KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

Vidmantas Rimavičius

KOMPIUTERIŲ HIERARCHINĖS ATMINTIES SISTEMOS TYRIMAS

Magistro darbas

Vadovas
doc. S. Maciulevičius

KAUNAS, 2005

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

TVIRTINU
Katedros vedėjas
prof. E. Kazanavičius
2005-05-23

KOMPIUTERIŲ HIERARCHINĖS ATMINTIES SISTEMOS TYRIMAS

Informatikos mokslo magistro baigiamasis darbas

Kalbos konsultantė
Lietuvių k. katedros lekt.
dr. J. Mikelionienė
2005-05-16

Recenzentas
doc. dr. R. Marcinkevičius
2005-05-19

Vadovas
doc. dr. S. Maciulevičius
2005-05-18

Atliko
IFM 9/1 gr. stud.
V. Rimavičius
2004-05-17

KAUNAS, 2005

The study of computer hierarchical memory

SUMMARY

The operating speed of computers tends to increase significantly, however, this process is not simple. It can be explained, that operating speed depends on how fast the computer facilities are as well as their balance. Modern processors can perform operations within several cycles meanwhile the selection time of big size main memory reaches tens and hundreds of cycles. Although the static memory able to operate at speed equal or close to processor's operating speed exists, it's using for main memory is expensive. Problem is solved by installing the small size *cache* between processor and main memory.

Relatively small but very fast memory called *cache* takes a specific position in modern computer memories system. Cache is a highest level of hierarchical memories system. Cache simulator for exploring of cache behaviour was developed. Cache's influence on computer efficiency from both theoretical and practical point of view, the latter to be supported with simulation results, is analysed in this master thesis. Comparing the theoretical and test results the influence of different factors to the operation of hierarchical memories system is evaluated.

The results of cache simulation show that the operation of hierarchical memory system is impacted by functioning of cache levels, the frequence of accesses to the memory, the hit rate (or miss rate), the cache organisation, line replacement algorithm, cache size, cache line size as well as specific properties of program executed.

TURINYS

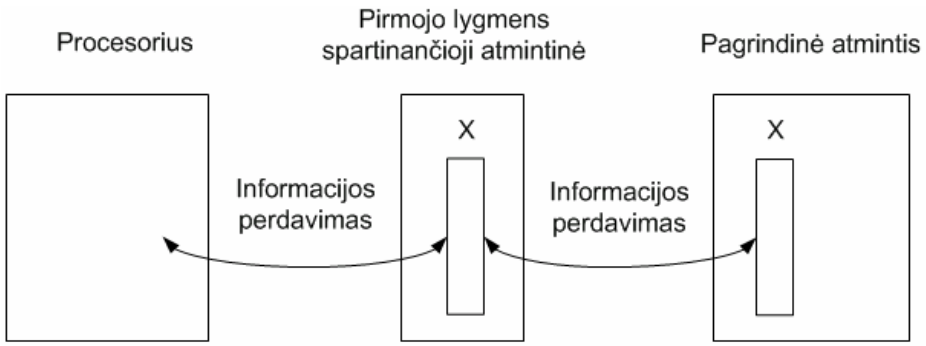
Lentelių sąrašas

Paveikslų sąrašas

ĮVADAS

Kompiuterių darbo sparta nuolat auga, tačiau šis procesas nėra paprastas. Tai paaiškinama tuo, kad darbo sparta priklauso beveik nuo visų kompiuterį sudarančių įtaisų spartos, nuo tinkamo jų subalansavimo.

Šiuolaikiniai procesoriai gali atlikti operacijas per keletą taktų, tuo tarpu didelės talpos pagrindinės atminties išrinkimo laikas siekia dešimtis ar net šimtus taktų. Nors ir egzistuoja tokia puslaidininkių atmintis, kuri gali dirbti sparta sulyginama su procesoriaus darbo sparta, tačiau ją panaudoti pagrindinei atminčiai neekonomiška. Problema sprendžiama tarp procesoriaus ir pagrindinės atminties įterpiant nedidelės apimties spartinančiosios atmintinės bloką, kaip parodyta 1.1 paveiksle.



1.1 pav. Hierarchinė atminties sistema

Spartinančiąją atmintinę sudaro labai sparti laisvai išrenkama atmintis (angl. *Random Access Memory - RAM*), kuri dirba procesoriui reikalinga sparta. Programos komandos ir duomenys yra perduodami į spartinančiąją atmintinę ir toliau jie patenka į procesorių. Paprastai originalios programos ir duomenys laikomi pagrindinėje atmintyje ir siekiama, kad pagrindinės atminties turinys atitiktų spartinančiojoje atmintinėje esančias komandų ir duomenų kopijas (1.1 paveiksle parodytas objektas X). Bet kurie procesoriuje pakeisti duomenys pirmiausia įrašomi į spartinančiąją atmintinę, į pagrindinę atmintį įrašant juos tuo pat metu arba atidedant įrašymą vėlesniam laikui, kai atitinkamą spartinančiosios atmintinės bloką (eilutę) reikia pakeisti nauja informacija.

Spartinančiosios atmintinės panaudojimo naudingumo kompleksinį įvertinimą galima gauti naudojant specialias testines programas (pvz.: *RightMark Memory Analyzer, SiSoftware Sandra 2005*) [2, 19, 20]. Dažnai gautus rezultatus tenka perskaičiuoti, jei norime juos pateikti įprastais pralaidumo vienetais (*MB/s, GB/s*). Be to, testinės programos neleidžia keisti spartinančiosios atmintinės parametrų

(spartinančiosios atmintinės tipą, krypčių skaičių, lygmenų talpos ir pan.), norint patikrinti jų įtaką bendrajam hierarchinės sistemos našumui. Tam reikėtų panaudoti hierarchinės atminties imitatorių.

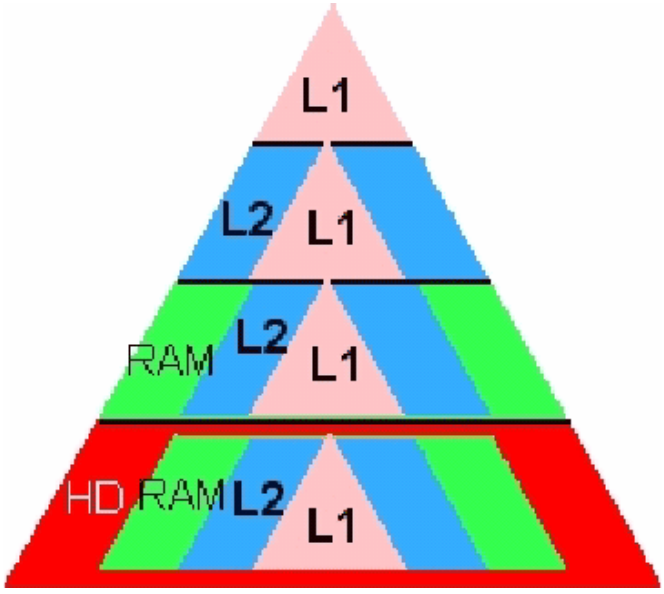
Šiame darbe bus analizuojama spartinančiosios atmintinės įtaka kompiuterio našumui tiek teoriniu požiūriu, tiek praktiniu, pastaruoju atveju bus pateikti našumo testų rezultatai. Gretindami teorinius ir testų rezultatus, įvertinsime įvairių faktorių įtaką hierarchinės atminčių sistemos darbui.

Skaitytas pranešimas tarpuniversitetinėje konferencijoje *Informacinės technologijos 2005* , kuri vyko Kauno Technologijos Universitete 2005 m. balandžio 29 d. Publikuotas straipsnis leidinyje *INFORMACINĖS TECHNOLOGIJOS 10-oje tarpuniversitetinė magistrantų ir doktorantų konferencija* tema „*Kompiuterių hierarchinės atminties sistemos tyrimas*“ (*žr.* 1 priedą) [14].

1. KOMPIUTERIŲ HIERARCHINĖS ATMINTIES ANALIZĖ

1.1. Kompiuterių atminčių hierarchinė sistema

Bet kurį kompiuterį sudaro trys pagrindinės dalys: procesorius, atmintis ir periferinė įranga. Šios dalys tarpusavyje sujungtos magistralėmis . Kompiuterių atminčių hierarchinę sistemą sudaro: procesoriuje įmontuota pirmojo ir antrojo lygmens spartinančiosios atmintinės (žymimos *L1* ir *L2*), pagrindinėje plokštėje sumontuota pagrindinė (darbinė) atmintis (žymima *RAM*) ir magnetinių kaupiklių atmintis (angl. *Hard Drive*, žymima *HD*) [16, 22, 27]. Žemiau pateiktame paveiksle parodyta kompiuterio atminčių hierarchinė struktūra.



1.2 pav. Kompiuterių hierarchinės atminties sistemos struktūra

Kompiuterių hierarchinės atminties sistemos struktūrą galima aprašyti formule:

$$MH = \{N_i\} \cup MV, \quad i = \overline{1,n} \qquad \textbf{(1.1)}$$

Kur N - atminčių aibė, $|N| = n$ - atminčių lygių skaičius, MV - atminties hierarchijos (angl. *Memory Hierarchy –MH*) valdymo mechanizmai.

Procesorius yra pagrindinis mazgas, kurio užduotis - vykdyti programos kodą. Šiuolaikinėje procesorių architektūroje toje pačioje mikroschemoje kartu su centriniu procesoriumi (angl. *Central Processing Unit - CPU*) yra montuojama ir spartinančioji atmintinė (angl. *Cache*). Procesoriuje yra nemažas skaičius registrų - bendrosios paskirties, komandų rodyklės, požymių (vėliavėlių), segmentų, sistemos valdymo [6]. Jų darbo sparta prilygsta procesoriaus spartai, nes jie realizuoti tame pačiame

kristale. Žemiau pateiktoje lentelėje pateikti duomenys apie kai kuriuos Intel procesorius: adresų ir duomenų magistralių pločiai, spartinančiosios atmintinės dydis, koprocesoriaus buvimas ir procesoriaus dažnis [10, 11].

1.1 lentelė. Intel procesorių charakteristikos

CPU	Magistralė		Spartinančioji atmintinė, KB	Koprocesorius	Dažnis, MHz
	Duomenų, bitais	Adresų, bitais			
8086	16	20	-	Nėra	4,77÷8
286	16	24	-	Nėra	6÷25
386SX	16	24	-	Nėra	16÷33
386DX	32	32	-	Nėra	25÷40
486SX	32	32	8	Nėra	16÷33
486DX	32	32	8	Yra	25÷50
486DX4	32	32	16	Yra	75÷120
Pentium	64	32	2x8	Yra	60÷200
Pentium MMX	64	32	2x16	Yra	166÷233
Pentium Pro	64	36	L1: 2x8 L2: 256	Yra	150, 166, 200
Pentium II	64	36	L1: 2x8 L2: 256	Yra	233, 266
Pentium III	64	36	L1: 2x16 L2: 256	Yra	500÷...
Pentium IV	64	36	L1: 16+96* L2: 512	Yra	2000÷...

*/ Šis procesorius turi trasų spartinančiąją atmintį, kurioje telpa 12000 mikrooperacijų. Manoma, kad tai atitinka 96 KB.

Lentelėje L1 reiškia spartinančiosios atmintinės pirmąjį lygmenį, o L2 – antrąjį. Lentelės stulpelyje „*spartinančioji atmintinė, KB*“ užrašas 2x8 reiškia, kad spartinančiojoje atmintinėje yra dvi atskiros 8 KB spartinančiosios atmintinės duomenims ir komandoms laikyti.

Bet kurių atminties posistemių parametrai yra šie :

- Talpa - saugomos informacijos kiekis. Didžiausia talpa pasižymi magnetinės juostos ir standieji diskai, po to seka pagrindinė atmintis.
- Išrinkimo laikas – vėlinimas nuo užklausos pateikimo iki informacijos išrinkimo. Mažiausia išrinkimo laiką turi spartinančioji atmintinė, po jos seka pagrindinė atmintis, standieji diskai, magnetinės juostos.
- Mainų sparta – apsikeitimo informacija sparta, perduodant ją srautu. Maksimalią mainų spartą turi spartinančioji atmintinė, po jos seka pagrindinė atmintis, standieji diskai.

- Informacijos laikymo kaina - sutarto kiekio (dažniausiai 1 MB) laikymo kaina. Mažiausia kaina saugomai informacijai turi juostiniai kaupikliai, juos vejasi diskiniai kaupikliai, o pati brangiausia yra spartinančioji atmintinė.

Išorinei atminčiai priskiriami įrenginiai, leidžiantys išsaugoti informaciją ilgesnį laiką. Šie įrenginiai gali naudoti skirtingus informacijos saugojimo principus: magnetinį, optinį. Charakteringas išorinės atminties bruožas yra tas, kad šie įrenginiai mainus atlieka informacijos blokais, o ne atskirais baitais ar žodžiais, kaip tai leidžia pagrindinė atmintis. Išorinių atminčių įrenginiams apibūdinti naudojami tokie parametrai:

- Pagrindinė įrenginių charakteristika yra talpa, matuojama megabaitais, gigabaitais ir terabaitais (MB, GB, TB).
- Išrinkimo laikas – tai laiko intervalas nuo pateiktos užklausos, kuomet reikia perduoti informacijos bloką, iki faktiškos perdavimo pradžios.
- Informacijos perdavimo sparta – nusako kokia sparta (MB/s) perduodama informacija.
- Kaupiklio sukimosi greitis. Skirtingų tipų kaupiklių sukimosi greičiai skiriasi. Magnetinių diskelių (angl. *Floppy Disk Drive - FDD*) sukimosi greitis yra 300 aps/min arba 360 aps/min. Standžiųjų diskų (angl. *Hard Disk Drive - HDD*) sukimosi greitis siekia 5400 aps/min, 7200 aps/min ir daugiau. Didesnis sukimosi greitis leidžia pasiekti ir didesnę informacijos mainų spartą [10, 11].

1.2 lentelė. Išorinių kaupiklių charakteristikos

Kaupiklio tipas	Talpa	Išrinkimo laikas, ms	Greitis, MB/s
FDD 5“ 360KB	360KB	100	0,027
FDD 5“ 1,2MB	1,2MB	100	0,045
FDD 3,5“	1,44MB	100	0,055
HDD IDE	20MB÷nx10GB	7,5÷40	0,2÷2,5
HDD SCSI	40MB÷nx10GB	7,5÷40	0,2÷2,0
CD-ROM 1-x	650MB	240	0,15
CD-ROM 4-x	650MB	240	0,6
CD-ROM 4-x	780M	200	0,7
CD-ROM 12-x	780MB	120	1,2
CD-ROM 24-x	780MB	120	3,0
Floptical	21MB	70	1,6
Iomega Zip	100MB	30	1,4
Iomega Jazz	1,1GB	12	5,5

Didelis kaupiklio sukimosi greitis susietas su kaupiklio tiksliu balansavimu.

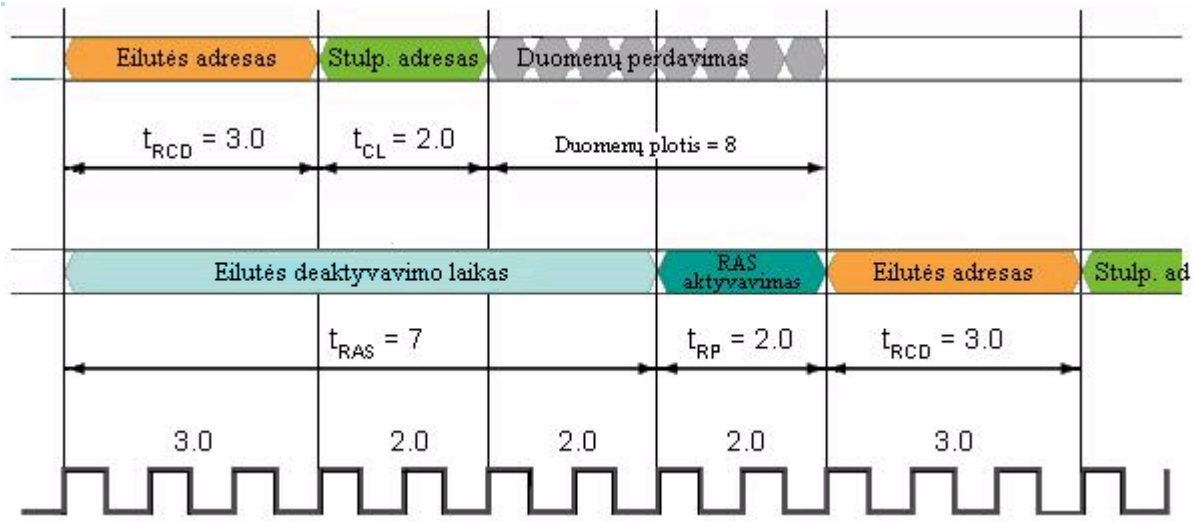
1.2. Atminties organizacija

Spartinančiosios atmintinės sparta priklauso nuo įvairių dalykų – jos talpos, organizavimo ypatumų, vykdomų programų charakteristikų.

Jeigu informacija perduodama iš pagrindinės atminties pirmiausia į spartinančiąją atmintinę ir tik po to naudojama procesoriuje, išrinkimo laikas būtų :

$$t_a = t_m + t_c , \qquad \textbf{(1.2)}$$

kur t_c - spartinančiosios atmintinės išrinkimo laikas, t_m - pagrindinės atminties išrinkimo laikas [25]. Pagrindinė atmintis yra charakterizuojama įvairiais vėlinimų laikais, kurie parodyti 1.3 paveiksle [10, 21 24].



1.3 pav. Pagrindinės atminties laikai

Aiškinant 1.3 paveiksle pateiktą diagramą reikia suvokti, kad pagrindinė atmintis organizuota kaip dvimatė matrica, kurioje išrinkus atitinkamą eilutę ir stulpelį, gausime atminties vietą su reikalinga informacija. Kiekvienos informacijos suradimas yra lydimas įvairiais laiko vėlinimais. 1.3 paveiksle parodyti sugaišti laikai taktais reikalingai informacijai surasti ir informacijai perduoti. Laikas t_{RCD} yra sugaištas laikas taktais pereinant nuo išrinktos eilutės prie stulpelio išrinkimo (angl. *Row Access Strobe to Column Access Strobe Delay*). Išrinkus atminties eilutę pereiname prie reikiamo stulpelio išrinkimo. Laikas t_{CL} - sugaištas laikas informacijai iš reikiamo stulpelio išrinkti (angl. *Column Access Strobe Latency*). Išrinkus atminties vietą toliau vyksta duomenų perdavimas. Laikas t_{RP} reikalingas, kad būtų

išrinkta kita eilutė ir kad pasibaigtų duomenų perdavimas (angl. *Row Access Strobe Prechange time*). Laikas t_{RAS} yra sugaištas laikas taktais pereinant nuo išrinktos eilutės prie naujos eilutės išrinkimo (angl. *Row Access Strobe*). Priklausomai nuo atminties tipo skiriasi ir laiko parametrai (taktų skaičius).

Akivaizdu, kad darbo sparta bus didesnė, jei reikalinga informacija (komandos ir duomenys) bus perkelta į spartinančiąją atmintinę. Tai atliekama pirmojo kreipinio į šią informaciją metu. Tikimybė, kad reikalingas žodis bus rastas spartinančiojoje atmintinėje, priklauso nuo vykdomos programos, spartinančiosios atmintinės dydžio ir struktūros. Paprastai 70-90% kreipinių metu reikalinga informacija randama spartinančiojoje atmintinėje [7, 13, 25]. Situacija, kai reikalingas žodis randamas spartinančiojoje atmintinėje, vadinama *pataikymu*; priešingu atveju sakoma, kad *nepataikoma* [9, 10, 25]. Pastaruoju atveju reikia kreiptis į pagrindinę atmintį.

Spartinančiosios atmintinės pataikymo dažnis h apibrėžiamas taip :

$$h = \frac{\textit{skaičius kreipinių, kai reikiami žodžiai randami spartinančiojoje atmintinėje}}{\textit{bendras kreipinių skaičius}} \qquad \textbf{(1.3)}$$

Įvertinę pataikymo dažnį, vidutinį informacijos išrinkimo iš hierarchinės atminties sistemos laiką galime išreikšti taip:

$$t_a = (1-h)t_m + ht_c, \qquad \textbf{(1.4)}$$

čia t_c - spartinančiosios atmintinės išrinkimo laikas, o t_m - pagrindinės atminties išrinkimo laikas [25, 26].

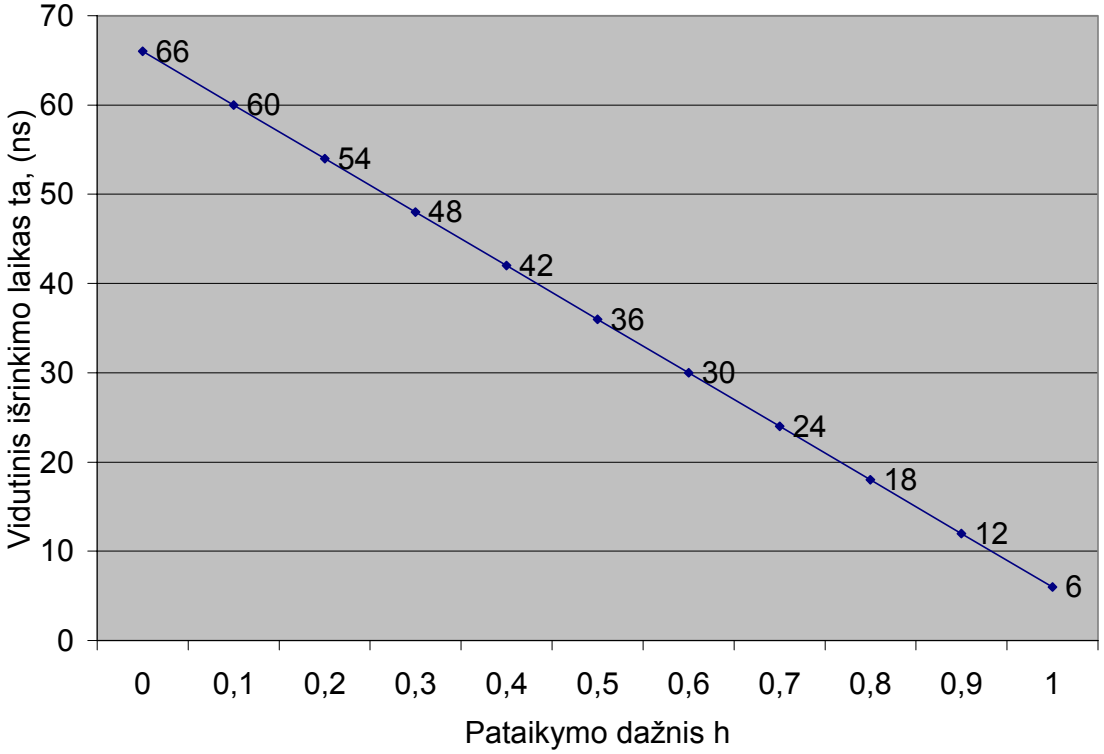
Jeigu nepataikymo atveju informacija pirmiausia perkeliama į spartinančiąją atmintinę ir tik po to siunčiama į procesorių, tuomet išrinkimo laikas būtų toks:

$$t_a = t_c + (1-h)t_m. \qquad \textbf{(1.5)}$$

Ši išraiška rodo, kad vidutinis kreipties laikas gali būti mažinamas:

1) mažinant informacijos išrinkimo iš spartinančiosios atmintinės laiką t_c;
2) didinant pataikymų dažnį *h*;
3) mažinant informacijos išrinkimo iš pagrindinės atminties laiką t_m.

Tarkime, kad $t_c = 6ns$, $t_m = 60ns$, o spartinančiosios atmintinės pataikymo dažnis h kinta nuo 0 iki 1 kas 0,1 . Tuomet vidutinio išrinkimo laiko t_a priklausomybę nuo pataikymų dažnio h gausime kaip parodyta 1.4 paveiksle.



1.4 pav. Vidutinio išrinkimo laiko priklausomybė nuo pataikymo dažnio

Aukščiau pateikta **(1.5)** išraiška gali būti detalizuota. Spartinančiosios atmintinės užklausos laiką sudaro spartinančiosios atmintinės išrinkimo laikas t_{ci} ir informacijos skaitymo iš spartinančiosios atmintinės laikas t_{cr} . Tuomet vidutinis išrinkimo laikas gaunamas :

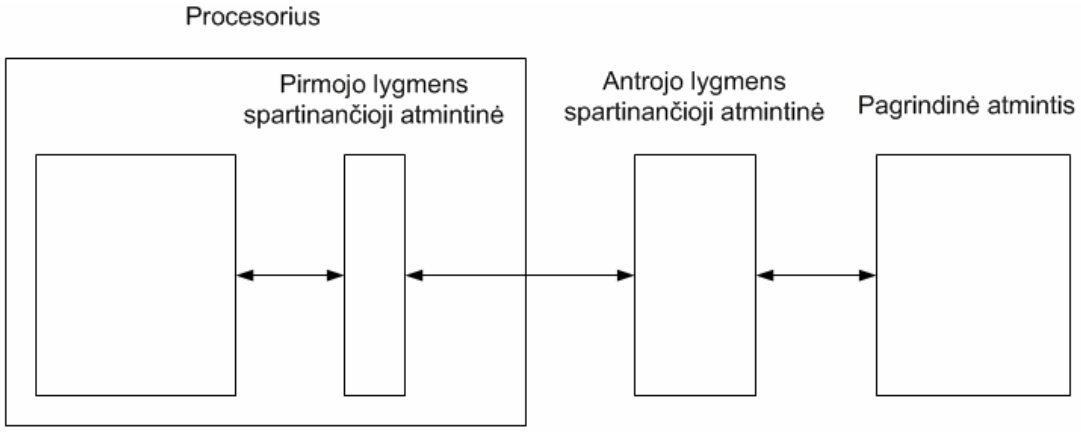
$$t_a = t_{ci} + ht_{cr} + (1-h)t_m \text{ arba} \qquad \textbf{(1.6)}$$

$$t_a = h(t_{ci} + t_{cr}) + (1-h)(t_{ci} + t_m) \text{ [25].} \qquad \textbf{(1.7)}$$

Išrinkimo laikas pataikymo atveju (kai duomenys yra spartinančiojoje atmintinėje) yra $t_{ci} + t_{cr}$, o nepataikymo atveju - $t_{ci} + t_m$. Aukščiau pateiktą **(1.7)** išraišką galima pertvarkyti taip:

$$t_a = t_{ci} + t_{cr} + (1-h)(t_m - t_{cr}). \qquad \textbf{(1.8)}$$

Dabartiniu metu kompiuteriuose naudojama dviejų ar net trijų lygmenų spartinančioji atmintinė. Didesnės talpos antrojo (ir trečiojo) lygmens spartinančioji atmintinė įterpiama tarp pirmojo lygmens spartinančiosios atmintinės ir pagrindinės atminties, kaip parodyta 1.5 paveiksle .



1.5 pav. Spartinančiosios atmintinės sistema su antrojo lygmens spartinančiąja atmintine

Vykdant komandas, procesorius pirma kreipiasi į pirmojo lygmens spartinančiąją atmintinę, jeigu reikiamos informacijos jis čia nerado, tuomet bus kreipiamasi į antrojo lygmens spartinančiąją atmintinę [1, 4, 17].

Įvertinę antrojo lygmens spartinančiąją atmintinę, vidutinį išrinkimo laiką galime užrašyti taip :

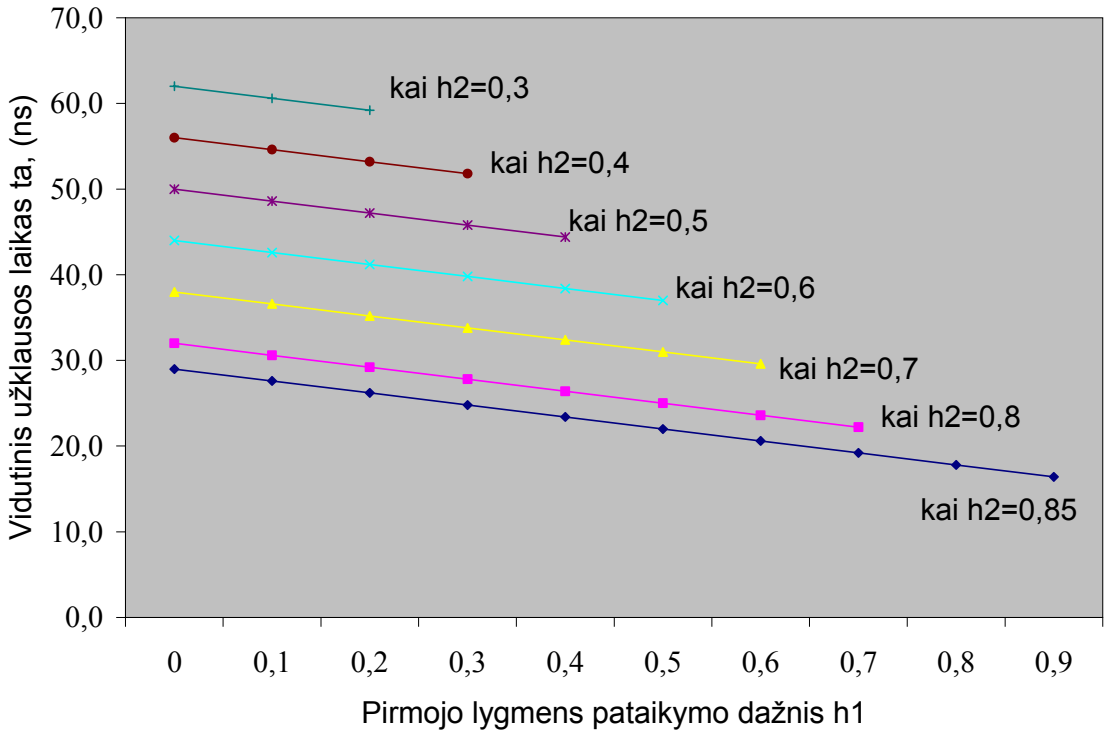
$$t_a = t_{c1} + (1 - h_1)t_{c2} + (1 - h_2)t_m , \qquad \textbf{(1.9)}$$

čia išskleidėme t_c, įvertindami du spartinančiosios atminties lygmenis:

$$t_c = t_{c1} + (1 - h_1)t_{c2} , \qquad \textbf{(1.10)}$$

kur t_{c1} yra pirmojo lygmens spartinančiosios atmintinės išrinkimo laikas, t_{c2} - antrojo lygmens spartinančiosios atmintinės išrinkimo laikas, t_m - pagrindinės atminties išrinkimo laikas, h_1 - pataikymo į pirmojo lygmens spartinančiąją atmintinę dažnis, h_2 - bendras pataikymo į dviejų lygmenų spartinančiąją atmintinę dažnis, kai šios spartinančiosios atmintinės sudaro vienalytę spartinančiųjų atmintinių sistemą [25]. Pataikymo dažnis h_1 bus didesnis negu h_2.

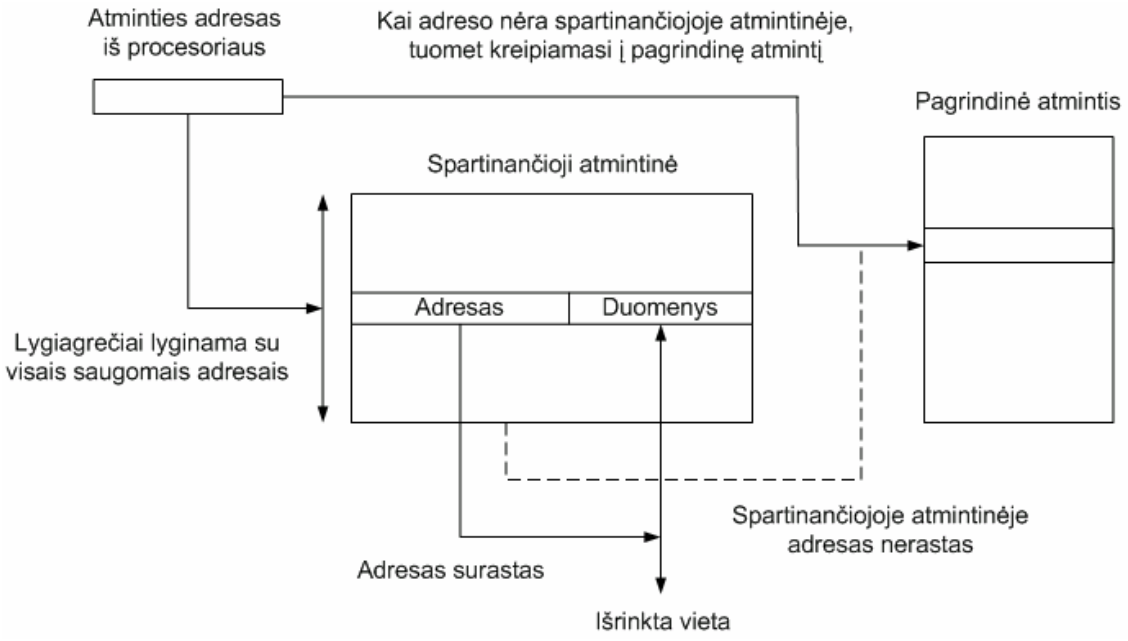
Pavyzdžiui, jeigu $t_{c1} = 2ns$, $t_{c2} = 10ns$, $t_m = 80ns$, spartinančiosios atmintinės pirmojo lygmens pataikymo dažnis h_1 kinta nuo 0 iki 0,9 kas 0,1 , o spartinančiosios atmintinės antrojo lygmens pataikymo dažnis h_2 kinta nuo 0,3 iki 0,85, tuomet vidutinį išrinkimo laiką t_a gausime kaip parodyta 1.6 paveiksle.



1.6 pav. Hierarchinės atminties su dviejų lygmenų spartinančiąja atmintine vidutinis išrinkimo laikas

Kadangi spartinančiosios atmintinės talpa būna šimtus ar net tūkstančius kartų mažesnė už pagrindinės atminties talpą, svarbus yra informacijos laikymo vietų pagrindinėje atmintyje ir spartinančiojoje atmintinėje atitikties nustatymas. Šis klausimas gali būti sprendžiamas trim būdais.

Turbūt lanksčiausias būdas susieti informacijos kopiją (angl. *Caching*) su jos laikymo vieta pagrindinėje atmintyje (jos adresu) yra kartu saugoti informaciją ir jos adresą spartinančiojoje atmintinėje. Tokio tipo atminties organizacija vadinama *visiškai asociatyviaja atmintimi* [10, 15, 25]. *Visiškai asociatyviojoje spartinančiojoje atmintinėje* informacija gali būti laikoma bet kurioje jos eilutėje. Aišku, tai reikalauja, kad kiekvienoje spartinančiosios atmintinės eilutėje būtų saugomi ne tik komandos ar duomenys, bet ir atitinkamo dydžio bloko adresas pagrindinėje atmintyje (vadinamas etikete, žyma, angl. *tag*). Kreipinio metu atminties adresą speciali logika lygiagrečiai lygina su kiekvienoje eilutėje saugomais adresais (žr. 1.7 pav.). Jeigu kreipinio adresas sutampa su kurioje nors eilutėje saugomu adresu (žyma), tuomet iš tos eilutės skaitomi atitinkami duomenys.



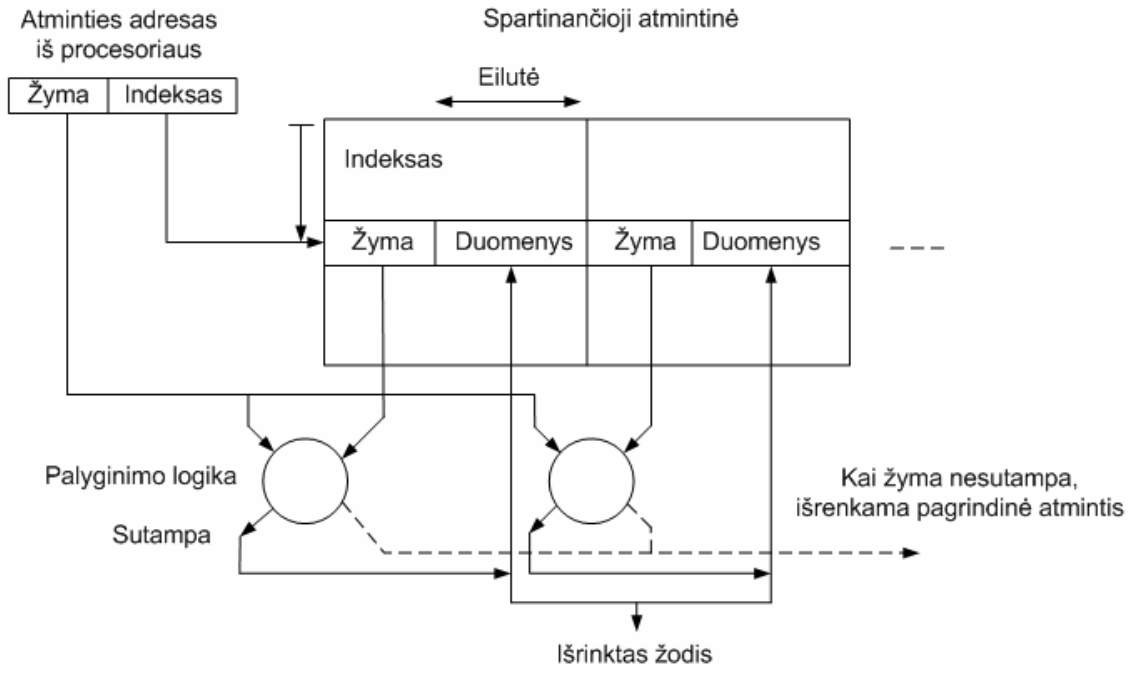
1.7 pav. Visiškai asociatyvios atminties struktūra

Visa eilutė gali būti perduodama į spartinančiąją atmintinę ir iš jos vienu perdavimu, jeigu pagrindinę atmintį su spartinančiąja atmintine jungia pakankamo pločio duomenų kelias (magistralė). Jei duomenų magistralės plotis nepakankamas, eilutė persiunčiama per kelis kartus.

Visiškai asociatyvi spartinančioji atmintinė suteikia galimybę lanksčiai panaudoti jos talpą informacijai laikyti. Tačiau tokio tipo atmintis yra brangiausia, kadangi reikalauja didelio skaičiaus palyginimo schemų. Be to, reikalingas efektyviai dirbantis algoritmas pakeičiamai eilutei parinkti, kai visos eilutės užimtos.

Ekonomiškai suformuoti visiškai asociatyvią spartinančiąją atmintinę galima tik esant mažai ar vidutinei talpai.

Iš dalies asociatyvioje atmintyje bet kuriam perkeltam iš pagrindinės atminties blokui skiriamas tam tikras skaičius eilučių, turinčių tą patį numerį (indeksą), bet skirtingas žymas [10, 15, 25]. Tai gali būti kompromisas tarp visiškai asociatyvios spartinančiosios atmintinės ir tiesioginio atitikimo spartinančiosios atmintinės. Schema pavaizduota 1.8 paveiksle.



1.8 pav. Iš dalies asociatyvios atminties struktūra

Spartinančioji atmintinė suskirstyta į eilučių rinkinius (pavyzdžiui, keturių krypčių iš dalies asociatyvi spartinančioji atmintinė turi keturias eilutes kiekviename rinkinyje). Eilučių skaičius rinkinyje vadinamas asociatyvumu arba rinkinio dydžiu. Kiekviena eilutė kiekviename rinkinyje turi savo žymą, kuri kartu su indeksu (rinkinio numeriu) identifikuoja eilutę. Vykdant programą procesorius suformuoja adresą, pagal kurį nustatomas rinkinio numeris, tuomet rinkinio eilučių žymos lyginamos su užklausos adreso aukščiausiais bitais. Jeigu randamas atitikmuo, informacija iš atitinkamos eilutės perduodama į procesorių, priešingu atveju užklausa perduodama į pagrindinę atmintį.

Iš dalies asociatyvioje spartinančiojoje atmintinėje palyginimo logikos elementų (komparatorių) skaičius lygus rinkinio eilučių skaičiui, o ne visų eilučių skaičiui, kaip būna visiškai asociatyviojoje atmintinėje. Todėl iš dalies asociatyvioje spartinančiojoje atmintinėje informacija randama greičiau, nei visiškai asociatyviojoje atmintinėje.

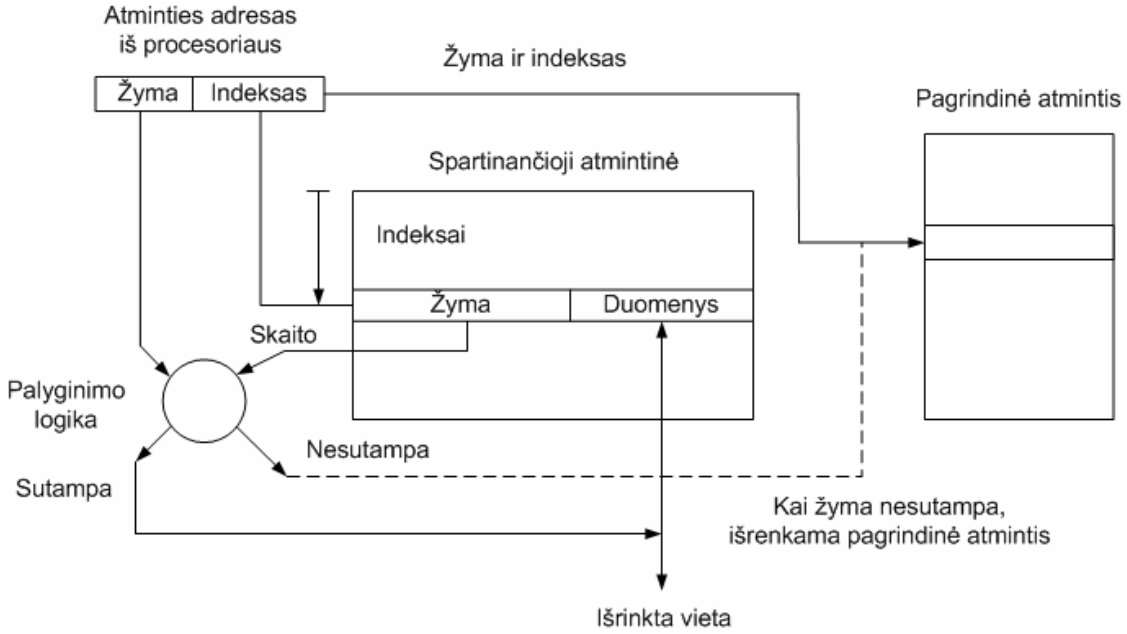
Iš dalies asociatyviosios atmintinės valdyme pakeitimo algoritmas atsižvelgia tik į eilutes, esančias viename rinkinyje, kadangi rinkinio parinkimą nulemia adrese esantis indeksas (rinkinio numeris). Tokiu būdu, jeigu kiekvieną rinkinį sudaro dvi eilutės, pakeičiamai eilutei identifikuoti kiekviename rinkinyje reikalingas tik vienas papildomas bitas; jei sudaro daugiau eilučių, tokių bitų skaičius didėja.

Jei rinkinį sudaro viena eilutė, turime tiesioginio atitikimo atminties valdymo struktūrą, jei visą spartinančiąją atmintinę sudaro vienas rinkinys, gauname visiškai asociatyvų atmintinės valdymą.

Turint tam tikrą eilučių skaičių, projektuojant yra galimybė rinktis - didinti rinkinių skaičių ar didinti

eilučių skaičių kiekviename rinkinyje, kadangi spartinančiosios atmintinės dydis lygus rinkinių skaičiui padaugintam iš eilučių skaičiaus kiekviename rinkinyje.

Tiesioginio atitikimo spartinančiojoje atmintinėje kiekvieną pagrindinės atminties adresą atitinka vienintelė spartinančiosios atmintinės eilutė [10, 15, 25]. Tai leidžia šios eilutės indeksą (numerį) nustatyti tiesiogiai iš žemesniųjų atminties adresų bitų. Likusieji aukštesnieji adreso bitai laikomi spartinančiojoje atmintinėje kaip žyma kartu su duomenimis. Paaiškinsime 1.9 paveiksle pateiktą struktūrą.

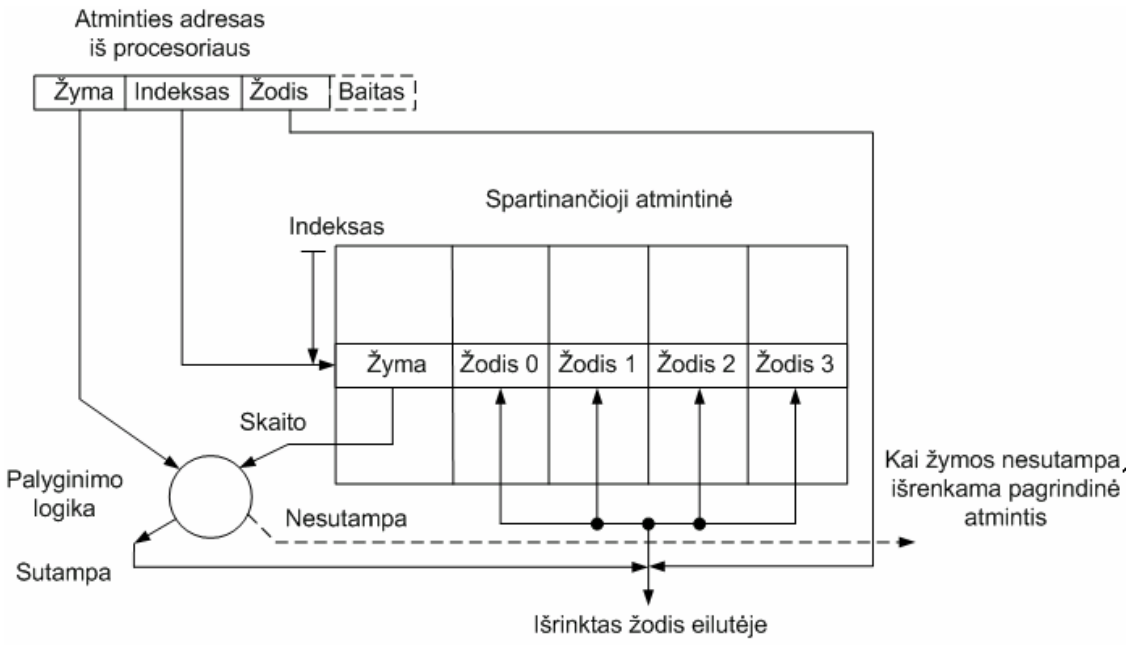


1.9 pav. Tiesioginės atminties valdymo struktūra

Iš procesoriaus atėjęs adresas padalinamas į du laukus, t. y. į žymą ir į indeksą. Žymą sudaro aukštesnieji adreso bitai, kurie saugomi su duomenimis. Indeksą sudaro žemesnieji adreso bitai, naudojami kreipiantis į spartinančiąją atmintinę. Kai nuoroda pateikiama į atmintį, pirmiausia išskiriamas indeksas žodžio vietai spartinančiojoje atmintinėje nustatyti.

1.10 paveiksle parodyta tiesioginio atitikimo spartinančioji atmintinė su eilute, kurią sudaro daugiau negu vienas žodis. Užklausos žodyje saugoma žyma perskaitoma ir palyginama su aukščiausiais adreso bitais; jeigu jie sutampa su žyma, užklausa pateikiama į spartinančiąją atmintinę. Tačiau jeigu jie nesutampa su žyma, reikalingo žodžio spartinančiojoje atmintinėje nėra, todėl jam surasti kreipinys pateikiamas į pagrindinę atmintį.

1.10 paveiksle parodyta tiesioginio atitikimo spartinančioji atmintinė, kurios eilutę sudaro keturi žodžiai.



1.10 pav. Tiesiogiai valdomos atminties ir eilutės struktūra

Pagrindinės atminties adresą sudaro žyma, indeksas ir žodžio adresas eilutėje. Visi spartinančiosios atmintinės eilutės žodžiai turi vienodą žymą. Indekso laukas yra naudojamas spartinančiosios atmintinės eilutei išrinkti, o joje saugojama žyma palyginama su reikalinga adreso žyma. Skaitymo operacijoje, jeigu žymos yra vienodos, iš eilutės parinktas žodis perduodamas į procesorių. Jeigu žymos eilutėje, kurioje yra reikalingas žodis, yra ne tokios pačios, eilutė perskaitoma iš pagrindinės atminties ir perduodama į spartinančiąją atmintinę.

Tiesioginio atitikimo atminties atveju atitinkamos eilutės su tais pačiais indeksais atitiks tą pačią eilutę spartinančiojoje atmintinėje, todėl spartinančiojoje atmintinėje tuo pačiu metu gali būti tik eilutės su skirtingais indeksais. Pakeitimo algoritmas yra nereikalingas, kadangi kiekvienai įeinančiai eilutei skirta tik viena vieta.

1.3. Eilutės pakeitimo algoritmai

Lengviausiai įgyvendinamas *atsitiktinio pakeitimo algoritmas* [10, 12, 18, 25]. Atsitiktinio pakeitimo algoritmas keičiamą eilutę parenka visiškai atsitiktinai, neatsižvelgiant į atminties nuorodas ar ankstesnius kreipinius. Praktiškai atsitiktinio pakeitimo algoritmas realizuojamas taip: visai spartinančiajai atmintinei yra skiriamas vienas skaitiklis ir jo turinys didinamas tam tikru dydžiu po kiekvieno takto arba po kiekvieno kreipinio, nepriklausomai nuo to, buvo pataikyta ar nepataikyta. Skaitiklyje laikoma reikšmė identifikuoja spartinančiosios atmintinės eilutę, jeigu spartinančioji atmintinė yra visiškai asociatyvi arba iš dalies asociatyvi (kelių krypčių). Skaitiklyje turėtų būti pakankamai bitų, kad jis galėtų identifikuoti bet kurią eilutę. Visiškai asociatyvios spartinančiosios atmintinės atveju reikalingas skaitiklis su n bitų, jeigu spartinančiojoje atmintinėje yra 2^n eilučių.

Pakeitimo eilės tvarkos (angl. *First Input First Output - FIFO - Pirmasis įėjęs - pirmas išėjo*) algoritmas pakeičia eilutę, kuri spartinančiojoje atmintinėje išbuvo ilgiausiai [10, 11, 25]. Šis algoritmas gali būti įgyvendinamas sudarant eilučių adresų eilę, tačiau paprastesnis jo įgyvendinimas siejamas su skaitiklio ar skaitiklių panaudojimu: vieno skaitiklio pakanka visiškai asociatyviai spartinančiajai atmintinei, o vienas skaitiklis kiekvienam rinkiniui reikalingas kelių krypčių spartinančiajai atmintinei; kiekvienas iš jų privalo turėti pakankamai bitų, kad galėtų identifikuoti keičiamą eilutę.

Paskutinės seniausiai naudotos eilutės (angl. *Least Recently Used - LRU*) algoritme spartinančiosios atmintinės eilutė, kuri ilgesnį laiką nebuvo užklausiama, yra keičiama [10, 12, 18, 25]. Šis algoritmas spartinančiosios atmintinės sistemose yra populiariausias. Šiam algoritmui realizuoti gali būti naudojamas skaitiklis, asocijuojamas su kiekviena spartinančiosios atmintinės eilute. Skaitiklio turinys yra didinamas reguliariais intervalais ir startuoja iš naujo, kai jam priskirta eilutė išrenkama, todėl kiekvieno skaitiklio reikšmė rodo santykinį laiką, kada eilutė buvo užklausta. Seniausia eilutė pakeičiama. Šį algoritmą modifikuojame taip: atsisakome skaitiklio ir esant pataikymui, *pataikyta eilutė* yra perkeliama į viršų, o prieš ją buvusios eilutės perstumiamos žemyn. Esant nepataikymui, jei spartinančioji atmintinė yra nepilna, visos spartinančiojoje atmintinėje esančios eilutės pastumiamos žemyn, o į laisvą pirmąją eilutę įrašoma iš atminties žemesnio lygmens nuskaityta eilutė. Esant nepataikymui, jei spartinančioji atmintinė yra pilna, žemiausiai esanti eilutė ir bus paskutinė vėliausiai naudota eilutė (ji visada būna

spartinančiosios atmintinės apačioje), ši eilutė iškeliama į žemesnio lygmens atmintį, virš jos buvusios eilutės perstumiamos žemyn, o viršuje į atsilaisvinusią eilutę įrašoma nauja eilutė .

Rečiausiai naudotos eilutės (angl. *Least Frequency Used - LFU*) algoritme spartinančiosios atmintinės eilutė, kuri ilgesnį laiką mažiausiai buvo užklausiama, yra keičiama (šis algoritmas labai panašus į *LRU* algoritmą) [10, 11, 25]. Šiame algoritme skaitiklis asocijuojamas su kiekviena spartinančiosios atmintinės eilute. Spartinančiojoje atmintinėje eilutės pataikymo atveju eilutės skaitiklis didinamas tam tikra reikšme. Ir bet kuriam eilutės skaitikliui (pataikymo atveju) pasiekus maksimalią reikšmę, likusių skaitiklių turinys sumažinamas atitinkamu dydžiu. Tokiu būdu mažiausiai naudotos eilutės skaitiklio reikšmė bus mažiausia, o tai ir bus rodiklis, parodantis, kurią eilutę reikia keisti.

1.4. Informacijos perrašymo mechanizmai

Kadangi reikalingo žodžio skaitymas nedaro įtakos spartinančiosios atmintinės turiniui, todėl spartinančiosios atmintinės žodis ir pagrindinėje atmintyje laikoma jo kopija sutampa. Tačiau tuomet, kai žodžiai įrašomi į spartinančiąją atmintinę, atsiranda tikimybė, kad spartinančiosios atmintinės žodis ir pagrindinėje atmintyje esanti kopija gali skirtis.

Jeigu ignoruosime duomenų grąžinimą į pagrindinę atmintį, tuomet vidutinis užklausos laikas gaunamas pagal formulę :

$$t_a = t_c + (1-h)t_m \qquad \textbf{(1.11)}$$

(su sąlyga, kad visos užklausos pataiko į spartinančiąją atmintinę). Vidutinis užklausos laikas, vykdant rašymo operaciją, prie šios formulės pridės papildomą laiką, priklausantį nuo mechanizmo, kuris užtikrina duomenų sinchronizaciją.

Dažniausiai taikomi mechanizmai pagrindinei atminčiai atnaujinti: *ištisinis įrašymas* (angl. *Write-through*) ir *atidėtas įrašymas* (angl. *Write-back*) [10, 25].

Ištisinio įrašymo mechanizme kiekviena įrašymo į spartinančiąją atmintinę operacija palydima įrašymu į pagrindinę atmintį, dažniausiai tai vykdoma tuo pat metu. Papildoma įrašymo operacija į pagrindinę atmintį, žinoma, užims daug daugiau laiko negu įrašymas į spartinančiąją atmintinę, ir tai rašymo operacijoje padidins užklausos trukmę. Atidėto įrašymo atveju vidutinis užklausos laikas, įvertinant nuostolius atliekant perdavimus iš pagrindinės atminties į spartinančiąją atmintinę, išreiškiamas taip:

$$t_a = t_c + (1-h)t_b + w(t_m - t_c) = (1-w)t_c + (1-h)t_b + wt_m = (1-w)t_c + (1-h+w)t_m, \text{ kai } t_b = t_m, \qquad \textbf{(1.12)}$$

kur t_b - eilutės perdavimo laikas į spartinančiąją atmintinę su sąlyga, kad visa eilutė turi būti perduodama iš karto, o w - įrašymo dažnis [25]. Laikotarpis $(t_m - t_c)$ yra papildomas laikas įrašyti žodį į pagrindinę atmintį ar tai būtų pataikyta, ar nepataikyta, su sąlyga, kad tiek spartinančiosios atmintinės, tiek pagrindinės atminties rašymo operacijos vyksta lygiagrečiai, tačiau įrašymo į pagrindinę atmintį operacija turi užsibaigti prieš tai, kai prasidės nauja spartinančiosios atmintinės skaitymo ar įrašymo į ją operacijos. Jeigu eilutės ilgis sutampa su atminties žodžio ilgiu ir duomenų magistralės pločiu, visa eilutė perduodama vienu metu, ir tuomet $t_b = t_m$. Jeigu eilutė yra ilgesnė negu atminties žodžio ilgis ar duomenų magistralės plotis, kiekvienai eilutei persiųsti reikalingi keli perdavimai, tada $t_b = bt_m$, kur b - perdavimų skaičius reikalingas perduoti visai eilutei.

Terminas *įrašymo iškvietimas* naudojamas apibūdinti situacijai, kai eilutė iš pagrindinės atminties perduodama į spartinančiąją atmintinę vykdant įrašymo operaciją, nes buvo užfiksuotas nepataikymas į spartinančiąją atmintinę [10, 25]. Atidėto įrašymo mechanizme kartais daroma kitaip – nepataikius į spartinančiąją atmintinę, informacija įrašoma tik į pagrindinę atmintį, bet nėra saugoma spartinančiojoje atmintinėje. Vidutinis rašymo laikas pastaruoju atveju gaunamas toks:

$$t_a = t_c + (1-w)(1-h)t_b + w(t_m - t_c) = (1-w)(t_c + (1-h)t_b) + wt_m . \qquad \textbf{(1.13)}$$

Pataikymo dažnis paprastai būna šiek tiek žemesnis negu įrašymo iškvietimo atveju, kadangi pakeistos eilutės nepateks į spartinančiąją atmintinę ir priklausomai nuo vykdomos programos jų gali prisireikti keliose skaitymo operacijose.

Ištisinio įrašymo mechanizme rašymo operacija į pagrindinę atmintį atliekama tik eilutės pakeitimo metu. Tuo metu, kai viena eilutė išstumia kitą eilutę, ši gali būti įrašyta į pagrindinę atmintį nepriklausomai nuo to ar eilutė buvo pakeista. Šiame algoritme vidutinis užklausos laikas yra:

$$t_a = t_c + (1-h)t_b + (1-h)t_b = t_c + 2(1-h)t_b , \qquad \textbf{(1.14)}$$

kur pirmasis dėmuo $(1-h)t_b$ atspindi eilutės skaitymą iš atminties, o kitas dėmuo $(1-h)t_b$ - eilutės įrašymą. Atidėto įrašymo mechanizme užrašomos tik tos spartinančiosios atmintinės eilutės, kurios buvo pakeistos. Šiam pakeitimo algoritmui įgyvendinti kiekvienoje spartinančiosios atmintinės eilutėje yra bitas, kuris parodo, ar eilutė yra pakeista ir ar reikia eilutę įrašyti atgal į pagrindinę atmintį. Šis bitas yra tikrinamas, kuomet eilutė yra pakeičiama. Tuomet vidutinis užklausos laikas bus:

$$t_a = t_c + (1-h)t_b + w_b(1-h)t_b = t_c + (1-h)(1-w_b)t_b , \qquad \textbf{(1.15)}$$

kur w_b yra tikimybė, kad eilutė arba jos dalis buvo pakeista. Tačiau pagal šį algoritmą visa eilutė yra įrašoma atgal į atmintį, net jeigu eilutėje buvo pakeistas vienas žodis. Jeigu atminties žodžio ilgis ar duomenų magistralės plotis mažesni nei eilutės ilgis, pakeitimui prireikia žymiai daugiau laiko.

Hierarchinės atminties sistemos našumą nusakanti išraiška (žr. **1.9** formulę) leidžia paskaičiuoti užklausos laiką, įvertinant atskirų lygmenų darbo spartą, pataikymo dažnį. Pagal tai gauti rezultatai rodo maksimalią spartą, neįvertinant techninės realizacijos detalių. Matematinės išraiškos parodo, kad sparčiausiai veikia pirmojo lygmens spartinančioji atmintis, lėtesnė antrojo lygmens spartinančioji atmintis, o lėčiausia – pagrindinė atmintis. Kompleksinį įvertinimą galima gauti naudojant specialias testines programas (pvz.: *RightMark Memory Analyzer, PC Mark2002, CPU-Z, PC Wizadr2004, Sandra2003, SiSoftware Sandra2005*) [2, 19, 20]. Gautus rezultatus reikia perskaičiuoti, jei norime juos

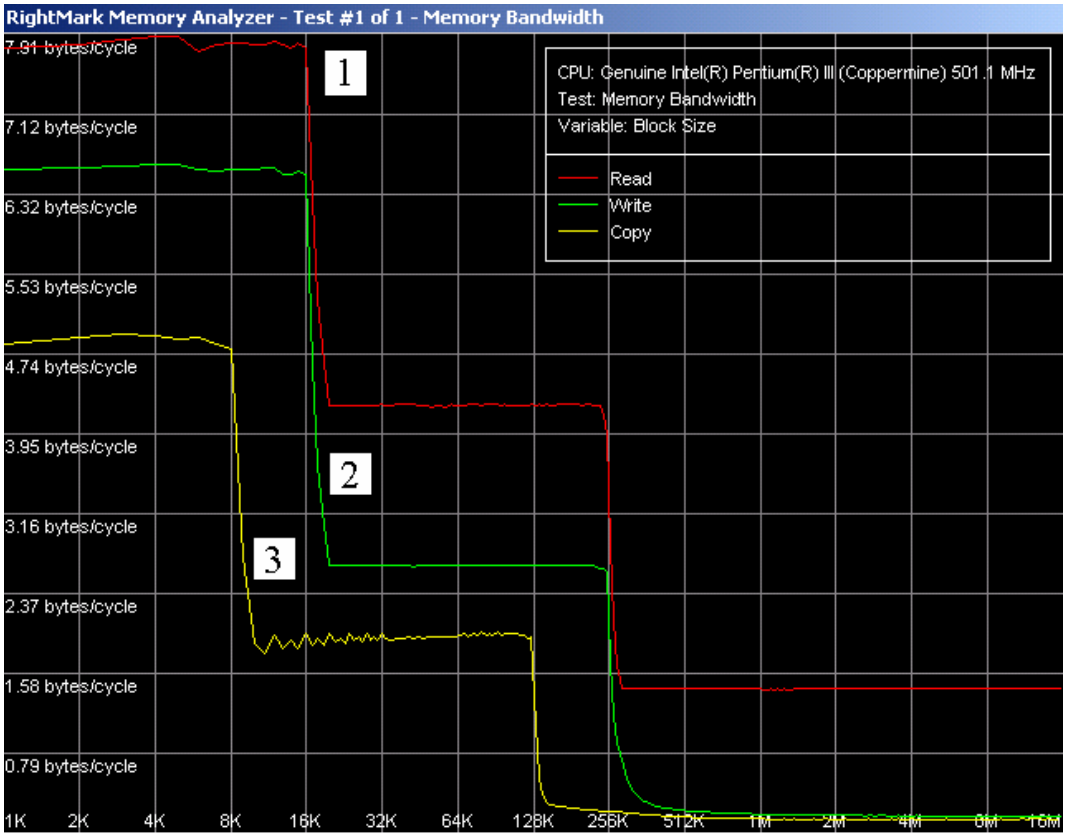
pateikti įprastais pralaidumo vienetais (B/ns, MB/s, GB/s). Be to, testinės programos neleidžia pasirinkti spartinančiosios atmintinės parametrų (informacijos pakeitimo taisyklių, informacijos perrašymo mechanizmų, lygmenų talpos ir pan.). Tam reikia sukurti hierarchinės atminties imitatorių.

Sukurtas spartinančiosios atminties imitatorius privalo turėti galimybę keisti spartinančiosios atmintinės eilutės struktūrą (žymos bitų skaičių, indekso bitų skaičių, duomenų baitų skaičių), informacijos pakeitimo algoritmus (*Random, FIFO, LRU, LFU*), spartinančiosios atminties tipą (visiškai asociatyvi atmintis, iš dalies asociatyvi atmintis, tiesioginio atitikimo atmintis), informacijos perrašymo mechanizmus (ištisinis įrašymas, atidėtas įrašymas).

2. HIERARCHINĖS ATMINTIES SISTEMOS TYRIMAS

2.1. Atminčių sistemos testavimas

Spartinančiosios atmintinės hierarchijos testavimas buvo atliktas su testine programas *RightMark Memory Analyzer 3.0* [2]. Testuotas personalinis kompiuteris Pentium III: taktinis dažnis 500 MHz, 128 MB pagrindinė atmintis ir funkcionuojančios pirmojo ir antrojo lygmenų spartinančiosios atmintinės. Testo rezultatai parodyti 2.1 paveiksle.



2.1 pav. Duomenų sparta, kai naudojami abu spartinančiosios atmintinės lygmenys

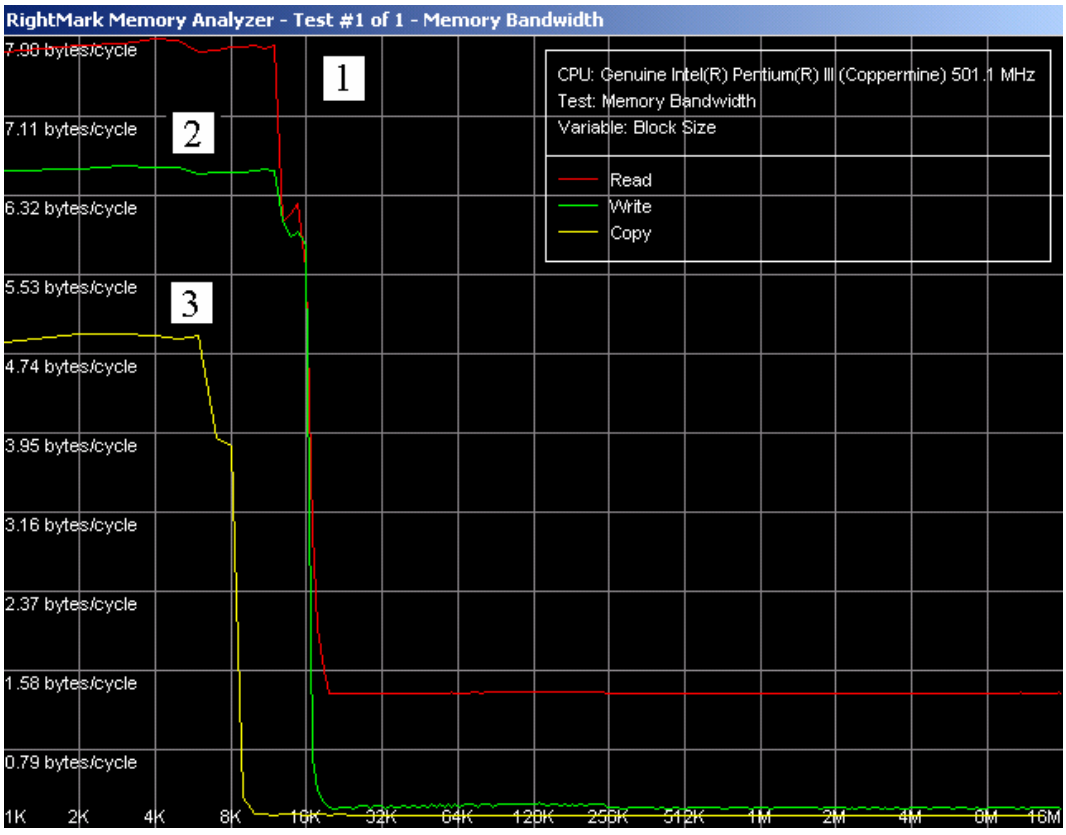
Šiame paveiksle pirma kreivė vaizduoja informacijos skaitymo spartą (informacijos kiekis per taktą).

Pirmojo lygmens spartinančiosios atmintinės talpa - 16 KB. Iš pateikto paveikslo matyti, kad daugiausia informacijos per taktą perskaitoma iš pirmojo lygmens spartinančiosios atmintinės (7.91 baitų duomenų). Antrojo lygmens spartinančioji atmintinė užima 256 KB, jos informacijos skaitymo sparta yra 4 B (žr. 2.1 pav.). Toliau testinė programa testavo duomenų skaitymą iš pagrindinės atmintinės, kurios sparta, lyginant

su spartinančiąja atmintine, yra gerokai mažesnė.

Antroji kreivė parodo duomenų rašymo į atmintį spartą. Iš grafiko matyti (žr. 2.1 pav.), kad rašymo operacijos sparta tuose pačiuose atmintinės lygmenyse yra mažesnė nei skaitymo operacijos sparta. Trečioji kreivė (žr. 2.1 pav.) parodo kopijavimo spartą. Testinė programa kopijavo (vykdė skaitymo ir rašymo komandas) duomenis tuose pačiuose atmintinių lygmenyse, iš pirmojo lygmens spartinančiosios atmintinės į pirmąjį lygmenį. Analogiškai informacijos kopijavimas atliktas antrojo lygmens spartinančiojoje atmintinėje ir pagrindinėje atmintyje.

Išjungus antrojo lygmens spartinančiąją atmintinę (iš pagrindinių kompiuterio nustatymų - *BIOS*), buvo atliktas analogiškas testas su ta pačia testavimo programa *RightMark Memory Analyzer 3.0*. 2.2 paveiksle parodyti testo rezultatai, kuomet testuojant antrasis spartinančiosios atmintinės lygmuo buvo išjungtas.



2.2 pav. Duomenų sparta išjungus antrojo lygmens spartinančiąją atmintinę

Spartinančiosios atminties hierarchijos testavimui buvo panaudotas kitas testavimo programinis paketas *CPU-Z* [5]. Testuotas personalinis kompiuteris Pentium III: taktinis dažnis 500MHz, 128 MB

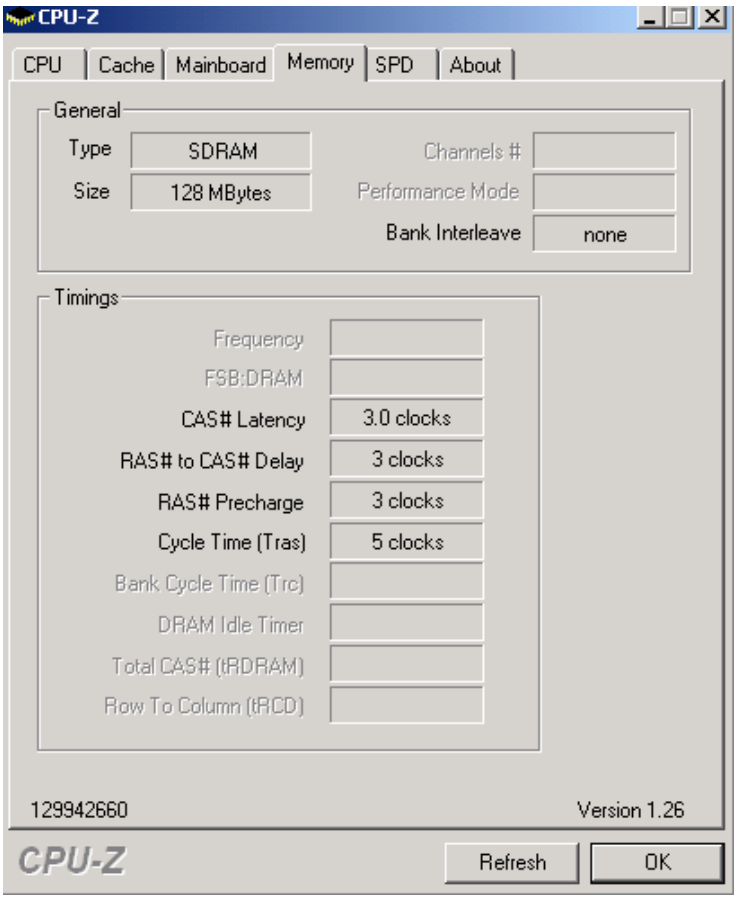
pagrindinė atmintinė ir funkcionuojančios pirmojo ir antrojo lygmenų spartinančiosios atmintinės. Testo rezultatai parodyti 2.3 paveiksle.

```
Cache latency computation, ver 1.0
www.cpuid.com

Computing ...

stride 4        8       16      32      64      128     256     512
size (Kb)
1       3       3       3       3       3       3       3       3
2       3       3       3       3       3       3       3       3
4       3       3       3       3       3       3       3       3
8       3       3       3       3       3       3       3       3
16      3       3       3       3       3       3       3       3
32      3       4       5       7       7       7       7       7
64      3       4       5       7       7       7       7       7
128     3       4       5       7       7       7       7       7
256     7       15      27      44      44      45      46      48
512     12      19      40      65      66      67      70      74
1024    13      22      41      66      66      67      70      74
2048    11      22      41      66      66      67      70      74
4096    13      22      41      67      67      67      70      74
8192    13      22      41      66      67      68      70      74
16384   12      22      41      66      67      68      70      74
32768   12      22      41      66      67      68      80      74

2 cache levels detected
Level 1                 size = 16Kb     latency = 3 cycles
Level 2                 size = 128Kb    latency = 7 cycles
```

2.3 pav. Atmintinių gaištis taktais, testuojant abu spartinančiosios atmintinės lygmenys

2.3 paveiksle parodyta spartinančiosios atmintinės sugaištas laikas taktais, kuomet duomenys į spartinančiąją atmintinę skaitomi blokais, kurių dydį kilobaitais rodo 2.3 paveikslo eilutė „*stride 4 8 16 32 64 128 256 512*“. Pirmojo lygmens spartinančioji atmintinė užima 16 KB. Spartinančiojoje atmintinėje pirmajame lygmenyje sugaištamas laikas yra lygus 3 taktams. Antrojo lygmens spartinančioji atmintinė užima 128 KB, iš jos skaityti duomenis sugaištama 7 taktus. Atlikto testo rezultatai parodo, kad pirmojo lygmens spartinančioji atmintinė duomenis perduoda daugiau nei 2 kartus sparčiau, negu antrojo lygmens spartinančioji atmintinė.

Testavimo programinis paketas *CPU-Z* taip pat parodė testuoto kompiuterio pagrindinės atminties darbo laikus, kurie pateikti 2.4 paveiksle.



2.4 pav. Pagrindinės atmintinės laikai

Matavimo vienetą „taktai“ galima nesunkiai perskaičiuoti į kitą laiko vienetą - „sekundės“ arba „nanosekundės“. Spartinančiojoje atmintinėje sugaištas laikas sekundėmis (arba sekundės dalimis) bus:

$$laikas(s) = \frac{1}{procesoriaus\ \ dažnis}. \qquad \textbf{(2.1)}$$

Įstatę į formulę procesoriaus taktinį dažnį (500 MHz) ir atlikę veiksmus gausime, kad spartinančiosios atmintinės vieno takto trukmė yra $2ns$. Įvertinus tai, kad spartinančiosios atmintinės pirmojo lygmens gaištis yra 3 taktai, o vieno takto trukmė $2ns$, pirmajame lygmenyje sugaištas laikas bus $6ns$. Spartinančiosios atmintinės atrojo lygmens gaištis yra 7 taktai, perskaičiavę taktų skaičių į nanosekundes gausime, kad spartinančiosios atmintinės antrajame lygmenyje sugaištamas laikas yra $14ns$.

Ištyrėme, kad pagrindinės atminties darbo dažnis yra 100 MHz, o duomenų išrinkimas (eilutės išrinkimas užima 3 taktus, o stulpelio išrinkimas, įskaitant ir vėlinimą, užima 3 taktus) užima 6 taktus (žr. 2.4 pav.). Pasinaudoję aukščiau pateikta formule **(2.1)** gausime, kad pagrindinėje atmintyje sugaištas

laikas yra 60*ns* .

Išjungus antrojo lygmens spartinančiąją atmintinę (iš pagrindinių kompiuterio nustatymų - *BIOS*), buvo atliktas analogiškas testas su ta pačia testavimo programa *CPU-Z*. 2.5 paveiksle parodyti testo rezultatai.

Cache latency computation, ver 1.0
www.cpuid.com

Computing ...

stride size (Kb)	4	8	16	32	64	128	256	512
1	3	3	3	3	3	3	3	3
2	3	3	3	3	3	3	3	3
4	3	3	3	3	3	3	3	3
8	3	3	3	3	3	3	3	3
16	3	3	3	3	3	3	3	3
32	13	22	41	67	67	68	70	74
64	23	22	41	67	78	68	70	73
128	13	24	41	67	67	68	70	84
256	13	21	41	67	67	68	70	73
512	13	22	41	67	67	68	70	74
1024	13	22	41	67	67	68	70	74
2048	13	22	41	67	67	68	70	75
4096	13	22	41	67	78	79	70	74
8192	13	22	41	67	67	69	70	74
16384	13	22	42	67	68	69	71	76
32768	13	22	51	67	67	69	71	76

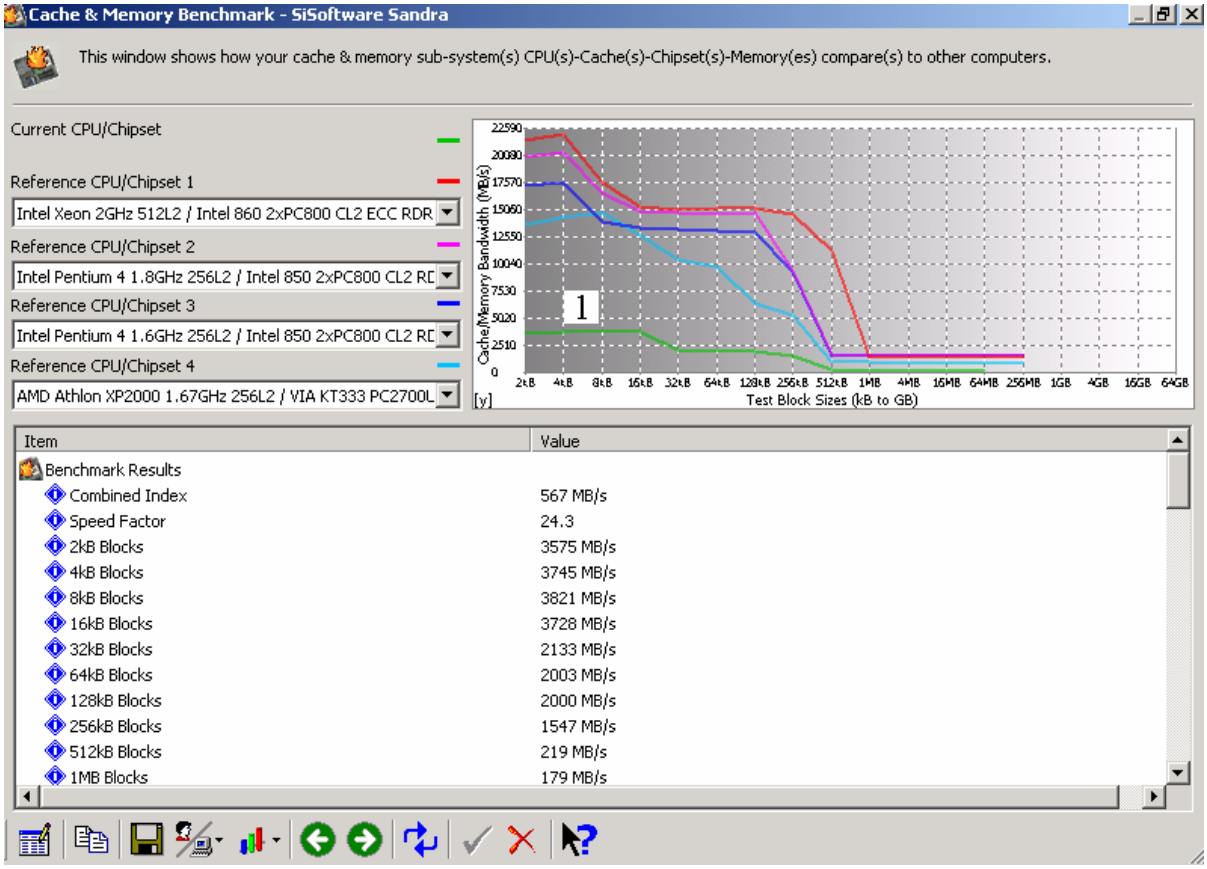
1 cache levels detected
Level 1 size = 16Kb latency = 3 cycles

2.5 pav. Atmintinių gaištis taktais, išjungus antrojo lygmens spartinančiąją atmintinę

Gauti testo rezultatai sulyginti su prieš tai atlikto *CPU-Z* testo rezultatais (žr. 2.4 pav.). Spartinančiosios atmintinės pirmajame lygmenyje testo rezultatai sutapo, kuomet testuojant antrasis spartinančios atmintinės lygmuo buvo išjungtas.

Spartinančiosios atmintinės hierarchijos testavimui buvo panaudotas kitas testavimo programinis paketas *SiSoftware Sandra 2005* [20]. Testuotas personalinis kompiuteris Pentium III: taktinis dažnis 500 MHz, 128 MB pagrindinė atmintinė ir funkcionuojančios pirmojo ir antrojo lygmenų spartinančiosios atmintinės. Testo rezultatai pateikti 2.6 paveiksle. Šiame paveiksle pirmoji kreivė vaizduoja informacijos skaitymo spartą (informacijos kiekis per sekundę), likusios kreivės parodo sulyginimui pasirinktų kompiuterių charakteristikas, kurios įrašytos į testavimo programos duomenų bazę. Pirmojo lygmens spartinančioji atmintinė užima 16 KB. Iš 2.6 paveikslo matyti, kad didžiausia informacijos skaitymo sparta (apie 3800 MB/s) pasiekiama pirmajame spartinančiosios atmintinės lygmenyje. Antrojo lygmens

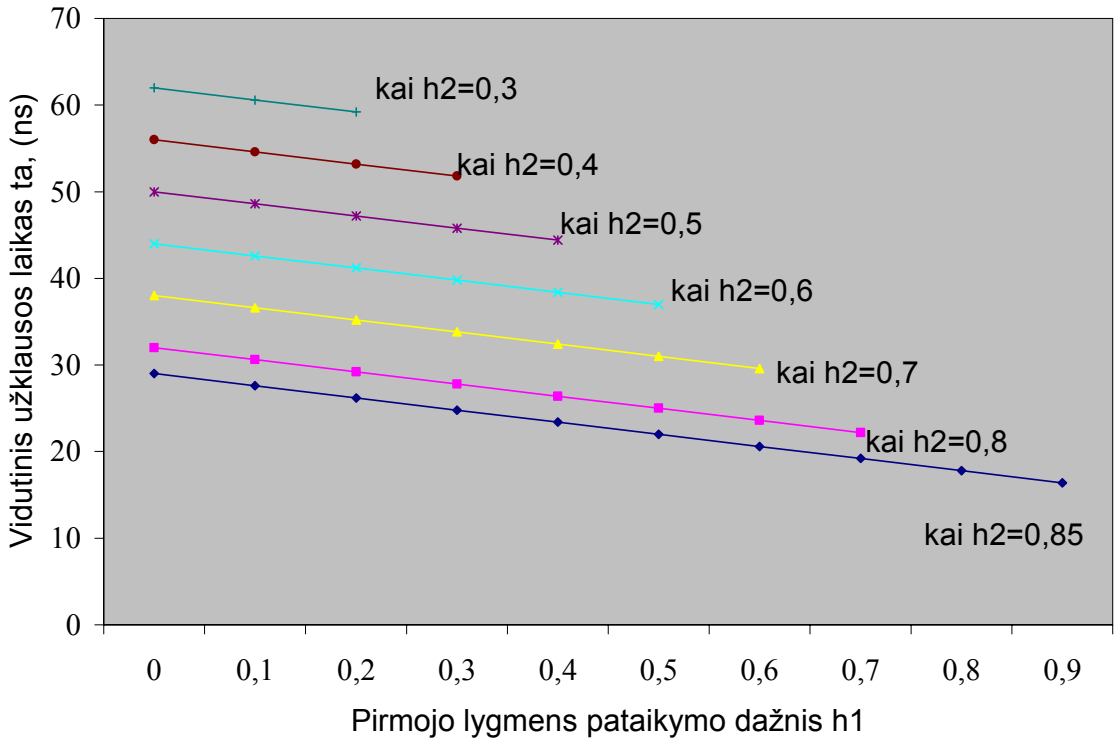
spartinančioji atmintinė užima 256 KB, josios informacijos skaitymo sparta yra mažesnė negu pirmosios (apie 2000 MB/s). Toliau testinė programa testavo duomenų skaitymą iš pagrindinės atmintinės, kurios sparta, lyginant su spartinančiąja atmintine, yra labai maža – tesiekia kelis šimtus MB/s.



2.6 pav. Hierarchinės atmintinės testo rezultatai gauti testavimo paketu *SiSoftware Sandra 2005*

2.2. Atminčių sistemos našumo įvertinimas

Pasinaudoję tyrimo rezultatais, paskaičiuosime vidutinį išrinkimo laiką, kuomet spartinančiosios atmintinės pirmojo lygmens sugaištas laikas $t_{c1} = 6ns$, antrojo lygmens $t_{c2} = 14ns$, o pagrindinės atminties $t_m = 60ns$. Pasirenkame, kad spartinančiosios atmintinės pirmojo lygmens pataikymo dažnis h_1 kinta nuo 0 iki 0,9 , o spartinančiosios atmintinės antrojo lygmens pataikymo dažnis h_2 kinta nuo 0,3 iki 0,85 , tuomet pritaikę **(1.9)** formulę vidutinį užklausos laiką gausime kaip parodyta 2.7 paveiksle.

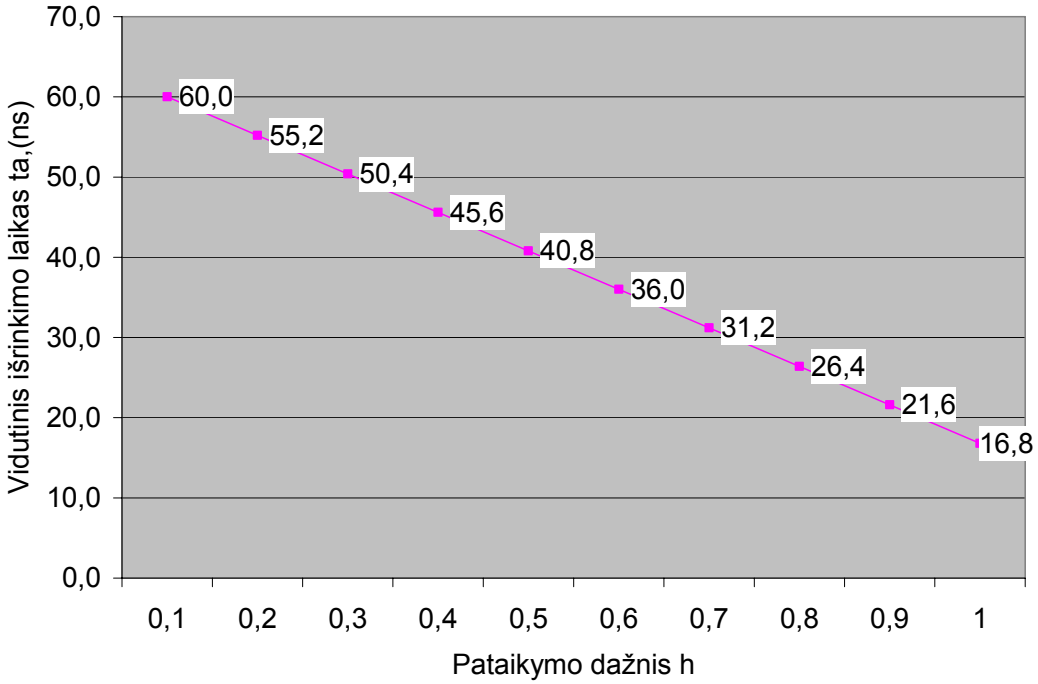


2.7 pav. Tiriamos hierarchinės atmintinės vidutinio išrinkimo laiko teorinis įvertinimas

2.7 paveikslas rodo, kaip mažėja atmintinės vidutinis užklausos laikas t_a, didėjant spartinančiosios atmintinės pirmojo lygmens pataikymo dažniui h_1 ir atvirkščiai.

Pasinaudoję tyrimo rezultatais, paskaičiuosime vidutinį išrinkimo laiką, kuomet spartinančiosios atmintinės gaišties laikas $t_c = 6ns$, o pagrindinės atminties $t_m = 60ns$. Pasirenkame, kad eilutės perdavimo į spartinančiąją atmintinę laikas $t_w = 60ns$, o perrašymo dažnis $w = 0,2$, spartinančiosios

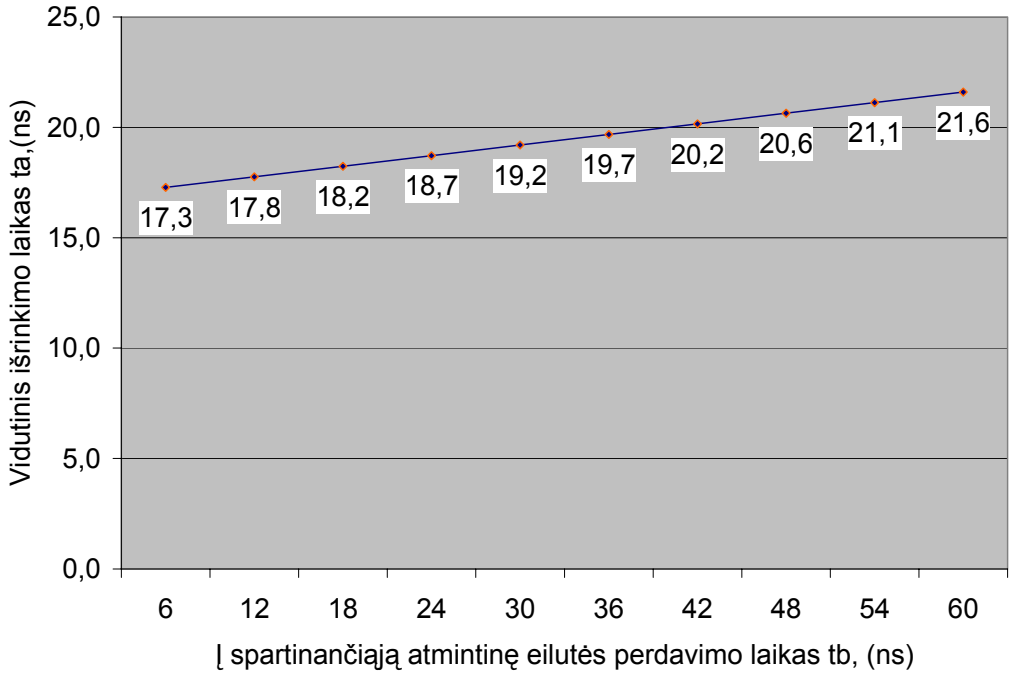
atmintinės pirmojo lygmens pataikymo dažnis h kinta nuo 0,1 iki 1 kas 0,1. Tuomet pritaikę **(1.12)** formulę vidutinį užklausos laiką gausime kaip parodyta 2.8 paveiksle.



2.8 pav. Vidutinis užklausos laikas atidėto įrašymo mechanizme kintant pataikymo dažniui

2.8 paveikslas rodo, kaip mažėja atmintinės vidutinis užklausos laikas t_a didėjant spartinančiosios atmintinės pataikymo dažniui h ir atvirkščiai.

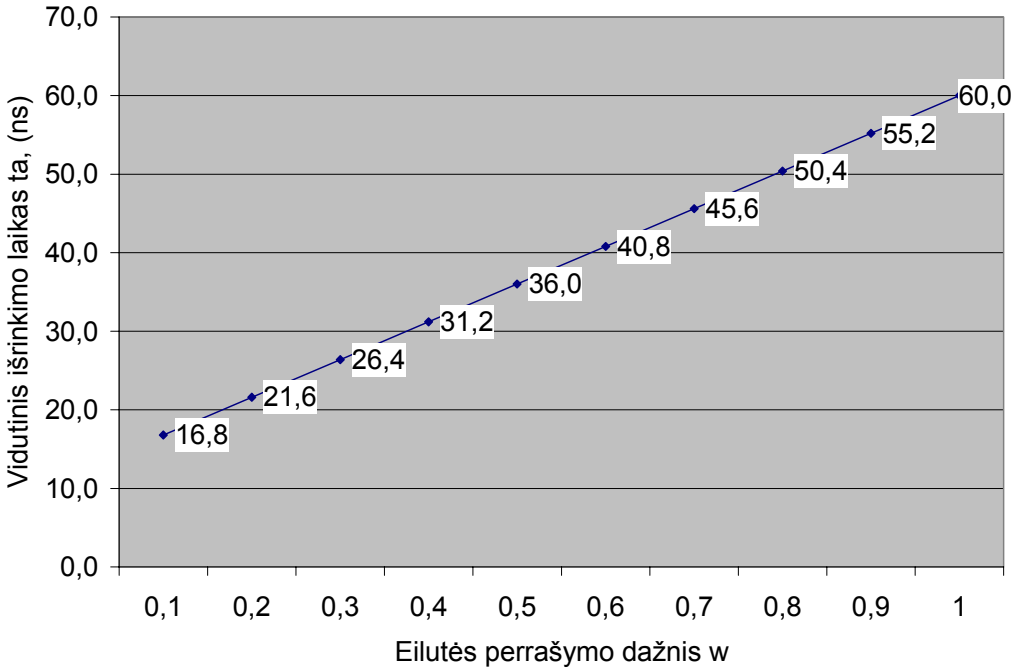
Dabar paskaičiuosime vidutinį išrinkimo laiką, kuomet spartinančiosios atmintinės gaišties laikas $t_c = 6ns$, o pagrindinės atminties $t_m = 60ns$. Pasirenkame, kad eilutės perdavimo į spartinančiąją atmintinę laikas t_w kinta nuo $6\,ns$ iki $60\,ns$ kas $6\,ns$, pataikymo dažnis $h = 0{,}9$, o perrašymo dažnis $w = 0{,}2$, tuomet pritaikę **(1.12)** formulę vidutinį užklausos laiką gausime kaip parodyta 2.9 paveiksle.



2.9 pav. Vidutinis užklausos laikas atidėto įrašymo mechanizme kintant eilutės perdavimo į spartinančiąją atmintinę laikui

Iš 2.9 paveikslo matyti, kad atmintinės vidutinis užklausos laikas t_a, didėja didėjant į spartinančiąją atmintinę eilutės perdavimo laikui t_b ir atvirkščiai, mažėjant t_b mažėja ir t_a.

Dabar paskaičiuosime vidutinį išrinkimo laiką, kuomet spartinančiosios atmintinės gaišties laikas $t_c = 6ns$, o pagrindinės atminties $t_m = 60ns$. Pasirenkame, kad eilutės perdavimo į spartinančiąją atmintinę laikas $t_w = 60ns$, pataikymo dažnis $h = 0{,}9$, o eilutės perrašymo dažnis w kinta nuo 0,1 iki 1 kas 0,1, tuomet pritaikę **(1.12)** formulę vidutinį užklausos laiką gausime kaip parodyta 2.10 paveiksle.

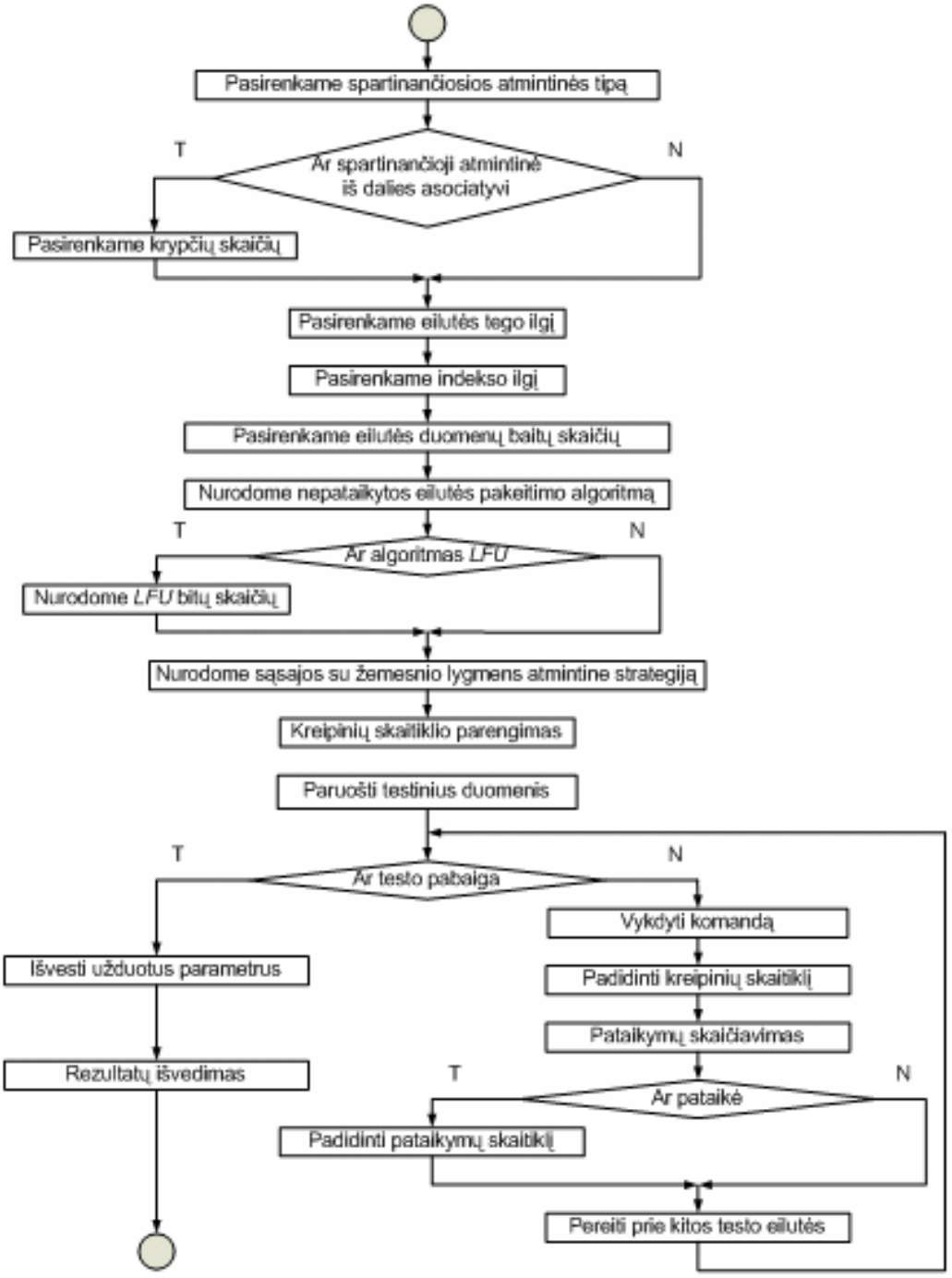


2.10 pav. Vidutinis užklausos laikas atidėto įrašymo mechanizme kintant eilutės perrašymo dažniui

Iš 2.10 paveikslo matyti, kad atmintinės vidutinis užklausos laikas t_a didėja, didėjant eilutės perrašymo dažniui w ir atvirkščiai.

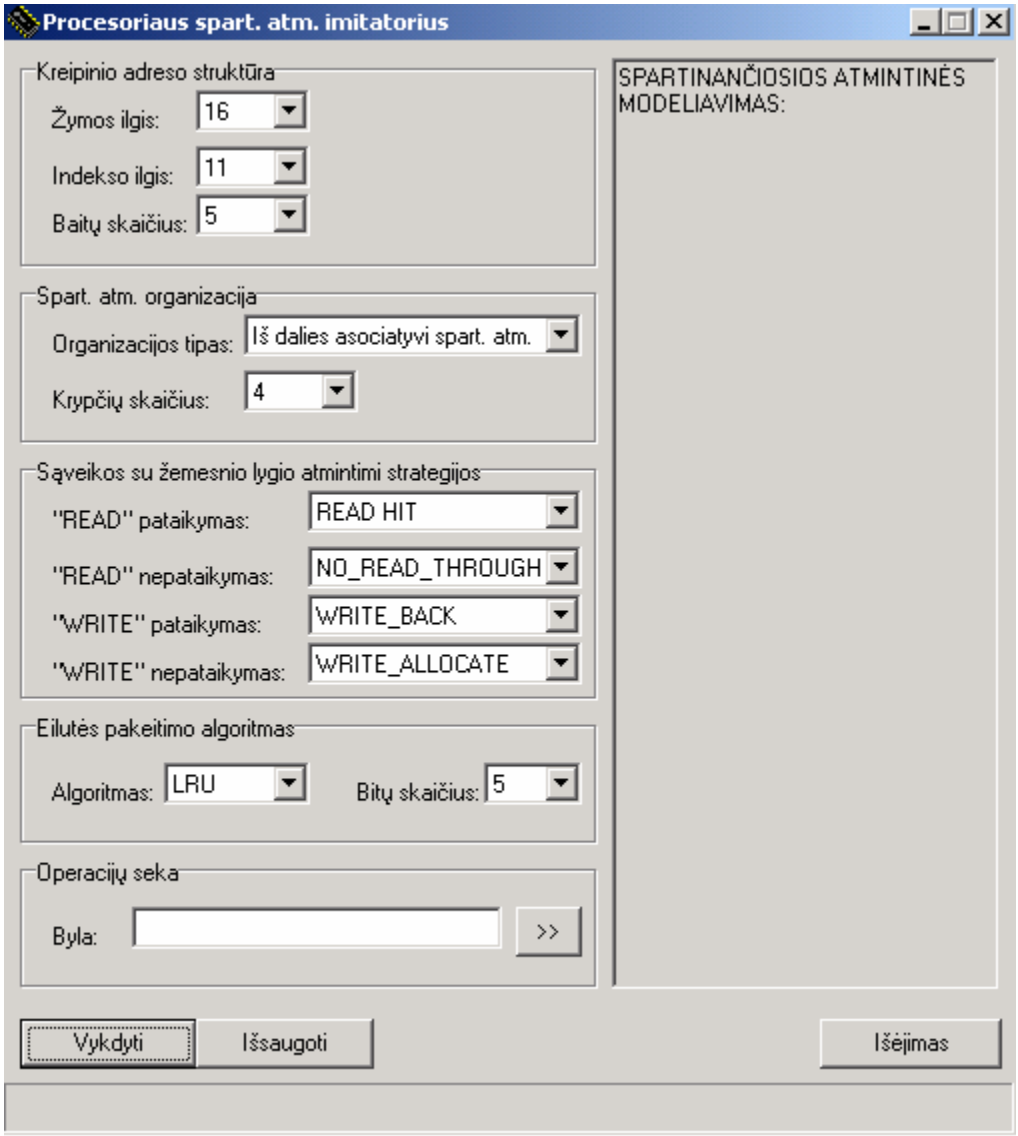
2.3. Spartinančiosios atmintinės imitatorius

Kadangi internete surastos spartinančiosios atmintinės testavimo programos mūsų norų netenkino, buvo nutarta sukurti spartinančiosios atmintinės imitatoriaus algoritmą ir parašyti programą. Spartinančiosios atmintinės imitatoriaus algoritmas pateiktas 2.11 paveiksle.



2.11 pav. Spartinančiosios atmintinės imitatoriaus algoritmas

Startavus spartinančiosios atmintinės imitatoriui, pasirenkame vieną iš trijų realizuotų imituojamos spartinančiosios atmintinės tipų: *visiškai asociatyvi, iš dalies asociatyvi, tiesioginio atitikimo*. Jeigu pasirinkome iš dalies asociatyvią spartinančiąją atmintinę, tuomet privalome nurodyti krypčių skaičių. Toliau reikia suformuoti spartinančiosios atmintinės talpą apibrėžiančius parametrus: pasirinkti žymos ir indekso ilgius bitais bei duomenų baitų kiekį. Reikia pasirinkti eilutės pakeitimo algoritmą (*Random, FIFO, LRU, LFU*). Pasirinkus *LFU* algoritmą, nurodyti bitų skaičių *LFU* reikšmei saugoti. Tuomet pasirenkama sąsajos su žemesnio lygmens atmintine strategija (*ištisinis įrašymas, atidėtas rašymas*). Sistema parengia kreipinių skaitiklį darbui, o mes turime nurodyti testinių duomenų bylą. Tuomet imitatoriui galima leisti dirbti. Sukurto imitatoriaus darbo lango vaizdas parodytas 2.12 paveiksle.



2.12 pav. Spartinančiosios atmintinės imitatoriaus bendras vaizdas

Darbo eigoje imitatorius skaičiuoja kreipinių į atmintinę skaičių. Jeigu paaiškėja, kad kreipinio į atmintį metu buvo pataikyta, sistema didina pataikymo skaitiklį, nepataikymo atveju šis skaitiklis nedidinamas. Programa analizuoja įrašus vienas po kito. Pasiekus bylos pabaigą, pateikiami užduoti pradiniai parametrai bei gauti rezultatai. Informacija pateikiama monitoriaus ekrane arba išsaugomi byloje.

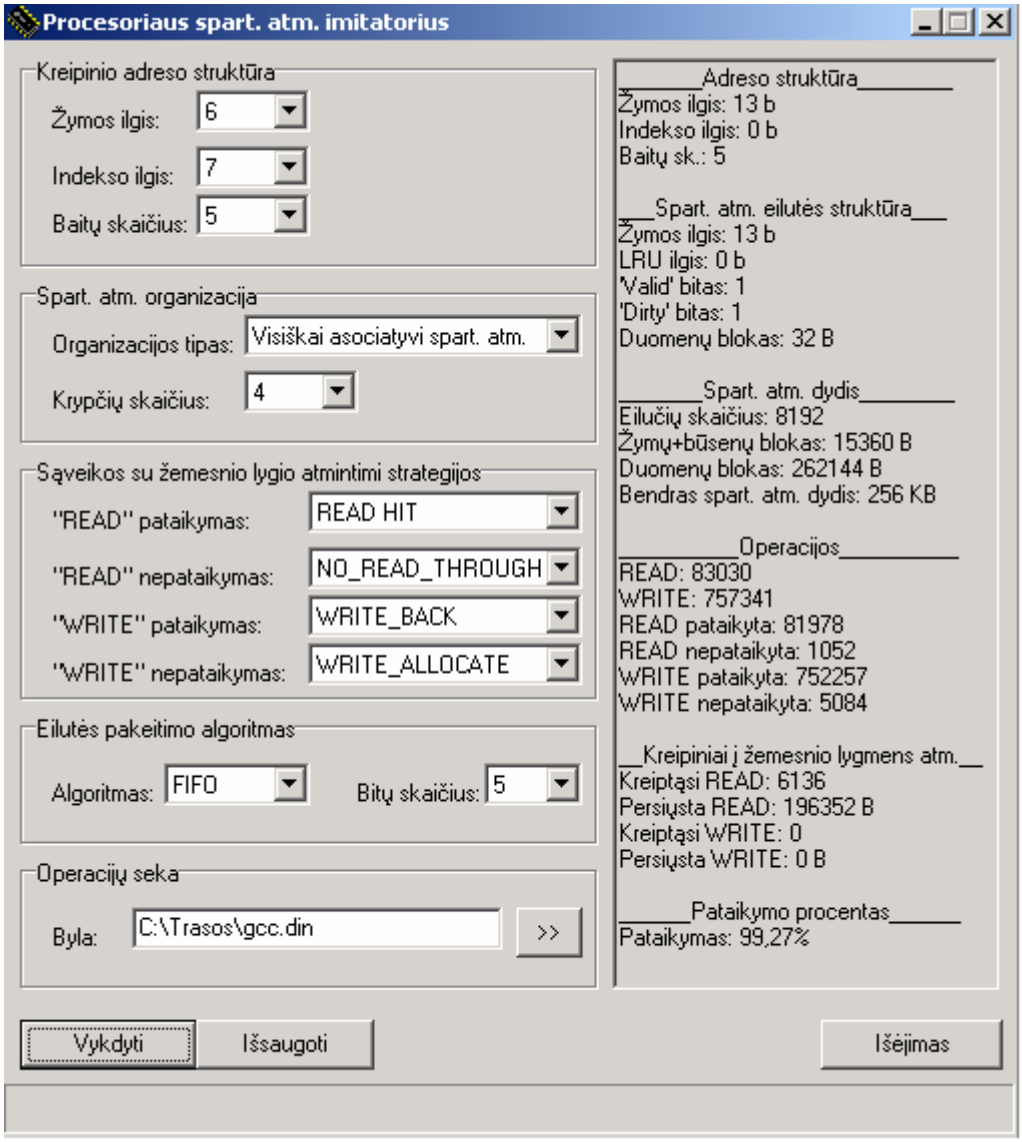
Spartinančiosios atmintinės imitatorius sukurtas su gerai žinomu ir plačiai taikomu programiniu paketu *Borland C++ Builder Enterprise Suite version 5.0* [3].

Spartinančiosios atmintinės imitatoriui testinius duomenų rinkinius naudosime iš interneto parsiųstas bylas (*gcc.din, spice.din, tex.din*) vadinamas trasomis [23]. Bylos *gcc.din* fragmentas pateiktas 2 priede. Kiekvienoje byloje yra apie milijoną testinių rinkinių. Testinę eilutę sudaro komandos kodas (0 – skaityti duomenis, 1 – įrašyti duomenis, 2 – komandos išrinkimas) ir atminties adresas šešioliktainėje formoje.

3. SPARTINANČIOSIOS ATMINTINĖS IMITATORIAUS ĮVERTINIMAS

Sukurtąjį spartinančiosios atmintinės imitatorių išbandėme, testavimui naudojome kompiuterio našumo įvertinimo testų *gcc*, *spice* ir *tex* trasas [23].

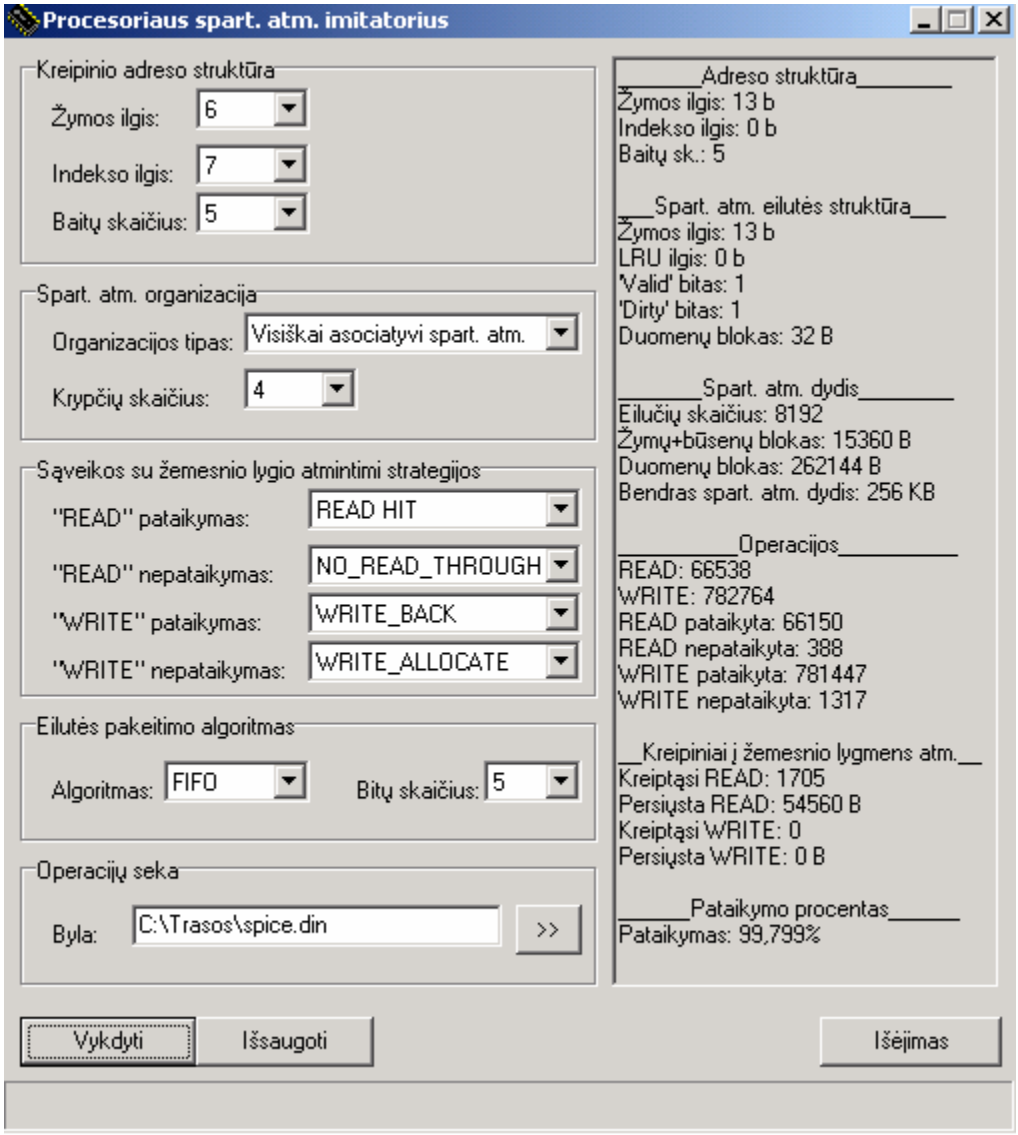
Startavus spartinančiosios atmintinės imitatoriui, pasirinkome: spartinančiosios atmintinės talpa yra 256KB, spartinančiosios atmintinės organizacija yra visiškai asociatyvioji atmintinė, eilutės pakeitimo algoritmas - *FIFO*, nurodėme testinių duomenų bylą *gcc.din* ir leidome imitatoriui dirbti. Imitatoriui baigus darbą, monitoriaus ekrane gauti rezultatai, kurie pateikti 3.1 paveiksle.



3.1 pav. Testo rezultatai visiškai asociatyviosios atmintinės, kai testo byla *gcc.din*

Pateiktame 3.1 paveiksle matyti pasirinkti parametrai ir gauti rezultatai (pataikymo procentas).

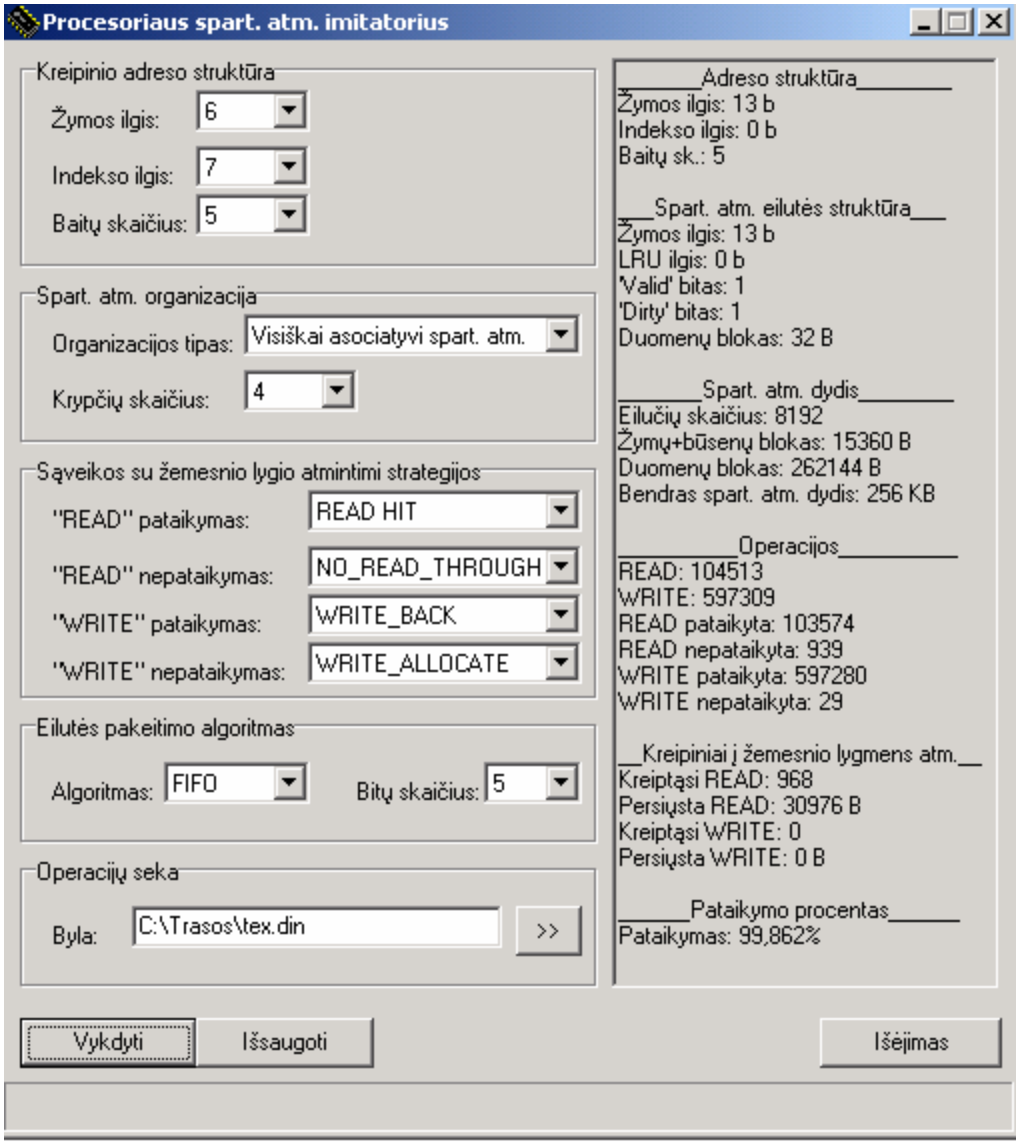
Nekeisdami atmintinės parametrų pasirinkome kitą testo bylą *spice.din* ir analogiškai atlikome tyrimą. Gauti rezultatai parodyti 3.2 paveiksle.



3.2 pav. Testo rezultatai visiškai asociatyviosios atmintinės, kai testo byla *spice.din*

Iš testų rezultatų matyti (žr. 3.1 ir 3.2 paveikslus), kad pakeitus tik testinius rinkinius, keičiasi pataikymo procentas.

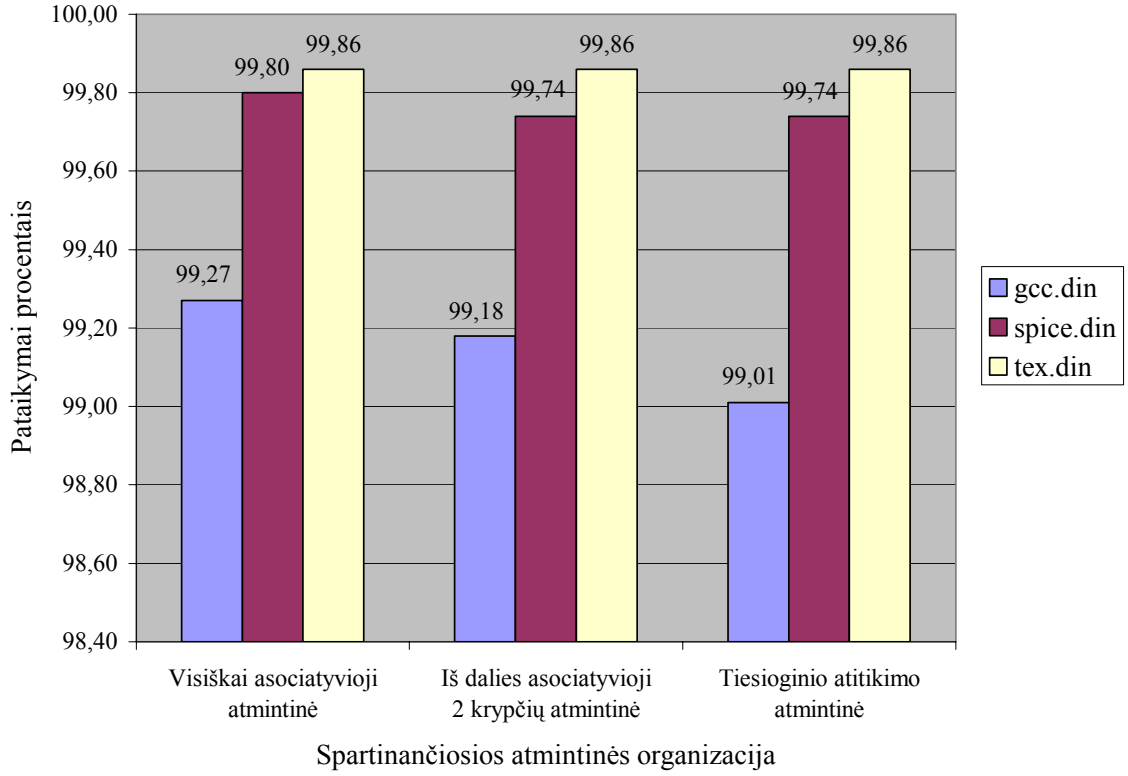
Toliau nekeisdami atmintinės parametrų pasirinkome kitą testo bylą *tex.din* ir atlikome tyrimą. Gauti rezultatai parodyti 3.3 paveiksle.



3.3 pav. Testo rezultatai visiškai asociatyviosios atmintinės, kai testo byla *tex.din*

Patogiausia gautus rezultatus vertinti grafiškai, todėl toliau atliktų testų gauti rezultatai pateikti grafiškai, kaip pataikymo dažnis priklauso nuo tiriamų spartinančiosios atmintinės parametrų (spartinančiosios atmintinės talpos, eilutės ilgio, spartinančiosios atmintinės organizacijos ir pan.).

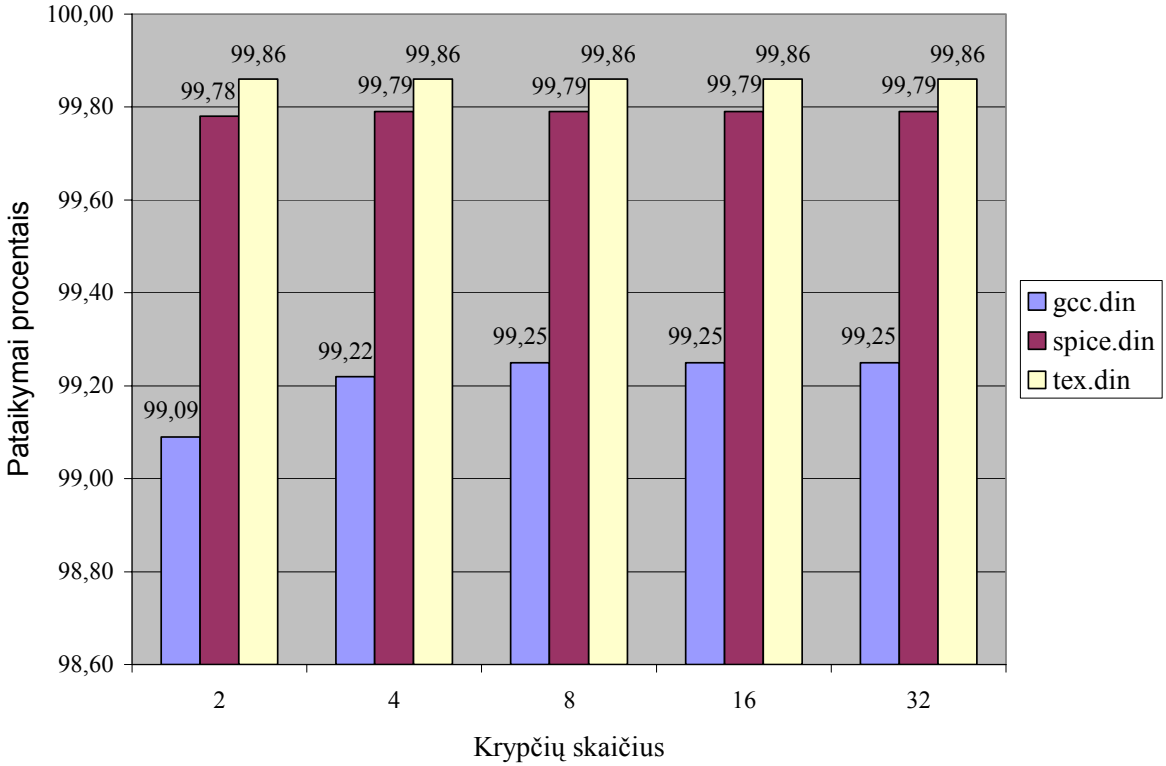
Nekeisdami spartinančiosios atmintinės parametrų, parinkome spartinančiosios atmintinės tipus: iš dalies asociatyvioji 2 krypčių atmintinė ir tiesioginio atitikimo atmintinė. Gauti rezultatai pateikti grafiškai 3.4 paveiksle.



3.4 pav. Pataikymų priklausomybė nuo spartinančiosios atmintinės organizacijos

Iš atlikto tyrimo rezultatų, parodytų 3.4 paveiksle, matyti, kad pataikymo dažnis priklauso ne tik nuo spartinančiosios atmintinės organizavimo, bet ir nuo testinių duomenų . Nors skirtumas yra nedidelis.

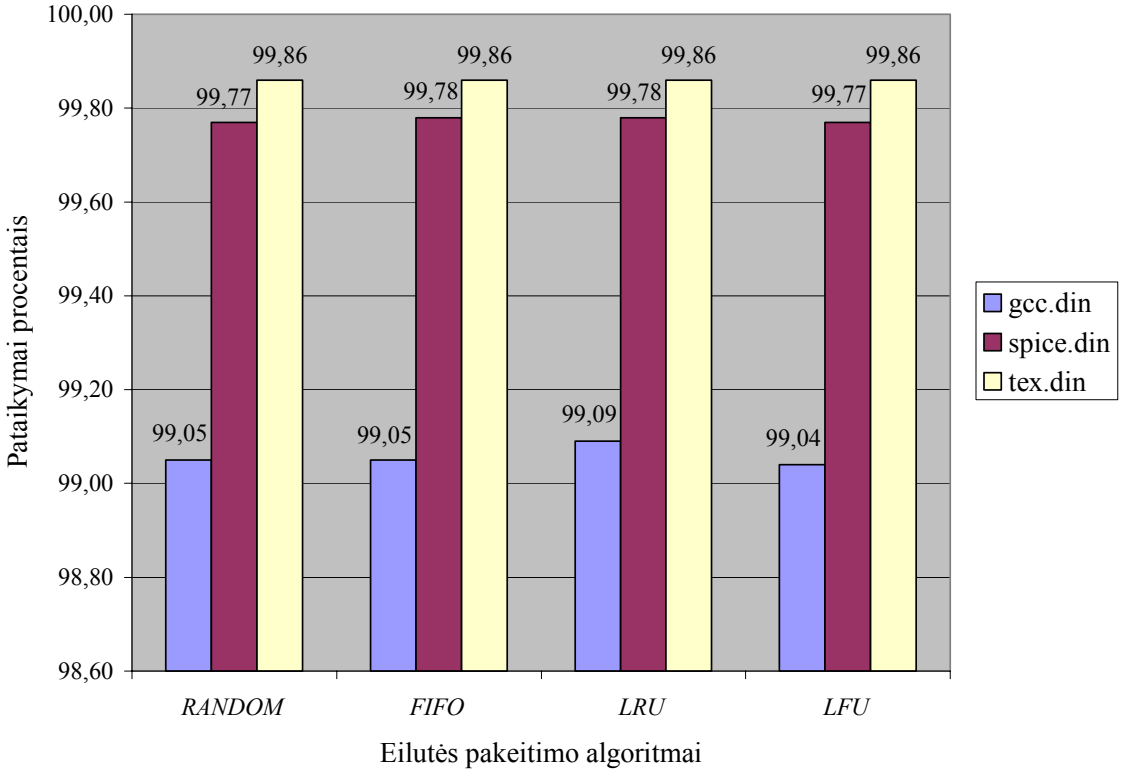
Keisdami krypčių skaičių testavome iš dalies asociatyviąją atmintinę ir grafiškai pavaizdavome pataikymo dažnio priklausomybę nuo iš dalies asociatyviosios atmintinės krypčių skaičiaus. Testas atliktas pasirinkus spartinančiosios atmintinės talpą - 256KB, o eilutės pakeitimo algoritmą - *FIFO*. Gauti rezultatai pateikti 3.5 paveiksle.



3.5 pav. Pataikymų priklausomybė nuo iš dalies asociatyviosios atmintinės krypčių skaičiaus

Iš atlikto tyrimo rezultatų, parodytų 3.5 paveiksle, matyti, kad pataikymo dažnis priklauso ne tik nuo iš dalies spartinančiosios atmintinės krypčių skaičiaus, bet ir nuo testinių duomenų . Tačiau skirtumas nėra didelis, jis yra iki 1%.

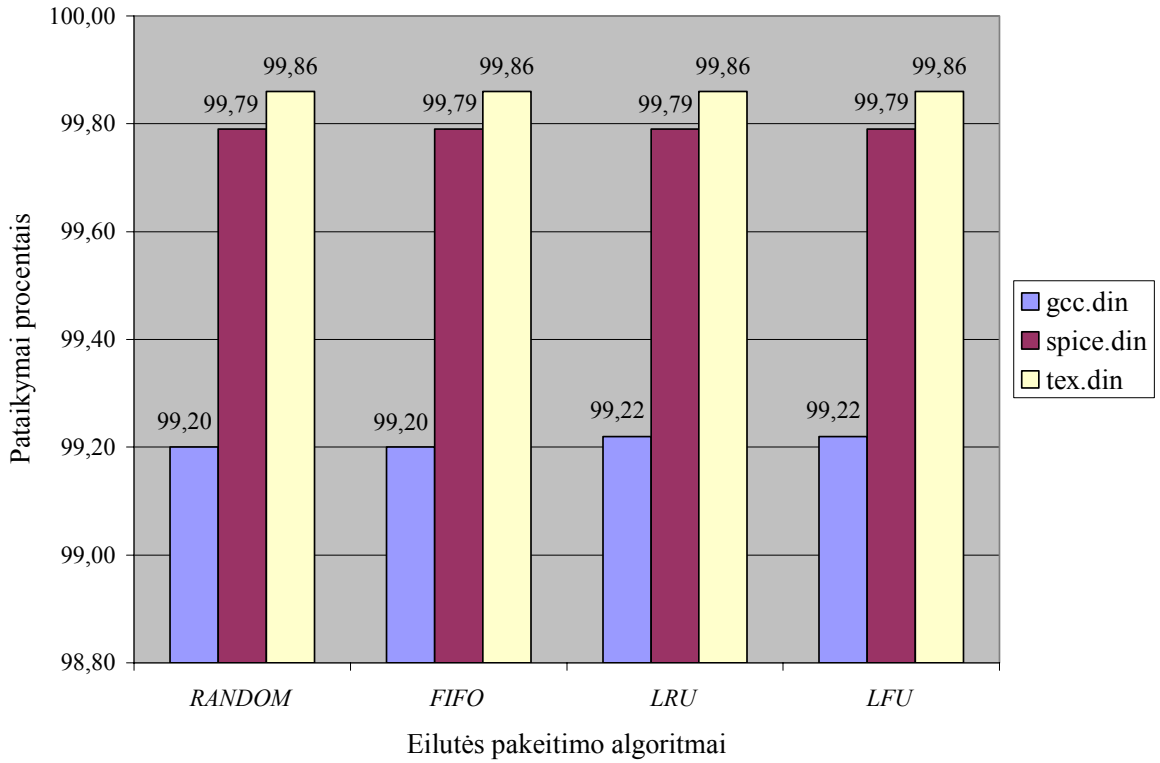
Toliau testavome iš dalies asociatyviąją 2 krypčių atmintinę, keičiant eilutės pakeitimo algoritmus ir grafiškai pavaizdavome pataikymo dažnio priklausomybę nuo iš dalies asociatyviosios 2 krypčių atmintinės eilutės pakeitimo algoritmų. Imitatoriuje pasirinkome, kad spartinančiosios atmintinės talpa yra 256KB. Gauti rezultatai pateikti 3.6 paveiksle.



3.6 pav. Pataikymų priklausomybė nuo iš dalies asociatyviosios 2 krypčių atmintinės eilutės pakeitimo algoritmų

Iš atlikto tyrimo rezultatų, parodytų 3.6 paveiksle, matyti, kad pataikymo dažnis priklauso nuo testinių duomenų (vykdomos programos kodo). Didelių skirtumų tarp testuotų algoritmų nematyti.

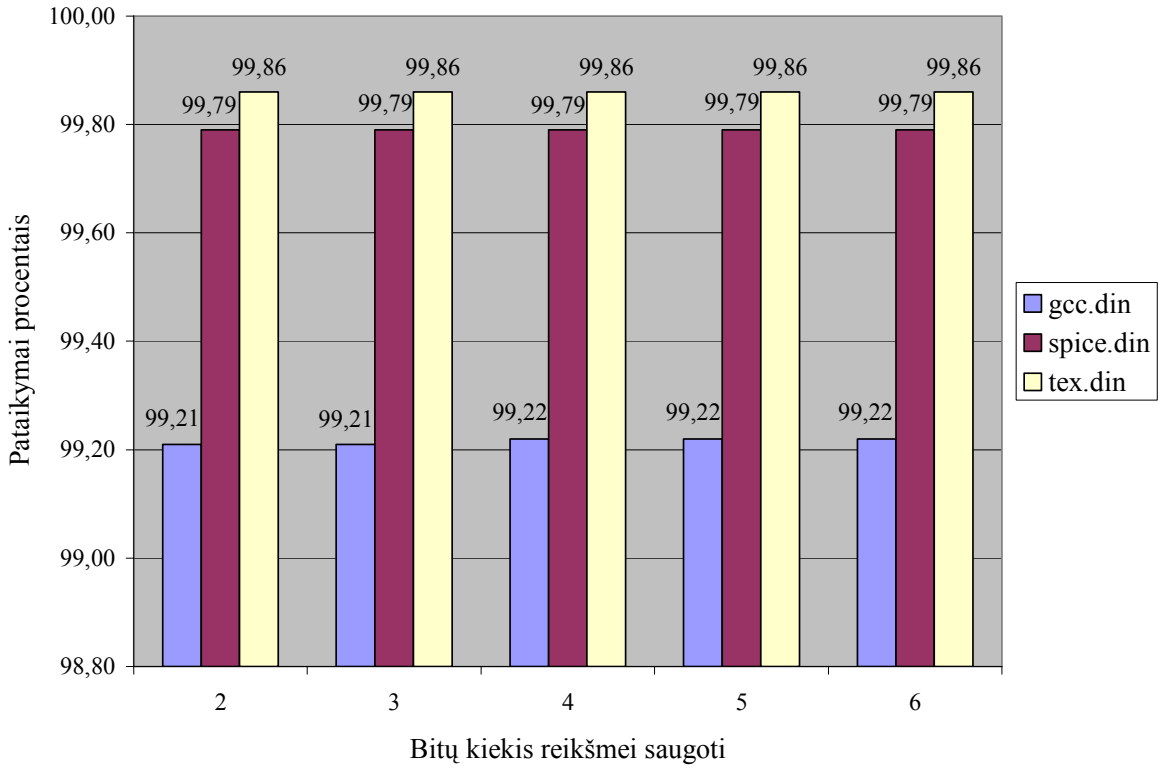
Toliau testavome iš dalies asociatyviąją 4 krypčių atmintinę, keičiant eilutės pakeitimo algoritmus ir grafiškai pavaizdavome pataikymo dažnio priklausomybę nuo iš dalies asociatyviosios 4 krypčių atmintinės eilutės pakeitimo algoritmų. Imitatoriuje pasirinkome, kad spartinančiosios atmintinės talpa - 256KB. Gauti rezultatai pateikti 3.7 paveiksle.



3.7 pav. Pataikymų priklausomybė nuo iš dalies asociatyviosios 4 krypčių atmintinės eilutės pakeitimo algoritmų

Gautus tyrimo rezultatus, parodytus 3.7 paveiksle, palyginus su 3.6 paveiksle pateiktais rezultatais matyti, kad pataikymo dažnis padidėjo tik viename testiniame rinkinyje (*gcc.din*), o kituose liko nepakitęs.

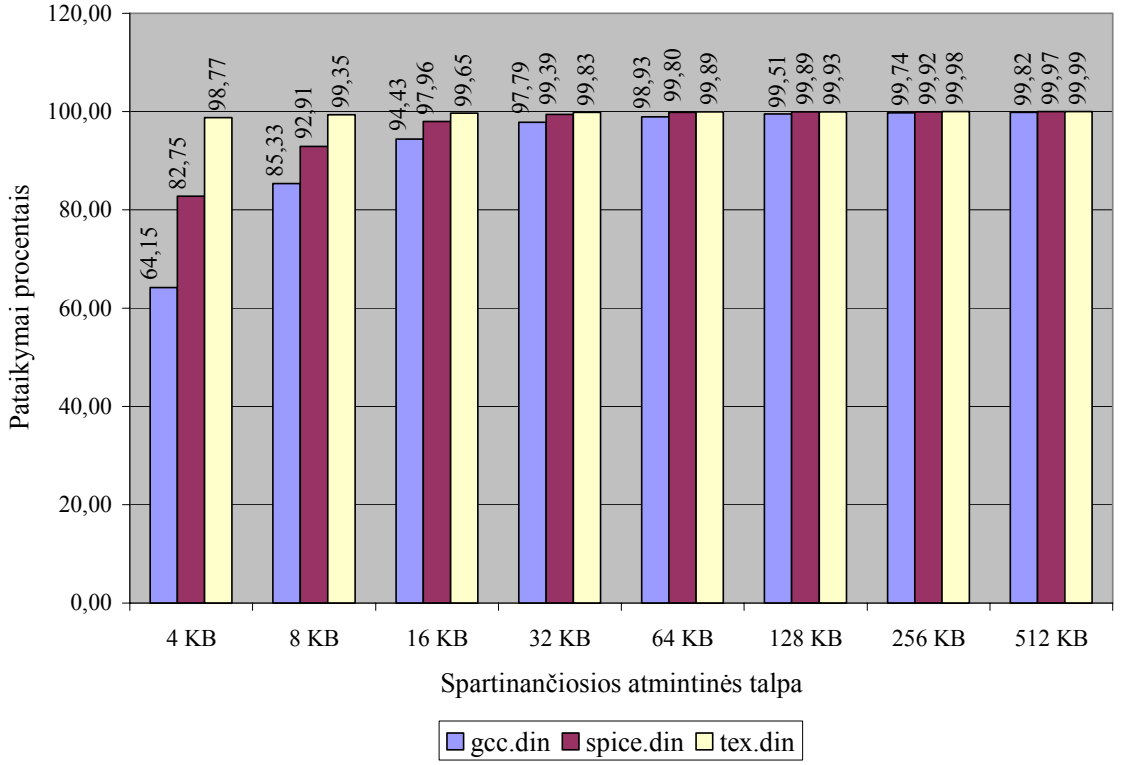
Toliau testavome iš dalies asociatyviąją 4 krypčių atmintinę, keičiant *LFU* algoritmo bitų skaičių reikšmei saugoti ir grafiškai pavaizdavome pataikymo dažnio priklausomybę nuo iš dalies asociatyviosios 4 krypčių atmintinės eilutės pakeitimo *LFU* algoritmo reikšmei saugoti bitų skaičiaus. Imitatoriuje pasirinkome, kad spartinančiosios atmintinės talpa - 256KB, o eilutės pakeitimo algoritmas - *LFU*. Gauti rezultatai pateikti 3.8 paveiksle.



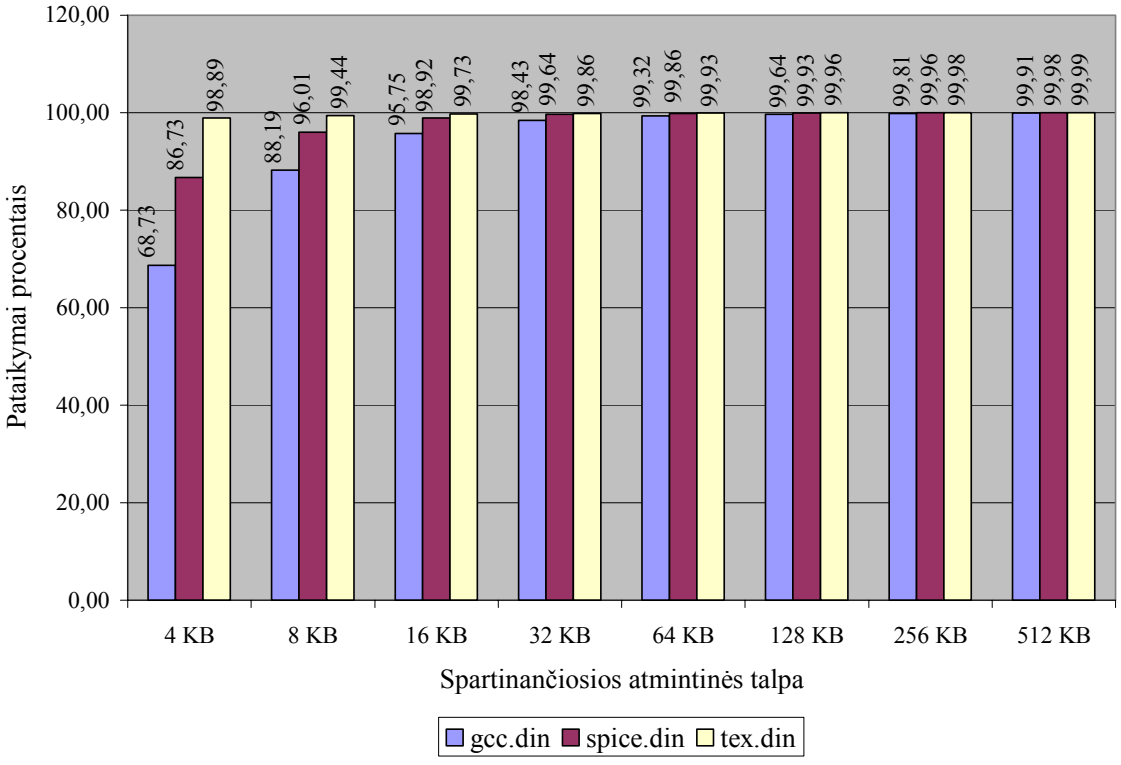
3.8 pav. Pataikymų priklausomybė nuo iš dalies asociatyviosios 4 krypčių atmintinės eilutės pakeitimo *LFU* algoritmo bitų skaičiaus

Iš atlikto tyrimo rezultatų, parodytų 3.8 paveiksle, matyti, kad pataikymo dažnis priklauso nuo testinių duomenų (vykdomos programos kodo) ir labai mažai priklauso nuo *LFU* algoritmui skirtų bitų kiekio reikšmei saugoti.

Toliau testavome tiesioginio atitikimo atmintinę ir iš dalies asociatyviąją 2 krypčių atmintinę, keisdami spartinančiosios atmintinės talpą. Grafiškai pavaizdavome pataikymo dažnio priklausomybę nuo spartinančiosios atmintinės talpos. Imitatoriuje buvo pasirinkta, kad eilutės pakeitimo algoritmas *LR*U. Gauti rezultatai pateikti 3.9 ir 3.10 paveiksluose.



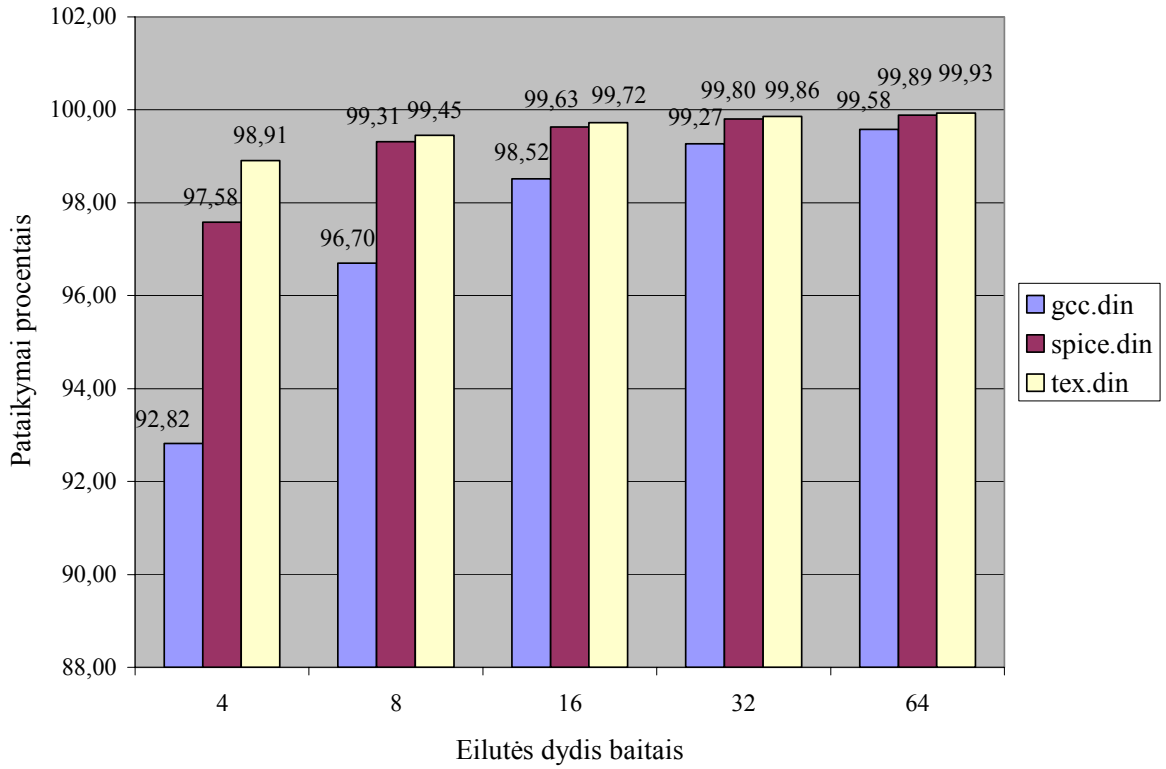
3.9 pav. Pataikymų priklausomybė nuo tiesioginio atitikimo atmintinės talpos



3.10 pav. Pataikymų priklausomybė nuo iš dalies asociatyvios 2 krypčių atmintinės talpos

Iš atlikto tyrimo rezultatų, parodytų 3.9 ir 3.10 paveiksluose, matyti, kad pataikymo dažnis priklauso ne tik nuo testinių duomenų (vykdomos programos kodo), bet ir nuo spartinančiosios atmintinės talpos, jos organizavimo.

Toliau testavome visiškai asociatyviąją atmintinę, kai kito eilutės dydis baitais, ir grafiškai pavaizdavome pataikymo dažnio priklausomybę nuo eilutės dydžio baitais. Imitatoriuje buvo pasirinkta, kad eilutės pakeitimo algoritmas - *FIFO*, o spartinančiosios atmintinės talpa - 256KB Gauti rezultatai pateikti 3.11 paveiksluose.



3.11 pav. Pataikymų priklausomybė nuo visiškai asociatyviosios atmintinės eilutės dydžio

Iš tyrimo rezultatų, parodytų 3.11 paveiksle, matyti, kad pataikymo dažnis didėja, didėjant spartinančiosios atmintinės eilutei. Tačiau pataikymo dažnis taip pat priklauso ir nuo testinių duomenų rinkinių ypatumų (vykdomos programos kodo).

PAGRINDIAI DARBO REZULTATAI IR IŠVADOS

1. Darbe analizuojama spartinančiosios atmintinės įtaka kompiuterio našumui tiek teoriniu, tiek praktiniu požiūriu, pastaruoju atveju pateikti našumo testų rezultatai. Gretinant teorinius ir testų rezultatus, įvertinta įvairių faktorių įtaką hierarchinės atminčių sistemos darbui.
2. Pateikta hierarchinės atminties sistemos našumą nusakanti išraiška leidžia paskaičiuoti užklausos laiką, įvertinti atskirų lygmenų darbo spartą, pataikymo dažnį. Pagal šią išraišką gauti rezultatai rodo maksimalią spartą, neįvertinant techninės realizacijos detalių. Matematinės išraiškos teigia ir gauti testiniai rezultatai rodo, kad sparčiausiai veikia pirmojo lygmens spartinančioji atmintinė, antrojo lygmens spartinančioji atmintinė lėtesnė už pirmąją 2 kartus (50%), o pagrindinės atminties sparta nusileidžia pastarajai apie 22 kartus. Pagrindinės atmintinės sparta 11 kartų lėtesnė už antrojo lygmens spartinančiosios atmintinės spartą.
3. Testavimo rezultatai rodo, kad sparčiausiai įvykdoma skaitymo operacija, lėčiau - rašymo operacija, lėčiausiai – kopijavimo, nes ji apima skaitymo ir rašymo operacijas.
4. Kompleksinį įvertinimą galima gauti naudojant specialias testines programas (pvz., *RightMark Memory Analyzer*). Gautus rezultatus reikia perskaičiuoti, jei norime juos pateikti įprastais pralaidumo vienetais (MB/s, GB/s). Be to, testinės programos neleidžia pasirinkti spartinančiosios atmintinės parametrų (lygmenų talpos ir pan.).
5. Pateikiamas hierarchinės atminties imitatorius, skirtas spartinančiosios atmintinės našumui įvertinti, keičiant jos pagrindinius parametrus (organizacijos būdą, lygmenų talpą ir pan.).
6. Pagal spartinančiosios atmintinės imitatoriaus gautus rezultatus galima teigti, kad hierarchinės atminties sistemos darbui turi įtakos visų spartinančiosios atminties lygmenų funkcionavimas, kreipinių į atmintį skaičius, pataikymo dažnis (arba nepataikymo dažnis), spartinančiosios atmintinės organizacija, eilutės pakeitimo algoritmo, spartinančiosios atmintinės talpa, eilutės dydis, bei vykdomos programos ypatumai.

LITERATŪRA

[1] An Overview Of Cache. **Embedded Intel Architecture Papers.** - [žiūrėta 2005 m. kovo 15 d.]. Prieiga per internetą: <http://www.intel.com/design/intarch/papers/cache6.pdf>.

[2] **Беседин Д.** RightMark Memory Analyzer 3.0. 2004.04.07 - [žiūrėta 2005 m. kovo 4 d.]. Prieiga per internetą: <http://cpu.rightmark.org/download.shtml>.

[3] Borland C++ Builder Enterprice Siute Version 5.0. – [žiūrėta 2005 m. sausio 24 d.]. Prieiga per internetą <http://www.borland.com>.

[4] **Dahlgren F.** Cache-only memory Architectures. 1999 - [žiūrėta 2005 m. vasario 3 d.]. Prieiga per internetą: <http://citeseer.ist.psu.edu/307564.html>.

[5] **Delattre F.** CPU-Z version 1.26 . 2004 December - [žiūrėta 2005 m. kovo 4 d.]. Prieiga per internetą: <http://www.cpuid.com/cpuz.php>.

[6] **Engelhart K.** CPU structure and Function. - [žiūrėta 2005 m. sausio 28 d.]. Prieiga per internetą: <http://www.ee.unb.ca/kengleha/courses/CMPE4233/Notes/Chapter11.pdf>.

[7] **Gavrichenkov I.** Chooeing Optimal Memory to Math Intel Pentium 4 Processor. 2003.09.15 - [žiūrėta 2005 m. vasario 18 d.]. Prieiga per internetą: <http://www.xbitlabs.com/articles/memory/display/p4-mem.html>.

[8] **Genther K., Reischuk R.** Analysing Data Access Strategies in Cache Coherent Architectures. 1999 - [žiūrėta 2005 m. kovo 15 d.]. Prieiga per internetą: <http://citeseer.ist.psu.edu/genther99analysing.html>.

[9] **Getov V.S.** Performance Characterisation of Cache Memory Effect. 1995 - [žiūrėta 2005 m. kovo 21 d.]. Prieiga per internetą: <http://citeseer.ist.psu.edu/315225.html>.

[10] **Гук М.** Аппаратные средства IBM PC. Энциклопедия. Питер Ком, 1999. 815с.

[11] **Гук М.** Процессоры Pentium II, Pentium Pro и просто Pentium. Питер Ком, 1999. 288с.

[12] Handling Memory Cache Policy. **Integer Points Countings Philippe Clauss ICPS.** 1997 - [žiūrėta 2005 m. balandžio 4 d.]. Prieiga per internetą: <http://citeseer.ist.psu.edu/556880.html**>.**

[13] **Harman N.A.** High Performance Microprocessors. - [žiūrėta 2005 m. kovo 1 d.]. Prieiga per internetą: <http://www.cs.swan.ac.uk/~csneal/HPM/reorder.html>.

[14] Konferencija „Informacinės Technologijos 10-oji tarpuniversitetinė magistrantų ir doktorantų konferencija“ : parn. medžiaga/ ats. red. Vrubliauskas A.: Technologija, Kaunas, 2005. // 229-232 p.

[15] **Mandhani A., Cook T.A., Kremer U.** Using Cache as a Local Memory. - [žiūrėta 2005 m. vasario 8 d.]. Prieiga per internetą: <http://citeseer.ist.psu.edu/523909.html>.

[16] **Mazzucco P.** The Fundamentals Of Cache. 2000.10.17 - [žiūrėta 2005 m. kovo 7 d.]. Prieiga per internetą: <http://systemlogic.net/articles/00/10/cache/print.php>.

[17] Notes on Cache Memory. - [žiūrėta 2005 m. kovo 15 d.]. Prieiga per internetą: <http://www.bowdoin.edu/~allen/courses/cs220/lab7/notes.html>.

[18] **Osorio R., Bruguera J.** Arithmetic Coding/decoding Architecture Based on a Cache Memory. 1998 - [žiūrėta 2005 m. balandžio 26 d.]. Prieiga per internetą: <http://citeseer.ist.psu.edu/112914.html**>.**

[19] PCMark 2002. 2002.03.13 - [žiūrėta 2005 m. kovo 7 d.]. Prieiga per internetą: <http://www.3dnews.ru/download/tests/pcmark/>.

[20] **Silasi C.A.** SiSoftware Sandra 2005. 2005.01.20 - [žiūrėta 2005 m. kovo 7 d.]. Prieiga per internetą: <http://www.sisoftware.net/sandra>.

[21] **Stokes J.** Ars Technica RAM Guide. - [žiūrėta 2005 m. kovo 7 d.]. Prieiga per internetą: <http://arstechnica.com/paedia/r/ram_guide/ram_guide.part3-3.html>.

[22] **Suh G.E., Rudolph L., Dekadas S**. Dynamic Partitioning of Shared Cache Memory. 2002 - [žiūrėta 2005 m. kovo 9 d.]. Prieiga per internetą: <http://citeseer.ist.psu.edu/504811.html>.

[23] Trace-based Cache Simulation. **DEE 1061 Computer Organization**. 2003.01.20 - [žiūrėta 2005 m. kovo 7 d.]. Prieiga per internetą: < http://twins.ee.nctu.edu.tw/~tjlin/courses/co02/hw.htm>.

[24] **Wang A., Kuenning G**. The Effects of Memory-Rich Environments on File System Microbenchmarks. 2003 - [žiūrėta 2005 m. kovo 22 d.]. Prieiga per internetą: <http://fmg-www.cs.ucla.edu/reiher/papers/spects03.pdf>.

[25] **Wilkinson B.** Computer Architecture Design and performance. Prentice Hall, 1996. 465p.

[26] **Zhang H., Martonosi M.** A Mathematical Cache Miss Analysis for Pointer Data Structures. 2001 - [žiūrėta 2005 m. vasario 25 d.]. Prieiga per internetą: <http://citeseer.ist.psu.edu/413069.html>.

[27] **Zheng Y., Davis B.** Performance Evaluation of Exclusive Cache Hierarchies. Electrical & Computer Engineering Department, Michigan Technological University. 2004 - [žiūrėta 2005 m. kovo 4 d.]. Prieiga per internetą: <http://www.ece.mtu.edu/faculty/btdavis/papers/ispass04.pdf>.

TERMINŲ IR SANTRUMPŲ ŽODYNAS

BIOS (angl. Basic Input Output System) – pagrindinės kompiuterio nuostatos.

Cache – spartinančioji atmintinė.

CPU (angl. Central Processing Unit) – centrinis procesorius.

FDD (angl. Floppy Disk Drive) – magnetinių diskelių įrenginys.

FIFO (angl. First Input First Output) – pakeitimo eilės tvarka, pirmasis įėjęs – pirmasis išėjo.

HDD (angl. Hard Disk Drive) – standžiųjų diskų įrenginys.

LFU (angl. Least Frequency Used) – rečiausiai naudotos eilutės pakeitimo algoritmas.

LRU (angl. Least Recently Used) – paskutinės seniausiai naudotos eilutės pakeitimo algoritmas.

MH (angl. *Memory Hierarchy*) - atminties hierarchija.

RAM (angl. Random Access Memory) – laisvai išrenkama atmintis.

Random – atsitiktinė pakeitimo eilutės tvarka.

Tag – žyma.

Write back – atidėtas įrašymas.

Write through – ištisinis įrašymas.

1 PRIEDAS. Skaityto pranešimo 10-oje tarpuniversitetinėje magistrantų ir doktorantų konferencijoje „*INFORMACINĖS TECHNOLOGIJOS* – 2005“ tekstas

Kompiuterių hierarchinės atminties sistemos tyrimas

Vidmantas Rimavičius

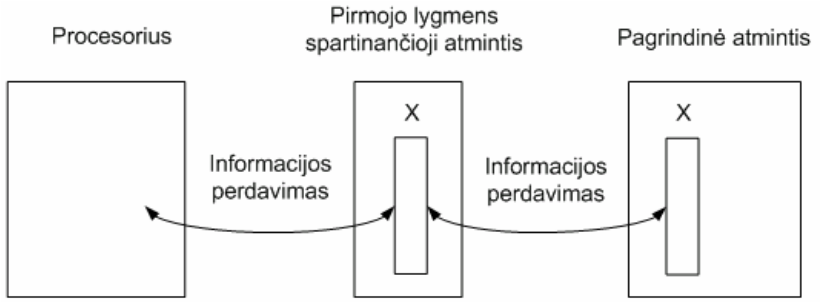
Kauno Technologijos Universitetas, Studentų 50, Kaunas

Šiuolaikinių kompiuterių atminčių sistemoje ypatingą vietą užima santykinai nedidelės talpos, bet didelės spartos atmintis, vadinama spartinančiąja atmintimi (angliškai cache). Spartinančioji atmintis yra aukščiausias hierarchinės atminčių sistemos lygmuo. Pranešime bus analizuojama jos įtaka kompiuterio našumui tiek teoriniu požiūriu, tiek praktiniu, pastaruoju atveju bus pateikti našumo testų rezultatai. Gretindami teorinius ir testinius rezultatus, įvertinsime įvairių faktorių įtaką hierarchinės atminčių sistemos darbui.

1 Kompiuterio darbo spartos problemos

Kompiuterių darbo sparta nuolat auga, tačiau šis procesas nėra paprastas. Tai paaiškinama tuo, kad darbo sparta priklauso beveik nuo visų kompiuterį sudarančių įtaisų spartos, nuo tinkamo jų subalansavimo.

Šiuolaikiniai procesoriai gali atlikti operacijas su operandais greičiau (dažnai per vietą taktą), tuo tarpu didelės talpos pagrindinės atminties sparta, apibūdinama išrinkimo laiku, gerokai atsilieka. Tiesa, yra sukurtos labai sparčios puslaidininkinės atmintinės, galinčios dirbti sparta sulyginama su procesoriaus darbo sparta, tačiau jos brangios ir todėl jas naudoti visuose kompiuteriuose neekonomiška. Problemą iš dalies galima išspręsti sukuriant hierarchinę atminties sistemą, greta pagrindinės atminties pridedant nedidelės apimties labai sparčios atminties bloką, vadinamą spartinančiąja atmintimi, įterpiamą tarp procesoriaus ir pagrindinės atminties, kaip parodyta 1 paveiksle.



1 pav. Hierarchinė atminties sistema.

Panagrinėsime, kaip kompiuterių darbo spartą įtakoja hierarchinė atminties sistema.

2 Hierarchinės atminties sistemos sparta

Akivaizdu, kad darbo sparta bus didesnė, jei reikalinga informacija (komandos ir duomenys) bus perkelta į spartinančiąją atmintį. Tai atliekama pirmojo kreipinio į šią informaciją metu. Tikimybė, kad reikalingas žodis bus rastas spartinančiojoje atmintyje, priklauso nuo vykdomos programos, spartinančiosios atminties dydžio ir struktūros. Paprastai 70-90% kreipinių metu reikalinga informacija randama spartinančiojoje atmintyje [3]. Situacija, kai reikalingas žodis randamas spartinančiojoje atmintyje, vadinama ***pataikymu***; priešingu atveju sakoma, kad ***nepataikoma*** [4]. Pastaruoju atveju reikia

kreiptis į pagrindinę atmintį.

Spartinančiosios atminties pataikymo dažnis h apibrėžiamas taip :

$$h = \frac{\textit{skaičius kreipinių, kai reikiami žodžiai randami spartinančiojoje atmintyje}}{\textit{bendras kreipinių skaičius}}$$

.

Įvertinę pataikymo dažnį, vidutinį informacijos išrinkimo iš hierarchinės atminties sistemos laiką galime išreikšti taip:

$t_a = (1-h)t_m + ht_c$;

čia t_c - spartinančiosios atminties išrinkimo laikas, o t_m - pagrindinės atminties išrinkimo laikas [1].

Jeigu nepataikymo atveju, informacija pirmiausia perkeliama į spartinančiąją atmintį ir tik po to siunčiama į procesorių, tuomet išrinkimo laikas būtų toks:

$t_a = t_c + (1-h)t_m$.

Ši išraiška rodo, kad vidutinis kreipties laikas gali būti mažinamas:

4) mažinant informacijos išrinkimo iš spartinančiosios atminties laiką;
5) pažinant nepataikymų dažnį;
6) spartinančiojoje atmintyje įvykus pataikymui.

Aukščiau pateikta išraiška gali būti detalizuota. Spartinančiosios atminties užklausos laiką sudaro: spartinančiosios atminties išrinkimo laikas t_{ci} ir informacijos nuskaitymo iš spartinančiosios atminties laikas t_{cr} . Tuomet vidutinis užklausos laikas gaunamas :

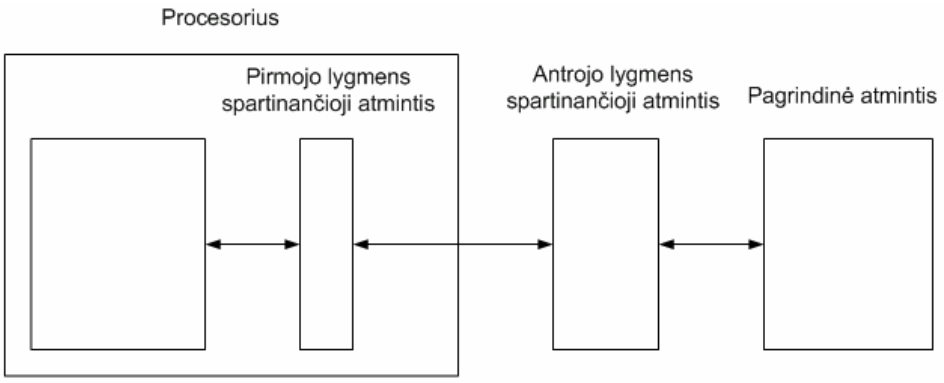
$t_a = t_{ci} + ht_{cr} + (1-h)t_m$ arba $t_a = h(t_{ci} + t_{cr}) + (1-h)(t_{ci} + t_m)$ [1].

Užklausos laikas pataikymo atveju (kai duomenys yra spartinančiojoje atmintyje) yra $t_{ci} + t_{cr}$, o nepataikymo atveju - $t_{ci} + t_m$. Aukščiau pateiktą išraišką galima pertvarkyti taip :

$$t_a = t_{ci} + t_{cr} + (1-h)(t_m - t_{cr}) .$$

3 Kelių lygmenų spartinančioji atmintis

Dabartiniu metu kompiuteriuose naudojama dviejų ar net trijų lygmenų spartinančioji atmintis. Didesnės talpos antrojo (ir trečiojo) lygmens spartinančioji atmintis įterpiama tarp pirmojo lygmens spartinančiosios atminties ir pagrindinės atminties, kaip parodyta 2 paveiksle .



2 pav. Spartinančiosios atmintis struktūra su antrojo lygmens spartinančia atmintimi.

Vykdant komandas, procesorius pirma kreipiasi į pirmojo lygmens spartinančiąją atmintį; jeigu reikiamos informacijos

jis čia nerado, tuomet bus kreipiamasi į antrojo lygmens spartinančiąją atmintį.

Įvertinę antrojo lygmens spartinančiąją atmintį, vidutinį užklausos laiką galime užrašyti taip :

$$t_a = t_{c1} + (1-h_1)t_{c2} + (1-h_2)t_m;$$

čia išskleidėme t_c, įvertindami du spartinančiosios atminties lygmenis:

$$t_c = t_{c1} + (1-h_1)t_{c2},$$

kur t_{c1} yra pirmojo lygmens spartinančiosios atminties užklausos laikas, t_{c2} - antrojo lygmens spartinančiosios atminties užklausos laikas, t_m - pagrindinės atminties užklausos laikas, h_1 - pataikymo į pirmojo lygmens spartinančiąją atmintį dažnis, h_2 - bendras pataikymo į dviejų lygmenų spartinančiąją atmintį dažnis, kai šios spartinančios atmintys sudaro vienalytę spartinančiųjų atminčių sistemą [1]. Pataikymo dažnis h_1 bus didesnis negu h_2.

4 Testai

Spartinančiosios atminties hierarchijos testavimui buvo panaudota testavimo programinis paketas „CPU-Z“ [2]. Testuotas personalinis kompiuteris Pentium III: taktinis dažnis 500MHz, 128 MB pagrindinė atmintis ir dviejų lygmenų spartinančioji atmintis.

```
Cache latency computation, ver 1.0
www.cpuid.com

Computing ...


stride 4       8       16      32      64      128     256     512
size (Kb)
1      3       3       3       3       3       3       3       3
2      3       3       3       3       3       3       3       3
4      3       3       3       3       3       3       3       3
8      3       3       3       3       3       3       3       3
16     3       3       3       3       3       3       3       3
32     3       4       5       7       7       7       7       7
64     3       4       5       7       7       7       7       7
128    3       4       5       7       7       7       7       7
256    7       15      27      44      44      45      46      48
512    12      19      40      65      66      67      70      74
1024   13      22      41      66      66      67      70      74
2048   11      22      41      66      66      67      70      74
4096   13      22      41      67      67      67      70      74
8192   13      22      41      66      67      68      70      74
16384  12      22      41      66      67      68      70      74
32768  12      22      41      66      67      68      80      74

2 cache levels detected
Level 1                size = 16Kb     latency = 3 cycles
Level 2                size = 128Kb    latency = 7 cycles
```

3 pav. Gaišties laikas taktais, kai naudojami abu spartinančiosios atminties lygmenys

3 paveiksle parodyta spartinančiosios atminties gaišties laikai taktais, kuomet duomenys į spartinančiąją atmintį skaitomi baitais, tai parodyta eilutėje „stride 4 8 16 32 64 128 256 512“. Pirmojo lygmens spartinančioji atmintis užima 16KB. Spartinančiosios atminties pirmojo lygmens gaišties laikas yra 3 taktai. Antrojo lygmens spartinančioji atmintis užima 128KB, josios duomenų skaitymo gaišties laikas yra 7 taktai. Atlikto testo gauti rezultatai parodo, kad sparčiausiai veikianti atmintis yra pirmojo lygmens, negu antrojo.

Matavimo vienetą „taktą“ galima nesunkiai perskaičiuoti kita laiko vienetą „sekundę“. Spartinančiosios atminties gaišties laikas sekundėmis (arba sekundės dalimis) bus:

$$laikas(s) = \frac{1}{procesoriaus\ \ dažnis}.$$

Įstatę į formulę procesoriaus taktinį dažnį (500MHz) ir atlikę veiksmus gausime, kad spartinančiosios atminties vieno takto trukmė yra $2ns$. Įvertinus tai, kad spartinančiosios atminties pirmojo lygmens taktų skaičius yra 3, o vieno takto trukmė $2ns$, tuomet pirmojo lygmens gaišties laikas bus $6ns$. Spartinančiosios atminties atrojo lygmes taktų skaičius yra 7, perskaičiavę

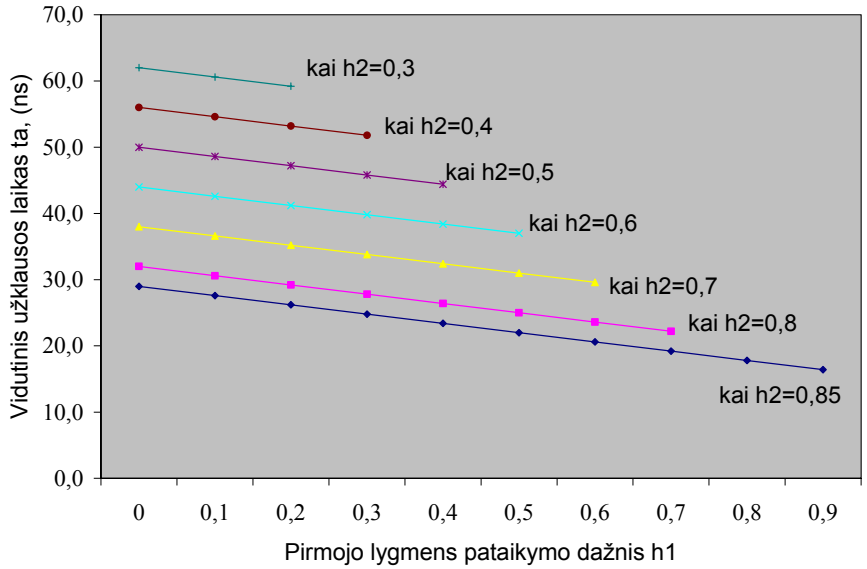
taktų skaičių į laiko vienetą sekundę gausime, kad spartinančiosios atminties antrojo lygmens gaišties laikas yra $14ns$.

Ištyrėme, kad pagrindinės atminties greitis yra 100MHz, o duomenų išrinkimas (eilutės išrinkimas užima 3 taktus, o stulpelio išrinkimas, įskaitant ir vėlinimą, užima 3 taktus)užima 6 taktus. Pasinaudoję aukščiau pateikta formulę gausime, kad pagrindinės atminties gaišties laikas yra $60ns$.

Pasinaudoję tyrimo rezultatais paskaičiuosime vidutinį išrinkimo laiką, kuomet spartinančiosios atminties pirmojo lygmens gaišties laikas $t_{c1} = 6ns$, antrojo lygmens $t_{c2} = 14ns$, o pagrindinės atminties $t_m = 60ns$. Pasirenkame, kad spartinančiosios atminties pirmojo lygmens pataikymo dažnis h_1 kinta nuo 0 iki 0,9 , o spartinančiosios atminties antrojo lygmens pataikymo dažnis h_2 kinta nuo 0,3 iki 0,85 , tuomet vidutinį užklausos laiką t_a pritaikę formulę

$$t_a = t_{c1} + (1 - h_1)t_{c2} + (1 - h_2)t_m$$

gausime kaip parodyta 4 paveiksle.



4 pav. Hierarchinės atminties su dviejų lygmenų spartinančiąja atmintimi vidutinis išrinkimo laikas

Išjungus antrojo lygmens spartinančiąją atmintį (iš pagrindinių kompiuterio nustatymų - BIOS), buvo atliktas analogiškas testas su ta pačia testavimo programa „CPU-Z“. Gauti testo rezultatai sulyginti su prieš tai atliktu testo rezultatais. Spartinančiosios atminties pirmajame lygmenyje testo rezultatai sutapo, kuomet testuojant antrasis spartinančios atminties lygmuo buvo išjungtas.

Tyrinėtas personalinis kompiuteris išjungti pirmojo spartinančiosios atminties lygmens neleido.

5 Išvados

Hierarchinės atminties sistemos našumą nusakanti išraiška leidžia paskaičiuoti užklausos laiką, įvertinant atskirų lygmenų darbo spartą, pataikymo dažnį. Pagal jas gauti rezultatai rodo maksimalią spartą, neįvertinant techninės realizacijos detalių. Matematinės išraiškos teigia ir gauti testiniai rezultatai rodo, kad sparčiausiai veikia pirmojo lygmens spartinančioji atmintis, lėtesnė antrojo lygmens spartinančioji atmintis, o lėčiausia – pagrindinė atmintis.

Kompleksinį įvertinimą galima gauti naudojant specialias testines programas (pvz., CPU-Z). Gautus rezultatus reikia perskaičiuoti, jei norime juos pateikti įprastais pralaidumo vienetais (ns, MB/s, GB/s). Be to, testinė programa neleidžia pasirinkti spartinančiosios atminties parametrų (pataikymo dažnio, lygmenų talpos ir pan.). Tam reikėtų sukurti hierarchinės atminties imitatorių.

Hierarchinės atminties sistemos darbui turi įtakos visų spartinančiosios atminties lygmenų funkcionavimas, kreipinių į atmintį skaičius, pataikymo dažnis (arba nepataikymo dažnis).

Literatūros sąrašas

[1] **Wilkinson B.** Computer Architecture Design and performance. Prentice Hall, 1996, 465p.

[2] **Delattre F.** CPU-Z version 1.26. 2004 December. – [žiūrėta 2005.03.04]. Prieiga per internetą <http://www.cpuid.com/cpuz.php>

[3] **Гук М.** Процессоры Pentium II, Pentium Pro и просто Pentium. Питер Ком, 1999, 288с.

[4] **Mazzucco P.** The Fundamentals Of Cache. 2000.10.17. [žiūrėta 2005.03.07] Prieiga per internetą <http://systemlogic.net/articles/00/10/cache/print.php>

The study of computer hierarchical memory

Annotation

Relatively small but very fast memory called *cache* takes a specific position in modern computer memories system. *Cache* is a highest level of hierarchical memories system. We analyze *cache's* influence on computer efficiency from both theoretical and practical point of view, the latter to be supported with efficiency test results. Comparing the theoretical and test results we will evaluate how different factors can impact operation of hierarchical memories systems.

2 PRIEDAS. Testinės bylos fragmentas

```
2 408ed4
0 10019d94
2 408ed8
1 10019d88
2 408edc
0 10013220
2 408ee0
2 408ee4
1 100230b8
2 408ee8
0 10013220
2 408eec
2 408ef0
2 408ef4
1 10013220
2 408ef8
1 100230bc
2 408efc
1 100230c0
2 408f00
0 7ffd8bec
2 408f04
2 408f08
2 408f0c
2 408f10
2 408f14
2 408f18
2 408f1c
2 408f20
2 408f24
2 408f28
2 408f2c
1 100230d0
2 408f30
2 408f34
2 408f24
2 408f28
2 408f2c
1 100230cc
2 408f30
2 408f34
2 408f24
2 408f28
2 408f2c
1 100230c8
2 408f30
2 408f34
2 408f24
2 408f28
2 408f2c
```