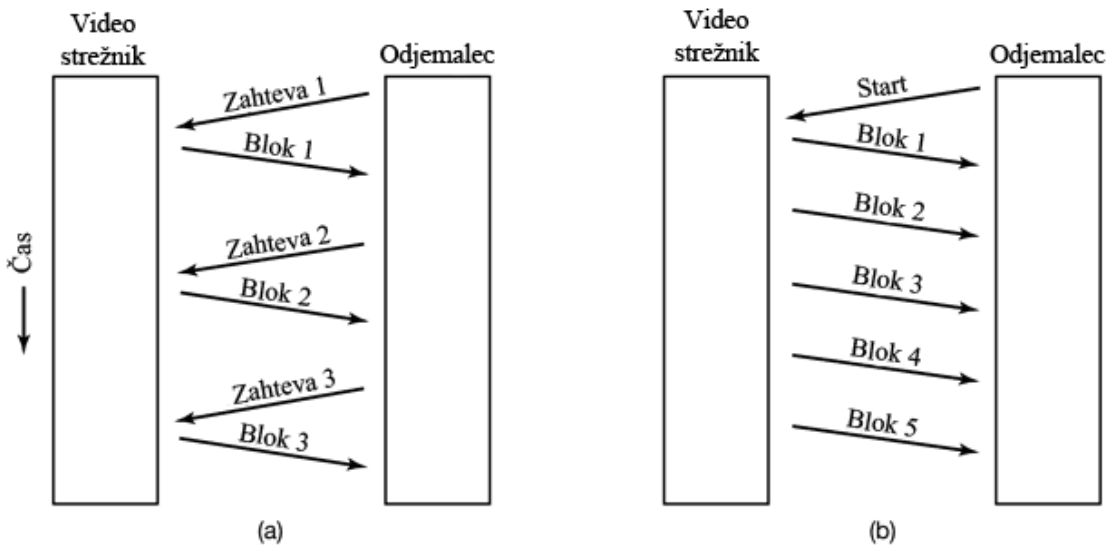
Za primerjavo, EDF venomer deluje za katerikoli razvrstljivi niz procesov in lahko doseže 100% izkoristek CPE. Cena za to je bolj kompleksen algoritem. Za dejanske video strežnike torej velja, da če je izkoristek CPE pod mejo, kjer deluje RMS, potem se lahko izbere RMS, drugače naj se izbere EDF.

4 Vzorec multimedijskega datotečnega sistema

Spregovorimo še nekaj besed o multimedijskih datotečnih sistemih, ki uporabljajo drugačen vzorec kot tradicionalni datotečni sistemi. Pri slednjih mora proces za dostop do datoteke najprej izdati sistemski klic open. Če to uspe, dobi klicalec neke vrste žeton, ki se imenuje datotečni deskriptor (angl. file descriptor) v operacijskem sistemu UNIX ali ročica (angl. handle) v Windows, za uporabo v prihodnjih klicih. Na tej točki lahko proces izda sistemski klic za branje read ter dobavi žeton, naslov medpomnilnika in število bajtov kot parametre. Operacijski sistem nato vrne zahtevane podatke v medpomnilnik. Lahko se naredijo dodatni klici za branje, dokler proces ni končan, ko kliče close za zapiranje datoteke in vrnitev njenih virov.

Ta model ne deluje dobro za multimedijo zaradi potrebe po realnočasovnem obnašanju. Še posebno slabo deluje za prikazovanje multimedijskih datotek, ki se vršijo na oddaljenem video strežniku. En problem je, da mora uporabnik delati klice za branje precej natančno razporejene v času. Drug problem pa je, da mora biti video strežnik sposoben dostavljati podatkovne bloke brez zakasnitve, kar je zanj težko, če prihajajo zahteve nenapovedano in če niso bili viri vnaprej rezervirani.

Za reševanje teh problemov uporabljajo multimedijski datotečni sistemi popolnoma drugačen vzorec: obnašajo se kot videorekorderji (angl. Video Cassette Recorders – VCRs). Uporabniški proces izda za branje multimedijske datoteke sistemski klic start s podatki, katera datoteka naj bo brana in z raznimi drugimi parametri, kot na primer, katere zvokovne sledi in podnapise uporabiti. Video strežnik nato začne odpošiljati sličice z zahtevano hitrostjo. Na uporabniku je, da ravna z njimi pri hitrosti, s katero prihajajo. Če postane uporabniku ob filmu dolgčas, sistemski klic stop konča tok. Datotečne strežnike s tem pretočnim modelom pogosto imenujemo **potisni strežniki** (angl. push servers), ker potisnejo podatke k uporabniku, v primerjavi s tradicionalnimi **poteznimi strežniki** (angl. pull servers), kjer mora uporabnik potegniti podatke v enem bloku naenkrat s ponavljajočim klicanjem klica read, da dobi en blok za drugim. Razlika med opisanima modeloma je prikazana na sliki 4.1.



Slika 4.1 (a) Potezni strežnik. (b) Potisni strežnik. Vir: [3].

5 Predpomnjenje

Običajno LRU (angl. least recently used) predpomnjenje datotek (angl. file caching) z multimedijskimi datotekami ne deluje dobro, saj se vzorci dostopa za filme razlikujejo od tistih za tekstovne datoteke. Ideja, ki stoji za običajnimi LRU predpomnilniki z medpomnilniki (angl. LRU buffer caches), je, da naj bi bil blok po uporabi zadržan v predpomnilniku še nekaj časa, če bi se v kratkem morda ponovno potreboval. Na primer, da med urejanjem datoteke, množica blokov, na kateri je zapisana datoteka, vseskozi teži k ponovni uporabi, dokler urejanje ni končano. Z drugimi besedami, če obstaja relativno visoka verjetnost, da bo blok ponovno uporabljen znotraj kratkega intervala, potem ga je vredno zadrževati v bližini, da se izločijo prihodnji dostopi do diska.

Pri multimediji je običajen vzorec dostopa, da se film gleda zaporedno od začetka do konca. Le malo je verjetno, da bo nek blok uporabljen še drugič, razen če uporabnik previje film nazaj, da ponovno vidi nek prizor. Posledično običajne tehnike predpomnjenja ne delujejo. Predpomnjenje je še vedno koristno, vendar le, če se ga uporabi drugače. V nadaljevanju je predstavljeno predpomnjenje za multimedijo.

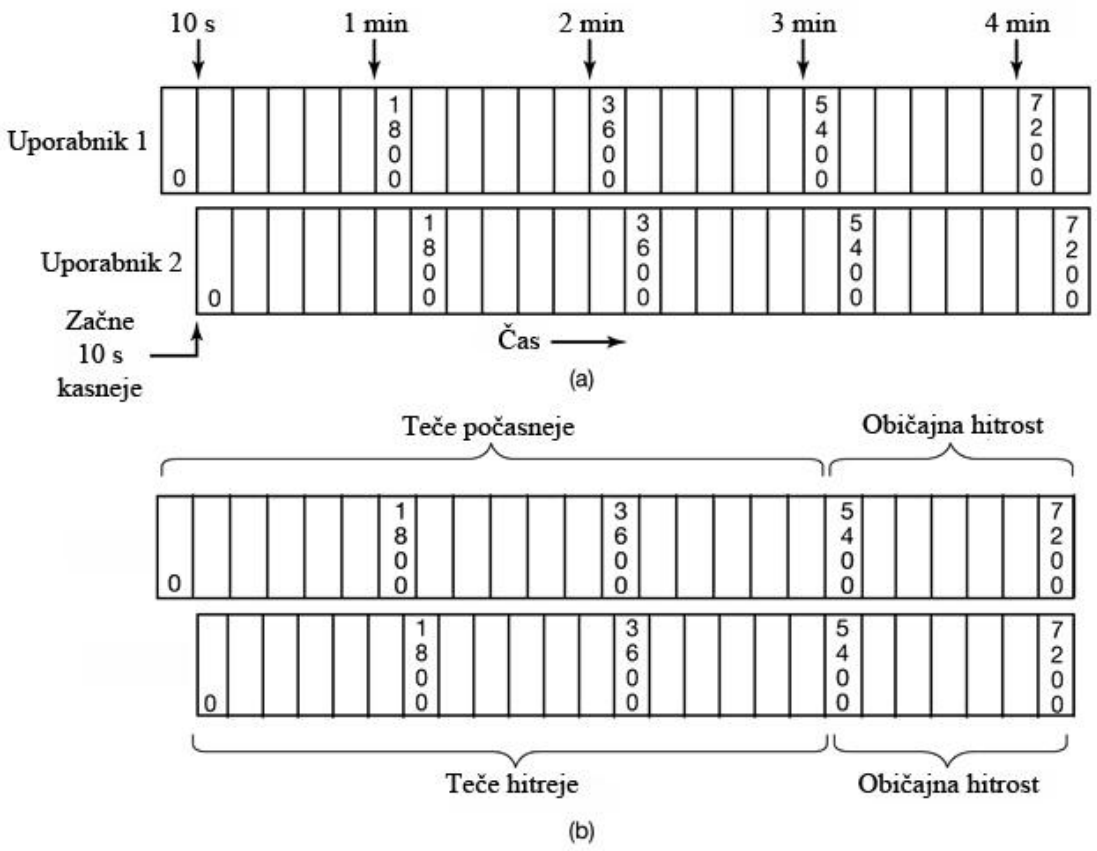
5.1 Predpomnjenje blokov

Čeprav je zadrževanje bloka v bližini v upanju, da bi ta lahko bil hitro znova uporabljen, nesmiselno, se lahko predvidljivost multimedijskih sistemov izkoristi za to, da se naredi predpomnjenje ponovno uporabno. Predpostavimo, da dva uporabnika gledata isti film, pri čemer ga je eden od njiju začel gledati dve sekundi kasneje kot drugi. Zatem, ko je prvi uporabnik prejel in pogledal nek blok, je zelo verjetno, da bo isti blok potreboval tudi drugi uporabnik dve sekundi kasneje. Sistem lahko enostavno spremlja, kateri filmi imajo le enega gledalca in kateri dva ali več gledalcev, ki si sledijo v kratkem časovnem razmaku.

Torej, kadarkoli je blok bran na račun filma, ki bo v kratkem zopet potreben, bi ga bilo smiselno držati v predpomnilniku, odvisno od tega, kako dolgo bi moral biti v predpomnilniku in kako tesen je pomnilnik. Namesto zadrževanja vseh diskovnih blokov v predpomnilniku in zamenjevanja tistih (ko se predpomnilnik napolni), do katerih dostop najdlje ni bil narejen, naj bi se uporabila drugačna strategija. Vsak film, ki ima drugega gledalca znotraj določenega časovnega obdobja ΔT od prvega gledalca, se lahko označi kot predpomnitven (angl. cacheable), njegovi bloki pa ostanejo v predpomnilniku, dokler jih drugi (in morda tretji) gledalec ne uporabi. Za ostale filme se predpomnjenje ne uporablja.

To zamisel je mogoče razviti še korak dlje. V nekaterih primerih bi bilo mogoče dva toka združiti. Predpostavimo, da dva uporabnika gledata isti film, vendar z 10 sekundnim razmakom. Držanje blokov v predpomnilniku za 10 sekund je mogoče, vendar trati pomnilnik. Alternativni, toda rahlo pritajeni, pristop je, da se skuša oba filma časovno uskladiti

(sinhronizirati). To se lahko doseže s spreminjanjem hitrosti sličic obeh filmov. Ideja je ponazorjena na sliki 5.1.



Slika 5.1 (a) Dva uporabnika gledata isti film z 10 sekundnim razmakom.
(b) Združevanje obeh tokov v enega.
Vir: [3].

Na sliki 5.1(a) oba filma tečeta pri standardni NTSC hitrosti 1800 sličic/min. Če začne uporabnik 2 gledati film z 10-sekundno zamudo, bo skozi celoten film zaostajal za 10 sekund glede na uporabnika 1. Na sliki 5.1(b) se tok uporabnika 1 upočasni, ko se pojavi uporabnik 2. Namesto, da bi tekel pri hitrosti 1800 sličic/min, teče naslednje tri minute pri hitrosti 1750 sličic/min. Po treh minutah je pri sličici 5550. Poleg tega se tok uporabnika 2 prve tri minute predvaja pri hitrosti 1850 sličic/min, tako da tudi sam pride do sličice 5550. Od te točke dalje se oba tokova predvajata pri običajni hitrosti.

Med periodo dohitevanja (angl. catch-up period) teče tok prvega uporabnika 2˙8% počasneje, tok drugega uporabnika pa 2˙8% hitreje. Zelo malo je verjetno, da bosta uporabnika to zaznala. Vendar, če nas to skrbi, se lahko perioda dohitevanja razširi na interval, ki je daljši od treh minut.

Alternativni način, da upočasnimo tok nekega uporabnika in ga spojimo z drugim, je, da ponudimo uporabnikom možnost oglasnih sporočil med predvajanjem njihovih filmov, domnevno za nižjo ceno gledanja kot pri filmih, ki oglasnih sporočil ne vsebujejo. Uporabnik lahko izbira tudi kategorijo oglaševanih izdelkov, tako da so oglasi manj vsiljivi in bolj gledani. S prikrojevanjem števila, dolžine in tempiranja oglasov se lahko tok dovolj dolgo zadrži, da se časovno uskladi z željenim tokom.

5.2 Predpomnjenje datotek

Predpomnjenje je lahko v multimedijskih sistemih uporabno tudi na drugačen način. Zaradi precejšnje velikosti večine filmov (npr. 2 GB ali več), video strežniki pogosto ne morejo hraniti vseh filmov na disku, zato se ti hranijo na DVD-jih ali trakovih. Ko se film potrebuje, se lahko vedno kopira na disk, vendar pa obstaja precejšnji zagonski čas za lociranje filma in njegovo kopiranje. Večina video strežnikov zato obdrži najbolj zahtevane filme v diskovnem predpomnilniku. Priljubljeni filmi so v celoti shranjeni na disku.

Drug način za uporabo predpomnjenja je, da na disku hranimo prvih nekaj minut vsakega filma. Tako se lahko začne posnetek, ko se pojavi zahteva po filmu, nemudoma predvajati iz datoteke na disku. Medtem se film prekopira z DVD-ja ali traku na disk. Z neprestanim shranjevanjem zadostnega dela filma na disk je mogoče doseči zelo visoko verjetnost, da bo naslednji kos filma izstavljen, preden se bo potreboval. Če gre vse po načrtih, bo celoten film na disku veliko prej, preden nastopi potreba po njem. Potem se film prebere v predpomnilnik. Za vsak primer ostane še nekaj časa na disku, če se kasneje pojavi še več zahtev po njem. Če mine preveč časa brez pojavitve nove zahteve, se film iz predpomnilnika odstrani, da se sprosti prostor za film, ki je priljubljenejši.

6 Razvrščanje zahtev za disk pri multimediji

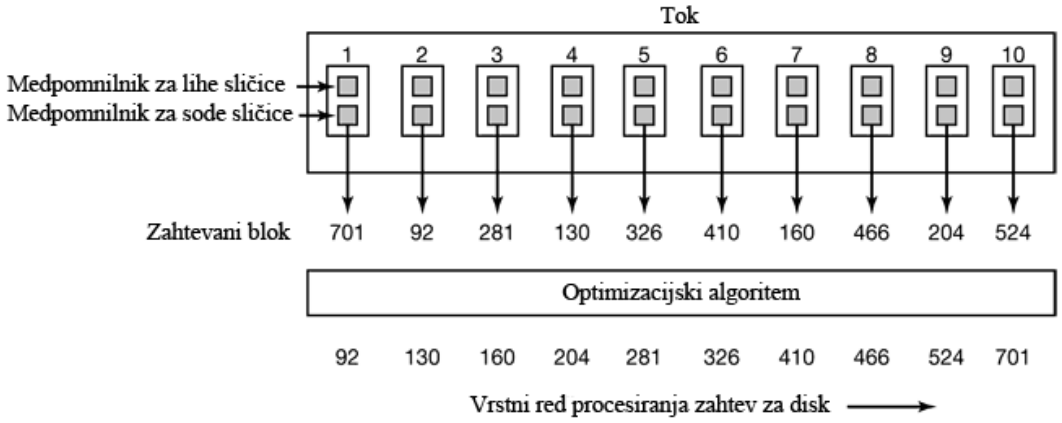
Multimedija nalaga diskom drugačne zahteve kot običajne tekstovno orientirane aplikacije, kot so npr. prevajalniki ali urejevalniki besedil. Še posebno zahteva ekstremno visok prenos podatkov in dostavo podatkov v realnem času, kar pa ni trivialno zagotoviti. Poleg tega se pri video strežnikih pojavlja hud ekonomski pritisk, tako da morajo posamezni strežniki sočasno ravnati s tisočimi strankami. Te zahteve močno vplivajo na celoten sistem.

6.1 Statično diskovno razvrščanje

Kljub enormnim realnočasovnim zahtevam ter zahtevam po visokem prenosu podatkov, ki jih multimedija nalaga vsem delom sistema, ima le-ta eno lastnost, ki olajšuje rokovanje v primerjavi z običajnim sistemom, in sicer predvidljivost. V običajnih operacijskih sistemih so zahteve za diskovne bloke narejene precej nepredvidljivo. Najboljše, kar lahko naredi diskovni podsistem je, da izvrši vnaprejšnje branje enega bloka za vsako odprto datoteko. Poleg tega je vse, kar lahko naredi, to, da čaka na prihajajoče zahteve in jih procesira na zahtevo. Pri multimediji je drugače. Vsak aktivni tok naloži sistemu dobro definirano breme, ki je zelo predvidljivo. Pri NTSC posnetku hoče vsaka posamezna stranka na vsakih 33˙3 ms naslednjo sličico v svoji datoteki, tako da ima sistem na voljo 33˙3 ms, da zagotovi vse sličice (sistem mora dati v medpomnilnik vsaj eno sličico na tok, da lahko poteka izstavljanje (angl. fetching) sličice $k + 1$ vzporedno s predvajanjem sličice k).

To predvidljivo breme se lahko uporabi za diskovno razvrščanje, ki uporablja algoritme, narejene za multimedijske operacije. Smatrali smo, da imamo en disk, čeprav lahko idejo posplošimo tudi na več diskov. V sledečem primeru smo predpostavili, da imamo 10 uporabnikov, izmed katerih vsak gleda drug film in da imajo vsi filmi enako ločljivost, hitrost sličic in ostale značilnosti.

Odvisno od preostalega sistema ima lahko računalnik 10 procesov, enega na video tok, ali en proces z 10 nitmi, ali celo en proces z eno nitjo, ki ravna z 10 tokovi na način krožnega dodeljevanja. Podrobnosti niso pomembne. Kar je pomembno, je to, da je čas razdeljen na **runde** (angl. rounds), kjer vsaka runda predstavlja čas sličice (33˙3 ms za NTSC, 40 ms za PAL). Na začetku vsake runde se na račun vsakega uporabnika gerenerira ena zahteva za disk, kot prikazuje slika 6.1.



Slika 6.1 V eni rundi vsak film zahteva eno sličico. Vir: [3].

Zatem, ko se na začetku runde pojavijo vse zahteve, disk ve, kaj mora narediti tekom te runde. Prav tako ve, da se ne bo pojavila nobena nova zahteva, dokler ne bodo tekoče zahteve obdelane in dokler se ne prične naslednja runda. Posledično lahko zahteve optimalno posortira, če je možno v cilindrskem vrstnem redu (čeprav je v nekaterih primerih sprejemljiv tudi sektorski vrstni red), in jih nato v optimalnem vrstnem redu tudi obdela. Na sliki 6.1 so zahteve sortirane v cilindrskem vrstnem redu.

Na prvi pogled se zdi, da optimizacija diska v tem pogledu ničesar ne doprinese, saj dokler disk zadovolji skrajnemu roku, je vseeno, ali ostane na voljo še 1 ms, ali pa 10 ms. Kakorkoli že, takšno predvidevanje ni pravilno. S tovrstno optimizacijo iskanj se povprečen čas procesiranja vsake zahteve zmanjša, kar pomeni, da lahko disk v povprečju upravlja z večimi tokovi na rundo. Drugače povedano, optimizacija diskovnih zahtev v tem oziru zvišuje število filmov, ki jih lahko strežnik simultano prenese. Prosti čas na koncu runde se lahko uporabi za servisiranje katerekoli druge zahteve, ki ne potrebuje obdelave v realnem času.

Če ima strežnik preveč tokov, se lahko občasno ob prevzemu sličic iz oddaljenega dela diska zgodi, da zamudi skrajni rok. Vendar dokler so zamujeni roki dovolj redki, se lahko tolerirajo v zameno za rokovanje z večimi tokovi naenkrat. Kar je pomembno, je število tokov, ki se prevzemajo. Če si dve ali več strank lasti isti tok, to ne vpliva na učinkovitost diska ali na razvrščanje.

Da ohranimo gladek tok podatkov vse do strank, potrebujemo v strežniku dvojno medpomnjenje (angl. double buffering). Tekom runde 1 je uporabljen en niz medpomnilnikov, in sicer en na tok. Ko je runda končana, se izhodni proces ali procesi odklenejo in zaukazano jim je, da prenesejo sličico 1. Istočasno se pojavijo nove zahteve za sličico 2 vsakega filma (lahko imamo diskovno nit in izhodno nit za vsak film). Tem zahtevam mora biti ugodeno z uporabo drugega niza medpomnilnikov, medtem ko so prvi medpomnilniki še vedno zasedeni. Ko se začne runda 3, je prvi niz medpomnilnikov zopet prazen in se lahko uporabi za prevzem sličice 3.

Predpostavili smo, da imamo eno rundo na sličico. Ta omejitev ni strogo potrebna. Lahko bi imeli dve rundi na sličico, da bi zmanjšali količino potrebnega medpomnilniškega prostora, vendar za ceno dvakratnega števila diskovnih operacij. Podobno bi lahko iz diska prevzeli dve sličici na rundo (če predpostavimo, da so si pari sličic na disku sosedni). V tem primeru bi se število diskovnih operacij razpolovilo na račun dvakratne količine medpomnilniškega prostora, ki bi bila potrebna. Z ozirom na relativno dosegljivost, zmogljivost in ceno pomnilnika v primerjavi z diskovnim V/I, se lahko izračuna in uporabi optimalna strategija.

6.2 Dinamično diskovno razvrščanje

V prejšnjem primeru smo predpostavili, da imajo vsi tokovi enako ločljivost, hitrost sličic in ostale lastnosti. Pa sedaj to predpostavko ovrzimo. Različni filmi imajo lahko sedaj različne hitrosti sličic, tako da ni mogoče imeti ene runde vsakih 33˙3 ms in prevzeti ene sličice za vsak tok. Zahteve za disk prihajajo bolj ali manj naključno.

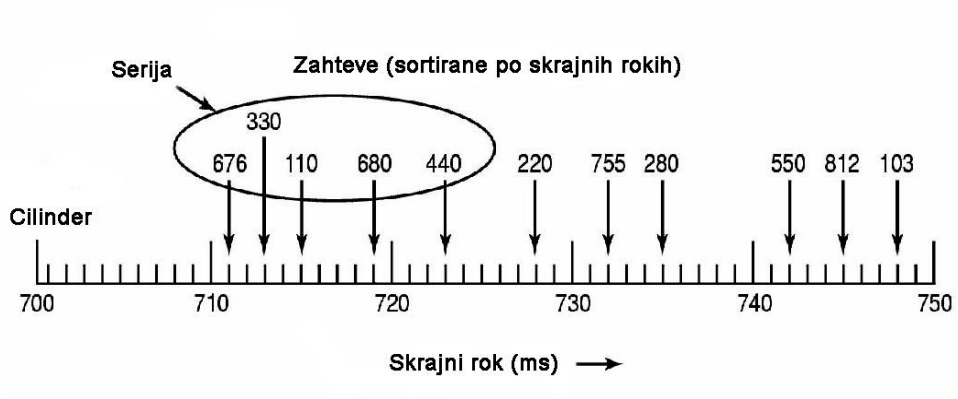
Vsaka bralna zahteva določi, kateri blok naj se prebere in tudi čas, kdaj se blok potrebuje (skrajni rok). Za poenostavitev lahko predpostavimo, da je dejanski strežni čas za vsako zahtevo enak (čeprav to vsekakor ne drži). Z ozirom na to lahko od vsake zahteve odštejemo fiksni strežni čas in tako dobimo najkasnejši možni čas, v katerem se še lahko sproži zahteva, da še doseže svoj skrajni rok. To močno poenostavi model, kajti diskovnemu razvrščevalniku je pomembno le, da ujame skrajni rok za razvrščanje zahteve.

Ko se sistem zažene, še ni nobene diskovne zahteve v teku. Ko prispe prva zahteva, je le-ta postrežena takoj. Ko poteka prvo iskanje, se lahko pojavijo nove zahteve, se pravi, da bo imel diskovni gonilnik, ko bo prva zahteva zaključena, na voljo, katero zahtevo naj začne naslednjo obdelovati. Neka zahteva je izbrana ter pognana in ko se konča, imamo zopet niz možnih zahtev: tistih, ki niso bile prvikrat izbrane, in tistih, ki so prišle na novo med procesiranjem druge zahteve. V splošnem, kadarkoli se diskovna zahteva zaključi, ima diskovni gonilnik v teku kakšen niz zahtev, med katerimi se mora odločiti. Vprašanje, ki se zastavlja, je, kakšen algoritem uporabiti za izbiro naslednje zahteve za strežbo.

Dva faktorja vplivata na izbiro naslednje diskovne zahteve, in sicer skrajni roki in cilindri. S performančnega vidika držanje zahtev sortiranih glede na cilinder in uporaba algoritma po principu dvigala (angl. elevator algorithm) minimizirata absolutni iskalni čas, toda lahko povzročita, da zahteve na oddaljenih cilindrih zgrešijo svoje skrajne roke. Z vidika realnega časa pa sortiranje zahtev po skrajnih rokih in njihovo izvrševanje v vrstnem redu glede na skrajni rok (EDF) minimizirata možnost zgrešitve skrajnega roka, a povečujeta absolutni iskalni čas.

Oba faktorja lahko upoštevamo, če uporabimo **prebirni-EDF algoritem** (angl. scan-EDF algorithm). Osnovna ideja tega algoritma je kopičiti zahteve, katerih skrajni roki so si

relativno blizu, v serije in le-te procesirati v cilindrskem vrstnem redu. Kot primer si poglejmo situacijo na sliki 6.2 pri t = 700 ms. Diskovni gonilnik ve, da je v teku 11 zahtev za različne skrajne roke in različne cilindre. Lahko se na primer odloči obravnavati pet zahtev z najbližjimi skrajnimi roki kot serijo, jih sortirati po cilindrski številki ter nato uporabiti algoritem po principu dvigala, da jih postreže v cilindrskem vrstnem redu. Vrstni red bi bil potem 110, 330, 440, 676 in 680. Dokler je vsaka zahteva zaključena pred svojim skrajnim rokom, se le-te lahko varno prerazporedi tako, da se minimizira zahtevani absolutni iskalni čas.



Slika 6.2 Prebirni-EDF algoritem uporablja za razvrščanje skrajne roke in številke cilindrov. Vir: [3].

Ker imajo različni tokovi različne hitrosti prenosa podatkov, se pojavi kočljivo vprašanje, ko se prikaže nova stranka, in sicer, ali naj bo stranka sprejeta ali ne. Če bo sprejetje stranke povzročilo, da bodo drugi tokovi začeli redneje zamujati svoje skrajne roke, potem se odgovor verjetno glasi ne. Obstajata dva načina, kako izračunati, ali sprejeti stranko ali jo zavrniti. Prvi način je, da predpostavimo, da vsaka stranka v povprečju potrebuje določeno količino resursov, npr. diskovno pasovno širino, medpomnilnike, CPE čas itd. Če je vsakega resursa še dovolj za povprečno stranko, potem se lahko nova stranka sprejme.

Drug algoritem je bolj izčrpen, saj mora imeti vpogled v specifičen film, ki ga stranka želi, in preveriti (vnaprej izračunano) hitrost prenosa podatkov za ta film, ki se razlikuje za črnobele filme v primerjavi z barvnimi, za risanke v primerjavi s filmi itd. Če ima strežnik dovolj kapacitete za specifičen film, ki si ga stranka želi, potem se sprejem stranke potrdi, v nasprotnem primeru pa zavrne.

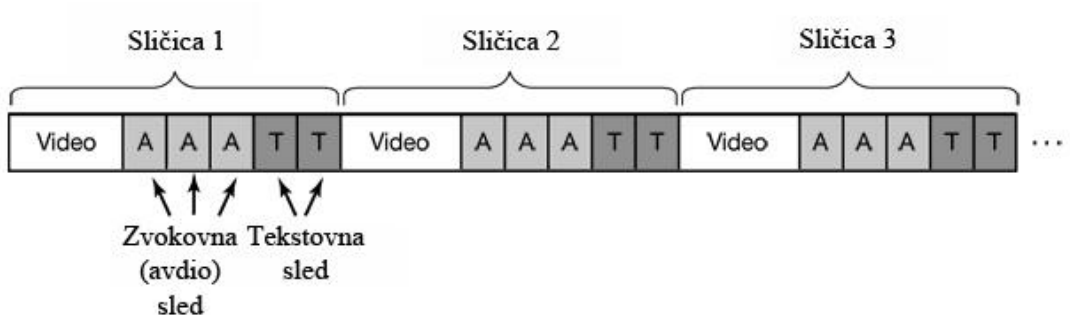
7 Razmeščanje multimedijskih datotek

Multimedijske datoteke so zelo velike. Ponavadi so zapisane le enkrat, a velikokrat prebrane, dostop do njih pa je zaporeden. Njihovo predvajanje mora ustrezati strogim kriterijem kvalitete storitve (angl. quailty of service - QoS). Skupaj te zahteve narekujejo drugačno zasnovo datotečnega sistema, kot jo uporabljajo tradicionalni operacijski sistemi.

7.1 Razmeščanje datoteke na enem disku

Najpomembnejša zahteva je, da se lahko podatki pretakajo k omrežju ali izhodni napravi z zahtevano hitrostjo in brez trepetanja. Iz tega razloga so mnogokratna iskanja v času trajanja ene sličice zelo nezaželena. En način, da se izognemo meddatotečnim (angl. intrafile) iskanjem na video strežnikih, je uporaba strnjenih (angl. contiguous) datotek. Običajno se rešitev s strnjenimi datotekami ne obnese dobro, toda na video strežniku, ki je skrbno vnaprej prednaložen s filmi, ki se naknadno ne spremenijo, se lahko izkaže za dobro.

Težava, ki se pojavlja, je tudi prisotnost ločenega videa, zvoka in teksta. Čeprav so video, zvok in tekst vsak shranjeni kot ločene strnjene datoteke, je potrebno iskanje za prehod iz video datoteke na zvokovno datoteko ter nato še na tekstovno, če se pojavi potreba. To narekuje drugo možno ureditev shranjevanja (angl. storage arrangement), in sicer tako, da se video, zvok in tekst prepletajo (angl. interleaving), vendar celotna datoteka ostaja strnjena, kar prikazuje slika 7.1. Videu za sličico 1 sledijo različne zvokovne sledi (angl. audio tracks) za isto sličico in nato razne tekstovne sledi, prav tako v okviru prve sličice. Odvisno od števila zvokovnih in tekstovnih sledi bi bilo verjetno najenostavneje prebrati vse kose za vsako sličico v enojni bralni operaciji diska in prenesti le potrebne dele do uporabnika.



Slika 7.1 Prepletanje videa, zvoka in teksta v eni strnjeni datoteki za vsak film. Vir: [3].

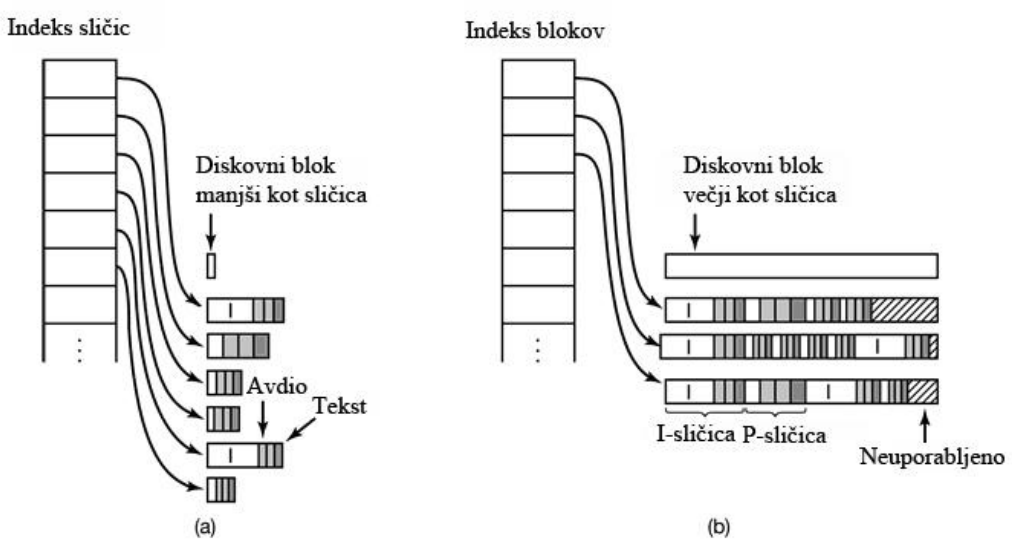
Takšna organizacija zahteva dodaten diskovni V/I za branje neželenega zvoka in teksta ter dodaten medpomnilniški prostor v pomnilniku za shranjevanje, odpravlja pa vsa iskanja (na

enouporabniškem sistemu) in ne terja nobene režije za vzdrževanje sledi, kje se katera sličica na disku nahaja, saj je celoten film shranjen v eni strnjeni datoteki. Naključni dostop pri tej ureditvi je izključen, toda če ni potreben, izgube zaradi tega niso velike. Podobno sta hitro previjanje nazaj in naprej neizvedljiva brez dodatnih podatkovnih struktur in kompleksnosti.

Prednost hrambe celotnega filma v eni strnjeni datoteki se izgubi na video strežniku z več sočasnimi izhodnimi tokovi, ker bo moral disk po branju sličice enega filma prebrati sličice še mnogih drugih filmov, preden se bo ponovno vrnil na prvega. Prav tako je za sistem, v katerem so filmi narejeni, kakor tudi brani (na primer sistem za video produkcijo ali urejanje), uporaba ogromnih strnjenih datotek težavna in ne ravno učinkovita.

7.2 Dve alternativni strategiji organizacije datotek

Zgornja opažanja vodijo do dveh različnih organizacij razmeščanja multimedijskih datotek. Prvo – model malih blokov – prikazuje slika 7.2(a). V tej organizaciji je izbrana velikost bloka na disku precej manjša od povprečne velikosti sličice, celo za P-sličice in B-sličice. Za MPEG-2 pri 4 MB/s s 30 sličicami/s meri povprečna sličica 16 KB, zato bo blok velikosti 1 KB ali 2 KB dobro deloval. Ideja je, da imamo za vsak film podatkovno strukturo – indeks sličic (angl. frame index). Vsaki sličici pripada en vnos v indeksu sličic, ki kaže na začetek sličice. Posamična sličica sestoji iz vseh video, zvokovnih in tekstovnih sledi za to sličico kot strnjen niz diskovnih blokov. Branje sličice *k* je tako sestavljeno iz indeksiranja v indeksu sličic, da najdemo *k*-ti vnos, in nato prebiranja celotne sličice v eni diskovni operaciji. Ker so različne sličice različno velike, potrebujemo velikost sličice (v blokih) v indeksu sličic, a tudi če imamo 1-KB diskovne bloke, 8-bitno polje zadošča za sličico velikosti do 255 KB, kar je dovolj za nezgoščeno NTSC sličico, tudi z mnogimi zvokovnimi sledmi.



Slika 7.2 Nestrnjena hramba filma. (a) Mali diskovni bloki. (b) Veliki diskovni bloki. Vir: [3].

Drug način za shranjevanje filma je uporaba velikih diskovnih blokov (recimo 256 KB) z večdelnimi sličicami v vsakem bloku, kakor je razvidno s slike 7.2(b). Indeks je še vedno potreben, toda sedaj imamo namesto indeksa sličic indeks blokov. Indeks je pravzaprav v osnovi enak indeksnemu vozlišču (angl. i-node), kot ga poznamo pri UNIX-u, z morebitno dodatno informacijo, ki nam pove, katera sličica je na začetku vsakega bloka, da je mogoče hitro najti dano sličico. V splošnem blok ne bo hranil celotnega števila sličic, tako da je potrebno v zvezi s tem nekaj ukreniti. Obstajata dve možnosti.

Prva možnost je ilustrirana na sliki 7.2(b) - kadar se naslednja sličica ne prilega v trenutni blok, ostane preostanek bloka prazen. Ta neizkoriščeni prostor je notranja fragmentacija (angl. internal fragmentation), kot jo imamo v sistemih z navideznim pomnilnikom s fiksno velikostjo strani. Iskanje sredi sličice v tem primeru ni nikoli potrebno.

Druga možnost je, da se blok napolni do konca in sličice razdelijo čez bloke. Ta možnost vpeljuje potrebo po iskanjih sredi sličic, kar lahko negativno učinkuje na delovanje, vendar prihrani prostor na disku z odpravo notranje fragmentacije.

Za primerjavo lahko ugotovimo, da uporaba malih blokov v modelu na sliki 7.2(a) prav tako zapravi nekaj prostora na disku, saj ostane del zadnjega bloka vsake sličice neizkoriščen. Pri velikosti bloka na disku 1 KB in dveurnem NTSC filmu, ki ga sestavlja 216.000 sličic, bo neizkoriščenih le okoli 108 KB diskovnega prostora od 3˙6 GB. Neizkoriščenost prostora je teže izračunati za model s slike 7.2(b), vendar bo ta bržkone veliko večja, saj bo občasno na koncu bloka ostalo 100 KB, čemur bo sledila I-sličica, večja od tega.

Po drugi strani pa je indeks blokov veliko manjši od indeksa sličic. Pri velikosti bloka 256 KB in povprečni velikosti sličice 16 KB se bloku prilega približno 16 sličic, tako da potrebuje film z 216.000 sličicami le 13.500 vnosov v indeksu blokov, medtem ko rabi v indeksu sličic 216.000 vnosov. Za doseganje večje učinkovitosti (angl. performance) naj bi indeks v obeh primerih vseboval vnose vseh sličic ali blokov (to je brez posrednih blokov kot pri UNIX-u). Ugotovimo lahko, da 13.500 8-bajtnih vnosov (4 B za naslov diska, 1 B za velikost sličice in 3 B za številko začetne sličice), ki zasedajo prostor v pomnilniku, v primerjavi z 216.000 5-bajtnimi vnosi (le naslov diska in velikost), prihrani skoraj 1 MB RAM-a med predvajanjem filma.

Že zapisane ugotovitve vodijo do naslednjih kompromisov:

1. Indeks sličic: večja poraba RAM-a med predvajanjem filma; majhna potrata diska.
2. Indeks blokov (brez delitve sličic prek blokov): majhna poraba RAM-a; večja potrata diska.
3. Indeks blokov (delitev sličic prek blokov je dovoljeno): majhna poraba RAM-a; ni potrate diska; dodatna iskanja.

Kompromisi tako vključujejo porabo RAM-a med predvajanjem posnetka, potrato diskovnega prostora ves čas in izgubo učinkovitosti (angl. performance loss) med predvajanjem zaradi dodatnih iskanj. Te probleme se lahko rešuje na različne načine. Poraba RAM-a se lahko

zmanjša z ostranjevanjem (angl. paging) v delih tabele sličic ravno pravočasno (angl. just in time). Iskanja med prenašanjem sličic se lahko maskirajo z zadostnim medpomnjenjem (angl. buffering), vendar to vpelje potrebo po dodatnem pomnilniku in po vsej verjetnosti dodatnem kopiranju. Dobro načrtovanje mora vsebovati natančno analizo vseh teh dejavnikov, da se lahko naredi dober izbor za priročno aplikacijo.

Pojavlja se še en dejavnik. Upravljanje shranjevanja na disk (angl. disk storage management) je bolj zapleteno v primeru s slike 7.2(a), saj shranjevanje sličice zahteva, da se najde prostor prave velikosti na disku za zaporeden niz blokov. V idealnem primeru naj ta niz blokov ne bi prekoračil meje diskovne sledi, če pa se uporabi zamik glave (angl. head skew), izgube zaradi tega niso velike. Prekoračenju meje cilindra naj bi se izogibali. Te zahteve pomenijo, da je bolje, če je prazen prostor na disku organiziran kot seznam lukenj različnih velikosti, kot pa enostaven seznam blokov ali bitni vzorec, kar se lahko oboje uporabi v primeru s slike 7.2(b).

Vsi navedeni primeri govorijo v prid dajanju blokov ali sličic filma znotraj ozkega področja, recimo nekaj cilindrov, kjer je mogoče. Takšna razmestitev pomeni, da bodo iskanja potekala hitreje in bo več časa ostalo za druge aktivnosti (ki ne potekajo v realnem času) ali podporo dodatnim video tokovom. Omejena razmestitev te vrste se lahko doseže z deljenjem diska na skupine cilindrov in vzdrževanjem ločenih seznamov ali bitnih vzorcev prostih blokov za vsako skupino. Če so uporabljene luknje, imamo lahko na primer en seznam za 1-KB luknje, enega za 2-KB luknje, naslednjega za luknje od 3 KB do 4 KB, še naslednjega za luknje velikosti 5 KB do 8 KB in tako naprej. Na ta način je enostavno najti luknjo primerne velikosti v dani skupini cilindrov.

Obravnavana pristopa razmeščanja se razlikujeta tudi v medpomnjenju (angl. buffering). Pri pristopu z malimi bloki dobi vsako branje natanko eno sličico. Enostavna strategija z dvojnim medpomnjenjem posledično dobro deluje: en medpomnilnik za predvajanje trenutne sličice in eden za prevzem naslednje. Če so uporabljeni fiksni medpomnilniki, mora biti vsak izmed njih dovolj velik za največjo možno I-sličico. Če pa je za vsako sličico dodeljen drug medpomnilnik iz zaloge (angl. pool) in je velikost sličice znana, preden je le-ta prebrana, se lahko uporabi majhen medpomnilnik za P- ali B-sličico.

Pri pristopu z velikimi bloki je potrebna kompleksnejša strategija, saj vsak blok vsebuje več sličic, po možnosti vključujoč delce sličic na vsakem koncu bloka (odvisno od predhodno izbrane možnosti). Če prikazovanje ali prenašanje sličic zahteva, da so te strnjene, se morajo kopirati, kopiranje pa je draga operacija, zato bi se mu morali, kjer je mogoče, izogniti. Če strnjenost ni zahtevana, so lahko sličice, ki se raztezajo prek meja blokov, odposlane preko omrežja ali do prikazovalne naprave v dveh kosih.

Dvojno medpomnjenje se lahko uporabi tudi pri velikih blokih, vendar uporaba dveh velikih blokov trati pomnilnik. En način, da se izognemo tratenju pomnilnika, je, da imamo na vsak tok krožni prenosni medpomnilnik (angl. circular transmission buffer), ki polni omrežje ali zaslon in je malo večji od diskovnega bloka. Ko pade vsebina medpomnilnika pod nek prag (oz. ko se ta do neke mere izprazni), se vanj iz diska prebere nov veliki blok, vsebina se nato

kopira v prenosni medpomnilnik, medpomnilnik za velike bloke pa se vrne v skupno zalogo medpomnilnikov. Velikost krožnega medpomnilnika mora biti izbrana tako, da je ob dosegu praga dovolj prostora za cel naslednji diskovni blok. Prebrano z diska ne more iti neposredno v prenosni medpomnilnik, saj se lahko zgodi, da mora ta prej zakrožiti (angl. wrap around). Kopiranje in raba pomnilnika sta tu vzročno povezana tako, da je npr. na račun večje rabe pomnilnika potrebno manj kopiranja in obratno.

Dejavnik, ki ga je pri primerjanju obeh pristopov razmeščanja tudi potrebno upoštevati, je diskovna učinkovitost (angl. disk performance). Pri uporabi velikih blokov deluje disk s polno hitrostjo, kar je pogosto glavna skrb, saj prebiranje majhnih P- in B-sličic kot posamičnih enot v bloke ni učinkovito. Poleg tega je razdeljevanje (angl. striping) velikih blokov preko več diskovnih enot mogoče, kar je predstavljeno v nadaljevanju, medtem ko razdeljevanje posameznih sličic ni.

Organizacija z malimi bloki s slike 7.2(a) je včasih poimenovana tudi kot organizacija **časovno nespremenljive dolžine** (angl. constant time lenght), saj vsak kazalec v indeksu predstavlja enako število milisekund predvajalnega časa. Organizaciji s slike 7.2(b) pa včasih pravimo organizacija **podatkovno nespremenljive dolžine** (angl. constant data lenght), saj so podatkovni bloki enako veliki.

Naslednja razlika med obema datotečnima organizacijama je, da je možno, če so tipi sličic shranjeni v indeksu s slike 7.2(a), izvajati hitro previjanje naprej le s prikazovanjem I-sličic. Vendar pa lahko, glede na to, kako pogosto se I-sličice pojavljajo v toku, zaznavamo hitrost kot prehitro ali prepočasi. Pri organizaciji s slike 7.2(b) hitro previjanje naprej na ta način ni mogoče. Pravzaprav zahteva zaporedno branje datoteke za izbiro željenih sličic ogromen diskovni V/I.

Drug pristop je uporaba posebne datoteke, ki nam ob predvajanju pri normalni hitrosti daje občutek hitrega predvajanja pri 10-kratni hitrosti (v njej je shranjena le vsaka deseta sličica). Ta datoteka je lahko strukturirana enako kot druge, pri čemer se lahko uporablja tako indeks sličic, kakor tudi indeks blokov. Pri odpiranju datoteke mora biti sistem sposoben najti datoteko za hitro previjanje naprej, če je ta potrebna. Če uporabnik pritisne gumb za hitro previjanje naprej, mora sistem nemudoma najti in odpreti datoteko za hitro previjanje naprej in nato skočiti na pravo mesto v datoteki. Kar ve, je številka sličice, pri kateri se nahaja, potrebuje pa še sposobnost iskanja ustrezne sličice v datoteki za hitro previjanje naprej. Če je recimo pri sličici 4816 in ve, da ima datoteka za hitro previjanje naprej faktor hitrosti 10, potem mora v tej datoteki poiskati sličico 482 in začeti predvajati od tam naprej.

Če se uporabi indeks sličic, je iskanje določene sličice enostavno – to je kar indeks v indeksu sličic. Če se uporabi indeks blokov, je potrebna dodatna informacija v vsakem vnosu, da se določi, katera sličica je v katerem bloku, prav tako pa se mora izvesti binarno iskanje po indeksu blokov. Hitro previjanje nazaj deluje podobno kot hitro previjanje naprej.

7.3 Razmeščanje več datotek na enem disku

Do sedaj je bilo govora o razmestitvi enega filma. Na video strežniku bo seveda veliko filmov. Če so ti naključno razsejani po celem disku, se bo veliko časa potratilo za premikanje diskovne glave od filma do filma, ko se bo mnogo filmov hkrati predvajalo s strani različnih strank.

To stanje je mogoče izboljšati z zapažanjem, da so nekateri filmi bolj priljubljeni kot drugi, in upoštevanjem priljubljenosti med razmeščanjem filmov na disk. Čeprav je v splošnem težko kaj reči o priljubljenosti konkretnih filmov (razen tega, da je morda film priljubljenejši, če v njem nastopajo zvezde svetovnega slovesa), pa se v splošnem da nekaj povedati o relativni priljubljenosti.

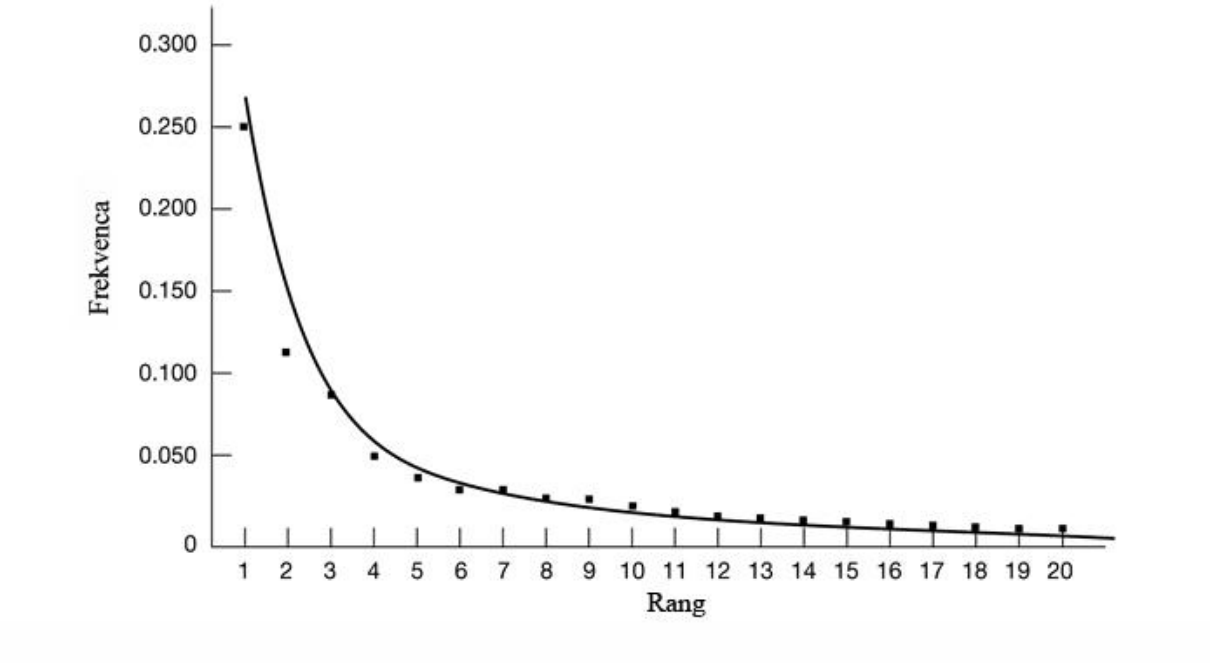
Za veliko načinov merjenj priljubljenosti, kot so izposojeni filmi, izposojene knjige iz knjižnic, sklici na spletne strani, število angleških besed, uporabljenih v romanu, ali populacija največjih mest, lahko vidimo, da iz sprejemljive aproksimacije relativne priljubljenosti sledi presenetljivo predvidljiv vzorec. Ta vzorec je odkril harvardski profesor jezikoslovja George Zipf (1902-1950) in ga imenujemo Zipfov zakon. Omenjeni zakon pravi, da če so filmi, knjige, spletne strani ali besede razvrščeni po svoji priljubljenosti, je verjetnost, da bo naslednja stranka izbrala k-ti primerek s seznama C/k, kjer je C normalizacijska konstanta.

Tako so deleži zadetkov za prve tri najpopularnejše filme $C/1$, $C/2$ in $C/3$, kjer se C izračuna tako, da je vsota vseh členov enaka 1. Z drugimi besedami, če imamo N filmov, potem

$$C/1 + C/2 + C/3 + C/4 + \cdots + C/\mathrm{N} = 1.$$

Iz te enačbe lahko enostavno izračunamo C. Vrednosti C za populacije z 10, 100, 1000 in 10000 primerki so 0˙134, 0˙067, 0˙045, 0˙034 in 0˙027.

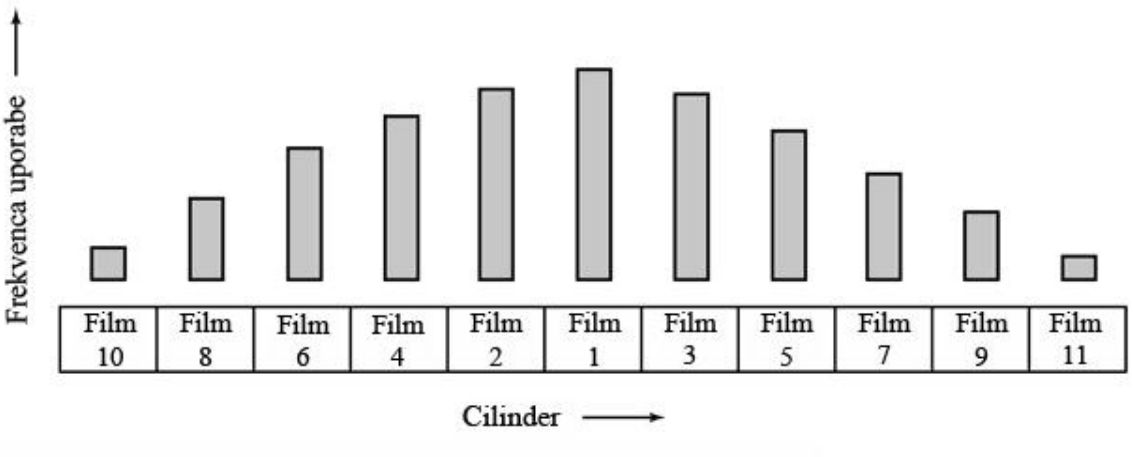
Zipfov zakon je ilustriran na grafu 7.1. Za primer je bil uporabljen na populaciji 20 največjih mest v ZDA. Zipfov zakon predvideva, da naj bi drugo največje mesto imelo število prebivalstva enako polovici števila prebivalstva največjega mesta, tretje največje mesto tretjini števila prebivalstva največjega mesta in tako dalje. Čeprav prileganje ni popolno, lahko rečemo, da je presenetljivo dobro.



Graf 7.1 Krivulja po Zipfovem zakonu za N = 20. Kvadratki predstavljajo populacijo 20 največjih mest v ZDA, rangiranih po številu prebivalcev (New York je 1, Los Angeles je 2, Chicago je 3 itd.). Vir: [3].

Za filme na video strežniku Zipfov zakon pravi, da je najpopularnejši film izbran dvakrat pogosteje kot drugi najpopularnejši, trikrat pogosteje kot tretji in tako dalje. Kljub dejstvu, da se porazdelitev na začetku dokaj hitro izjalovi, ima še dolg rep. Na primer, priljubljenost filma 50 je $C/50$ in filma 51 $C/51$, se pravi, da priljubljenost filma 51 znaša 50/51 priljubljenosti filma 50, kar je le 2% razlike. Bolj ko se pomikamo naprej po repu, manjša je razlika v odstotkih med zaporednimi filmi. Zaključimo lahko, da mora biti na strežniku shranjenih veliko filmov, saj je precejšnje povpraševanje tudi po filmih izven deseterice najpriljubljenejših.

Poznavanje relativne priljubljenosti za različne filme nam omogoča, da modeliramo učinkovitost video strežnika in da dobljeno informacijo uporabimo za razmeščanje datotek. Študije so pokazale, da je najboljša strategija presenetljivo enostavna, porazdelitev pa neodvisna. Strategijo razmeščanja imenujemo algoritem orgelskih piščali (angl. organ-pipe algorithm). Poteka tako, da se na sredino diska razmesti najprej najpriljubljenejši film, nato drugi in tretji film po priljubljenosti na vsako stran prvega, zatem sledita na vsaki strani četrti ter peti film in tako naprej, kar prikazuje slika 7.3. Takšna razmestitev deluje najbolje, če je vsak film strnjena datoteka tipa, ki je prikazan na sliki 7.1, uporabi pa se lahko do neke mere tudi, če je vsak film omejen na ozko območje cilindrov. Ime algoritma izhaja iz dejstva, da je histogram verjetnosti podoben malce nagnjenim orgelskim piščalim.



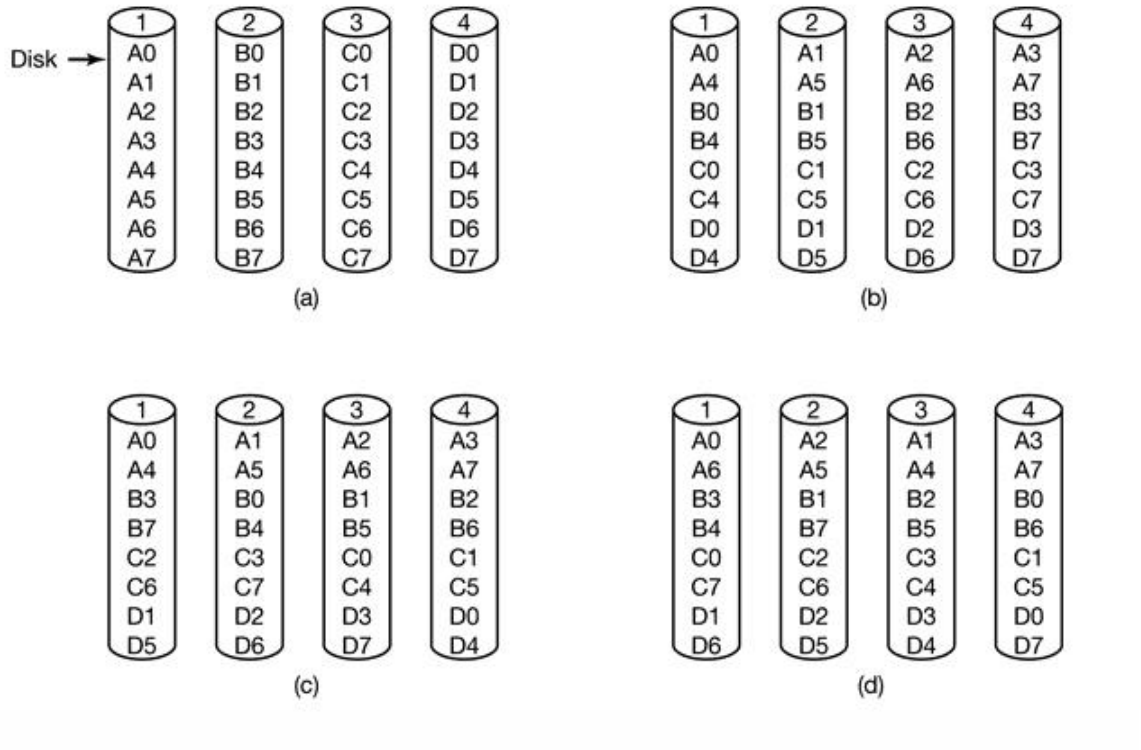
Slika 7.3 Datoteke na video strežniku, porazdeljene po porazdelitvi orgelskih piščali (angl. organ-pipe distribution). Vir: [3].

Algoritem deluje tako, da skuša zadržati diskovno glavo na sredini diska. Pri 1000 filmih in porazdelitvi po Zipfovem zakonu predstavlja prvih pet filmov skupno verjetnost 0˙307, kar pomeni, da bo diskovna glava ostala v cilindrih, ki so dodeljeni za prvih pet filmov, približno 30% časa, kar je pri 1000 razpoložljivih filmih presenetljivo veliko.

7.4 Razmeščanje datotek na več diskov z razdeljevanjem datotek

Za dosego večje učinkovitosti imajo video strežniki pogosto veliko diskov, ki lahko tečejo vzporedno. Včasih se uporabljajo diskovna polja (angl. redundant array of independent disks – RAID), vendar ne pogosto, saj ponujajo višjo zanesljivost na račun učinkovitosti. Video strežniki običajno zahtevajo visoko učinkovitost in se ne menijo dosti za popravljanje bežnih napak. Tudi RAID krmilniki so med drugim lahko ozko grlo, če morajo ravnati s preveč diski naenkrat.

Bolj pogosta konfiguracija je veliko število diskov, kar je včasih imenovano kot **diskovna farma** (angl. disk farm). Diski se ne rotirajo na sinhroniziran način in ne vsebujejo nobenih paritetnih bitov, tako kot RAID. Ena izmed možnih konfiguracij je, da damo film *A* na disk 1, film *B* na disk 2 in tako dalje, kot to prikazuje slika 7.4(a). V praksi se lahko pri modernih diskih razmesti na vsak disk po več filmov.



Slika 7.4 Štirje načini organiziranja multimedijskih datotek preko več diskov. (a) Brez razdeljevanja. (b) Enak razdelitveni vzorec za vse datoteke. (c) Cikcak razdeljevanje. (d) Naključno razdeljevanje. Vir: [3].

Takšna organizacija je nezahtevna za implementacijo, posledice okvar pa so jasne - če en disk odpove, postanejo vsi filmi na tem disku nedostopni. Potrebno je vedeti, da podjetje, ki izgubi disk poln filmov, ni niti približno v tako slabem položaju, kot podjetje, ki izgubi disk poln pomembnih podatkov, saj se filmi lahko enostavno ponovno naložijo na nadomestni disk z DVD-jev. Pomanjkljivost takšnega pristopa je, da utegne biti breme slabo uravnoteženo. Če imajo nekateri diski v lasti filme, po katerih je v danem trenutku veliko povpraševanja, ostali diski pa hranijo manj priljubljene filme, potem sistem ne bo popolnoma izkoriščen. Ko so enkrat pogostosti uporabe filmov znane, obstaja možnost, da nekaj filmov premestimo in s tem ročno uravnotežimo breme.

Druga možna organizacija je, da vsak film razdelimo preko več diskov (štirih v primeru na sliki 7.4(b)). Za trenutek predpostavimo, da so vse sličice enako velike (npr. nezgoščene). Fiksno število bajtov filma *A* je zapisano na disk 1, nato enako število bajtov na disk 2 in tako dalje, dokler ni dosežen zadnji disk (v tem primeru z enoto *A3*). Razdeljevanje (angl. striping) se nato nadaljuje ponovno na prvem disku z *A4* in tako naprej, dokler ni zapisana celotna datoteka. Filmi *B*, *C* in *D* so razdeljeni z uporabo enakega vzorca.

Možna slabost takšnega vzorca razdeljevanja je, da utegne biti breme med diski neuravnoteženo, saj se vsi filmi začnejo na prvem disku. En način za boljšo razporeditev bremena je cikcak razdeljevanje (angl. staggered striping), kot ga prikazuje slika 7.4(c). Še

ena možnost, da bolje uravnotežimo breme, pa je uporaba naključnega vzorca razdeljevanja za vsako datoteko, kakor je razvidno s slike 7.4(d).

Do sedaj smo predpostavljali, da so vse sličice enako velike. Pri MPEG-2 filmih je ta predpostavka napačna, saj so I-sličice dosti večje od P-sličic. Obstajata dva načina za rešitev zapleta, in sicer razdeljevanje blokov (angl. striping by blocks) ali razdeljevanje sličic (angl. striping by frames). Pri razdeljevanju sličic gre prva sličica filma *A* na disk 1 kot strnjena enota, neodvisno od tega, kako velika je. Naslednja sličica gre na disk 2 in tako dalje. Film *B* je razdeljen na podoben način, bodisi se začne na istem disku, na naslednjem disku (pri cikcak razdelitvi), ali pa na naključnem disku. Ker so sličice brane ena naenkrat, ta vzorec razdeljevanja ne pohitri branja nobenega danega filma. Vendar pa razporedi breme med diske precej bolje, kot v primeru na sliki 7.4(a), kjer lahko pride do neželenega obnašanja, če se na primer veliko ljudi odloči, da si bo ogledalo film *A* zvečer ob približno enakem času, nihče pa si ne želi ogledati filma *C*. V splošnem razporejanje bremena med vse diske omogoča boljšo izrabo pasovne širine celotnega diska in s tem povečuje število strank, ki se jih lahko postreže.

Drug način je razdeljevanje blokov. Za vsak film so enote fiksne velikosti zapisane na vsakega od diskov zaporedoma (ali naključno). Vsak blok vsebuje eno ali več sličic ali delce teh. Sistem lahko sedaj izda zahteve za več blokov naenkrat za isti film. Vsaka zahteva zaprosi za branje podatkov v drug medpomnilnik, toda na tak način, da ko so vse zahteve končane, se v pomnilniku sestavi strnjen kos filma (ki vsebuje mnogo sličic). Te zahteve lahko potekajo vzporedno. Ko je izpolnjena zadnja zahteva, se lahko procesu zahtevanja sporoči, da je delo opravljeno. Nato se lahko začne prenašanje podatkov k uporabniku. Mnogo sličic kasneje, ko je v medpomnilniku še zadnjih nekaj sličic, se več zahtev izda za to, da se prednaloži (angl. preload) drug medpomnilnik. Ta pristop uporablja velike količine pomnilnika za medpomnjenje z namenom, da bi bili diski zasedeni. Na sistemu s 1000 aktivnimi uporabniki in 1-MB medpomnilniki (na primer, da se uporabljajo 256-KB bloki na vsakem od štirih diskov), je potreben 1 GB RAM-a za medpomnilnike. Takšna količina je na strežniku s 1000 uporabniki malenkost in ne bi smela biti problem.

Še zadnje vprašanje, ki se tiče razdeljevanja, je, na koliko diskov razdeliti. Ena skrajnost je, da je vsak film razdeljen čez vse diske. Na primer pri 2-GB filmih in 1000 diskih bi lahko bil blok velikosti 2 MB zapisan na vsak disk tako, da noben film ne bi uporabljal istega diska dvakrat. Druga skrajnost pa je, da so diski razdeljeni v majhne skupine (kot na sliki 7.4), vsak film pa je omejen na posamezen razdelek (angl. partition). Prvi pristop, imenovan **široko razdeljevanje** (angl. wide striping), dobro opravi svoje delo pri uravnoteževanju bremena po diskih. Njegov glavni problem je v tem, da vsak posamezen film uporablja vse diske, in če en disk odpove, potem ne more biti prikazan noben film. Drugi pristop, imenovan **ozko razdeljevanje** (angl. narrow striping), lahko trpi na račun vročih točk (angl. hot spots) oz. priljubljenih razdelkov, vendar izguba enega diska uniči le tiste filme, ki so v njegovem razdelku.

7.5 Razmeščanje datotek na več diskov z repliciranjem datotek s shemo CLB

V prejšnjem poglavju smo ugotovili, da se pri ozkem razdeljevanju multimedijskih datotek pojavlja problem vročih točk, kjer velika večina zahtev pripada majhni podmnožici diskov, kjer so razmeščeni najpriljubljenejši filmi. Ker lahko vsak disk postreže le majhno število sočasnih tokov, lahko ostane veliko zahtev po priljubljenih filmih nezadovoljenih, medtem ko so diski z manj priljubljenimi filmi manj izkoriščeni, kot bi lahko bili. Če problem vročih točk rešujemo s širokim razdeljevanjem, pa naletimo na očiten problem zanesljivosti. Tega lahko ublažimo z uporabo RAID tehnologije.

Za video strežnike, ki zagotavljajo visoko stopnjo interaktivnosti, pa razdeljevanje povzroča še bolj zahrbten problem, ki se nanaša na zmogljivost – manjše enote (za razdeljevanje) povečujejo režijo relativnega iskanja (angl. relative seek overhead) in rotacijsko režijo (angl. rotational overhead) ter s tem zmanjšujejo efektivno propustnost vsakega diska v diskovnem polju. Zmanjšanje propustnosti je lahko precejšnje, ko so zakasnitve pri interakcijah uporabnikov s sistemom (vključno z zakasnitvami po začetnem zagonu filma, po pritisku gumba za premor/nadaljevanje in po začasnem skoku) omejene na manj kot sekundo (več o tem v [12]). Za visoko interaktivne video strežnike se zato, v primeru razdeljevanja, priporoča razdeljevanje ozke širine (npr. razdeljevanje čez dva diska oz. širine 2). Še bolj pa je priporočljiva uporaba razmeščanja datotek z repliciranjem, ki je predstavljeno v nadaljevanju. V literaturi (npr. v [9]) je predstavljenih več shem dodeljevanja datotek (angl. file assignment schemes), vendar smo se v diplomskem delu omejili le na eno najučinkovitejših, in sicer na kombinacijsko izravnavo bremen (angl. combination load balancing – CLB), v okviru katere imamo tri različne sheme izbire virov (angl. resource selection schemes).

7.5.1 Uvod h kombinacijski izravnavi bremen

Na račun pripadajočih omejitev pristopa z razdeljevanjem, morajo biti uporabniške zahteve za filmski naslov v interaktivnem VOD sistemu v večini primerov povezane z enim diskom, kjer je datotečna kopija zahtevanega filma razmeščena. Posledica je repliciranje priljubljenih filmskih naslovov na več diskov tako, da se poveča število sočasnih video tokov, ki so lahko sistemsko podprti, da zadovoljijo uporabniške zahteve po teh priljubljenih filmih. Ker so filmske povezave med uporabniki in diski dolgo vzpostavljene, so VOD sistemi pogosto modelirani kot izgubni sistemi (angl. loss systems). To pomeni, da je uporabniška zahteva, ki želi dostopati do filmskega naslova, blokirana, če je sistem ne more nemudoma postreči. Podobno kot pri mnogih drugih storitvenih sistemih se tudi pri načrtovanju in delovanju VOD sistema pojavlja pomembno vprašanje, kako enakomerno porazdeliti prometno breme filmov med diske v sistemu. Bremensko uravnotežen (angl. load balanced) VOD sistem vodi do učinkovitega delovanja in minimalne *verjetnosti blokiranja zahtev* (angl. request blocking probability - RBP) oz. verjetnosti zavrnitve zahteve.

Če bi lahko vsak filmski naslov imel datotečno kopijo shranjeno na vseh diskih v VOD sistemu, potem bi bilo prometno breme, ki ga uporabniki povzročajo z zahtevami po predvajanju filmov, brez težav uravnoteženo med diski (obremenitev diskov bi bila enakomerna). V takšnem idealnem scenariju *popolne replikacije* je uporabniška zahteva po kateremkoli filmskem naslovu blokirana le, če je kapaciteta video tokov celotnega sistema na osnovi prihajanja zahtev popolnoma izkoriščena. Zaradi omejitev diskovnega prostora za shranjevanje je zelo malo verjetno, da bi bilo to praktično izvedljivo v zelo obsežnem VOD sistemu, ki podpira veliko knjižnico filmskih vsebin. Realni VOD sistemi tipično podpirajo selektivno datotečno replikacijo. Za dani primerek datotečne replikacije osnovanje pogoja, da datotečna alokacija doseže izravnavo bremen in s tem minimalno verjetnost blokiranja zahtev (RBP), v tem tipu sistemov ni trivialna naloga, razen v določenih poenostavljenih situacijah, kjer so uporabniške zahteve po filmih z več kopijami obravnavane v skladu s shemo izbire virov (angl. resource selection scheme), ki jo imenujemo *enkratni naključni poskus* (angl. single random trial – SRT).

Za SRT je značilno, da se ob prihodu uporabniške zahteve po filmu z več kopijami naključno izbere eden izmed diskov, ki hranijo datotečno kopijo zahtevanega filmskega naslova. Če je disk v celoti zaseden, se zahteva enostavno blokira brez nadaljnjih poskusov pri drugih diskih, ki hranijo datotečno kopijo istega filmskega naslova. Little in Venkatesh sta pokazala, da je SRT sistem bremensko uravnotežen takrat, ko so filmske datoteke optimalno alocirane tako, da za vsak homogeni disk obstaja enaka verjetnost dostopa. Predpostavila sta, da je pri tem bremensko uravnoteženem stanju, ki ga imenujemo izravnava diskovnih bremen (angl. disk load balancing – DLB), verjetnost blokiranja zahtev (RBP) VOD sistema minimizirana. Če DLB ni mogoče doseči v praksi, je odličnost suboptimalne rešitve datotečne alokacije, ki je opredeljena kot njena oddaljenost do DLB v smislu RBP sistema, povezana s tem, kako enakomerno je prometno breme filmov porazdeljeno v primerjavi z enakomerno porazdelitvijo.

V primerjavi z drugimi shemami izbire virov je shema SRT sama po sebi neučinkovita v izkoriščanju sistemskih virov z danimi filmi z več kopijami. Učinkovitejša je uporaba *izčrpnih* (angl. exhaustive) shem izbire virov, pri katerih se v primeru, da je prva izbira blokirana, dostopa do diskov, ki hranijo alternativne datotečne kopije zahtevanega filmskega naslova. Dve takšni izčrpni shemi sta *ponovni naključni poskusi* (angl. repeated random trials – RRT) in *ujemanje najmanj zasedenega* (angl. least busy fit – LBF). V obeh naštetih shemah se uporabniška zahteva blokira le pod pogojem, da so vsi diski (v izčrpnem smislu), ki hranijo zahtevano filmsko datoteko, v celoti zasedeni. RRT je naravna razširitev sheme SRT, saj se s ponovnimi naključnimi poskusi nadaljuje, dokler niso vsi diski preizkušeni. Z uporabo razpoložljivih informacij o stanju sistema LBF vedno usmeri uporabniško zahtevo po filmu z več kopijami k najmanj zasedenemu disku (z največjim številom razpoložljivih logičnih kanalov), kjer je razmeščena datotečna kopija zahtevanega filmskega naslova.

Dokazano je bilo, da LBF v primerjavi s SRT in RRT zagotavlja izredno učinkovitost ne le za filme z več kopijami, ampak tudi za filme z eno samo kopijo. To je v skladu z ugotovitvami s področja vodovno komutiranih omrežij, da sheme usmerjanja glede na najmanjšo

obremenjenost (angl. least loaded routing schemes) zagotavljajo boljšo učinkovitost kot sheme naključnega izmeničnega usmerjanja (angl. random alternate routing schemes).

7.5.2 Model sistema

Naj bo sistem videa na zahtevo sestavljen iz množice $\mathcal{D}$, ki vsebuje J homogenih diskov, označenih z $1, 2, \ldots, J$. Vsak disk ima omejen prostor za shranjevanje velikosti C enot (ena enota diskovnega prostora meri npr. 1 GB). Smatramo, da so neodvisni video tokovi, ki izhajajo iz diska, statistično približno enakovredni. Vsak disk naj bi podpiral do N sočasnih video tokov (logičnih kanalov). V primerih, kjer sistem sestoji iz heterogenih diskov, predpostavljamo uporabo tehnik združevanja diskov na tak način, da je iz polja heterogenih diskov mogoče sestaviti logično zbirko J homogenih diskov.

Sistem ponuja veliko knjižnico filmskih vsebin, ki vsebuje M različnih filmskih naslovov, označenih z $1, 2, \ldots, M$. Množica teh M filmskih naslovov je označena s $\mathcal{F}$. Velikost filmske datoteke filma m je L_m enot. Torej potrebujemo $L = \sum_{m \in \mathcal{F}} L_m$ enot diskovnega prostora za alokacijo (angl. allocation) ene kopije vsakega filma iz $\mathcal{F}$. Predpostavljamo $max_{m \in \mathcal{F}} L_m \ll C$, tako da lahko vsak disk hrani številne filmske datoteke. Prav tako predpostavljamo $L < JC$, tako da ima sistem še razpoložljiv diskovni prostor za hrambo več kopij določenih filmov iz $\mathcal{F}$. Množica filmskih datotek, ki so nameščene na disku j, je označena z Φ_j.

Da bi lahko iz dodelitve filmske datoteke (angl. movie file assignment) izluščili informacijo, kako je vsak posamezen filmski naslov repliciran ter kje so filmska datoteka in njene kopije (če je replicirana) alocirani, smo definirali dva koncepta. Naj bo *primerek replikacije datotek* (angl. file replication instance) definiran z vektorjem $\mathbf{n} = (n_1, n_2, \ldots, n_M)$, kjer n_m, $m \in \mathcal{F}$, označuje celo število kopij filmske datoteke filma m in velja $1 \leq n_m \leq J$. Filmski naslov, ki ima c datotečnih kopij, imenujemo film tipa c. Naj bo *primerek alokacije datotek* (angl. file allocation instance) definiran kot razporeditev diskovnih lokacij $\mathbf{\Omega} = (\Omega_1, \Omega_2, \ldots, \Omega_M)$ za množico filmskih datotek, ki so navedene v primerku replikacije datotek $\mathbf{n}$, za katere velja omejitev prostora za shranjevanje na vsakem disku. Vsak izmed elementov Ω_m, $m \in \mathcal{F}$, opisuje množico n_m različnih diskov, kjer je shranjenih n_m datotečnih kopij filma m. Da se primerek replikacije datotek smatra za izvedljivega, zahtevamo, da se da vsaj en veljaven primerek alokacije datotek realizirati za primerek replikacije datotek glede na omejitev diskovnega prostora za shranjevanje.

V statističnem smislu je ustvarjanje zahteve za film v VOD sistemu podobno ustvarjanju klica v telefoniji, kjer je široko sprejeta Poissonova predpostavka. Ta predpostavka je tudi utemeljena, saj je bilo ugotovljeno, da so medprihodni časi uporabniških zahtev pri pretočnih multimedijskih sistemih (angl. streaming multimedia systems) eksponentno porazdeljeni. Privzemamo, da agregirani prihodi zahtev za vse filmske naslove sledijo Poissonovemu procesu z intenzivnostjo λ zahtev na časovno enoto. Procesi prihajanja zahtev za različne filmske naslove so medsebojno neodvisni Poissonovi procesi.

Čas povezave filma m (čas strežbe), če upoštevamo interaktivno obnašanje uporabnikov, sledi logaritemsko normalni porazdelitvi s srednjo vrednostjo $1/\mu_m$ časovnih enot (utemeljitev lahko najdemo v [6]). Brez izgube splošnosti predpostavljamo, da je vrednost povprečnega časa povezave $1/\mu_m$ za film m enaka vrednosti njegove datotečne velikosti L_m, standardni odklon časa povezave filma m pa je enakovreden njegovi povprečni vrednosti.

Intenzivnost zahtevanja za film m omogoča ustvariti njegov profil priljubljenosti p_m, ki je definiran kot relativna verjetnost, da stranka zahteva film m, pri čemer velja $\sum_{m\in\mathcal{F}} p_m = 1$. Za dani primerek replikacije datotek **n** je profil priljubljenosti $\hat{p}_c$ za filme tipa c dobljen z enačbo $\hat{p}_c = \sum_{m\in\mathcal{F},n_m=c} p_m$. Povprečni čas povezave $1/\hat{\mu}_m$ za film tipa c je tako podan z

$$\frac{1}{\hat{\mu}_c} = \frac{1}{\hat{p}_c} \sum_{m\in\mathcal{F},n_m=c} \frac{p_m}{\mu_m}. \tag{7.1}$$

V praksi se profili priljubljenosti filmov redno posodabljajo, da zajamejo spremenljivost povpraševanja uporabnikov. V časovnem intervalu med takimi posodobitvami je intenzivnost prihajanja zahtev (angl. request arrival rate) za film m dana z λp_m. Torej je prometno breme A_m filma m podano z $\lambda p_m/\mu_m$ (faktor obremenitve strežne enote s strani filma m). Skupno prometno breme (angl. aggregate traffic load) $\hat{A}_c$ za vse filme tipa c dobimo z $\sum_{m\in\mathcal{F},n_m=c} A_m$. Skupno prometno breme A za vse filme v $\mathcal{F}$ je izračunano z $\sum_{m\in\mathcal{F}} A_m$.

V našem modelu privzemamo, da so profili priljubljenosti filmskih naslovov v VOD sistemu porazdeljeni tako, da velja

$$p_m = \frac{m^{-\zeta}}{\sum_{k=1}^{M} k^{-\zeta}} \tag{7.2}$$

za $m \in \mathcal{F}$. Parameter ζ v (7.2) določa stopnjo nesimetričnosti (angl. skewness) porazdelitve. Ta porazdelitvena funkcija je splošno znana kot Zipfovi podobna porazdelitev (angl. Zipf-like distribution), ki pri vrednosti $\zeta = 1$ postane Zipfova porazdelitev. Opisana porazdelitev z $\zeta = 0{\cdot}271$ statistično ustreza frekvencam dostopov strank do različnih filmskih naslovov, kar je bilo ugotovljeno iz poslovanja video izposojevalnic (angl. video rental business).

7.5.3 Izravnava diskovnih bremen

Če je uporabniška zahteva po filmu m z več kopijami obravnavana v skladu s SRT, potem je naključno odposlana k le enemu disku iz množice Ω_m. Nobenega prizadevanja ni, da bi bila zahteva učinkoviteje obravnavana med n_m diski v Ω_m. Glede na to, da je proces prihajanja zahtev za vsakega izmed M filmskih naslovov v sistemu Poissonov, je proces prihajanja zahtev za film m enostavno razstavljen na n_m neodvisnih Poissonovih procesov, vsak izmed njih pa ima intenzivnost $\lambda p_m/n_m$ ob predpostavki enake verjetnosti. Vsaka datotečna kopija filma m tako generira prometno breme A_m/n_m za disk, kjer je shranjena.

Pod pogojem, da so vsi diski glede na prostor za shranjevanje in tokovno kapaciteto (angl. stream capacity) homogeni, sta Little in Venkatesh predpostavila, da je verjetnost blokiranja zahteve (RBP) za SRT sistem minimalna, če in samo če so lahko filmske datoteke optimalno alocirane tako, da je prometno breme na vsakem izmed homogenih diskov v SRT sistemu enako. Pri takem bremensko uravnoteženem stanju je prometno breme na vsakem disku enako A/J, tako da je lahko RBP SRT sistema, ki doseže DLB, natanko podan z Erlangovo B formulo

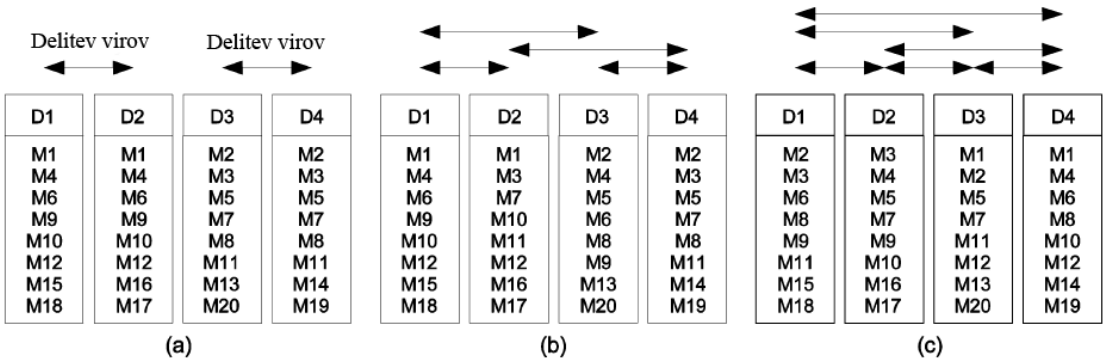
$$RBP \stackrel{\text{def}}{=} E\left(\frac{A}{J}, N\right) = \frac{\left(\frac{A}{J}\right)^N / N!}{\sum_{i=0}^{N} \left(\frac{A}{J}\right)^i / i!}. \tag{7.3}$$

Naslednji računski primer prikazuje, da pogoj o datotečni alokaciji za dosego izravnave bremen v SRT sistemu ne odraža pravega pomena izravnave bremen v LBF sistemu. Za lažjo predstavo je podan primer za majhen sistem s štirimi diski in z 20 različnimi filmskimi naslovi. Vsak disk v tem primeru ima prostor za shranjevanje velikosti osem enot in podpira do deset sočasnih video tokov. Vsak filmski naslov ima datoteko velikosti ene enote. Posledično je povprečni čas povezave kateregakoli filmskega naslova ena časovna enota. Profili priljubljenosti za teh 20 filmskih naslovov so v naraščajočem vrstnem redu podani v tabeli 7.1. Za ta konkreten primer je bila predpostavljena skupna intenzivnost $\lambda = 24$ zahtev na časovno enoto.

Tabela 7.1 Porazdelitev priljubljenosti filmov za primer s štirimi diski. Vir: [11].

Filmi	1	2	3	4	5
Priljubljenost	0.08655	0.07173	0.06427	0.05945	0.05596
Filmi	6	7	8	9	10
Priljubljenost	0.05326	0.05108	0.04927	0.04772	0.04638
Filmi	11	12	13	14	15
Priljubljenost	0.04519	0.04414	0.04319	0.04233	0.04155
Filmi	16	17	18	19	20
Priljubljenost	0.04083	0.04016	0.03954	0.03897	0.03843

Določen je bil specifičen primerek datotečne replikacije, kjer imajo filmi od 1 do 12 po dve datotečni kopiji, filmi od 13 do 20 pa po eno kopijo. Iz veliko možnih ureditev diskovnih lokacij za teh 32 filmskih datotek so bili izbrani trije veljavni primerki datotečne alokacije, kot prikazuje slika 7.5.



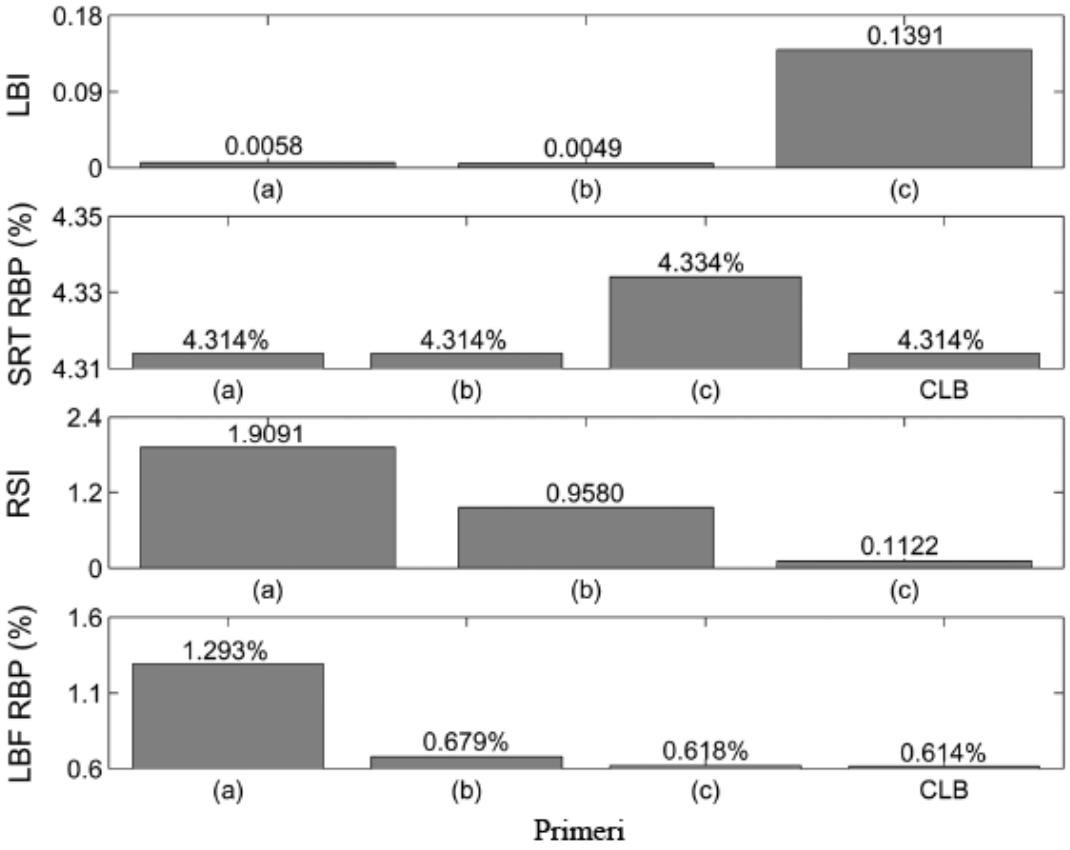
Slika 7.5 Primerki alokacije filmskih datotek za primer s štirimi diski: (a) Le dva para diskov si delita promet filmov z več kopijami; (b) Le štirje pari diskov si delijo promet filmov z več kopijami; (c) Vsak par diskov si deli promet filmov z več kopijami. Vir: [11].

Kot je bilo že povedano, je v SRT sistemu primerek datotečne alokacije optimalen, če je možno skupno prometno breme filmov A enakomerno porazdeliti med J diskov v sistemu. Po (7.3) je minimalni RBP za ta primer, ki deluje po SRT, 4˙314%. Za razlikovanje in primerjavo stopnje izravnave bremen med temi tremi primerki datotečne alokacije v SRT sistemu služi *indeks izravnave bremen* (angl. load balancing index – LBI), ki je podan z

$$LBI = \sqrt{\frac{1}{J}\sum_{j\epsilon\mathcal{D}}\left(\sum_{m\epsilon\,\Phi_j}\frac{A_m}{n_m} - \frac{A}{J}\right)^2}\,. \tag{7.4}$$

Za primerek datotečne alokacije LBI meri, kako enakomerno je prometno breme filmov razširjeno med J diski v SRT sistemu. To je ekvivalentno standardni deviaciji. Manjša vrednost LBI kaže na bolj uravnoteženo prometno porazdeljenost v SRT sistemu. Po izračunih po (7.4) za vsakega izmed treh primerkov datotečne alokacije, upoštevanih na sliki 7.5, rezultati LBI na sliki 7.6 (ti so bili pridobljeni s simulacijami) kažejo, da sta tako (a) kot (b) blizu enakomerni porazdelitvi in da (c) ni tako bremensko izravnan kot sta (a) in (b).

Za namen preverjanja kakovosti vsakega primerka datotečne alokacije v SRT sistemu v primerjavi s tem, kar prikazuje LBI, ter za ugotavljanje, če se uspešnost datotečne alokacije, vpeljane v SRT sistemu, odraža tudi v LBF sistemu, je bila narejena simulacijska študija diskretnih dogodkov posamično za SRT in LBF sistem. V tipičnem teku simulacijskega testa je vsak izmed sto milijonov naključnih poizkusov predstavljal bodisi prihod uporabniške zahteve bodisi prekinitev filmske povezave. RBP je bil pridobljen s štetjem celotnega števila prihodov zahtev in celotnega števila izgubljenih zahtev. Za zagotovitev zaupanja v simulacijske ocene, je bil simulacijski test ponovljen z mnogimi neodvisnimi teki.



Slika 7.6 Rezultati LBI, SRT RBP, RSI in LBF RBP za primer s štirimi diski. Vir: [11].

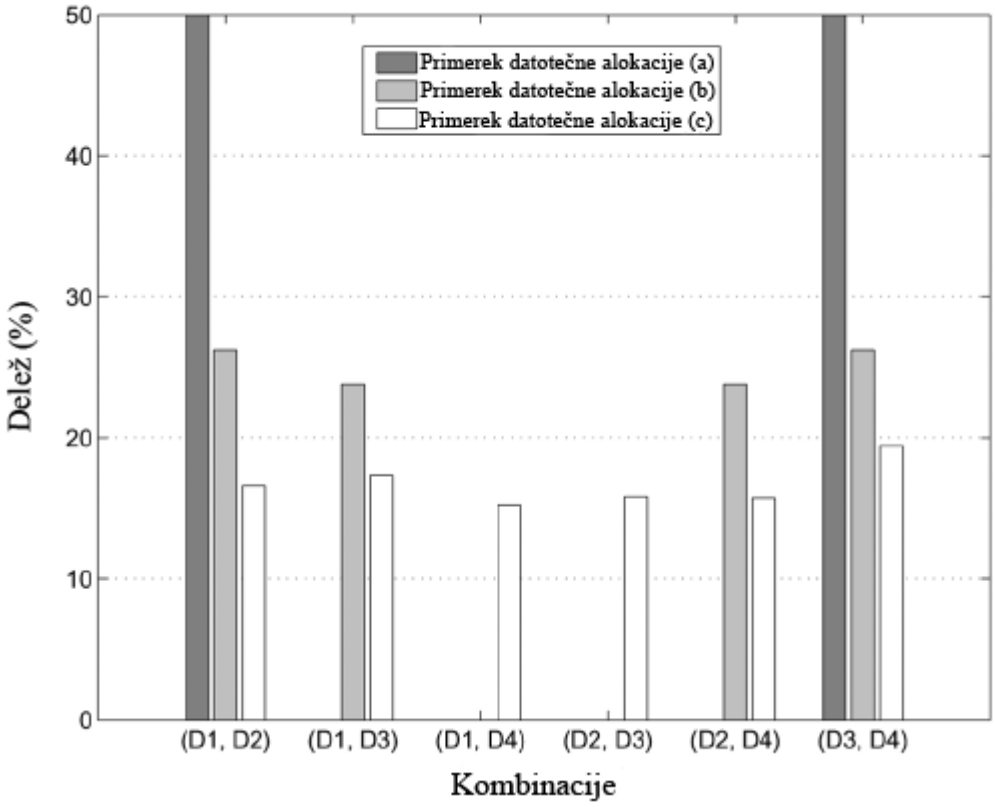
Iz rezultatov SRT RBP na sliki 7.6 vidimo, da se RBP SRT sistema ujema z domnevo Littla in Venkatesha. Tako (a) kot (b) dosežeta DLB v SRT sistemu in dajeta minimalni RBP, kot je bil izračunan z (7.3). Vendar pa prinašata bistveno drugačne rezultate za RBP v LBF sistemu. Kot lahko opazimo iz rezultatov LBF RBP s slike 7.6, uporabniška zahteva izkusi veliko manjši RBP v (b) in (a), ko je v delovanju LBF. Med drugim lahko ugotovimo, da je LBF RBP še manjši v (c), ne glede na njegov slab LBI rezultat v SRT sistemu. Jasno je, da je LBI neprimeren za razlago resnične vrednosti datotečne alokacije v LBF sistemu.

7.5.4 Kombinacijska izravnava bremen

Podrobnejši pregled slike 7.5 razkrije, da so v vseh treh primerkih datotečne alokacije diskovne lokacije filmov z eno kopijo enake in da je prometno breme filmov z eno kopijo skoraj enakomerno porazdeljeno. Vendar pa so diskovne lokacije filmov tipa 2 zelo različne. V (a) so filmi tipa 2 alocirani tako, da si le dva para diskov delita skupne filme tipa 2. Tako si disk 1 deli promet filmov tipa 2 le z diskom 2, disk 3 pa si deli promet filmov tipa 2 le z diskom 4. V (b) so filmi tipa 2 prepleteni na tak način, da obstajajo štirje različni pari diskov, ki si delijo promet filmov tipa 2 znotraj vsakega para. V (c) je stopnja prepleta še večja. Vsak

izmed šestih parov ima skupne filme tipa 2, tako da si vsak disk v sistemu deli promet filmov tipa 2 z vsakim izmed preostalih treh diskov.

Ker mora biti v tem primeru katerikoli par diskov v eni izmed $\binom{4}{2}$ kombinacijskih skupin z dvema diskoma, naštetima v množici $\mathcal{D}$, lahko natačno izračunamo delež prometa filmov tipa 2, ki dosega vsako izmed šestih kombinacijskih skupin z dvema diskoma. Na grafu 7.2 vidimo, da je promet filmov tipa 2 med šestimi kombinacijskimi skupinami bolj izravnan v (b) kot v (a), tisti v (c) pa je še bolj izravnan.



Graf 7.2 Delež prometa filmov tipa 2 na vsaki kombinacijski skupini dveh diskov za primer s štirimi diski. Vir: [11].

Iz teh treh primerkov datotečne alokacije opazimo, da je v LBF sistemu količina delitve diskovnih virov za promet filmov z več kopijami tista, ki ima največji vpliv na RBP sistema. Večje je število parov diskov, ki imajo skupne filme tipa 2 in bolj ko je izravnana delitev diskovnih virov za promet filmov tipa 2, manjši je RBP sistema. Podobno bi za sistem, ki vsebuje filme tipa c, $c \geq 2$, pričakovali, da je potrebno maksimizirati stopnjo delitve diskovnih virov pri strežbi prometa filmov tipa c, če bi lahko imeli enakomerno delitev virov za prometno breme filmov tipa c znotraj vsake možne kombinacijske skupine c diskov. Tako pridemo do naslednje domneve o tem, kako naj bo prometno breme filmov (tako za filme z več kopijami kot za filme z eno kopijo) idealno porazdeljeno za dosego bremenske izravnave in za minimizacijo RBP LBF sistema.

Domneva: LBF sistem je bremensko izravnan, če je za vsak c, $c \geq 1$, promet, ki želi dostopati do filmov tipa c, enakomerno porazdeljen med vsemi $\binom{J}{c}$ skupinami c diskov, izbranih iz množice $\mathcal{D}$, v kateri se nahaja vseh J diskov sistema. Takšno stanje imenujemo *kombinacijsko bremensko izravnano* (angl. combination load balanced) in domnevamo, da je RBP LBF sistema v tem stanju minimiziran.

Ta domneva napeljuje na to, da za LBF sistem s specifičnim primerkom datotečne replikacije, primerek datotečne alokacije, ki idealno doseže CLB, vedno daje spodnjo mejo za RBP. Kot poseben primer se ta domneva pridružuje tudi domnevi za SRT sistem. Pod CLB je, za vsak c, $c \geq 1$, prometno breme filmov tipa c enakomerno porazdeljeno med $\binom{J}{c}$ kombinacijskimi skupinami c diskov, ki so našteti v množici $\mathcal{D}$. Če je katerikoli disk iz $\mathcal{D}$ v $\binom{J-1}{c-1}$ kombinacijskih skupinah, potem je promet filmov tipa c, ki želi dostopati do kateregakoli diska, ob predpostavki SRT, natanko podan z

$$\frac{\binom{J-1}{c-1}\hat{A}_c}{c\binom{J}{c}} = \frac{\hat{A}_c}{J},$$

prometno breme na vsakem disku pa je, na račun vseh tipov filmov, natanko podano z

$$\sum_c \frac{\hat{A}_c}{J} = \frac{A}{J}.$$

To dokazuje, da je CLB-SRT (SRT sistem, ki doseže CLB) ena izmed implementacij DLB.

7.5.4.1 Simulacijska študija CLB

Za oceno natančnega rezultata CLB-LBF, kakor tudi za preveritev, da je CLB-SRT ena izmed realizacij DLB, je bila opravljena simulacijska študija posamično za CLB-SRT in CLB-LBF. V ta namen je bila prilagojena simulacija diskretnih dogodkov (ki smo jo že opisali v poglavju 7.5.3) na način, ki sledi. Če je bil med vsakim tekom simulacije naključni dogodek uporabniška zahteva, smo gledali le tip filma, ki ga je zahtevala stranka. Če je šlo za film tipa c, $c \geq 1$, smo nato ugotovili, na kateri kombinacijski skupini c diskov je shranjenih c datotečnih kopij filma. Če je pod CLB promet filmov tipa c bremensko izravnan med $\binom{J}{c}$ kombinacijskimi skupinami c diskov, potem za vsako kombinacijsko skupino obstaja enaka verjetnost, da se bo do nje naredil dostop. Ko je enkrat takšna kombinacijska skupina generirana, za ravnanje z uporabniško zahtevo nadaljujemo s SRT ali LBF shemo. Nadaljnji postopki v procesiranju vsake uporabniške zahteve in pridobivanje rezultata RBP sistema enostavno sledijo temu, kar je bilo opisano v poglavju 7.5.3.

7.5.4.2 Utemeljitev študije

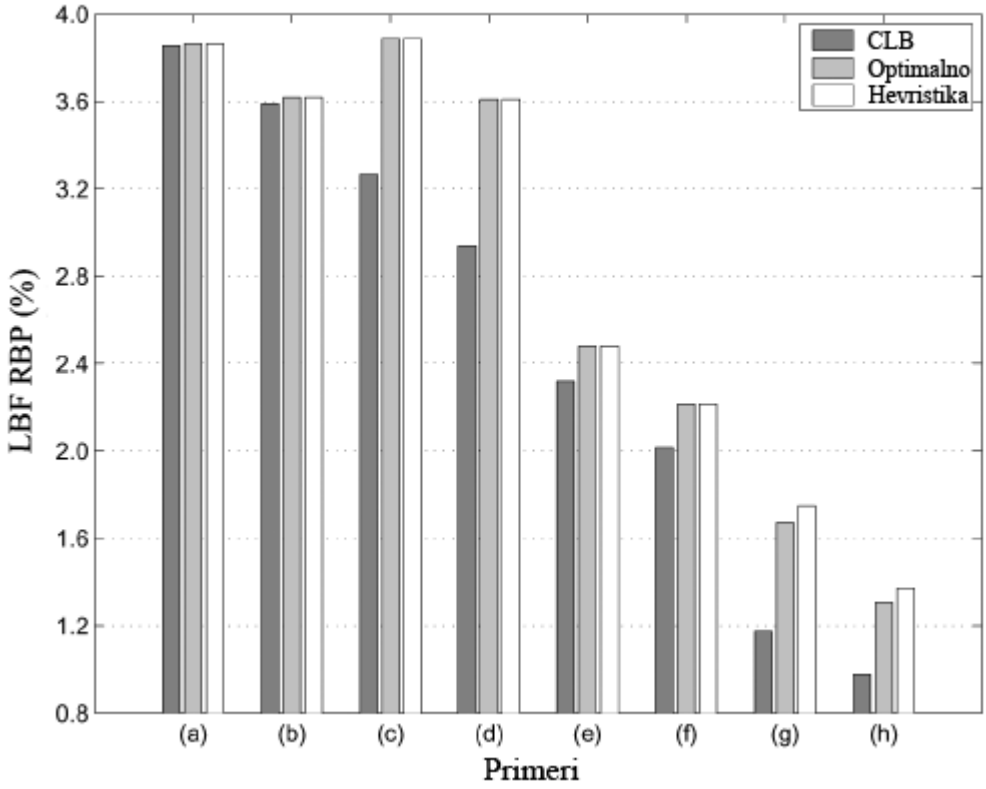
Za primer malega sistema, obravnavanega v poglavju 7.5.3, je bila opravljena simulacijska študija CLB. Rezultati RBP so za CLB-SRT in CLB-LBF podani na sliki 7.6. Ti potrjujejo, da CLB-SRT doseže natanko enak rezultat RBP, kot je bilo izračunano s (7.3). V primerjavi s suboptimalnimi rešitvami (a), (b) in (c), obravnavanimi na sliki 7.5, CLB-LBF jasno daje minimalni RBP LBF sistema. Poleg tega se v tem konkretnem primeru rezultat RBP LBF sistema na račun primerka datotečne alokacije (c) skoraj ne razlikuje glede na rezultat CLB.

Opozoriti velja, da CLB ni mogoče doseči v situacijah, kjer bodisi prometno breme filmov z več kopijami ne more biti enakomerno razdeljeno v vsako od povezanih kombinacijskih skupin diskov bodisi porazdeljenost prometa filmov z eno kopijo močno odstopa od enakomerne porazdelitve. V takšnih situacijah se za CLB smatra, da daje spodnjo mejo za RBP LBF sistema.

Čeprav strog matematični dokaz za dano trditev ni podan, je le-ta preverjena skozi veliko število simulacijskih eksperimentov za številne VOD sisteme različnih obsegov, kjer se ni našlo nobenega protiprimera. Z namenom utemeljitve je v tabeli 7.2 podanih osem primerov majhnih sistemov. Za vsakega od primerov in za dani testni primerek datotečne replikacije so izčrpno našteti vsi veljavni primerki, ki zadostijo omejitvi diskovnega prostora za shranjevanje. RBP rezultati optimalne rešitve za LBF sistem in za CLB so prikazani na grafu 7.3. Ti rezultati jasno prikazujejo vlogo CLB za zagotavljanje efektivne spodnje meje za RBP LBF sistema.

Tabela 7.2 Eksperimentalni primeri za utemeljitev CLB. Vir: [11].

Primeri	M	J	$\mathbf{n}$	λ	ζ
(a)	3	2	(2, 1, 1)	5	0.271
(b)	3	2	(2, 1, 1)	5	0.500
(c)	4	3	(2, 1, 1, 1)	7	0.271
(d)	4	3	(2, 1, 1, 1)	7	0.500
(e)	5	3	(2, 2, 1, 1, 1)	7	0.271
(f)	5	3	(2, 2, 1, 1, 1)	7	0.500
(g)	6	4	(3, 3, 2, 2, 1, 1)	10	0.271
(h)	6	4	(3, 3, 2, 2, 1, 1)	10	0.500



Graf 7.3 Eksperimentalni rezultati za utemeljitev CLB. Hevristični rezultati so pridobljeni s požrešno metodo za alokacijo datotek, ki je predstavljena v poglavju 7.5.5. Vir: [11].

7.5.4.3 Indeks delitve virov

Kot je že bilo pojasnjeno, ima faktor delitve diskovnih virov za promet filmov z več kopijami velik vpliv na RBP LBF sistema. V nadaljevanju je predstavljena učinkovita mera, ki omogoča za dani primerek datotečne alokacije numerično oceniti, kako enakomerno je prometno breme filma z več kopijami deljeno med diski v primerjavi z idealnim primerkom datotečne alokacije, ki doseže CLB.

Za primerek datotečne alokacije, ki vsebuje filme z več kopijami, je promet, ki želi dostopati do filma m z več kopijami, enakomerno porazdeljen med skupino n_m diskov v množici Ω_m. Tako je količina prometa, ki prihaja k disku $i \in \Omega_m$ in je generirana s strani filma m, enaka A_m/n_m. Poleg tega je količina prometa, ki prihaja k disku i in je generirana s strani filmov, ki se nahajajo tudi na disku j, enaka $\sum_{m \in \Phi_i \cap \Phi_j} A_m/n_m$.

Po drugi strani pa, če primerek datotečne alokacije idealno doseže CLB, potem je za vsak c, $c > 1$, skupno prometno breme $\hat{A}_c$ za filme tipa c enakomerno porazdeljeno med vsemi $\binom{J}{c}$ kombinacijskimi skupinami c diskov, ki so našteti v množici $\mathcal{D}$. Če diska i in j soobstajata v

$\binom{J-2}{c-2}$ kombinacijskih skupinah c diskov, potem disk i prejme promet filmov tipa c, ki je podan z

$$\frac{\binom{J-2}{c-2}\hat{A}_c}{c\binom{J}{c}} = \frac{(c-1)\hat{A}_c}{J(J-1)}$$

s strani filmov, ki so shranjeni tudi na disku j.

Na enak način, kot je definiran LBI v poglavju 7.5.3, je definiran *indeks delitve virov* (angl. resource sharing index – RSI), ki je podan z

$$RSI = \sqrt{\frac{2}{J(J-1)} \sum_{\substack{i,j \epsilon \mathcal{D} \\ i<j}} \left(\sum_{m \epsilon \Phi_i \cap \Phi_j} \frac{A_m}{n_m} - \sum_{c>1} \frac{(c-1)\hat{A}_c}{J(J-1)} \right)^2} \qquad (7.5)$$

kot mero, kako enakomerno primerek datotečne alokacije porazdeli prometno breme filma z več kopijami. Kot pri LBI, manjša vrednost RSI kaže na boljšo alokacijo filmskih datotek z več kopijami v LBF sistemu.

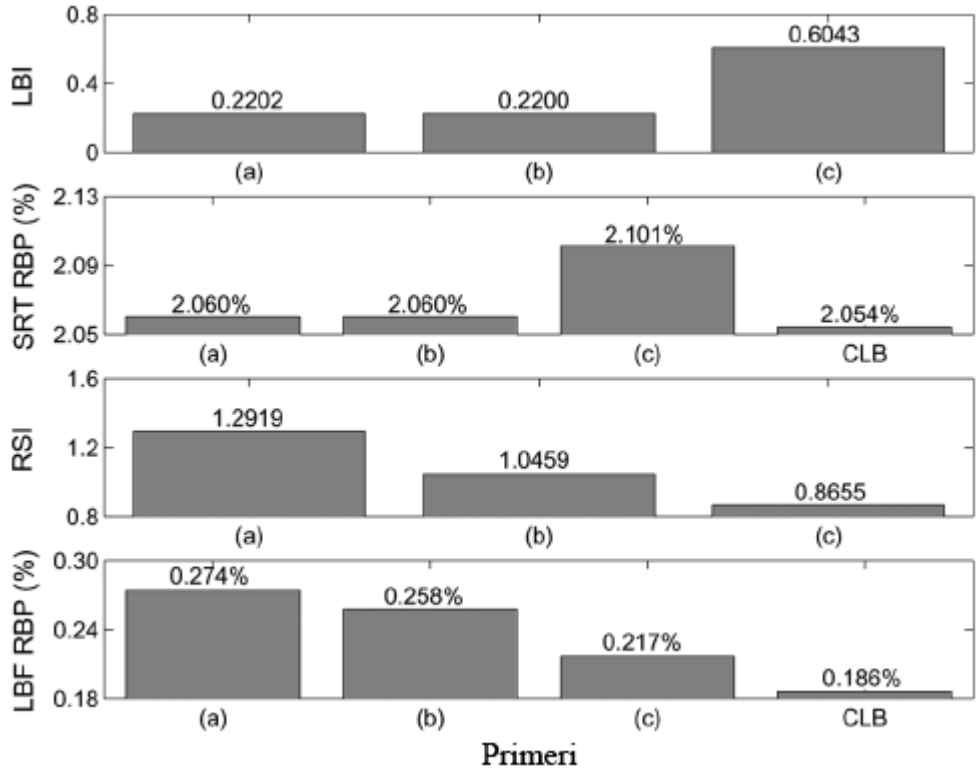
Z uporabo (7.5) na treh primerkih datotečne alokacije, ki smo jih obravnavali na sliki 7.5, smo dobili vrednosti RSI, kot jih prikazuje slika 7.6. Ti rezultati RSI zagotavljajo resnično vrednost alokacije filmov z več kopijami za tri primerke datotečne alokacije, ki smo jih obravnavali v poglavju 7.5.3.

7.5.4.4 Numerični rezultati za velik sistem

Velikost zelo obsežnega VOD sistema, ki omogoča dostop na zahtevo do več sto različnih filmskih naslovov, je ponavadi reda nekaj deset diskov. Poleg tega tipično vsebuje različne tipe filmov z več kopijami na račun kakovosti storitve in potrebe po zanesljivosti ter za učinkovito izkoriščanje neuporabljenega diskovnega prostora za shranjevanje. Na koncu podpoglavja je utemeljena domneva CLB, kakor tudi mera RSI, z upoštevanjem primera velikega sistema z 20 diski in 200 različnimi filmskimi naslovi. Vsak disk v tem primeru ima prostor za shranjevanje velikosti 14 enot in podpira do 30 sočasnih video tokov. Vsak filmski naslov ima datoteko velikosti ene enote. Povprečni čas povezave kateregakoli filmskega naslova je zato ena časovna enota. Profili priljubljenosti za 200 filmskih naslovov so izračunani po (7.2) z $\zeta = 0{\cdot}271$. Po (7.3) je minimalni RBP SRT sistema v tem primeru $2{\cdot}054\%$.

Za namen tega primera je bil upoštevan primerek datotečne replikacije, kjer so za vsakega izmed prvih treh filmskih naslovov alocirane štiri datotečne kopije, tri datotečne kopije za filme od 4 do 25, dve datotečni kopiji za filme od 26 do 50 in ena datotečna kopija za preostalih 150 filmskih naslovov. Med mnogimi veljavnimi primerki alokacije tega primerka

replikacije so bili izbrani trije. Njihove LBI in RSI vrednosti prikazuje slika 7.7. Rezultati RBP SRT in LBF sistema so pridobljeni s simulacijsko študijo in so prikazani na sliki 7.7 za primerjavo z LBI in RSI.



Slika 7.7 Rezultati LBI, SRT RBP, RSI in LBF RBP za primer z 20 diski. Vir: [11].

Zopet lahko v vseh primerih opazimo, da je minimalni RBP sistema dosežen s CLB. Poleg tega RSI pravilno napove kvaliteto alokacije filmskih datotek z več kopijami in tako nazorno prikaže vpliv delitve diskovnih virov za promet filmov z več kopijami na RBP LBF sistema.

7.5.5 Hevristična alokacija datotek

Za dani primerek replikacije datotek je problem iskanja primerka alokacije datotek, ki doseže kombinacijsko izravnavo bremen CLB (če je dosegljiva), NP-težek. Koncept deljenja diskovnih virov daje motivacijo za osnovanje požrešne metode za alokacijo datotek, katere namen je enakomerna delitev virov za promet filmov z več kopijami (glej sliko 7.8).

Naj O_j šteje nakopičene enote (angl. cumulative units), ki zasedajo prostor za shranjevanje na disku j, T_j naj beleži nakopičeno prometno breme (angl. cumulative traffic load) na disku j, S_{ij} pa naj beleži nakopičeno prometno breme, ki prihaja k disku i s strani filmov, ki se nahajajo tudi na disku j. Ko je datotečna kopija filma m razmeščena na disk j, povečamo O_j

za L_m enot in T_j za A_m/n_m. Če ima film m v sistemu alociranih več datotečnih kopij, za vse $i, j \in \Omega_m$, $i \neq j$, potem povečamo tako S_{ji} kot S_{ij} za A_m/n_m.

Proceduro za alokacijo datotek določimo s sortiranjem filmskih datotek z več kopijami v nenaraščajočem vrstnem redu glede na A_m/n_m, za vse $m \in \mathcal{F}$ in $n_m > 1$. Podobno uredimo filmske datoteke z eno kopijo v nenaraščajočem vrstnem redu glede na A_m, za vse $m \in \mathcal{F}$ in $n_m = 1$. Najprej razmestimo filmske datoteke z več kopijami zaradi stroge zahteve, da moramo vedno najti n_m različnih diskov, ki imajo dovolj prostora za shranjevanje, da lahko razmestimo n_m datotečnih kopij filma m. To bi bilo manj verjetno izvedljivo, če bi vnaprej alocirali filmske datoteke z eno kopijo.

Glavni koraki požrešne metode za alokacijo datotek se nadaljujejo na sledeč način: (1) Za alokacijo prve datotečne kopije filma z več kopijami m izberemo disk j tako, da je T_j najmanjši, pod pogojem $O_j + L_m \leq C$. Nato za vsako preostalo datotečno kopijo filma m izberemo disk i, $i \neq j$, tako da je S_{ij} najmanjši, pod pogojem, da ni na disku i shranjena že kakšna datotečna kopija filma m in da velja $O_i + L_m \leq C$; (2) Za alokacijo filma m z eno kopijo uporabimo običajno metodo »najmanj neobremenjen najprej«. Zopet izberemo disk j, tako da je T_j najmanjši, pod pogojem $O_j + L_m \leq C$.

Te korake ponavljamo, dokler niso vse filmske datoteke, ki so navedene v primerku replikacije datotek, uspešno alocirane, ali če se na katerikoli stopnji izkaže, da noben disk v $\mathcal{D}$ nima zadostnega prostora za shranjevanje za razmestitev filmske datoteke. V tem primeru se primerek replikacije datotek tretira kot neizvedljiv na račun nezmožnosti požrešne metode, da najde veljavni primerek hevristične alokacije datotek.

Za alokacijo prve datotečne kopije vsakega filmskega naslova v $\mathcal{F}$ moramo torej izvesti iskanje med J diski v $\mathcal{D}$ za disk j, ki ima dovolj prostora za shranjevanje in najmanjšo možno vrednost T_j. Za alokacijo vsake preostale datotečne kopije filma m z več kopijami moramo zopet izvesti iskanje med največ $J-1$ diski v $\mathcal{D}$ za disk i, ki ima dovolj prostora za shranjevanje in najmanjšo možno vrednost S_{ij}, z ozirom na to, da je prva datotečna kopija filma m razmeščena na disk j. Z upoštevanjem, da je število datotečnih kopij za vsak filmski naslov v takšnem sistemu največ J, je kompleksnost požrešne metode $O(MJ^2)$.

Psevdokoda procedure, ki implementira požrešno metodo za hevristično alokacijo datotek je prikazana na sliki 7.8. Za potrebe simulacijskega modela kombinacijske izravnave bremen v Simprocessu smo požrešno metodo realizirali v programskem jeziku Java. Pridobljeni primerki alokacije datotek služijo kot vhodni podatki simulacijskega modela, ki je obravnavan v poglavju 8.

Procedura: požrešna alokacija datotek

nastavi $O_j = 0$ za vse $j \in \mathcal{D}$;
nastavi $T_j = 0$ za vse $j \in \mathcal{D}$;
nastavi $S_{ij} = 0$ za vse $i, j \in \mathcal{D}, i \neq j$;
nastavi $\Omega_m = \emptyset$ za vse $m \in \mathcal{F}$;
za vsak $m \in \mathcal{F}$, $n_m > 1$, v nenaraščajočem vrstnem redu glede na $\frac{A_m}{n_m}$ ponavljaj
    nastavi $\mathcal{D}^* = \mathcal{D}$;
    če $O_j + L_m > C$ za vse $j \in \mathcal{D}^*$
        vrni NEIZVEDLJIVO;
    najdi $j \in \mathcal{D}^*$ z najmanjšim T_j in $O_j + L_m \leq C$;
    povečaj O_j za L_m;
    povečaj T_j za $\frac{A_m}{n_m}$;
    nastavi $\mathcal{D}^* = \mathcal{D}^* - \{j\}$;
    nastavi $\Omega_m = \Omega_m + \{j\}$;
    nastavi $k = 2$;
    dokler $k \leq n_m$ ponavljaj
        če $O_i + L_m > C$ za vse $i \in \mathcal{D}^*$
            vrni NEIZVEDLJIVO;
        najdi $i \in \mathcal{D}^*$ z najmanjšim S_{ij} in $O_i + L_m \leq C$;
        povečaj O_i za L_m;
        povečaj T_i za $\frac{A_m}{n_m}$;
        povečaj $S_{ij'}$ in $S_{j'i}$ za $\frac{A_m}{n_m}$ za vse $j' \in \Omega_m$;
        nastavi $\mathcal{D}^* = \mathcal{D}^* - \{i\}$;
        nastavi $\Omega_m = \Omega_m + \{i\}$;
        povečaj k za 1;
    konec
konec
za vsak $m \in \mathcal{F}$, $n_m = 1$, v nenaraščajočem vrstnem redu glede na A_m ponavljaj
    če $O_j + L_m > C$ za vse $j \in \mathcal{D}$
        vrni NEIZVEDLJIVO;
    najdi $j \in \mathcal{D}$ z najmanjšim T_j in $O_j + L_m \leq C$;
    povečaj O_j za L_m;
    povečaj T_j za A_m;
    nastavi $\Omega_m = \Omega_m + \{j\}$;
konec
vrni IZVEDLJIVO;

Slika 7.8 Psevdokoda procedure požrešne metode za hevristično alokacijo datotek. Vir: [11].

8 Simulacija kombinacijske izravnave bremen v Simprocessu

Eden izmed poglavitnih ciljev diplomskega dela je bil sestaviti simulacijski program bremensko dobro uravnoteženega diskovnega podsistema videa na zahtevo. V VOD sistemih ustvarjajo prometno breme uporabniki VOD storitev s svojimi zahtevami po predvajanju filmskih datotek. Dobro uravnotežen podsistem terja takšno razmestitev filmskih datotek po diskih, da je prometno breme karseda enakomerno porazdeljeno in da je zmogljivost diskovnega podsistema pri neki konfiguraciji diskov čim večja. Za izdelavo simulacijskega programa je potrebno opraviti analizo zmogljivosti obravnavanega diskovnega podsistema, ki terja sistematičen pristop, saj nesistematična obravnava pogosto vodi do napačnih zaključkov, ki so lahko posledica nejasno definiranih ciljev, napravilno izbranih metrik, nereprezentativnega bremena, spregledanja kakšnih pomembnih parametrov in podobno. Za zagotavljanje sistematičnosti sta bila upoštevana predvsem vira [1] in [5].

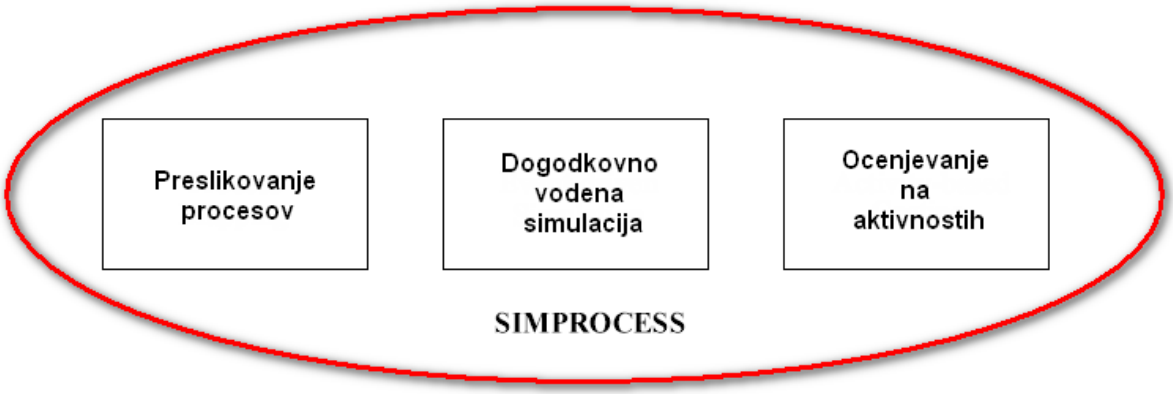
Za čim enakomernejšo porazdelitev prometnega bremena med diske v sistemu je potrebno izbrati ustrezno obliko razmeščanja datotek. V poglavju 7 smo obravnavali razmeščanje z razdeljevanjem datotek, kjer po diskih porazdelimo dele datotek, in razmeščanje z repliciranjem datotek, kjer za priljubljenejše filmske datoteke naredimo več kopij in te porazdelimo po diskih. Po pregledu strokovne literature in člankov (npr. [12]) smo ugotovili, da je za visoko interaktivne VOD sisteme, kjer so zakasnitve pri interakcijah uporabnikov s sistemom omejene na manj kot sekundo, nesporna uporaba razmeščanja z repliciranjem datotek. V okviru slednjega smo obravnavali zelo učinkovito shemo dodeljevanja datotek – kombinacijsko izravnavo bremen, ki je bila predmet simulacije. Za izdelavo simulacijskega programa smo uporabili simulacijsko orodje Simprocess.

8.1 Predstavitev simulacijskega orodja Simprocess

Simprocess je hierarhično in integrirano orodje za simulacijo procesov, ki močno izboljša produktivnost pri procesnem modeliranju in analizi. Namenjen je BPR in IT strokovnjakom industrijskih in storitvenih podjetij, ki morajo zmanjšati čas in tveganja, ki se pojavljajo pri nudenju storitev strankam, pri zadovoljevanju povpraševanja in razvoju novih proizvodov. Simprocess integrira preslikovanje procesov, hierarhično dogodkovno vodeno simulacijo in ocenjevanje na aktivnostih v samostojno orodje (glej sliko 8.1). S pomočjo njegovih gradnikov in funkcij lahko ustvarimo abstraktni model obravnavanega procesa in spremljajočo dokumentacijo (diagrame, opise, statistična poročila, grafe ipd.).

Namen procesnega modeliranja v Simprocessu je ustvariti poenostavljen in hkrati uporaben model poslovnega sistema, ki bo služil ugotavljanju ozkih grl v sistemu, povečanju učinkovitosti procesa, odkrivanju in odpravljanju zmogljivostnih problemov sistema, ocenitvi stroškov, načrtovanju novega, izboljšanega procesa itd. Simprocess upošteva časovno

spremenljivo naravo procesov, nelinearne interakcije med elementi procesa, naključno obnašanje procesov in nepričakovane dogodke v poslovnem okolju. Je orodje za dinamično modeliranje poslovnih procesov, ki za opis obnašanja sistema uporablja simulacijo diskretnih dogodkov. Za načrtovanje simulacijskega modela služijo razni gradniki, kot so procesi, resursi, entitete (pretočni objekti), aktivnosti, povezave ter vhodi/izhodi. Naključnost v modelu dosežemo z uporabo različnih verjetnostnih porazdelitev ter z naključnimi generatorji. Simprocess vsebuje 215 različnih naključnih generatorjev, vsak izmed njih pa uporablja drugačno seme, kar je potrebno za doseganje statistične neodvisnosti generiranih nizov števil v modelu. Pri izvajanju poizkusov nam Simprocess omogoča animiran prikaz izvajanja simulacije, ki nam daje dodaten vpogled v delovanje simuliranega sistema. Animacija pretakanja entitet skozi aktivnosti procesa nam olajša tudi preverjanje pravilnosti delovanja simulacijskega programa.



Slika 8.1 Hierarhično in integrirano simulacijsko orodje Simprocess. Vir: [21].

Glede na zmogljivost, funkcionalnost in namembnost simulacijskega orodja Simprocess lahko ugotovimo, da le-ta zadostuje potrebam izdelave simulacijskega modela kombinacijske izravnave bremen za VOD sistem. Podrobnosti o orodju Simprocess je moč najti v [21].

8.2 Definicija problema in načrt simulacije

Namen praktičnega dela diplomske naloge je bil narediti uporaben simulacijski model, ki bi lahko služil načrtovanju bremensko dobro (karseda enakomerno) uravnoteženega diskovnega podsistema VOD. Dobra uravnoteženost je pomembna, saj znatno vpliva na prepustnost sistema (tj. število obdelanih zahtev v časovni enoti) in posledično na zmogljivost celotnega VOD sistema. Pri slabi uravnoteženosti bi hitro prišlo do ozkih grl v diskovnem podsistemu in s tem zavračanja zahtev, kar bi imelo za posledico negativen odziv uporabnikov, česar si ne želi noben ponudnik storitev.

Simulacija je poleg analitičnih metod in meritev eden izmed temeljnih postopkov pri določanju zmogljivosti računalniških sistemov. Primerjava omenjenih treh metod je podana v tabeli 8.1.

Tabela 8.1 Primerjava različnih metod analize zmogljivosti. Vir: [5].

Kriterij	Analitični model	Simulacija	Meritve
Stanje sistema	Poljubno	Poljubno	Najmanj prototip
Čas za izvedbo	Majhen	Srednji	Spremenljiv
Potrebna orodja	Strokovnjaki	Programska orodja	Posebna oprema
Verodostojnost rezultatov	Nizka	Srednja	Spremenljiva
Študij vpliva parametrov	Enostaven	Srednji	Težak
Cena izvedbe analize	Nizka	Srednja	Visoka
Prodajna cena rezultatov	nizka	Srednja	Visoka

Simulacija se uporablja predvsem, kadar ni možnosti za izvedbo testov na realnem sistemu in če nimamo vpogleda v sistem. Ker pri izdelavi diplomskega dela ni bilo možnosti izvajanja meritev na realnem VOD sistemu ali na njegovem prototipu, je bila izbira metode simulacije ustrezna. Kljub temu, da izdelava simulacije terja več časa kot izdelava analitičnega modela, lahko rezultatom simulacije bolj zaupamo (so verodostojnejši kot pri analitičnem modelu), saj pri izdelavi modela sprejmemo manj poenostavitev in predpostavk.

Osnova za simulacijo je simulacijski program. V njem je opisan model sistema, ki zajema vse pomembne lastnosti sistema ter se med simulacijo obnaša kot opazovani sistem. Med potekom simulacije je spreminjanje različnih parametrov modela sistema ter vhodnih parametrov enostavno, kar je za namen tega diplomskega dela ugodno, saj lahko brez velikih dodatnih stroškov, ki bi jih v realnosti terjali posegi v sistem ob preizkušanju različnih rešitev, primerjamo različne konfiguracije diskovnega podsistema VOD, dobimo jasen vpogled v delovanje podsistema in tako lažje načrtujemo željeno zmogljivost sistema. S pomočjo simulacije lahko ugotovimo, ali bi predvidena oprema uspešno opravljala zastavljene naloge, oziroma kakšno opremo bi potrebovali, da bi zadostili potrebam gledalcev, ter kakšno opremo bi bilo potrebno zagotoviti za nemoteno delovanje v prihodnosti.

Simulacije se glede na stanje sistema delijo na zvezne in diskretne, glede na čas pa na časovno zvezne in časovno diskretne. Za simulacijo kombinacijske izravnave bremen smo uporabili model z zveznim časom in diskretnimi stanji. Model je tudi dinamičen, saj se stanje sistema s časom spreminja. Rezultat simulacije je verjetnosten, saj v modelu sistema nastopajo naključne vrednosti. Opazovani diskovni podsistem VOD je odprtega tipa, saj prihajajo zahteve v sistem neodvisno od strežbe. Modeliran je tudi kot izgubni sistem, kjer ni čakalnih vrst za resurse, tako da se zahteva v primeru, ko ne more biti nemudoma postrežena, enostavno zavrne.

Po začetni predstavitvi problema in nekaterih značilnostih obravnavanega sistema ter utemeljitvi izbire metode, je potrebno določiti še zmogljivostne zahteve, opredeliti opravila, ki jih sistem izvaja ter pričakovane rezultate, izbrati metrike za medsebojno primerjavo veličin, opredeliti parametre, določiti breme (množico zahtev, ki jih mora sistem obdelati) ter sprejeti razne predpostavke in poenostavitve.

Za opis modela sistema smo zaradi nazornosti uporabili enake oznake za parametre, kot so uporabljeni v poglavju 7.5.2. Parametri s pogoji in predpostavkami, ki so predmet zmogljivostne analize in so uporabljeni v nadaljevanju, so našteti po vrsticah:

$\mathcal{D}$ – množica, ki vsebuje J homogenih diskov, označenih z $1, 2, \dots, J$;

C – velikost diskovnega prostora za shranjevanje (enota diskovnega prostora je npr. 1 GB);

N – število sočasnih video tokov (logičnih kanalov), ki jih podpira disk;

$\mathcal{F}$ – množica M različnih filmskih naslovov, označenih z $1, 2, \dots, M$;

L_m – velikost filmske datoteke filma m;

$L = \sum_{m \in \mathcal{F}} L_m$ – število enot diskovnega prostora za alokacijo ene kopije vsakega filma iz $\mathcal{F}$;

$max_{m \in \mathcal{F}} L_m \ll C$ – vsak disk lahko hrani številne filmske datoteke;

$L < JC$ – sistem ima še razpoložljiv diskovni prostor za hrambo več kopij filmov iz $\mathcal{F}$;

Φ_j – množica filmskih datotek, ki so nameščene na disku j;

$\mathbf{n} = (n_1, n_2, \dots, n_M)$ – primerek replikacije datotek, kjer n_m, $m \in \mathcal{F}$, označuje celo število kopij filmske datoteke filma m in velja $1 \leq n_m \leq J$;

$\boldsymbol{\Omega} = (\Omega_1, \Omega_2, \dots, \Omega_M)$ – primerek alokacije datotek, definiran kot razporeditev diskovnih lokacij za množico filmskih datotek, ki so navedene v primerku replikacije datotek $\mathbf{n}$;

Ω_m, $m \in \mathcal{F}$ – množica n_m različnih diskov, kjer je shranjenih n_m datotečnih kopij filma m;

λ – intenzivnost prihajanja zahtev na časovno enoto za vse filmske naslove;

$1/\mu_m$ – povprečni čas povezave za film m (strežni čas filma m);

p_m – profil priljubljenosti filma m, definiran kot relativna verjetnost, da stranka zahteva film m, pri čemer velja $\sum_{m \in \mathcal{F}} p_m = 1$;

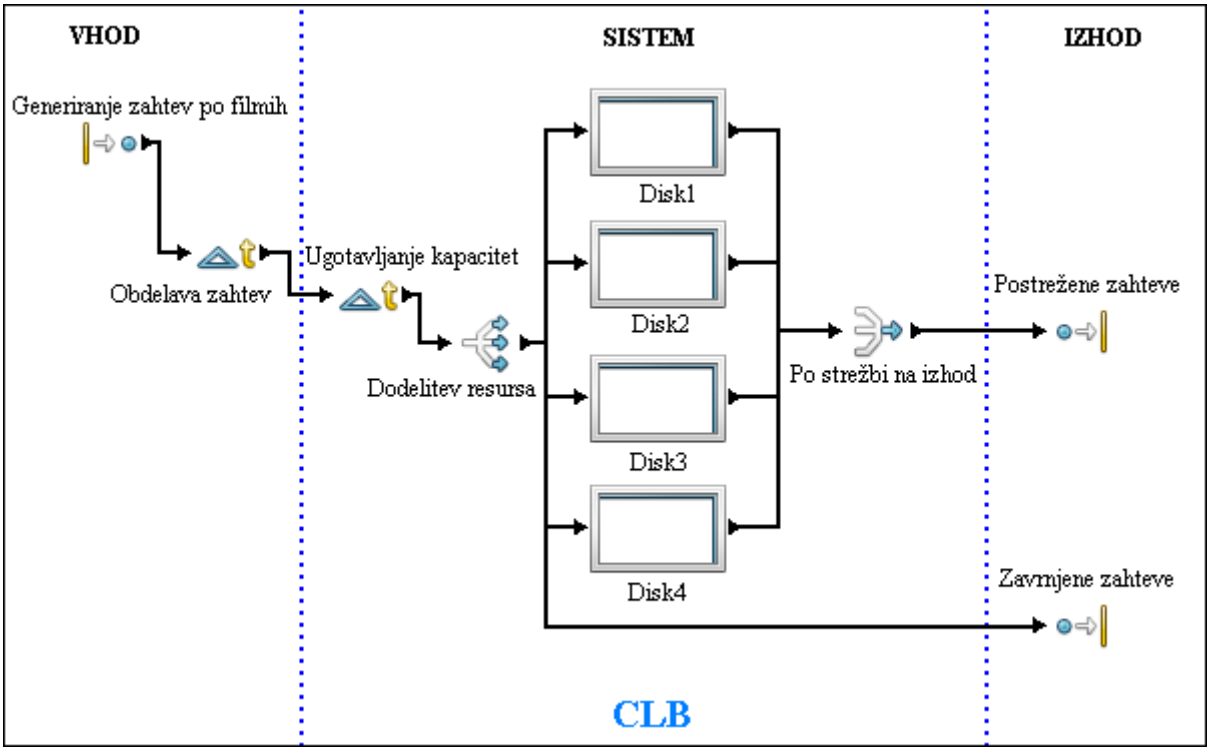
$A_m = \lambda p_m / \mu_m$ – prometno breme filma m (faktor obremenitve strežne enote s strani filma m);

$A = \sum_{m \in \mathcal{F}} A_m$ – skupno prometno breme za vse filme v $\mathcal{F}$.

Za model obravnavanega sistema smo predvideli naslednja opravila:

- generiranje zahtev za konkretne filmske naslove,
- ugotavljanje trenutne zasedenosti resursov (diskov),
- dodeljevanje prostih resursov zahtevam po izbrani shemi izbire resursov (LBF, RRT),
- usmerjanje zahtev na dodeljene proste resurse ali zavrnitev zahtev v primeru zasedenosti resursov,
- strežbo zahtev,
- usmerjanje postreženih zahtev na izhod in njihovo končanje.

Na sliki 8.2 je podan blokdiagram manjšega modela sistema v Simprocessu za primer s štirimi diski, ki predstavlja grobo predstavitev procesa. Prikazani so vsi osnovni gradniki opazovanega procesa. Gradniki so med seboj povezani s puščicami, ki ponazarjajo smer toka v diagramu. Podrobnejši opis posameznih gradnikov in programske logike, ki se nahaja v ozadju, je podan v poglavju 8.4.



Slika 8.2 Blokdiagram manjšega modela sistema v Simprocessu za primer s 4 diski.

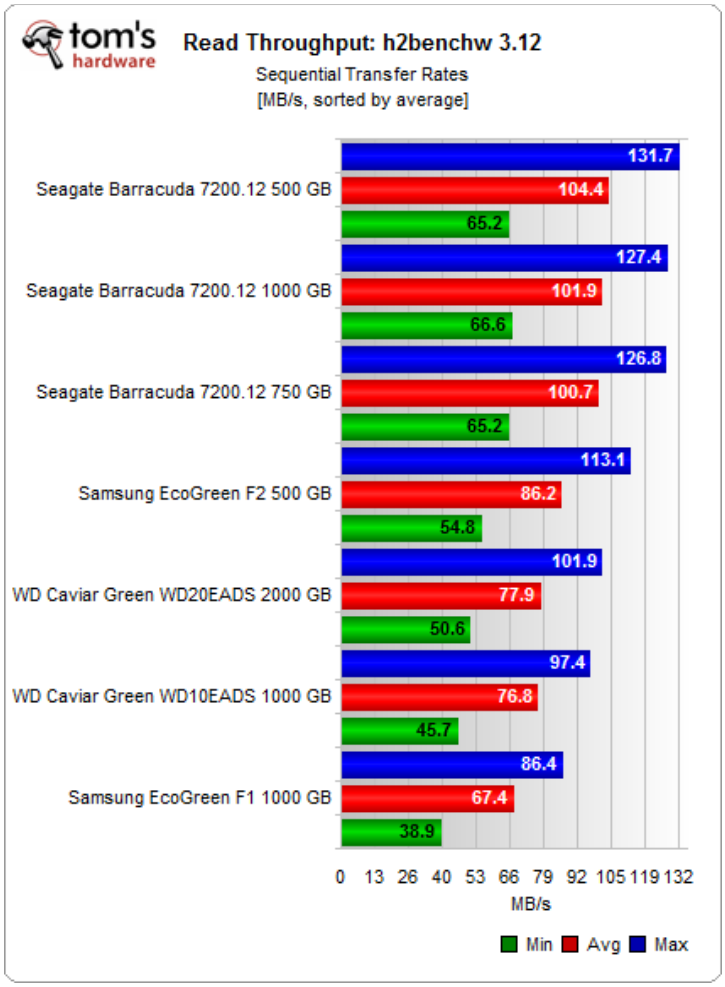
Diskovni podsistem VOD pri analizi zmogljivosti obravnavamo kot množico povezanih resursov, ki servisirajo zahteve po predvajanju filmov, katere se pojavijo na vhodu vsakega diska. Množica vhodnih zahtev po konkretnih filmskih naslovih predstavlja realno breme, ki obremenjuje strežniške sposobnosti opazovanega diskovnega podsistema. Na tem mestu velja omeniti, da ima pri ugotavljanju zmogljivosti model bremena prednost pred realnim bremenom, saj zagotavlja ponovljivost meritev, kar pri realnem bremenu ne drži. Prav tako realno breme ni primerno za primerjavo sistemov med seboj, saj je nemogoče zagotoviti enake testne pogoje. Pri načrtovanju modela smo uporabili parametrično umetno breme, ki je podano zgolj s parametri in je zato za simulacijo dobra izbira. Obremenjenost posameznih diskov v diskovnem podsistemu VOD je neposredno odvisna od značilnosti bremena, ki so pogojene s časi prihajanja zahtev ter z njihovo časovno porazdelitvijo. V nadaljevanju so podane časovne porazdelitve, ki smo jih predpostavili pri načrtovanju modela.

Agregirani prihodi zahtev za vse filmske naslove sledijo Poissonovemu procesu z intenzivnostjo λ zahtev na časovno enoto, zato smo pri generiranju zahtev uporabili Poissonovo porazdelitev. Porazdelitev medprihodnih časov uporabniških zahtev pri pretočnih multimedijskih sistemih, kamor spada tudi VOD, je eksponentna, zato smo pri modeliranju časa med zahtevami upoštevali eksponentno porazdelitev. Čas povezave filma m (čas strežbe zahteve), če upoštevamo interaktivno obnašanje uporabnikov, najbolje opisuje logaritemsko normalna porazdelitev s srednjo vrednostjo $1/\mu_m$ časovnih enot, kar je utemeljeno v [6]. Predpostavili smo, da je standardni odklon časa povezave filma m enakovreden povprečni vrednosti povezavnega časa. Profili priljubljenosti filmov, ki so upoštevani pri generiranju zahtev, so pridobljeni z Zipfovi podobno porazdelitvijo z vrednostjo parametra $\zeta = 0{\cdot}271$, kar statistično ustreza frekvencam dostopov strank do različnih filmskih naslovov.

Pri določanju zmogljivostnih zahtev sistema se je potrebno ozirati na kriterije, ki jih sprejemljiva zmogljivostna kapaciteta sistema mora zadovoljevati. Ti so izvrševanje dogovorjenih nivojev storitev, upoštevanje tehnologije in standardov ter upoštevanje stroškovnih zahtev. Za diskovni podsistem VOD smo določili pogoj o nivoju storitev, da mora biti verjetnost zavrnitve zahteve za film, z drugimi besedami verjetnost blokiranja zahteve (RBP), v povprečju manjša ali enaka kot 1%, kar zagotavlja še zadovoljivo kakovost storitve.

Zaradi lažje predstave in večje aktualnosti smo si za zmogljivostno obravnavo zamislili VOD sistem, ki hrani v svoji zbirki video vsebin 1000 različnih filmskih naslovov ($M = 1000$), kar približno odseva stanje ob koncu leta 2009 pri vseh štirih slovenskih operaterjih, ki ponujajo VOD storitve v okviru IPTV. V diplomskem delu se v okviru zmogljivostne analize s poslovnimi modeli nismo ukvarjali, a smo vseeno zaradi smotrnosti (zaradi nižjih stroškov opreme) za diskovni podsistem izbrali magnetne trde diske (angl. hard disk drives – HDDs) in ne denimo polprevodniških diskov (angl. solid-state drives – SSDs), ki so trenutno tudi do deset in večkrat dražji od primerljivih magnetnih glede na zmogljivost in kapaciteto. Privzeli smo, da so diski homogeni. V primeru nehomogenosti realnega sistema bi lahko uporabili tehnike združevanja diskov, tako da bi iz polja heterogenih diskov sestavili logično zbirko J homogenih. Za diske so na razpolago natančne specifikacije zmogljivostnih parametrov, ki se jih da meriti z mnogimi programskimi orodji. Za primerjanje sistemov so najbolj verodostojni namenski programi za meritve (angl. benchmarks). Primerjava stalnih prenosnih hitrosti (angl. sequential transfer rates – STR) pri branju za nekatere aktualne magnetne diske, ki je bila narejena s testnim programom h2benchw 3.12, je podana na grafu 8.1. Stalna prenosna hitrost je hitrost branja ali pisanja podatkov (samo efektivni podatki) z upoštevanjem časov za preklop glave (površine) in premikov na novo sled. Pri izboru diskov smo upoštevali večjo povprečno stalno prenosno hitrost pri branju (na grafu označeno z rdečo barvo). Glede velikosti pomnilniškega prostora za shranjevanje smo predpostavili, da je bolje izbrati manjše diske, saj nudijo za manjšo ceno približno enako bralno prepustnost (angl. read throughput) kot enkrat večji. Po pregledu trenutnih cen na trgu smo npr. ugotovili, da je različica diska Seagate Barracuda 7200.12 z velikostjo pomnilniškega prostora 500 GB za polovico cenejša od različice, ki lahko hrani 1000 GB podatkov. Dva 500-GB diska Seagate Barracuda 7200.12 imata skupaj torej približno dvakrat večjo povprečno bralno prepustnost kot en 1000-GB disk

pri enaki ceni, zato smo v modelu diskovnega podsistema VOD predpostavili 500-GB različico omenjenih diskov, ki med vsemi navedenimi diski na grafu 8.1 nudi tudi največjo povprečno stalno prenosno hitrost pri branju.



Graf 8.1 Primerjava stalnih prenosnih hitrosti pri branju (v MB/s) za različne magnetne diske. Vir: [18].

Za določitev števila sočasnih video tokov, ki jih lahko nek disk sprejme, je potrebno poleg bralne prepustnosti diska, ki predstavlja pasovno širino diska, sprejeti odločitev o formatu kodiranja videa in določiti bitno hitrost prenosa video vsebin. Za video kompresijo smo predvideli standard, ki uporablja napredno video kodiranje, in sicer MPEG-4 AVC (angl. advanced video coding), ki je znan tudi pod imenom H.264, saj zaradi svoje izredne učinkovitosti kodiranja na trgu že močno izpodriva starejši MPEG-2 standard. MPEG-4 AVC namreč uporabi le 50% bitne hitrosti glede na MPEG-2 pri enaki velikosti oz. kvaliteti slike in je zelo primeren za prenos video vsebin visoke ločljivosti. Video na zahtevo uporablja stalno bitno hitrost (angl. constant bit rate - CBR), kar operaterjem daje vnaprejšnjo informacijo, koliko pasovne širine zasedajo posamezni video tokovi (več o tem je moč prebrati v [13]). Za video vsebine visoke ločljivosti ponujajo napredni video kodeki stalno bitno hitrost od 6 do 9

Mb/s. Za lažje računanje smo izbrali CBR 8 Mb/s oz. 1 MB/s, kar bi ustrezalo tudi za kvaliteten MPEG-2 signal standardne ločljivosti, in predpostavili, da uporablja začrtan VOD sistem izbrano bitno hitrost kodiranja za vse filme, ki jih hrani diskovni podsistem. Pri deljenju pasovne širine izbranega diska (zaokrožene na 104 MB/s) s CBR dobimo število sočasnih video tokov (logičnih kanalov), ki jih podpira disk, N = 104.

Ob predpostavki, da trajajo filmi v povprečju 2 h (privzeli smo, da sta 2 h ena časovna enota povprečnega povezavnega časa, torej $1/\mu_m = 1$), je velikost povprečne filmske datoteke, ki je kodirana z izbranim CBR, enaka 7˙2 GB. Za potrebe simulacije smo privzeli, da so vse filmske datoteke enako velike. Za enoto diskovnega prostora smo izbrali kar velikost povprečne filmske datoteke 7˙2 GB, tako da velja $L_m = 1$ in $L = 1000$. Na en disk velikosti 500 GB lahko tako shranimo 69 filmskih datotek (C = 69), hkrati pa je izpolnjen pogoj $max_{m \in \mathcal{F}} L_m \ll C$. Za definiranje modela sistema smo morali določiti tudi minimalno število diskov (ob upoštevanju pogoja $L < JC$), ki lahko hranijo vse filmske naslove z vsemi kopijami, da pa je bilo to mogoče, smo morali prej določiti še primerek replikacije datotek $\mathbf{n} = (n_1, n_2, \ldots, n_M)$. Koliko datotečnih kopij n_m bo imel vsak posamezen filmski naslov m, smo določili glede na njegov profil priljubljenosti. Po normiranju profilov priljubljenosti smo določili primerek replikacije, kot ga prikazuje tabela 8.2. Primerek datotečne replikacije: $\mathbf{n} = (n_1, n_2, n_3, n_4, n_5, n_6, \ldots, n_{10}, n_{11}, \ldots, n_{25}, n_{26}, \ldots, n_{50}, n_{51}, \ldots, n_{250}, n_{251}, \ldots, n_{1000}) = (7, 7, 6, 6, 6, 5, \ldots, 5, 4, \ldots, 4, 3, \ldots, 3, 2, \ldots, 2, 1, \ldots, 1)$. Skupno število vseh kopij vseh filmskih naslovov, ki jih lahko hrani zasnovan diskovni podsistem VOD, je tako 1342. Če to število delimo s C, dobimo (po zaokrožitvi navzgor) število potrebnih diskov, in sicer J = 20.

Tabela 8.2 Določitev primerka replikacije datotek $\mathbf{n} = (n_1, n_2, \ldots, n_M)$ za $M = 1000$.

Filmski naslovi (m)	**Število kopij (n_m)**
1-2	7
3-5	6
6-10	5
11-25	4
26-50	3
51- 250	2
251-1000	1

Za simulacijski model je ključno določiti primerek alokacije filmskih datotek **Ω** za izbran primerek replikacije **n**, saj nam ta daje informacijo, kako naj bodo datotečne kopije filmov razmeščene po diskih, da bo prometno breme s strani uporabniških zahtev po ogledih filmov čim enakomerneje porazdeljeno med diske. Primerek alokacije smo določili s pomočjo požrešne metode za hevristično alokacijo datotek, ki je opisana v poglavju 7.5.5. Delni zapis primerka alokacije datotek, ki ga dobimo s hevristično alokacijo (vrstice M1 - M1000 predstavljajo zaporedne filmske datoteke, ki se nahajajo na navedenih diskih):

M1: 1 2 3 4 5 6 7,
…
M16: 6 7 8 11,
…
M1000: 13,

$\boldsymbol{\Omega} = (\Omega_1, \dots, \Omega_{16}, \dots, \Omega_{1000}) = ((1,2,3,4,5,6,7), \dots, (6,7,8,11), \dots, (13)).$

V okviru kombinacijske izravnave bremen (CLB) smo predstavili tri sheme izbire resursov – SRT, RRT in LBF, izmed katerih sta zadnji dve izčrpni. Predstavljene sheme so v simulacijskem modelu potrebne za usmerjanje uporabniških zahtev na proste diske, kjer se nahajajo željene filmske datoteke. Ugotovitve s področja vodovno komutiranih omrežij kažejo, da sheme usmerjanja glede na najmanjšo obremenjenost zagotavljajo boljšo učinkovitost kot sheme naključnega usmerjanja, zato smo se odločili, da s simulacijo naredimo primerjavo učinkovitosti algoritmov, ki realizirata izčrpni shemi RRT in LBF. Predpostavili smo, da bodo rezultati potrdili večjo učinkovitost sheme LBF.

Pri načrtovanju novega VOD sistema ponavadi nimamo natančnih podatkov o velikosti bremena, pa tudi, če bi bili na razpolago, se skozi čas spreminjajo. Za simulacijski model smo predvideli več različnih intenzivnosti prihajanja zahtev (λ), s katerimi smo preverili obnašanje sistema pri različnih obremenitvah. Ker smo domnevali, da hevristična metoda dobro porazdeli prometno breme med diske in da bodo diski ves čas precej enakomerno obremenjeni, smo se odločili preveriti delovanje sistema pri intenzivnostih prihajanja zahtev, ki obremenjujejo resurse nad 80% (ob upoštevanju, da bi bilo pri popolni zasedenosti resursov zasedenih vseh 2080 logičnih kanalov). Tako smo sklenili preveriti obnašanje sistema pri λ_1 = 1700 zahtev/2 h, λ_2 = 1800 zahtev/2 h, λ_3 = 1900 zahtev/2 h in λ_4 = 2000 zahtev/2 h. Cilj je bil ugotoviti maksimalno vhodno intenzivnost, pri kateri je še izpolnjen pogoj RBP $\leq$ 1%, ki še zagotavlja sprejemljivo kakovost storitve, ter zasedenost resursov pri tej intenzivnosti. RBP dobimo, če delimo število zavrnjenih zahtev v opazovanem časovnem obdobju s številom zahtev, ki so v tem obdobju vstopile v sistem.

8.3 Zbiranje in analiza podatkov

Pri gradnji simulacijskega modela je treba določiti potrebne podatke. Najprej se določi podatke na vhodu sistema, kjer zahteve vstopajo v sistem. Potreba po podatkih se kasneje izkaže tudi pri sami izgradnji modela.

Večino podatkov, ki so nujni za izgradnjo obravnavanega simulacijskega modela, smo določili in utemeljili v prejšnjem poglavju. Za obravnavo smo določili model z naslednjimi podatki:

$M = 1000,$
$J = 20,$
$C = 69,$
$N = 104,$
$L_m = 1, \forall m \in \mathcal{F},$
$\zeta = 0\dot{2}71,$
$\mathbf{n} = (n_1, n_2, n_3, n_4, n_5, n_6, \dots, n_{10}, n_{11}, \dots, n_{25}, n_{26}, \dots, n_{50}, n_{51}, \dots, n_{250}, n_{251}, \dots, n_{1000})$
$= (7, 7, 6, 6, 6, 5, \dots, 5, 4, \dots, 4, 3, \dots, 3, 2, \dots, 2, 1, \dots, 1).$

Ti podatki so služili kot vhodni podatki za program, ki realizira požrešno metodo za hevristično alokacijo datotek. Izhodni podatki tega programa, ki smo jih potrebovali za izdelavo simulacijskega modela, so primerek alokacije datotek **Ω** in delne vsote profilov priljubljenosti p_m, pri čemer velja $\sum_{m \in \mathcal{F}} p_m = 1$. Delne vsote profilov priljubljenosti določajo intervale verjetnosti, ki predstavljajo inverzno porazdelitveno funkcijo Zipfovi podobne verjetnostne porazdelitve. Le-to potrebujemo za izračun naključnih števil (s pomočjo naključnega generatorja z enakomerno porazdelitvijo), ki v simulacijskem modelu predstavljajo zahteve po predvajanju posameznih filmov. Simprocess namreč ne pozna Zipfove in njej podobnih porazdelitev, zato smo si morali pri prirejanju naključnih vrednosti po želeni porazdelitvi pomagati z opisano klasično metodo. Primerek alokacije datotek in delne vsote profilov priljubljenosti smo zapisali v Excelovi preglednici (Diski.xls in Filmi.xls), ki sta služili za vhodne podatke simulacijskega modela v Simprocessu. Vhodni podatki modela so tudi verjetnostne porazdelitve, ki smo jih opisali v prejšnjem poglavju, ter:

λ_1 = 1700 zahtev/2 h,
λ_2 = 1800 zahtev/2 h,
λ_3 = 1900 zahtev/2 h,
λ_4 = 2000 zahtev/2 h,
$1/\mu_m$ = 2 h.

Vse ostale podatke, po katerih se je pojavila potreba pri sami izgradnji modela, so opisani v poglavju 8.4. Prav tako smo pri izdelavi modela definirali spremenljivke, entitete in resurse.

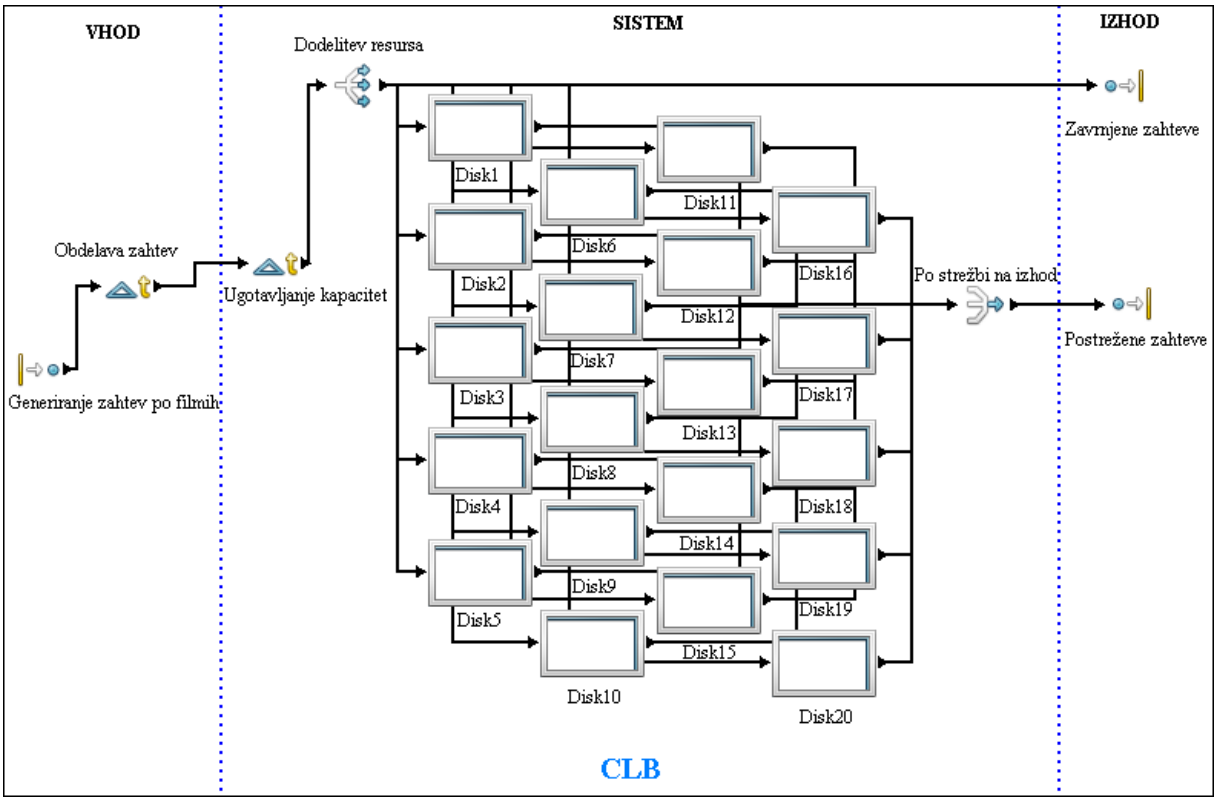
8.4 Izdelava modela

Vsak simulacijski model temelji na določenih predpostavkah in poenostavitvah, zato ne moremo nikoli za noben izdelani model trditi, da je pravilen ali nepravilen. Model je lahko kvečjemu uporaben ali neuporaben. Za uporabnega smatramo tistega, ki uporabniku posreduje koristne informacije.

Pri izgradnji lahko kompleksnost modela zelo hitro narašča, zato je zaradi lažje obvladljivosti in lažjega odkrivanja morebitnih napak priporočljivo začeti z gradnjo majhnega modela, ki mora vsebovati že vse bistvene lastnosti sistema, ki ga simulira. Tako smo najprej izdelali

model za primer s štirimi diski, katerega blokdiagram smo predstavili na sliki 8.2. Zanj smo uporabili podatke iz študije [11], saj smo želeli na podlagi že opravljene študije potrditi svoj model. Prednost začetnega izvajanja poizkusov na majhnem modelu je tudi v manjši časovni porabi, kar je zelo priročno, saj smo s časom pri izdelavi modela ponavadi omejeni.

Kompleksnejši model za primer z 20 diski in 1000 filmskimi naslovi, ki je bil predmet simulacije, prikazuje blokdiagram na sliki 8.3. Pri izdelavi simulacijskega modela v Simprocessu smo definirali 20 resursov ($disk1 - disk20$) s kapaciteto 104 enote (kapaciteta diskov v modelu pomeni število logičnih kanalov $N = 104$), ki predstavljajo diske v diskovnem podsistemu VOD. Definirali smo tudi entiteto $zahteva$, ki ima entitetne atribute $film$, $disk$ in tip. Atributu $film$ smo v izrazih (angl. expressions) v generatorju zahtev priredili naključno številko filmskega naslova po Zipfovi podobni porazdelitvi, tako da je zgenerirana entiteta predstavljala uporabniško zahtevo po točno določenem filmskem naslovu. Porazdelitev zagotavlja, da bodo zahteve po priljubljenejših filmih zgenerirane večkrat kot zahteve po manj priljubljenih. Atribut tip se določi z izrazi v gradniku Ugotavljanje kapacitet glede na to, katerega tipa c je zgenerirana filmska datoteka, ali z drugimi besedami, koliko datotečnih kopij ima. V istem gradniku se z izrazi, ki s pomočjo vhodnih podatkov o diskovnih lokacijah Ω_m za film m, realizirajo eno izmed shem izbire resursov (RRT, LBF), določi atribut $disk$, ki služi za usmerjanje zahteve na ustrezen prosti disk ali na izhod, če so vsi diski, na katerih so kopije zahtevanega filma, polni.



Slika 8.3 Blokdiagram modela sistema v Simprocessu za primer z 20 diski in 1000 filmskimi naslovi.

V simulacijskem modelu smo določili tudi globalne atribute modela, in sicer $maxKopij$, ki predstavlja največje število kopij, ki jih ima najpriljubljenejši film (ta jih ima vedno največ pri Zipfovi podobni porazdelitvi), $stFilmov$, ki pove, koliko različnih filmskih naslovov vsebuje sistem ($M = 1000$), tabelo $verjetnost$ velikosti M, ki je namenjena hranjenju delnih vsot profilov priljubljenosti p_m, pridobljenih iz Excelove preglednice Filmi.xls za realizacijo Zipfovi podobne porazdelitve. Definirali smo objekta ExcelDiski in ExcelFilmi, ki služita za rokovanje (odpiranje, branje, zapiranje datotek) z Excelovima preglednicama Filmi.xls in Diski.xls, ter objekt $ArrayDiski$, ki je namenjen kreiranju dinamične dvodimenzionalne tabele velikosti $stFilmov \times maxKopij$, v kateri po vrsticah hranimo elemente primerka datotečne alokacije $\Omega_m, m \in \mathcal{F}$, prebrane iz preglednice Diski.xls (če ima nek filmski naslov manjše število kopij od $maxKopij$, je vrstica tabele, ki hrani diskovne lokacije kopij izbranega filmskega naslova, dopolnjena z ničlami). S pomočjo tabele $ArrayDiski$ je v izrazu aktivnosti Ugotavljanje kapacitet realizirano usmerjanje zahtev na ustrezne nezasedene diske. Določili smo tudi globalni atribut $kapacitetaDiskov$, ki ponazarja število logičnih kanalov posameznega diska ($N = 104$) ter enodimenzionalno tabelo $zasedenostDiskov$ (velikosti $J = 20$), v kateri se nahaja sistemska informacija o zasedenosti posameznega diska, ki služi usmerjevalnemu algoritmu za usmerjanje zahtev na izbrani resurs po določeni shemi izbire resursov (RRT, LBF). Pri izdelavi različice modela s shemo RRT smo definirali še objekt $randDisk$, ki smo ga uporabili za kreiranje enodimenzionalne dinamične tabele, namenjene permutaciji diskovnih lokacij, da je bila zagotovljena naključna izbira prostega diska.

Diagram na sliki 8.3 prikazuje smer toka entitet, in sicer od generiranja zahtev na vhodni strani, vstopanja le-teh v sistem, njihove strežbe, do zapuščanja obdelanih ali zavrnjenih zahtev iz sistema na izhodni strani. V izrazih modela (angl. model expressions) smo v izrazu Start Simulation() (ki se izvede ob začetku simulacije) realizirali odpiranje vhodnih datotek Diski.xls in Filmi.xls, ki vsebujeta vnaprej pripravljene vhodne podatke za simulacijski model. V istem izrazu se nahaja tudi programska skripta, ki prebere vhodne podatke v tabeli $verjetnost$ ter $ArrayDiski$, v izrazu End Simulation() pa se izvrši zapiranje datotek. Simulacija se začne v aktivnosti Generiranje zahtev po filmih. V izrazu Release Entity() te aktivnosti se določi entiteta zahteva, ki se ji po Zipfovi podobni porazdelitvi (s pomočjo naključnega generatorja in tabele $verjetnost$) dodeli številka filma, ki predstavlja uporabniško zahtevo po predvajanju točno določenega filma. Zahteve se generirajo po Poissonovi porazdelitvi z intenzivnostjo λ, s čimer dobimo željeno število zahtev na časovno enoto. Aktivnost Obdelava zahtev smo uporabili za modeliranje časa med zahtevami. Upoštevali smo, da je verjetnostna porazdelitev medprihodnih časov uporabniških zahtev eksponentna.

Zahteva nato vstopi v sistem. Aktivnost Ugotavljanje kapacitet je namenjena realizaciji usmerjevalnega algoritma, ki zahtevi dodeli ustrezen prosti disk po določeni shemi izbire resursov. V izrazu Accept Entity() se najprej ugotovi, koliko datotečnih kopij ima zahtevani film, to število se priredi entitetnemu atributu tip, ki določa tip filma, nato pa programska skripta na podlagi globalne tabele $zasedenostDiskov$ pri različici sistema CLB-LBF preveri trenutno zasedenost vseh diskov, kjer se nahajajo kopije zahtevanega filma, in zahtevi preko

njenega atributa $disk$ dodeli najmanj zaseden disk, pri različici sistema CLB-RRT pa preverja zasedenost diskov s ponavljajočimi naključnimi poskusi in zahtevi dodeli prvi prosti disk. V primeru, da so vsi diski zasedeni, se zahtevi priredi neobstoječ disk 0, kar vodi do zavrnitve zahteve. Aktivnost Dodelitev resursa je potrebna za pravilno usmerjanje zahtev na diske oziroma na aktivnost Zavrnjene zahteve, če so vsi pripadajoči diski v danem trenutku zasedeni. Usmerjanje se vrši s pomočjo aktivnosti in iz nje izhajajočih povezav preko entitetnega atributa $disk$.

Diski so namenjeni strežbi uporabniških zahtev. Modelirali smo jih z gradniki procesov, v katerih se nahaja aktivnost Predvajanje filmov (glej sliko 8.4), ki je realizirana kot zakasnitev. V tej aktivnosti uporabniška zahteva zasede en logični kanal pasovne širine diska (eno enoto resursa). Programska skripta v izrazu Accept Entity() aktivnosti Predvajanje filmov skrbi, da se ob prihodu zahteve na disk (in s tem zaseženju logičnega kanala) v tabeli $zasedenostDiskov$ poveča število zasedenih enot resursa za eno enoto, v izrazu Release Entity() pa, da se to število ob zapuščanju entitete po koncu strežbe zmanjša za ena. S tem se ohranja in osvežuje informacija o trenutni zasedenosti diskov, ki jo potrebuje usmerjevalni algoritem za usmerjanje, poleg tega pa zagotavlja, da ni čakalnih vrst za strežbo. Povprečni čas povezave $1/\mu_m$ pri predvajanju filma m (oziroma čas strežbe uporabniške zahteve) je zaradi interaktivnega obnašanja uporabnikov modeliran po logaritemsko normalni porazdelitvi s standardnim odklonom, ki je enakovreden povprečni vrednosti povezavnega časa. Predpostavili smo, da filmi v povprečju trajajo 2 h. Tako smo v porazdelitvi upoštevali $1/\mu_m$ = 2 h.

Slika 8.4 Aktivnost Predvajanje filmov znotraj procesov Disk1 – Disk20.

Entitete po strežbi potujejo v aktivnost Postrežene na izhod, ki služi usmerjanju postreženih zahtev na izhodno stran v aktivnost Postrežene zahteve, kjer se zahteve sprostijo oziroma končajo. Aktivnost Zavrnjene zahteve je namenjena sprostitvi zavrnjenih zahtev iz sistema. Pri modeliranju naključnosti smo uporabljali različne generatorje naključnih števil, za vsakega izmed njih pa smo izbrali drugačno seme in tako zagotovili statistično neodvisnost generiranih nizov števil.

8.5 Preverjanje in potrditev modela

Simulacijski model mora pred uporabo prestati še fazo preverjanja ali verifikacije in fazo potrjevanja ali validacije. Verifikacija preveri, ali je model pravilno implementiran, se pravi pravilnost delovanja, validacija pa potrdi, ali rezultati simulacije uspešno predstavljajo realen

sistem (sovpadajo z njegovo dinamiko), oziroma če se skladajo s sprejetimi predpostavkami. Glede na ta dva koncepta lahko uvrstimo simulacijski model v eno izmed štirih kategorij – simulacijski model je lahko nepreverjen in napačen, nepreverjen in veljaven, preverjen in napačen, ali pa preverjen in pravilen.

Verifikaciji bi lahko z drugimi besedami rekli iskanje in odpravljanje napak (angl. debugging), včasih pa se uporablja tudi izraz razhroščevanje. Za preverjanje obstaja ponavadi kar nekaj tehnik, kot so testni izpisi, delovanje korak za korakom, prekinitve, spremljanje toka dogodkov itd.

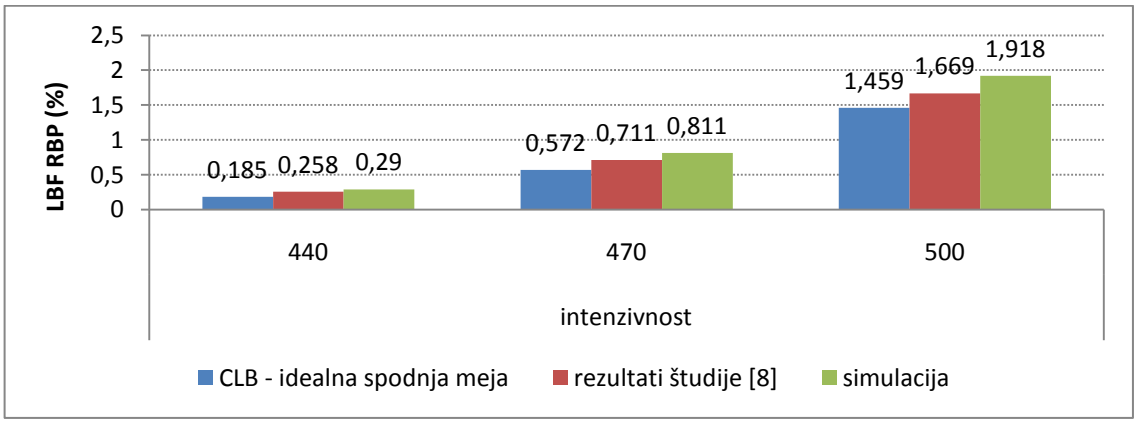
Pravilnost delovanja smo morali preveriti tako za javanski program za hevristično alokacijo datotek, ki simulacijskemu modelu daje vhodne podatke, kakor tudi za sam model, ki smo ga implementirali v Simprocessu. Hevristično metodo smo sprogramirali s pomočjo razvojnega orodja Eclipse, ki nudi mnogo različnih mehanizmov za iskanje in odpravljanje napak, kot so testni izpisi, lovljenje izjem, izvajanje korak za korakom, vstavljanje kontrolnih točk, JUnit testi, razhroščevanje itd., ki so nam zelo olajšali odpravo vseh napak v programski kodi. Po odpravi napak smo morali potrditi še ustreznost izhodnih podatkov. Pri izpisu delnih vsot profilov priljubljenosti ni bilo težko preveriti pravilnosti podatkov, saj je moral biti izpolnjen pogoj $\sum_{m \in \mathcal{F}} p_m = 1$, kar se je izkazalo za resnično. Pravilen seštevek vseh verjetnosti p_m, $m \in \mathcal{F}$, se je odražal v zadnji delni vsoti za najpriljubljenejši filmski naslov ($m = 1$). Tudi izpis primerka alokacije datotek $\boldsymbol{\Omega}$ se je izkazal za pravilnega. Preverjanje, če je sistem kombinacijsko bremensko izravnan, je zaradi kombinacijske kompleksnosti lažje preveriti na manjšem sistemu, npr. za primer s štirimi diski. Izpis je potrdil kombinacijsko izravnavo bremen, saj se je delitev resursov skladala z delitvijo, ki je prikazana na sliki 7.5(c).

Verifikacija simulacijskega modela je bila nekoliko težavnejša, saj Simprocess ne nudi toliko možnosti za iskanje napak kot Eclipse. Pri iskanju napak smo si pomagali predvsem s testnimi izpisi vrednosti v datoteko, počasnim in animiranim izvajanjem simulacije ter z uporabo manjših modelov zaradi lažje obvladljivosti. Pri preverjanju modela smo se opirali tudi na pričakovane rezultate in predpostavke, ki smo jih sklenili pri načrtovanju modela. Tako je bilo na primer za fazo verifikacije pomembno, da so rezultati pokazali, da za resurse med izvajanjem simulacije ni bilo čakalnih vrst in da so bili resursi približno enakomerno obremenjeni, kar je bilo v skladu s pričakovanji. Model je uspešno prestal fazo verifikacije.

Validacija na drugi strani ugotavlja, ali so predpostavke, ki so bile uporabljene pri načrtovanju modela, sprejemljive. Simulacijski model je potrjen, če se rezultati pravilno implementiranega modela skladajo s predpostavkami in se dobro prilagajajo rezultatom realnega sistema ali rezultatom, ki so pridobljeni npr. z analitično metodo. Validacija modela sestoji iz potrjevanja predpostavk, vhodnih vrednosti in porazdelitev ter izhodnih vrednosti in zaključkov. Test veljavnosti lahko temelji na intuiciji eksperta, meritvah realnega sistema ali teoretičnih rezultatih.

Vhodni podatki za simulacijski model so izhodni podatki programa, ki realizira hevristično alokacijo datotek, in so že prestali fazo validacije. Ker meritve realnega sistema niso bile

mogoče, saj dejanskega VOD sistema nismo imeli na voljo, smo se pri potrjevanju modela opirali na teoretične rezultate, ki so bili objavljeni v študijah [7, 8, 11]. Rezultate, ki smo jih pridobili z izvajanjem simulacije, smo primerjali s teoretičnimi rezultati in ugotavljali njihovo sovpadanje. Za večje zaupanje v pridobljene rezultate smo preizkusili različno velike modele z vsemi tremi shemami izbire resursov (SRT, RRT, LBF) na podatkih, ki so bili uporabljeni v omenjenih študijah. Popolno ujemanje simulacijskih in teoretičnih rezultatov zaradi naključnosti v realnih procesih ni mogoče, zato se je potrebno zadovoljiti s približkom ujemanja. V vseh primerjavah teoretičnih rezultatov s simulacijskimi se je izkazalo, da se slednji precej dobro prilagajajo teoretičnim. V nobenem primeru se tudi ni pripetilo, da bi simulacijski rezultati bili pod teoretično spodnjo mejo kombinacijske izravnave bremen, kar bi nakazovalo na njihovo nepravilnost. Za ponazoritev ujemanja simulacijskih rezultatov s teoretičnimi smo eno izmed primerjav upodobili na grafu 8.2. Rezultati se nanašajo na RBP LBF sistema z 20 diski in 200 različnimi filmskimi naslovi ter intenzivnostmi prihajanja zahtev λ_1 = 440, λ_2 = 470 in λ_3 = 500. Zaradi dobrega ujemanja primerjanih rezultatov lahko zaključimo, da je model potrjen. Simulacijski model je torej preverjen in pravilen.



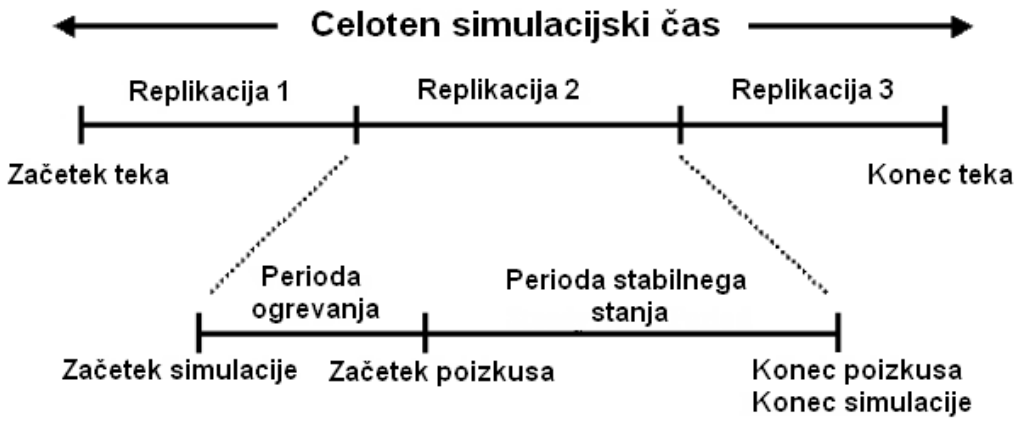
Graf 8.2 Primerjava ujemanja teoretičnih in simulacijskih rezultatov za RBP LBF sistema pri različnih vhodnih intenzivnostih prihajanja zahtev.

8.6 Izvajanje poizkusov in analiza rezultatov

Po preverbi in potrditvi modela lahko pričnemo z izvajanjem simulacijskih poizkusov. Izvajanja nam dajejo koristne informacije o simuliranem sistemu. Rezultati nam služijo za načrtovanje ustrezno dimenzioniranega sistema, dajejo informacije o obnašanju sistema pri spremembi parametrov ter o obnašanju pri različnih obremenitvah. Z njimi lahko ugotovimo, kje v sistemu se nahajajo ozka grla, pri kateri obremenitvi resursov je sistem še kos zadanim nalogam in kakšna je izkoriščenost posameznih resursov pri tej obremenitvi. Naredimo lahko primerjavo različnih algoritmov (v našem primeru sta predmet obravnave algoritma za usmerjanje zahtev na diske – LBF in RRT) ter ugotovimo, v kakšni meri se rezultati skladajo s hipotezami, ki smo jih postavili pri načrtovanju modela.

V simulacijskem modelu nastopajo stohastični elementi, ki odražajo naključnost opazovanega procesa, posledica tega pa je, da so simulacijski rezultati po več izvajanjih simulacije pri istih parametrih različni. Simulacijo je potrebno večkrat ponoviti, da pridobimo ustrezne statistične podatke o procesu, kot sta srednja vrednost in interval zaupanja. Simprocess nudi kar nekaj možnosti za statistično obdelavo pridobljenih podatkov. Zelo priročna so t.i. standardna poročila, ki jih Simprocess zgenerira po izvajanjih simulacije. Z njimi lahko med drugim prikažemo povprečne vrednosti izvajanj ter izračunamo 90%, 95% in 99% intervale zaupanja.

Pri izvajanju simulacije je potrebno vedno upoštevati zagonski čas simulacije, ki je v Simprocessu poimenovan kot perioda ogrevanja. Simulacija se ponavadi začne z neobremenjenim sistemom, v katerem še ni zahtev, in šele čez nekaj časa preide v normalno delovno stanje oziroma periodo stabilnega stanja, v kateri lahko opazujemo povprečno zasedenost resursov itd. Simulacijske rezultate, ki izvirajo iz periode ogrevanja, je potrebno preskočiti, saj ti ne dajejo prave slike o delovanju sistema v stabilnem stanju. V Simprocessu je slednje zelo enostavno narediti, saj lahko periodo ogrevanja nastavimo pred samim začetkom simulacije in s tem zagotovimo, da se podatki začnejo beležiti šele na začetku periode stabilnega stanja, ki hkrati predstavlja začetek poizkusa (glej sliko 8.5). V našem primeru smo za periodo ogrevanja nastavili začetni dan. Izvajanje simulacije imenujemo tudi replikacija simulacije. Vsaka replikacija vsebuje tako periodo ogrevanja, kakor tudi periodo stabilnega stanja. Za analizo obravnavanega diskovnega podsistema smo določili po pet simulacijskih replikacij za vsak poizkus. Za pridobitev verodostojnih rezultatov je potrebno zagotoviti tudi primerno dolžino simulacije, ki pa je povezana z vhodnimi parametri. Tako smo glede na vhodne intenzivnosti prihajanja zahtev in velikost sistema določili različne periode simulacij, in sicer od enega do več mesecev. Izvajanje simulacijskega programa se je izkazalo kot časovno zelo potratno, saj smo za pridobitev rezultatov za en poizkus porabili tudi po več ur za izvajanje na enem osebnem računalniku, kar je slabost v primerjavi z analitičnimi metodami. Prednost simulacijskega pristopa pred analitičnim pa je pridobitev verodostojnejših rezultatov, saj pri simulaciji sprejmemo manj poenostavitev.



Slika 8.5 Izvajanje simulacij v programskem orodju Simprocess. Vir: [21]

1. NIZ POIZKUSOV

Simulacija ni nikoli sama sebi namen, saj se na podlagi dobljenih rezultatov odločamo o nadaljnjih korakih, ki jih bomo storili bodisi pri načrtovanju novega sistema bodisi pri nadgradnji obstoječega realnega sistema. Smiselno je izvesti več poizkusov, saj različne variante olajšujejo odločitveni proces in dajejo temeljitejši vpogled v delovanje sistema. Pri načrtovanju modela simulacije smo si za cilj zadali karseda enakomerno zasedenost resursov. Predpostavili smo izgradnjo diskovnega podsistema VOD z 20 500-GB diski, ki lahko hranijo 1000 različnih filmskih naslovov. Želeli smo ugotoviti, pri kateri vhodni intenzivnosti prihajanja zahtev je še izpolnjen pogoj RBP $\leq 1\%$, ki zagotavlja še sprejemljivo kakovost storitve, ter zasedenost resursov pri tej intenzivnosti. Prav tako smo hoteli primerjati učinkovitost dveh izčrpnih shem izbire resursov za kombinacijsko izravnavo bremen, in sicer LBF in RRT, pri čemer smo predpostavili večjo učinkovitost LBF. Določili smo, da bo to kar prvi niz poizkusov. V tabeli 8.3 navajamo vrednosti parametrov, ki smo jih uporabili pri 1. nizu poizkusov.

Tabela 8.3 Določitev parametrov za 1. niz poizkusov.

Parametri	Opis parametrov
$M = 1000$	število različnih filmskih naslovov, ki jih hrani diskovni podsistem VOD
$J = 20$	število homogenih diskov za shranjevanje filmskih datotek
$C = 69$	maksimalno število prostorskih enot za shranjevanje na posameznem disku
$N = 104$	število sočasnih video tokov (logičnih kanalov), ki jih podpira vsak disk
$L_m = 1\ (= 7{,}2\ GB)$, $\forall m \in \mathcal{F}$	velikost vsake filmske datoteke m je ena prostorska enota
$1/\mu_m = 1\ (= 2\ h)$	povprečni povezavni čas filma m je ena časovna enota (strežni čas filma m)

Učinkovitost algoritmov za usmerjanje zahtev na diske (LBF, RRT) smo preverili pri več vhodnih intenzivnostih λ. Zaradi domneve, da bo breme po diskih dobro uravnoteženo, smo sklenili preveriti obnašanje sistema pri intenzivnostih, ki obremenjujejo resurse nad 80%. Tako smo za 1. niz poizkusov določili intenzivnosti prihajanja zahtev od 1700 zahtev/2 h do 2000 zahtev/2 h s korakom 100.

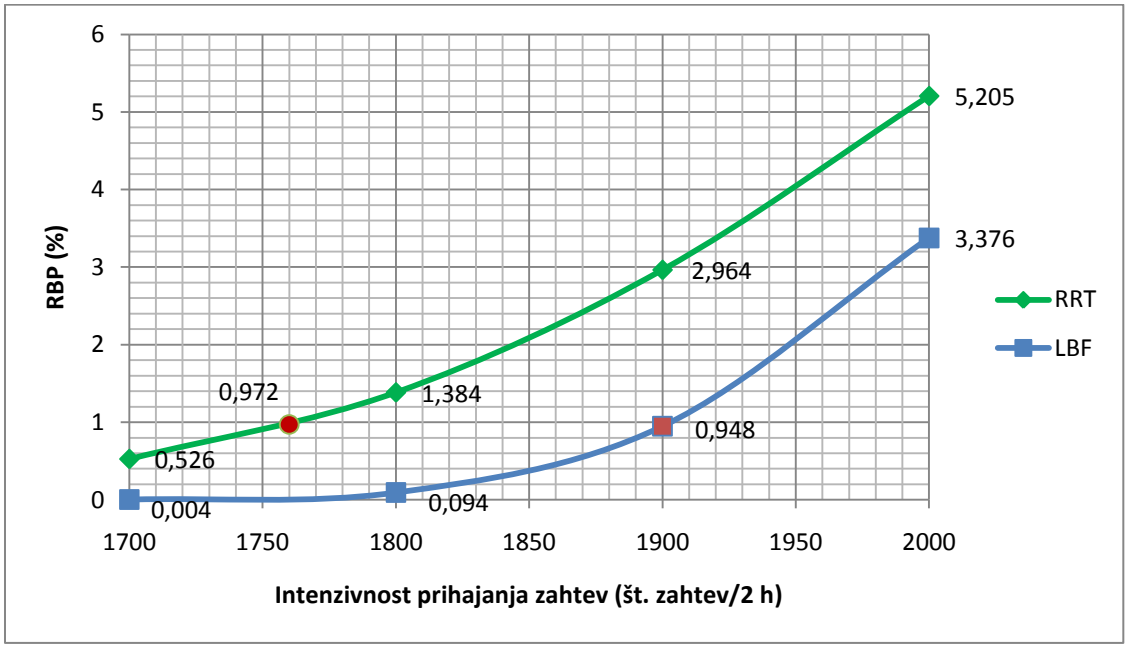
Rezultati so pokazali zelo enakomerno izkoriščenost resursov, kar potrjuje domnevo, da hevristična alokacija datotek za dosego kombinacijske izravnave bremen dobro porazdeli breme med diske. Slika 8.6 prikazuje izsek iz standardnega poročila Simprocessa za CLB-LBF sistem pri λ = 1900 zahtev/2 h, iz katerega je razvidno, da so diski pri tej intenzivnosti približno 90% obremenjeni (stolpec "Busy"). Rezultate izvajanj simulacij pri različnih λ za sistema CLB-LBF in CLB-RRT smo upodobili na grafu 8.3. Izkazalo se je, da je pri λ = 1900 zahtev/2 h pri CLB-LBF sistemu dobro izpolnjen tudi pogoj RBP $\leq 1\%$, kar pomeni, da lahko zagotovimo ustrezno kakovost VOD storitve pri visoki 90˙3% obremenjenosti resursov. Na grafih smo pridobljene rezultate, ki predstavljajo približno mejo za dosego zahtevane

kakovosti storitve, označili z rdečimi oznakami. Primerjava učinkovitosti obeh sistemov je po pričakovanjih pokazala večjo učinkovitost LBF sheme izbire resursov pred RRT shemo. Za določitev meje še zadovoljive kakovosti storitve pri RRT shemi smo si pomagali z grafom. Ocenili smo, da se bo ta meja nahajala nekje pri λ = 1760 zahtev/2 h, kar se je izkazalo za pravilno. Povprečna zasedenost resursov pri tej vhodni intenzivnosti je bila 83˙8%. Manjša učinkovitost RRT sheme izvira iz same narave algoritma RRT in se kaže tudi v večjem standardnem odklonu σ od povprečne obremenjenosti diskov – pri RRT imamo σ = 0˙592426, pri LBF pa σ = 0˙233043. Rezultat RBP za CLB-LBF sistem je pri λ = 1700 zahtev/2 h in 81˙6% obremenjenosti resursov skoraj ničeln, kar znova potrjuje izredno učinkovitost LBF sheme.

Resource : Percent Utilization By State

Resource Names	Idle	Busy
disk1	9,239%	90,760%
disk10	9,747%	90,253%
disk11	9,812%	90,185%
disk12	9,657%	90,343%
disk13	9,690%	90,309%
disk14	9,862%	90,136%
disk15	9,693%	90,307%
disk16	9,862%	90,137%
disk17	9,959%	90,040%
disk18	9,914%	90,085%
disk19	9,969%	90,029%
disk2	9,374%	90,625%
disk20	10,055%	89,944%
disk3	9,405%	90,594%
disk4	9,417%	90,582%
disk5	9,425%	90,573%
disk6	9,505%	90,494%
disk7	9,478%	90,521%
disk8	9,535%	90,464%
disk9	9,646%	90,354%

Slika 8.6 Odstotki izkoriščenosti resursov za CLB-LBF sistem pri λ = 1900 zahtev/2 h.



Graf 8.3 Primerjava učinkovitosti CLB-LBF in CLB-RRT sistema pri različnih vhodnih intenzivnostih prihajanja zahtev za 1. niz poizkusov.

2. NIZ POIZKUSOV

Denimo, da je CLB-LBF sistem z 20 diski iz prejšnjega niza poizkusov že precej obremenjen in bi radi povečali število možnih sočasnih dostopov strank pri enaki kakovosti storitve. Prav tako bi radi razširili ponudbo filmskih vsebin. Ker so rezultati pri 1. nizu poizkusov potrdili večjo učinkovitost LBF sheme delitve resursov v primerjavi z RRT shemo, smo pri nadaljnjih nizih poizkusov podrobneje obravnavali le CLB-LBF sisteme. Za namen 2. niza poizkusov smo diskovnemu podsistemu VOD dodali dva diska, kar prikazuje tabela 8.4(a). S tem smo povečali število vseh logičnih kanalov za 10% glede na sistem iz 1. niza poizkusov, in sicer na 2288 kanalov. Takšen sistem lahko sočasno sprejme več strank kot prejšnji pri enaki kakovosti storitve. Predpostavili smo, da lahko ob 10% povečanju logičnih kanalov sprejme približno 10% več strank. Ocenili smo, da se meja sprejemljive kakovosti storitve nahaja približno pri λ = 2090 zahtev/2 h. Ocena se je izkazala za pravilno, kar je razvidno z grafa 8.4.

V diskovnem podsistemu VOD, ki smo ga uporabili v 1. nizu poizkusov, je bilo shranjenih 1342 datotečnih kopij filmov. Ker lahko na 20 diskih hranimo največ $J \cdot C$ = 1380 datotečnih kopij, je bila zasedenost diskovnega prostora za shranjevanje 97˙25%. V 2. nizu poizkusov smo število diskov povečali na 22. Diskovni podsistem lahko v tem primeru hrani 1518 datotečnih kopij, zato je pri M = 1000 zasedenost diskovnega prostora 88˙41%. To nam daje možnost, da shranimo v diskovni podsistem dodatne filmske naslove ter s tem razširimo ponudbo filmskih vsebin. Odločili smo se, da bomo v diskovni podsistem dodali 150 filmskih naslovov, kar prikazuje tabela 8.4(b). Določiti smo morali tudi nov primerek datotečne replikacije **n** za M = 1150 (glej tabelo 8.4(c)), ki je služil za določitev novega primerka alokacije datotek **Ω** s hevristično metodo. Število vseh datotečnih kopij filmov, ki sledi iz **n**, je 1517, zasedenost diskovnega prostora pri 1150 filmskih naslovih in 22 diskih pa je 99˙93%. Rezultati izvajanja 2. niza poizkusov z dodanimi filmskimi naslovi se dobro prilagajajo rezultatom brez dodanih filmskih vsebin, meja še sprejemljive kakovosti storitve (1% RBP) pa se še vedno giblje okoli λ = 2090 zahtev/2 h (glej graf 8.4) in 90% obremenjenosti diskov.

Tabela 8.4 Določitev parametrov za 2. niz poizkusov.
(a) Povečanje števila diskov za 2.
(b) Povečanje števila filmskih naslovov za 150.
(c) Določitev primerka replikacije datotek $\mathbf{n} = (n_1, n_2, \ldots, n_M)$ za $M = 1150$.

(a)

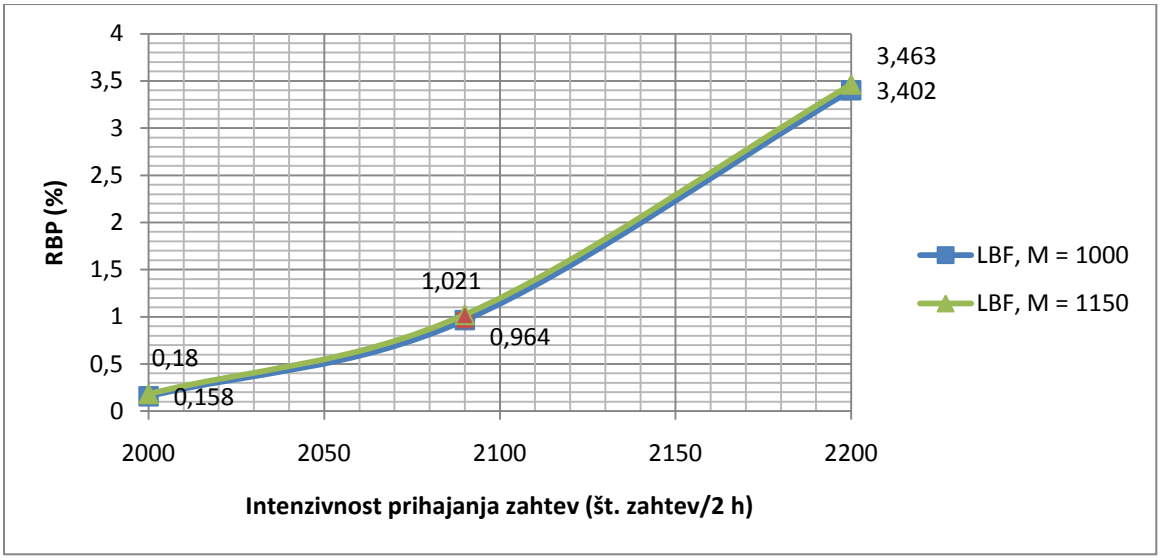
Parametri
$M = 1000$
$\boldsymbol{J = 22}$
$C = 69$
$N = 104$
$L_m = 1\ (= 7{,}2\ GB)$, $\forall m \in \mathcal{F}$
$1/\mu_m = 1\ (= 2\ h)$

(b)

Parametri
$\boldsymbol{M = 1150}$
$\boldsymbol{J = 22}$
$C = 69$
$N = 104$
$L_m = 1\ (= 7{,}2GB)$, $\forall m \in \mathcal{F}$
$1/\mu_m = 1\ (= 2\ h)$

(c)

Filmski naslovi (m)	Število kopij (n_m)
1-2	7
3-5	6
6-10	5
11-25	4
26-50	3
51-275	2
276-1150	1



Graf 8.4 Rezultati izvajanja 2. niza poizkusov za CLB-LBF sistem.

V 2. nizu poizkusov smo z dodatnima diskoma povečali število možnih sočasnih dostopov strank pri enaki kakovosti VOD storitve glede na CLB-LBF sistem iz 1. niza poizkusov. Dodajanje novih 150 filmskih naslovov v diskovni podsistem VOD ni vodilo k večjemu zavračanju zahtev in s tem slabšanju kakovosti storitve pri določeni vhodni intenzivnosti prihajanja zahtev, kar kaže na dobro bremensko izravnanost dopolnjenega sistema, ki je skoraj identična izravnanosti nedopoljnjenega sistema.

3. NIZ POIZKUSOV

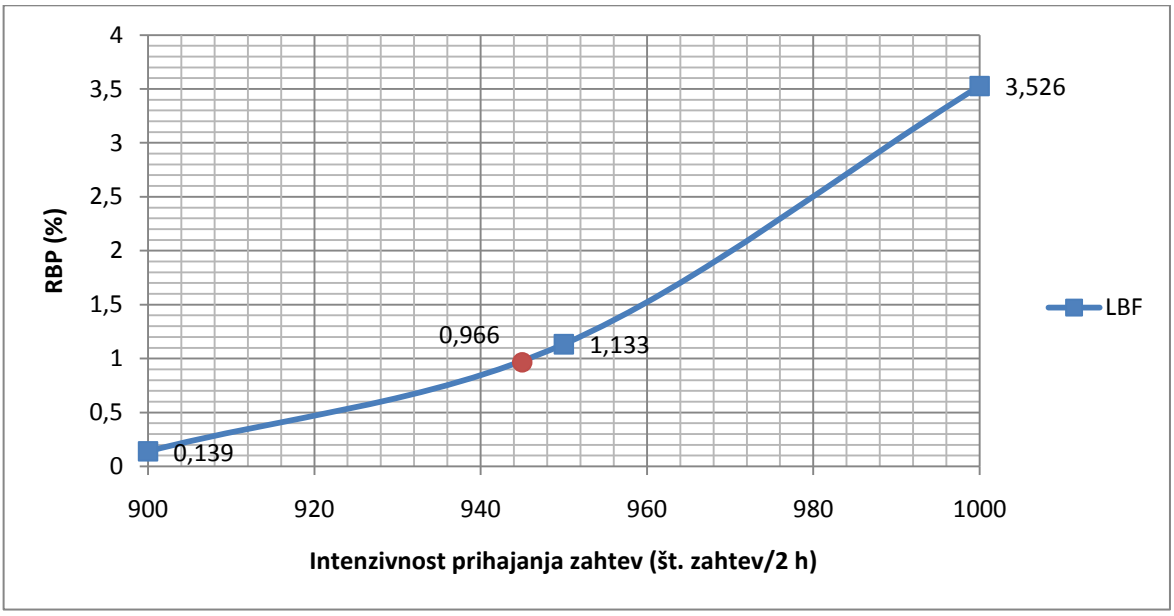
V 3. nizu poizkusov smo se odločili razpoloviti število diskov na 10 glede na sistem iz 1. niza poizkusov, in sicer z uporabo večjih 1-TB diskov. Tako smo za posamezen disk podvojili velikost prostora za shranjevanje (glej tabelo 8.5). Spomniti velja, da se je z uporabo polovičnega števila (dvakrat dražjih) diskov, razpolovilo tudi skupno število vseh logičnih kanalov. Ta varianta sistema je zato slabša od tiste, ki smo jo uporabili v 1. nizu poizkusov, saj ima sistem pri enako velikem skupnem prostoru za shranjevanje datotek (ter enaki ceni) na voljo le polovično število logičnih kanalov za strežbo uporabniških zahtev. Čeprav smo slednjo ugotovitev upoštevali že pri načrtovanju sistema, smo s 3. nizom poizkusov želeli preveriti, če je sistem s polovičnim številom logičnih kanalov pri istem primerku replikacije datotek **n** za M = 1000 sposoben pri polovični λ nuditi enako kvaliteto storitve, kar naj bi se kazalo v enakem RBP. Predpostavili smo, da bi lahko bili rezultati nekoliko slabši, saj pri manj diskih obstaja manj možnih kombinacij za izravnavo bremen. Rezultati, ki smo jih prikazali na grafu 8.5, so pri λ = 950 zahtev/2 h dali malo višji RBP kot rezultati, ki smo jih pridobili pri 1. nizu poizkusov pri λ = 1900 zahtev/2 h. Ugotovili smo, da lahko sistem nudi ustrezno kakovost storitev pri λ = 945 zahtev/2 h pri približno 90% zasedenosti resursov. Pri podvojenen diskovnem podsistemu VOD iz 3. niza poizkusov bi lahko tako sprejeli približno

10 zahtev/2 h manj kot pri sistemu iz 1. niza poizkusov. Takšno odstopanje je precej majhno, skoraj zanemarljivo, kar lahko utemeljimo s tem, da nudi 10 diskov za izbran primerek replikacije datotek še vedno dovolj kombinacijskih možnosti za enakomerno izravnavo bremen, poleg tega pa je majhna velikost datotek v primerjavi s pomnilniško velikostjo diska ugodna za porazdeljevanje bremena po diskih. Več manjših datotek je namreč dosti lažje enakomerno porazdeliti, kot manj večjih.

Ponavadi želimo, da lahko VOD sistem sočasno sprejme čim več uporabniških zahtev, zato je bolje, da diskovni podsistem (pri določeni vrednosti $J \cdot C$) sestavljajo manjši diski (glede na velikost pomnilniškega prostora), saj le-ti ponujajo več logičnih kanalov na enoto pomnilniškega prostora kot večji diski. Najboljše razmerje nudijo (dragi) polprevodniški diski. Na ta način se lahko znatno poveča prepustnost sistema. Če bi želeli imeti pri določenem številu logičnih kanalov na voljo čim več pomnilniškega prostora za pestrejšo ponudbo video vsebin, potem bi bili v tem primeru priročnejši večji in cenejši trdi magnetni diski.

Tabela 8.5 Določitev parametrov za 3. niz poizkusov.

Parametri
$M = 1000$
$\boldsymbol{J = 10}$
$\boldsymbol{C = 138}$
$N = 104$
$L_m = 1\ (= 7{,}4\ GB)$, $\forall m \in \mathcal{F}$
$1/\mu_m = 1\ (= 2\ h)$



Graf 8.5 Rezultati izvajanja 3. niza poizkusov za CLB-LBF sistem.

4. NIZ POIZKUSOV

S 4. nizom poizkusov smo želeli preveriti obnašanje sistema v primeru, da je povprečna dolžina filma 90 min in ne 120 min, kot je veljalo pri CLB-LBF sistemu iz 1. niza poizkusov. Povprečno trajanje filmov je precej odvisno od filmskega žanra. Tako so npr. komedije, dokumentarni filmi ter animirani filmi v povprečju krajši kot npr. zgodovinski filmi in trilerji. Ta ugotovitev nam daje motivacijo, da preverimo delovanje sistema pri krajši povprečni dolžini filma. Ker smo ohranili način kodiranja video vsebin, se je zaradi krajšega trajanja filmov zmanjšala tudi povprečna velikost filmskih datotek. Novo velikost datotek in nov povezavni čas prikazuje tabela 8.6(a).

Kot do sedaj smo rezultate izvajanja poizkusov primerjali z rezultati, ki smo jih pridobili za CLB-LBF sistem iz 1. niza poizkusov. Predpostavili smo, da se bo krajše trajanje filmov izražalo v večji prepustnosti sistema, kar se je izkazalo za resnično. Rezultati izvajanja 4. niza poizkusov (glej graf 8.6) so pokazali, da se meja še sprejemljive kakovosti storitve VOD nahaja pri λ = 2500 zahtev/2 h, kar pomeni, da lahko sedaj sistem v časovnem okviru 2 h sprejme 31˙6% strank več kot prvoten sistem (600 zahtev/2 h več). Na tem mestu omenimo, da se je pri ugotavljanju omenjene vhodne intenzivnosti grafična metoda zopet izkazala za zelo enostavno in učinkovito.

Ker smo v modelu sistema za 4. niz poizkusov uporabili manjšo povprečno velikost filmskih datotek, je zasedenost diskovnega prostora pri M = 1000 znašala 72˙9%. To nam je omogočalo, da smo lahko sistem dopolnili z novimi video vsebinami, podobno kot smo to storili pri sistemu iz 2. niza poizkusov. Sistem smo se odločili dopolniti s 400 novimi filmskimi naslovi, kar znaša 40% filmskih vsebin več kot pri prvotnem sistemu (glej tabelo 8.6(b)). Določiti smo morali tudi nov primerek replikacije datotek **n** za M = 1400 (tabela 8.6(c)). Z dodanimi datotečnimi kopijami smo dosegli 100% zasedenost diskovnega prostora.

Tabela 8.6 Določitev parametrov za 4. niz poizkusov.
(a) Manjša velikost filmskih datotek in krajši povezavni čas.
(b) Povečanje števila filmskih naslovov za 400.
(c) Določitev primerka replikacije datotek $\mathbf{n} = (n_1, n_2, \ldots, n_M)$ za $M = 1400$.

(a)

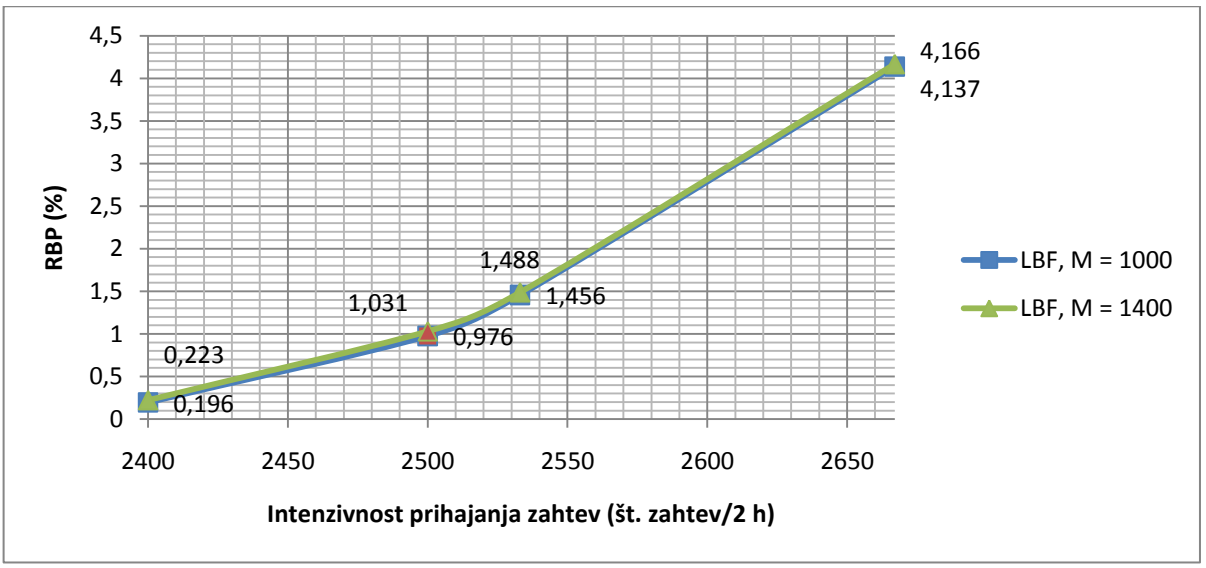
Parametri
$M = 1000$
$J = 20$
$C = 69$
$N = 104$
$\boldsymbol{L_m = 0,75\ (= 5,4\ GB)}$, $\boldsymbol{\forall m \in \mathcal{F}}$
$\boldsymbol{1/\mu_m = 0,75\ (= 1,5\ h)}$

(b)

Parametri
$\boldsymbol{M = 1400}$
$J = 20$
$C = 69$
$N = 104$
$\boldsymbol{L_m = 0,75\ (= 5,4\ GB)}$, $\boldsymbol{\forall m \in \mathcal{F}}$
$\boldsymbol{1/\mu_m = 0,75\ (= 1,5\ h)}$

(c)

Filmski naslovi (m)	**Število kopij (n_m)**
1-2	7
3-5	6
6-10	5
11-25	4
26-60	3
61-338	2
339-1400	1



Graf 8.6 Rezultati izvajanja 4. niza poizkusov za CLB-LBF sistem.

Rezultati izvajanja 4. niza poizkusov za CLB-LBF sistem z dodanimi filmskimi naslovi se zelo dobro prilagajajo rezultatom sistema brez dodanih filmskih vsebin, meja še sprejemljive kakovosti storitve (1% RBP) pa se še vedno giblje okoli λ = 2500 zahtev/2 h (glej graf 8.6). Do večjega zavračanja zahtev torej ni prišlo, kar kaže na dobro bremensko izravnanost dopolnjenega sistema. V 4. nizu poizkusov smo z uporabo krajših (90 min trajajočih) filmov povečali število možnih sočasnih dostopov strank v časovnem okviru 2 h pri enaki kakovosti VOD storitve glede na CLB-LBF sistem iz 1. niza poizkusov, s tem pa se je povečala tudi prepustnost sistema.

DODATEN KOMENTAR K REZULTATOM

Namen izvajanja različnih simulacijskih poizkusov je bil pridobiti uporabne informacije glede obnašanja modeliranega sistema pri različnih konfiguracijah diskovnega podsistema VOD. Pri modeliranju nas niso zanimale absolutne vrednosti dobljenih rezultatov, ker bi se rezultati realnega sistema gotovo razlikovali od tistih, ki smo jih pridobili z izvajanjem simulacijskih poizkusov, saj v simulacijskem modelu nastopajo določene potrebne poenostavitve. Zanimala so nas predvsem razmerja med pridobljenimi rezultati, ki so kazala na učinkovitost različnih konfiguracij. Simulacijski model se je izkazal za ravno prav natančnega in kompleksnega. Z njim smo uspeli pridobiti željene informacije o zastavljenem sistemu, hkrati pa smo sprejeli ravno toliko poenostavitev, kot je bilo potrebno. Modela sistema nismo nepotrebno zapletali z modeliranjem različnih velikosti filmskih datotek (uporabili smo povprečno vrednost) in različnih stalnih bitnih hitrosti kodiranja (izbrali smo eno za vse filme), saj bi to vsekakor zameglilo vpliv ostalih opazovanih parametrov na rezultate in otežilo analizo zmogljivosti sistema. Namen simulacije je bil namreč preveriti učinkovitost kombinacijske izravnave bremen (CLB) v diskovnem podsistemu VOD in ne načrtovanje nadgradnje nekega obstoječega realnega sistema. V primeru nadgradnje realnega sistema bi bil kompleksnejši in

natančnejši model povsem na mestu, saj bi bili premalo natančni oz. preveč poenostavljeni rezultati neuporabni za planiranje kapacitet novega sistema ali uglaševanje obstoječega. V praksi se npr. profili priljubljenosti filmov, da zajamejo spremenljivost povpraševanja uporabnikov, redno posodabljajo, kar bi bilo v modelu realnega sistema potrebno upoštevati. Ker podatkov realnega sistema nismo imeli na voljo, smo v modelu upoštevali ustrezne porazdelitve. Načrtan simulacijski model je zelo prilagodljiv za dodajanje kompleksnosti in novih resursov. Prilagodljivost modela je posledica tega, ker smo večino funkcionalnosti realizirali s programsko kodo v izrazih aktivnosti in manj z gradniki Simprocessa. Če želimo npr. dodati nov disk v sistem, lahko to hitro in enostavno naredimo z vstavitvijo novega procesa, ki ponazarja željen disk, in s spremembo nekaterih parametrov (globalnih spremenljivk). Prav tako bi lahko enostavno dodali generator zahtev, ki bi predstavljal generiranje zahtev z dodatno stalno bitno hitrostjo, tako da bi model opisoval sistem z dvema stalnima bitnima hitrostma prenosa podatkov – npr. ena bi bila za SDTV in druga za HDTV.

Simulacijski rezultati 1. niza poizkusov so za zasnovan sistem (s 1000 filmskimi naslovi in 20 500-GB diski) potrdili predpostavko, da se med izčrpnimi shemami izbire resursov bolje odreže LBF shema v primerjavi z RRT shemo. Dobili smo oceno, da je lahko CLB-LBF sistem za približno 6˙5% bolj obremenjen kot CLB-RRT sistem pri maksimalni obremenitvi, pri kateri je še izpolnjen pogoj sprejemljive kakovosti storitve (RBP $\leq$ 1%). CLB-LBF sistem daje torej večjo prepustnost kot CLB-RRT sistem pri določeni intenzivnosti prihajanja zahtev λ. Izkazalo se je, da se meja še sprejemljive kakovosti storitve za CLB-LBF sistem nahaja pri visoki 90% povprečni zasedenosti resursov oz. logičnih kanalov (pričakovali smo, da se bo ta meja nahajala pri zasedenosti logičnih kanalov nad 80%). Iz rezultatov je razvidna zelo enakomerna povprečna obremenjenost diskov, kar potrjuje predpostavko o učinkovitosti kombinacijske izravnave bremen. Pri CLB-RRT sistemu je slabša izravnava bremen, ki je posledica delovanja RRT algoritma (naključne izbire prostih diskov), rezultirala v slabši prepustnosti sistema, kar se je odrazilo v večjem standardnem odklonu σ od povprečne obremenjenosti diskov v primerjavi z σ pri CLB-LBF sistemu. Nadaljnji poizkusi so pokazali, da se da propustnost sistema povečati še na tri načine, in sicer z dodajanjem novih diskov, z uporabo več manjših diskov (glede na velikost pomnilniškega prostora), ki nudijo več logičnih kanalov na enoto pomnilniškega prostora, ter s hrambo krajših filmov (glede na trajanje) v diskovnem podsistemu VOD.

Pri 2. nizu poizkusov smo predpostavili, da je naš sistem že precej obremenjen, zato smo ga želeli nadgraditi z dvema dodatnima diskoma. Hoteli smo povečati število možnih sočasnih dostopov strank ter razširiti ponudbo filmskih vsebin pri ohranjeni kakovosti storitve. Z 10% povečanjem števila logičnih kanalov in velikosti diskovnega prostora smo dosegli 10% povečanje števila možnih sočasnih dostopov strank, kar je izpolnilo naša pričakovanja. Z novim primerkom replikacije datotek **n** smo uspeli povečati število filmskih naslovov v sistemu za 15% (na 1150), kar je glede na povečanje diskovnega prostora ugoden rezultat.

S 3. nizom poizkusov smo želeli preveriti obnašanje CLB-LBF sistema s polovičnim številom dvakrat večjih diskov (velikosti 1 TB). Hoteli smo ugotoviti, ali je sistem s polovičnim številom logičnih kanalov pri istem primerku replikacije datotek **n** za M = 1000 sposoben pri

polovični vhodni intenzivnosti λ nuditi enako kvaliteto storitve kot prvoten sistem, kar naj bi se kazalo v enakem RBP. Predpostavili smo, da bi lahko bili rezultati nekoliko slabši, saj pri manj diskih obstaja manj možnih kombinacij za izravnavo bremen. Rezultati so res pokazali malo večji RBP, vendar je bilo odstopanje zanemarljivo. To smo utemeljili z ugotovitvijo, da nudi 10 diskov za izbran primerek replikacije datotek še vedno dovolj kombinacijskih možnosti za enakomerno izravnavo bremen, poleg tega pa je majhna velikost datotek v primerjavi s pomnilniško velikostjo diska ugodna za porazdeljevanje bremena po diskih, saj je namreč več manjših datotek dosti lažje enakomerno porazdeliti, kot manj večjih. Tudi v tem primeru so bili diski pri meji še ustrezne kakovosti storitve v povprečju 90% (in enakomerno) obremenjeni. Opozoriti velja, da takšnega rezultata ne gre posploševati za vsak CLB-LBF sistem. Če je razmerje med velikostjo filmskih datotek in velikostjo posameznega diska v sistemu za razmeščanje datotečnih kopij na diske manj ugodno, tako da lahko na vsak disk razmestimo le manjše število velikih datotek, potem je lahko rezultat tudi slabši. Za primer lahko damo CLB-LBF sistem z 20 450-GB diski (npr. Hitachi Ultrastar 15K450 [18]), ki lahko hranijo 200 dveurnih Blu-ray filmov z velikostjo datoteke 32 GB in stalno bitno hitrostjo 36 Mbit/s. Tak primer smo navedli zato, ker smo za navedena razmerja parametrov že pridobili simulacijske rezultate pri fazi potrjevanja simulacijskega modela (delno jih prikazuje graf 8.2). Pri meji še zadovoljive kakovosti storitve (RBP <= 1%) je bila povprečna obremenjenost diskov 79˙07%, standardni odklon σ od povprečne obremenjenosti pa 0˙720183, kar kaže na slabšo izravnavo bremen kot pri CLB-LBF sistemu iz 1. niza poizkusov.

Namen 4. niza poizkusov je bil preveriti obnašanje CLB-LBF sistema v primeru, da filmi v povprečju trajajo 90 min in ne 120 min, kot je veljalo v prejšnjih primerih. Ugotovili smo, da lahko takšen sistem postreže 31˙6% uporabniških zahtev v 2 h več kot prvotni sistem iz 1. niza poizkusov. Ker diskovni prostor zaradi manjše povprečne velikosti filmskih datotek ni bil v celoti zaseden, smo lahko sistem s pomočjo novega primerka replikacije datotek **n** dopolnili s 400 novimi filmskimi naslovi, kar predstavlja 40% novih filmskih vsebin. Ta rezultat nas je pozitivno presenetil, saj bi ga brez uporabe programskega orodja težko napovedali. Prazen pomnilniški prostor smo uspeli dopolniti z dodatnimi filmskimi vsebinami brez poslabšanja kakovosti storitve, za kar je zaslužna hevristična metoda za alokacijo datotek, ki odlično izravna breme v sistemu.

Simulacija kombinacijske izravnave bremen je povsem izpolnila naša pričakovanja, saj smo z njo pridobili želene informacije o obnašanju diskovnega podsistema VOD pri različnih parametrih. Simulacijski rezultati so potrdili postavljene predpostavke, z njimi pa smo pridobili tudi ocene zmogljivosti sistema, ki bi jih bilo zaradi kompleksnih interakcij v sistemu težko ali nemogoče napovedati. Pri simulaciji razmeščanja datotek z repliciranjem smo za dosego kombinacijske izravnave bremen pri različnih konfiguracijah sistema predvideli za približno tretjino več datotečnih kopij, kot bi jih vseboval sistem brez repliciranja datotek. To je zadostovalo, da smo uspeli sestaviti simulacijski model enakomerno obremenjenega diskovnega podsistema VOD, ki je potrdil izredno učinkovitost hevristične metode za alokacijo datotek.

9 Povzetki in sklepne ugotovitve

Storitve VOD so se v razvitih državah že precej razširile in predstavljajo priljubljen način preživljanja prostega časa. Video na zahtevo predstavlja vrhunec na področju multimedije, ki zahteva zelo hiter prenos podatkov ter izvajanje v realnem času. Zaradi njene zahtevnosti potrebujemo zelo zmogljive računalnike (video strežnike) ter posebej prilagojene operacijske sisteme, ki podpirajo rokovanje z multimedijo. V diplomskem delu smo se orientirali predvsem na video strežnike in njihove operacijske sisteme, oziroma konkretneje - na algoritme, ki jih uporabljajo multimedijski operacijski sistemi za razvrščanje procesov, za razvrščanje zahtev za disk ter za razmeščanje filmskih datotek po diskih. Slednjemu problemu smo se še posebej posvetili in v praktičnem delu realizirali simulacijski program kombinacijske izravnave bremen, ki predstavlja enega izmed učinkovitejših načinov razmeščanja datotek.

Multimedija potrebuje realnočasovno razvrščanje procesov, da zadosti svojim skrajnim rokom. Navadno se uporabljata dva algoritma. Prvi je razvrščanje po naraščajoči periodi (RMS), ki je statični prekinjevalni algoritem, ki procesom priredi fiksne prioritete glede na njihove periode. Drugi je razvrščanje po najbližjih skrajnih rokih (EDF), ki je dinamičen algoritem, ki vedno izbere proces z najbližjim skrajnim rokom. Drugi algoritem je kompleksnejši, vendar lahko doseže 100% izkoristek, kar prvi ne more. Ugotovili smo, da je na račun manjše kompleksnosti bolje izbrati RMS, če je izkoristek CPE pod mejo, kjer deluje RMS, drugače je potrebno izbrati EDF.

Multimedijski datotečni sistemi običajno uporabljajo potisni pretočni model namesto poteznega. Ko začne tok podatkov na zahtevo uporabnika teči, prihajajo biti z diska brez nadaljnjih uporabniških zahtev. Ta pristop se bistveno razlikuje od pristopa, ki ga uporabljajo tradicionalni operacijski sistemi, vendar je potreben, da zadosti zahtevam v realnem času.

Pri razvrščanju zahtev za disk je predvidljivost tista lastnost multimedije, ki diskom olajšuje rokovanje v primerjavi z običajnimi tekstovno orientiranimi aplikacijami. Torej breme, ki ga vsak aktivni tok naloži sistemu, je dobro definirano in predvidljivo, zato lahko sistem zahteve optimalno posortira in jih zatem v optimalnem vrstnem redu obdela. Pri statičnem algoritmu, kjer smo predpostavili, da imajo vsi tokovi enako ločljivost, hitrost sličic in ostale lastnosti, potrebujemo dvojno medpomnenje v strežniku, da ohranimo gladek tok podatkov do strank. Pri dinamičnem algoritmu smo predpostavko o enakem bremenu tokov ovrgli, zagotoviti pa smo hoteli še dvema kriterijema – minimizirati absolutni iskalni čas ter minimizirati možnost zgrešitve skrajnega roka. Odgovor na zahteve je prebirni-EDF algoritem, ki kopiči zahteve, katerih skrajni roki so si blizu, v serije in jih procesira v cilindrskem vrstnem redu.

Datoteke so lahko shranjene strnjeno ali nestrnjeno. V slednjem primeru je lahko enota spremenljive dolžine (en blok sestavlja ena sličica) ali fiksne dolžine (en blok sestavlja več sličic). Ti pristopi vodijo do različnih kompromisov, ki se nanašajo na porabo RAM-a med

predvajanjem posnetka, potrato diskovnega prostora in izgubo učinkovitosti med predvajanjem zaradi dodatnih iskanj.

Razporeditev datotek na disku vpliva na učinkovitost. Če se na disku nahaja več datotek, se včasih uporabi algoritem orgelskih piščali. Za izboljšanje učinkovitosti se pogosto uporabljajo tudi strategije predpomnjenja blokov in datotek. Za razmeščanje filmskih datotek na več diskov se lahko uporabi razdeljevanje datotek (široko ali ozko) ali pa repliciranje datotek. Pri visoko interaktivnih VOD sistemih pride pri pristopu z razdeljevanjem, ko so zakasnitve pri interakcijah uporabnikov s sistemom omejene na manj kot sekundo, do precejšnjega zmanjšanja propustnosti, zato je potrebno pri video strežnikih, ki zagotavljajo visoko stopnjo interaktivnosti, uporabiti razmeščanje z repliciranjem datotek.

V okviru slednjega smo podrobno obravnavali kombinacijsko izravnavo bremen (CLB), ki je ena izmed učinkovitejših shem dodeljevanja datotek. Za izbiro diskovnih virov uporablja enega izmed treh mehanizmov: SRT, RRT ali LBF, izmed katerih sta zadnja dva izčrpna, zadnji pa je najučinkovitejši. Faktor delitve diskovnih virov za promet filmov z več kopijami ima velik vpliv na verjetnost blokiranja oz. zavrnitve zahteve (RBP) v VOD sistemu. CLB predpostavlja, kako izravnati prometno breme filmov z več kopijami med kombinacijskimi skupinami diskov, da je stopnja delitve diskovnih virov maksimalna. Za dani primerek replikacije datotek CLB v splošnem daje efektivno spodnjo mejo za RBP sistema. Problem iskanja primerka alokacije datotek, ki doseže CLB, je za izbrani primerek replikacije datotek NP-poln. Za učinkovito hevristično rešitev je zato potrebna uporaba požrešne metode za alokacijo datotek, katere namen je doseči tako enakomerno delitev virov za promet filmov z več kopijami, kakor tudi enakomerno porazdelitev prometa filmov z eno kopijo.

Cilj praktičnega dela diplomske naloge je bil narediti uporaben simulacijski program kombinacijske izravnave bremen za diskovni podsistem VOD, ki bi služil načrtovanju VOD sistema s široko ponudbo filmskih vsebin (s 1000 različnimi filmskimi naslovi). Simulacijo smo uspešno realizirali s simulacijskim orodjem Simprocess, vhodne podatke pa smo pridobili z javanskim programom, v katerem smo implementirali požrešno metodo za hevristično alokacijo datotek. Simulacijski model je uspešno prestal fazi verifikacije in validacije. Izvajanje poizkusov nam je poleg potrditve postavljenih predpostavk dalo podrobnejši vpogled v delovanje diskovnega podsistema VOD, dobili pa smo tudi dobro oceno izkoriščenosti diskovnih resursov pri maksimalni obremenitvi, pri kateri je še izpolnjen pogoj sprejemljive kakovosti storitve (RBP $\leq$ 1%). Za zasnovan CLB-LBF sistem so rezultati npr. pokazali, da je meja še sprejemljive kakovosti storitve dosežena pri visoki 90% povprečni zasedenosti logičnih kanalov. Različne konfiguracije diskovnih podsistemov, ki smo jih obravnavali pri izvajanju poizkusov, ter pripadajoč simulacijski model lahko služijo za načrtovanje ustrezno dimenzioniranega sistema VOD ter za njegovo uglaševanje. Če sistemska informacija o trenutni zasedenosti resursov, ki jo zahteva LBF algoritem, v realnem sistemu ni na voljo, se lahko namesto LBF uporabi RRT algoritem, ki je manj učinkovit, a še vedno lahko povsem zadovoljiv.

Simprocess se je izkazal kot primerno simulacijsko orodje za zastavljeno simulacijo diskretnih dogodkov kombinacijske izravnave bremen. Večjih težav pri uporabi orodja ni bilo. Omejitve univerzitetne licence Simprocessa (največ 50 modeliranih aktivnosti in procesov) nas pri načrtovanju niso ovirale, saj smo za diskovni podsistem VOD s 1000 filmskimi naslovi potrebovali gradnike le za 20 diskov. Pri obsežnejših modelih bi težave bržkone nastopile in bi bila potrebna uporaba profesionalne različice Simprocessa. Pomankljivost Simprocessa, ki smo jo med drugim zasledili, je predstavitev realnih števil v enojni natančnosti. Pri zelo veliki bazi različnih filmskih naslovov enojna natančnost ne bi zadostovala za predstavitev vseh profilov priljubljenosti filmov. Kot alternativo Simprocessu bi lahko uporabilili kakšno drugo simulacijsko orodje, npr. OMNeT++, NS-2 ipd.

Čeprav dobimo s simulacijsko metodo vpogled v zapletene interakcije med uporabniškimi zahtevami po filmih z več kopijami in v izbire diskovnih resursov za strežbo teh zahtev, je problem simulacijskega pristopa v primerjavi z analitičnim večja časovna kompleksnost. Da so se rezultati izvajanja simulacijskih poizkusov ustalili, smo morali čakati tudi po več ur. Za hitro in učinkovito oceno spodnje meje RBP LBF sistema bi bila zato na mestu uporaba analitične metode, ki je znana kot Erlangova aproksimacija fiksne točke (angl. Erlang's fixed-point approximation), na sistemu enačb iz [7, 11]. Izračun suboptimalne rešitve se lahko še nekoliko pohitri z uporabo evolucijskega optimizacijskega programa, predlaganega v [10].

Realiziran simulacijski model je zelo prilagodljiv in se ga da s spreminjanjem parametrov enostavno prilagoditi za različne konfiguracije diskovnega podsistema VOD. Z manjšimi spremembami se ga da uporabiti tudi kot model za druge, VOD-u podobne sisteme, kot so npr. igre na zahtevo, določeni načini e-izobraževanja, razne storitve v okviru računalništva v oblaku (angl. Cloud Computing) in podobno.

Seznam slik

Seznam grafov

Seznam tabel

Seznam literature in virov

KNJIGE

[1] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*, New York: Wiley-Interscience, 1991.

[2] W. Simpson, H. Greenfield, *IPTV and Internet Video: Expanding the Reach of Television Broadcasting*, Oxford: Focal Press, 2007, pogl. 2, 3, 5.

[3] A. S. Tanenbaum, *Modern Operating Systems*, 3. izdaja, New Jersey: Upper Saddle River, 2007, pogl. 7.

[4] P. K. C. Tse, *Multimedia Information Storage and Retrieval: Techniques and Technologies*, Hershey, New York: IGI Publishing, 2008.

[5] N. Zimic, M. Mraz, *Temelji zmogljivosti računalniških sistemov*, 1. izdaja, Ljubljana: Fakulteta za računalništvo in informatiko, 2006.

ČLANKI

[6] P. Branch, G. Egan, B. Tonkin, "Modeling interactive behaviour of a video based multimedia system," *Proceedings of IEEE International Conference on Communications 1999 (ICC 99)*, Vancouver, BC, Kanada, zv. 2, str. 978-982, junij 1999.

[7] J. Guo, S. Chan, E. W. M. Wong, M. Zukerman, P. Taylor, K. S. Tang, "On blocking probability evaluation for video-on-demand systems," *Proceedings of 18th International Teletraffic Congress (ITC18)*, Berlin, Nemčija, zv. 5a, str. 211–220, september 2003.

[8] J. Guo, P. G. Taylor, E. W. M. Wong, S. Chan, M. Zukerman, K. S. Tang, "Performance Benchmarks for an Interactive Video-on-Demand System," *Proceedings of IEEE International Conference on Communications 2004 (ICC 04)*, Pariz, Francija, zv. 4, str. 2189-2193, junij 2004.

[9] J. Guo, P. Taylor, M. Zukerman, S. Chan, K. S. Tang, E. W. M. Wong, "On the efficient use of video-on-demand storage facility," *Proceedings of IEEE International Conference on Multimedia and Expo. 2003 (ICME 03)*, Baltimore, MD, ZDA, zv. 2, str. 329–332, julij 2003.

[10] J. Guo, Y. Wang, K. S. Tang, S. Chan, E. W. M. Wong, P. Taylor, M. Zukerman, "Evolutionary Optimization of File Assignment for a Large-Scale Video-on-Demand System," *IEEE Transactions on Knowledge Data Engineering*, št. 6, zv. 20, str. 836-850, junij 2008.

[11] J. Guo, E. W. M. Wong, S. Chan, P. Taylor, M. Zukerman, K. S. Tang, "Combination Load Balancing for Video-on-Demand Systems," *IEEE Transactions on Circuits and Systems for Video Technology*, št. 7, zv. 18, str. 937-948, julij 2008.

[12] M. Reisslein, K. W. Ross, S. Shrestha, "Striping for Interactive Video: Is it Worth it?," *Proceedings of IEEE International Conference on Multimedia Computing and Systems (ICMCS 99)*, Firence, Italija, zv. 2, str. 635-640, junij 1999.

SPLETNI VIRI (dostopno januarja 2010)

[13] *Attack the HDTV Bandwidth Challenge with a Powerful Technology: VBR/StatMux for Digital Broadcast, VOD and SDV*, Imagine Communications, februar 2008. Dostopno na:

- http://www.imaginecommunications.com/pdfs/whitepapers/Bandwidth%20Expansion_02202008.pdf

[14] HDMI. Dostopno na:

- http://en.wikipedia.org/wiki/HDMI
- http://www.hdmi.org/

[15] *Poročilo o razvoju trga elektronskih komunikacij za drugo četrtletje 2009*, Ljubljana: Agencija za pošto in elektronske komunikacije Republike Slovenije (APEK), september 2009. Dostopno na:

- http://www.apek.si/sl/2009
- http://www.apek.si/datoteke/File/2009/telekomunikacije/Kvartalno%20poro%C4%8Dilo%20Q2%202009.pdf

[16] Slovarji. Dostopno na:

- http://dis-slovarcek.ijs.si/
- http://www.islovar.org/iskanje_enostavno.asp?reset=true
- http://slovar.ltfe.org/
- http://www.ijs.si/cgi-bin/rac-slovar?w=load
- http://evroterm.gov.si/
- http://translate.google.com/#

[17] *Television across Europe Follow-up Reports 2008: Overview and country reports*, Budimpešta: Open Society Institute, 2008. Dostopno na:

- http://www.mediapolicy.org/tv-across-europe/follow-up-reports-2008-country/media-followup-full-web.pdf/view?searchterm=None

[18] Testi povprečnih bralnih prepustnosti diskov s spletne strani Tom’s hardware. Dostopno na:
- http://www.tomshardware.com/reviews/2tb-hdd-caviar,2261-7.html
- http://www.tomshardware.com/charts/2009-3.5-desktop-hard-drive-charts/h2benchw-3.12-Avg-Read-Throughput,1010.html
- http://www.tomshardware.com/charts/enterprise-hard-drive-charts/Average-Read-Transfer-Performance,701.html

[19] *Understanding and Using High-Definition Video, A Digital Video Primer*. Dostopno na:
- http://www.adobe.com/products/premiere/pdfs/hdprimer_0306.pdf

[20] Unicast in multicast način prenosa podatkov. Dostopno na:
- http://research.ncku.edu.tw/re/articles/e/20071123/5.html
- http://en.wikipedia.org/wiki/Multicast
- http://en.wikipedia.org/wiki/Unicast

[21] Uporabniški priročnik za simulacijsko orodje Simprocess. Dostopno na:
- http://www.simprocess.net/docs/SP45/SP40_UserGuide.pdf

[22] Uporabniški vmesnik SIOL videoteke. Dostopno na:
- http://itm.siol.net/doc/2009/4/01_videoteka_tipi_iskanja.jpg
- http://www.siol.net/pomoc_in_podpora/televizija/siol_tv/prenovljen_vmesnik_siol_tv_plus.aspx
- http://www.siol.net/televizija/videoteka.aspx

[23] Video na zahtevo. Dostopno na:
- http://www.itvdictionary.com/vod.html
- http://en.wikipedia.org/wiki/Video_on_demand