KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

Rima Strelčiūnienė

Miglotosios logikos grandžių ir struktūrų modelių sudarymas

Magistro darbas

Darbo vadovas
Doc. V.Petrauskas

Kaunas
2004

Turinys

1. ĮVADAS

Kasdieniniame gyvenime dažnai susiduriame su negriežtai apibrėžtais reiškiniais, kuriuose sunku įžvelgti ribą tarp tiesos ir netiesos, t. y. nėra taip, kad reiškinys arba vyksta arba ne, jis gali vykti ir iš dalies. Tokius netikslius, neaiškius reiškinius aprašyti aibių teorija pagrįsta matematika yra sudėtinga, o be tikslaus matematinio modelio valdyti neaiškius procesus yra sudėtinga arba visai neįmanoma. Tokiu atveju išeitis gali būti rasta panaudojant miglotąją logiką, kuri leidžia matematiškai apibrėžti dalinį teisingumą, netikslumą panaudojant lingvistinius kintamuosius. Pasitelkiant miglotojų aibių teoriją, lingvistinę, kasdieninės kalbos informaciją galima perteikti kompiuteriui, o tai leidžia lengviau ir suprantamiau formuoti įvairias valdymo ir kitas užduotis.

Vienas pačių svarbiausių miglotosios logikos elementų yra lingvistiniai kintamieji, kurie yra suformuojami iš eilės persidengiančių miglotojų aibių, fizikine prasme apibūdinančių tą kintamąjį. Pvz. lingvistinis kintamasis gali būti sudaromas iš miglotojų aibių „Žemas“, „Vidutinis“ ir „Aukštas“. Atskirose situacijose miglotosios aibės gali turėti skirtingas reikšmių sritis, skirtingas savybes, nes pvz. „Vidutinis“ slėgis kalbant apie orą tikrai fizikine prasme nesutaps su „Vidutiniu“ slėgiu kalbant apie slėgį dujų balionuose.

Pastaruoju metu sudėtingoms automatizuoto valdymo sistemoms valdyti dažnai naudojami valdikliai su miglotąja logika. Tai įgalina pasiekti žymiai geresnių valdymo charakteristikų nei tradiciniais metodais. Kodėl miglotoji logika tinkamesnė sudėtingiems procesams valdyti? Miglotosios logikos teorija buvo pradėta vystyti 1965m. Berklio Universitete (JAV). Prieš tai valdymo logika buvo kuriama Aristotelio mokymo pagrindu, kad bet kuris elementas arba yra, arba nėra aibės narys. Tačiau realiai dažnai elementai negali būti griežtai apibrėžti ir suklasifikuoti pagal kažkokį kriterijų. Tokio griežto klasifikavimo nereikalauja miglotoji logika ("fuzzy" ang. reiškia neaiškus, neraiškus). Ir nors miglotoji logika yra griežtai apibrėžta matematiškai, ji leidžia sistemų valdymui naudoti lingvistines sąvokas. Kadangi lingvistinis sistemos apibrėžimas yra lengvesnis nei matematinis modelis, miglotoji logika yra patrauklesnė sudėtingose valdymo sistemose. Ja galima aprašyti tokius parametrus kaip „lėtai“, „neseniai“, „sumažinti“ ir pan.

Panašiai kaip tradicinėje aibių teorijoje, miglotosios logikos aibėms yra taikomi tokie veiksmai, kaip aibių pjūvis, suma ir paneigimas. Visi veiksmai atliekami ne su pagrindiniais aibės porų elementais, o su priklausomumo aibei reikšmėmis. Tradicinėje aibių teorijoje sudėtingi veiksmai atliekami naudojantis kalkuliatoriais. Norint geriau suvokti veiksmus su

miglotosios logikos aibėmis, sukurtas miglotosios logikos kalkuliatorius. Darbo „Miglotosios logikos grandžių ir struktūrų modelių sudarymo“ paskirtis – sumodeliuoti kalkuliatorių atliekanti aritmetinius veiksmus su miglotosios logikos elementais. Ši programa leistų atlikti tokius veiksmus su miglotosios logikos elementais :
sudėties (+) , atimties (-), daugybos (*) , ir dalybos (:).

Darbo „Miglotosios logikos grandžių ir struktūrų modelių sudarymo“ paskirtis – sumodeliuoti kalkuliatorių atliekanti aritmetinius veiksmus su miglotosios logikos elementais. Ši programa leistų atlikti tokius veiksmus su miglotosios logikos elementais :
sudėties (+) , atimties (-), daugybos (*) , dalybos (:).

Darbe pateiktos penkios pagrindinės dalys: analitinė, projektine, vartotojo dokumentacija, miglotosios logikos kalkuliatoriaus įvertinimas, išvados.

Pirmoje dalyje atlikta išsami miglotosios aritmetikos veiksmų analizė, išskirti jos veiksmai taikytini projektuojant kalkuliatorių.

Antroje dalyje išanalizuoti vartotojo reikalavimai bei išskirti funkciniai reikalavimai. Parengta sistemos specifikacija. Sudarytas projekto atlikimo grafikas. Nustatytos projekto rizikos ir jų išvengimo priemonės. Aprašytas atliktas testavimas. Pateiktas sistemos vystymo planas.

Trečioje dalyje parašyta vartotojo dokumentacija.

Ketvirtoje dalyje atlikta išsami kalkuliatoriaus veikimo analizė, išskirti jos privalumai bei trūkumai, taikymo sritys.

Penktoje dalyje pateiktos išvados.

2. MIGLOTOSIOS LOGIKOS GRANDŽIŲ IR STRUKTŪRŲ MODELIŲ ANALIZĖ

2.1. Uždavinio analizė

Pagrindinis darbo uždavinys: sumodeliuoti kalkuliatorių atliekanti veiksmus panaudojant miglotosios logikos aritmetinius principus.

Tam, kad galėtume kurti programinės įrangos projektą, pirmiausiai turime išsianalizuoti miglotosios logikos principus bei galimybes.

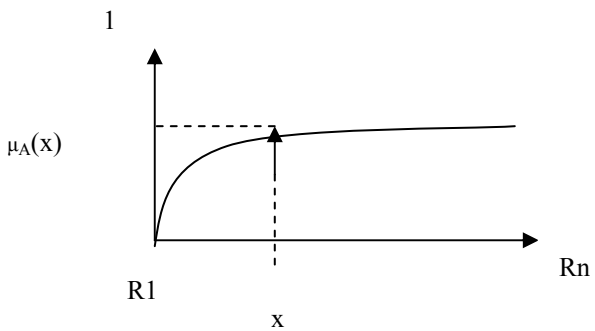
Žemiau pateikiami miglotosios logikos teoriniai aspektai.

2.1.1. Miglotosios aibės

Tradicinėje aibių teorijoje elementas gali arba priklausyti konkrečiai aibei, arba ne. Jis negali tik iš dalies priklausyti tai aibei. Tuo tarpu neapibrėžtumų logika leidžia elementui ir iš dalies priklausyti aibei, suteikiant jam atitinkamą priklausomumo koeficientą. Elemento priklausomybė konkrečiai aibei yra išreiškiama priklausomumo funkcija μ(), kuri gali įgyti reikšmes iš intervalo [0,1], kur 0 reiškia visišką nepriklausomybę,1 – visišką priklausomybę, o tarpinės reikšmės – dalinę priklausomybę. Miglotoji aibė A yra sutvarkytų porų aibė:

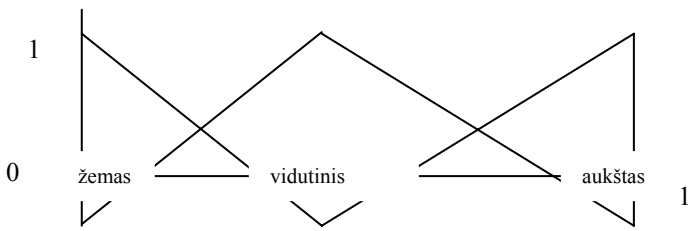
$$A=\{(x,\ \mu_A(x))|\ x \in X\} \qquad (1)$$

kur x yra aibės X elementas, o $\mu_A(x)$ yra funkcija, nusakanti elemento x priklausomumą aibei X, kitaip dar vadinama miglotosios aibės paviršiumi [1].



1 pav. Aibės elementų priklausomybė nuo realiai didėjančių skaičių

1 pav. pavaizduota miglotoji aibė susideda iš trijų komponenčių: aibės reikšmių srities, kur $R_{1...n}$ yra realūs didėjantys skaičiai (horizontali ašis), aibės elemento priklausomumo tai aibei (vertikali ašis) ir neaiškios aibės paviršiaus, kuris jungia reikšmių srities elementą su priklausomumo aibei dydžiu [3].



2 pav. Miglotosios aibės reikšmių sritis

Aibės reikšmių srities elementai gali būti tiek teigiami, tiek neigiami, priklausomai nuo to, kokį fizinį dydį ta aibė aprašo. Pavyzdžiui lingvistinio kintamojo galimų reikšmių intervalas yra nuo 0 iki 1, tuomet šio kintamojo miglotosios aibės ,,Žemas“ reikšmių sritis gali būti nuo 0 iki 0,5 , aibės ,,Vidutinis” reikšmių sritis gali būti nuo 0 iki 1, o aibės ,,Aukštas” - nuo 0,5 iki 1 (2 pav.).

Kiekviena miglotoji aibė turi atskirą, jos paviršių aprašančią t.y. elementų priklausomumo aibei dydžius nustatančią funkciją μ(x), kuri parenkama priklausomai nuo norimo tikslumo, sistemos jautrumo įėjimo signalams ir paprastumo. Miglotosios logikos aibės gali būti trikampio, trapecijos, S raidės, varpo ir kitokių formų [2] . Populiariausias dėl skaičiavimo paprastumo yra trikampio formos paviršius, kurį apsprendžia trys taškai, kairys pagrindo taškas, centras ir dešinys pagrindo taškas.

Tegul F yra visų lingvistinio kintamojo galimų reikšmių aibės *U* poaibis, o x $\in U$. Parenkam tris taškus a,b ir c iš reikšmių aibės *U*, kur $a < b < c$, kur taškai *a* ir *c* yra simetriški taškui *b* ir yra lingvistinio kintamojo vienos iš miglotojų aibių reikšmių intervalo ribinės reikšmės. Tuomet elemento x *U* priklausomumo miglotajai aibei reikšmė bus apskaičiuojama pagal formulę [3]:

$\mu_A = 0$, jei $x<a$,

=(x-a)/(b-a), jei a<x<b, (2)

=(c-x)/(c-b),jei b<x<c,

=0, jei x>c.

2.1.2.Veiksmai su aibėmis

Panašiai kaip tradicinėje aibių teorijoje, miglotosioms aibėms yra taikomi tokie veiksmai, kaip aibių pjūvis, suma ir paneigimas. Visi veiksmai atliekami ne su pagrindiniais aibės porų elementais, o su priklausomumo aibei reikšmėmis.

Apibrėžkime dvi miglotąsias aibes *A* ir *B,* kurių galimų reikšmių intervalas yra *X*, o $x \in X$.

- Aibių *A* ir *B* **pjūvis** yra nauja aibė, priklausanti tam pačiam reikšmių intervalui *X*, kurios elementų priklausomumo aibei reikšmės nustatomos skaičiuojant minimumą:

$$\mu_{A \cap B}(x) = \min\{\mu_A(x), \mu_B(x)\} \quad (3)$$

arba sandaugą:

$$\mu_{A \cap B}(x) = \{\mu_A(x).\ \mu_B(x)\} \quad (4)$$

Šis operatorius yra ekvivalentus loginiam operatoriui IR, nes rezultatas gaunamas paėmus mažesniąją iš reikšmių.

- Aibių *A* ir *B* **suma** yra nauja aibė iš to pačio galimų reikšmių intervalo *X*, kurios elementų priklausomumo reikšmės formuojamos skaičiuojant maksimumą:

$$\mu_{A \cup B}(x) = \max\{\mu_A(x), \mu_B(x)\} \quad (5)$$

arba algebrinę sumą:

$$\mu_{A \cup B}(x) = \{\mu_A(x) + \mu_B(x) - \mu_A(x).\ \mu_B(x)\} \quad (6)$$

Šis operatorius yra ekvivalentus loginiam operatoriui ARBA.

- Miglotosios aibės *A* **paneigimas** yra nauja aibė iš galimų reikšmių intervalo *X*, kurios elementų priklausomumas nustatomas taip:

$$\mu_{\sim A}(x) = 1 - \mu_A(x) \quad (7)$$

Šis operatorius yra ekvivalentus loginiam paneigimui NE.

Aukščiau paminėti operatoriai yra priimti kaip standartiniai miglotojų aibių operatoriai, tačiau egzistuoja ir visa eilė alternatyvių IR, ARBA operatorių, kuriais galima reguliuoti operatorių griežtumą [3] .

Intervalo pasirinkimas. Pasirenkame analizuojamą intervalą $A=[a_1, a_2]$, tarkime kad $a_1 \le a_2$.

Tai: $A=[a_1, a_2] = \{x \mid a_1 \le x \le a_2\}$; (8)

uždarytas intervalas $[a_1, a_2] = \{x \mid a_1 \le x \le a_2\}$;

uždarytas iš kairės ir atidarytas iš dešinės $[a_1, a_2) = \{x \mid a_1 \le x < a_2\}$;

atidarytas iš kairės ir uždarytas iš dešinės $(a_1, a_2] = \{x \mid a_1 < x \le a_2\}$;

atidarytas iš kairės ir atidarytas iš dešinės $(a_1, a_2) = \{x \mid a_1 < x < a_2\}$.

Sudėtis (+) ir atimtis (-). Kai: $A=[a_1, a_2]$; $B=[b_1, b_2]$ ir $x \in [a_1, a_2]$; $y \in [b_1, b_2]$, tai $x+y \in [a_1+b_1, a_2+b_2]$.

Gauname sumos išraišką: $A (+) B = [a_1, a_2] (+) [b_1, b_2] = [a_1+b_1, a_2+b_2]$. (9)

Atimtis gaunama: $A (-) B = [a_1, a_2] (-) [b_1, b_2] = [a_1-b_2, a_2-b_1]$. (10)

Daugyba (·) ir dalyba (:). Kai $A=[a_1, a_2]$; $B=[b_1, b_2]$, tai :

$A (\cdot) B = [a_1, a_2] (\cdot) [b_1, b_2] = [a_1 \cdot b_1, a_2 \cdot b_2]$, (11)

$k (\cdot) A = [k, k] (\cdot) [a_1, a_2] = [ka_1, ka_2]$, (12)

$A (:) B = [a_1, a_2] (:) [b_1, b_2] = [a_1 : b_2, a_2 : b_1]$, (13)

jeigu $b_1=b_2= 0$ tai dalant gauname $+\infty$.

Max (v) ir min (^) operacijos :

Maksimumą turime $A (v) B = [a_1, a_2] (v) [b_1, b_2] = [a_1 v b_1, a_2 v b_2]$. (14)

Minimumą turime $A (\wedge) B = [a_1, a_2] (\wedge) [b_1, b_2] = [a_1 \wedge b_1, a_2 \wedge b_2]$. (15)

Pagrindiniai dėsniai:

Komutatyvumo	$A (+) B = B (+) A$	$A (\cdot) B = B (\cdot) A$	(16)
Asociatyvumo	$(A (+) B) (+)C= A (+)(B(+)C)$	$(A (\cdot) B) (\cdot)C= A (\cdot)(B(\cdot)C)$	(17)
Neutralumo	$A (+) 0 = 0 (+) A = A$	$A (\cdot) 1 = 1 (\cdot) A= A$	(18)
Inversijos	$A (+) \bar{A} = \bar{A} (+) A \neq 0$	$A (\cdot) A^{-1} = A^{-1} (\cdot) A \neq 1$	(19)

Skaičiavimai:

Pvz.:turime $A = [1.23, 4.56]$, $B = [2.45, 6.26]$, $C = [-3.12, 5.64]$, $D = [-4.02, -1.27]$, $E = [2, 4]$,

$F = [-4, 6]$ $G = [-6, -2]$.

Skaičiavimai:

A (+) B= [1.23, 4.56] (+) [2.45, 6.26] = [1.23+2.45, 4.56+6.26] = [3.68, 10.82],

A (-) C= [1.23, 4.56] (-) [-3.12,5.64] = [1.23-5.64, 4.56+3.12] = [-4.41, 7.68],

(A (+) B) (-)C=[3.68, 10.82] (-) [-3.12,5.64] = [-1.96, 13.94],

A (·) B = [1.23, 4.56] (·) [2.45,6.26] = [3.0135, 28.5456],

A (:) B = [1.23, 4.56] (:) [2.45,6.26] = [1.23/6.26,4.56/2.45] = [0.1965, 1.8612],

E (^) F = [2,4] (^) [-4,6] = [2^-4, 4 ^6] = [-4,4],

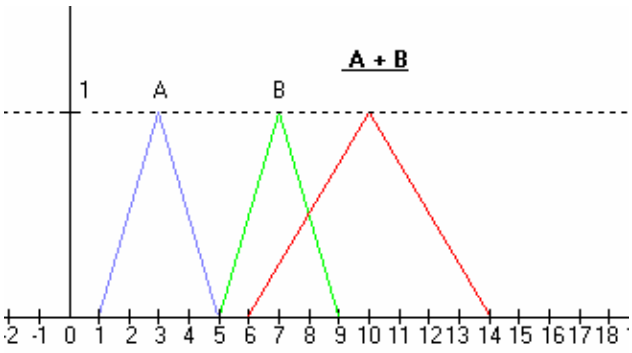
E (v) F = [2v-4, 4 v6] = [2,6],.

F (^) G = [-4^-6, 6 ^-2] = [-6,-2],

2.1.3.Grafinis veiksmų vaizdavimas

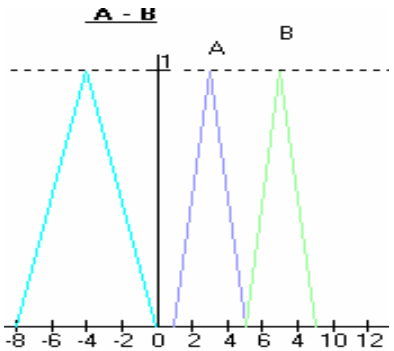
Kai A=[1, 5]; B=[5,9], tai :

Gauname sumos išraišką A (+) B = [1, 5] (+) [5,9] = [1+5, 5 +9] = [6,14].



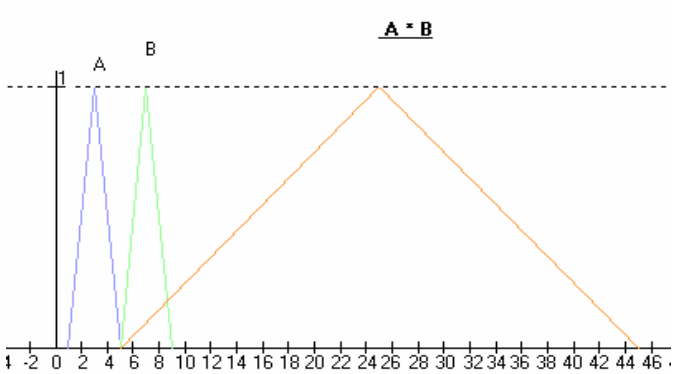
3 pav. Grafinis sumos vaizdavimas

Atimtis gaunama: A (-) B = [1, 5] (-) [5,9] = [1.9, 5 -5] = [-8,0].



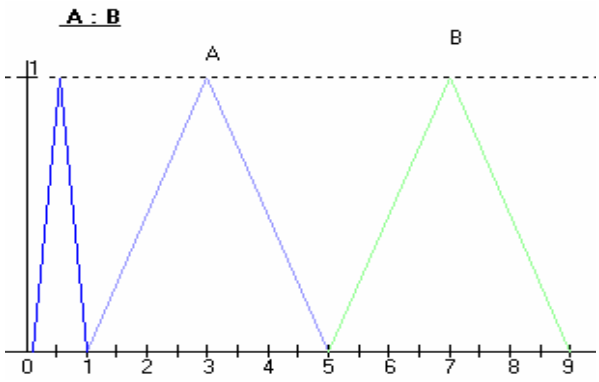
4 pav. Grafinis atimties vaizdavimas

Daugyba gaunama A (·) B = [1, 5] (·) [5,9] = [1·5, 5 ·9] = [5, 45],



5 pav.Grafinis daugybos vaizdavimas

Dalyba gaunama A (:) B = [1, 5] (:) [5,9] = [1:9, 5 :5] = [0.1, 1],



6 pav.Grafinis dalybos vaizdavimas

2.3. Miglotosios logikos kalkuliatoriaus modelio koncepcija

Vis sparčiau vystantis kompiuterinei grafikai atsirado sąvoka „erdvinis pasaulis". Kartu atsirado galimybės erdvinio pasaulio duomenis išsaugoti bylose, duomenų bazėse, juos standartizuoti. Žinoma, buvo numatytas ir jo panaudojimas inžinerijai, vizualizacijai, moksliniams , mokymo tikslams bei kitose srityse.

Kadangi į visą virtualų pasaulį galime žvelgti kaip į objektų, bei jų sąveikų visumą, natūralu, kad jį patogu realizuoti panaudojant objektinį modelį. Kalkuliatoriaus modeliavimui naudojama Visual Basic 6.0 programavimo kalba.

Pirmoji programavimo kalbos BASIC versija sukurta 1963 metais. „Microsoft" objektinė programavimo kalba VISUAL BASIC 6.0 - 1998 metais.VISUAL BASIC yra vidinė naujausių Microsoft paketų programavimo kalba. Pirmoji ir trečioji VB versijos buvo skirtos Microsoft Windows 3.x (16 bitų mašininio žodžio ilgio) terpei .

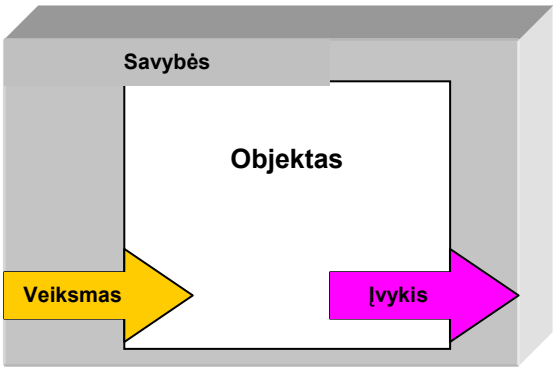
- Ji artimesnė šnekamajai kalbai nei kitos programavimo kalbos,
- ją nesunku išmokti,

- ja lengva rašyti programas, bei jas suprasti ir tobulinti,
- VB 6.0 supranta senesnėmis VB versijomis parašytas programas, o ir kitų Basic versijų programas nesunku perkelti į 6-tos versijos terpę ir suderinti jas vykdymui.

Programuojant VISUAL BASIC 6.0 galima naudotis kitų Windows programų funkcijomis, paimti iš jų duomenis ir net juos keisti, rašyti programas bendravimui Internetu, bei valdyti multimedijos įrangą. Taikomosios programos dažniausiai modeliuoja realų gyvenimą, o objektinis programavimas šiam tikslui tinkamesnis. Rašant OP programą, daug dėmesio skiriama objektams kurti, o visos programos algoritmas realizuojamas jungiant objektus į vientisą sistemą.

Kuriamojoje kalkuliatoriaus sistemoje numatoma panaudoti objektinį modelį. Šio modelio panaudojimo esmė yra perteikti miglotosios logikos aritmetinius veiksmus grafiniais modeliais. Tai savotiška dinamiška duomenų struktūra. Atskiras šios struktūros vienetas yra objektas. Objektų modelį sudaro1 pav.:

- *Veiksmai (Methods)* – Poveikis objektui.
- *Savybės (Properties)* – Objekto savybės.
- *Įvykiai (Events)* – Objekto reakcija.



7 pav. Objekto modelis

VB objektinis modelis savyje apima pagrindines objektinių modelių savybes išvardintas žemiau:

- *Lankstumas* –lengvai modifikuojamas įvedant arba išmetant objektus.
- *Paveldimumas* – hierarchiniu požiūriu žemesni objektai gali paveldėti aukštesniųjų objektų charakteristikas.
- *Grupavimas* –atskiri komponentai gali būti sugrupuoti, pavyzdžiui, žmogaus modelio grupė sudaryta iš atskirų objektų – rankų, kojų, liemens, galvos ir kt.

3. MIGLOTOSIOS LOGIKOS KALKULIATORIAUS PROJEKTAVIMAS

3.1.Programinės įrangos projektas

Pagrindiniai projekto tikslai yra šie:

- Sukurti nesunkiai įsisavinamą, nebrangią miglotosios logikos modeliavimo įrangą (kalkuliatorių), leidžiančią vartotojui modeliuoti miglotosios logikos aritmetinius veiksmus.
- Sukurti kalkuliatoriaus modelį, kurio veikimai turi kuo tiksliau atspindėti realų kalkuliatorių.
- Sukurti paprastą ir patogią vartotojo sąsają, kuri būtų suprantama, bet kuriam eiliniam vartotojui, norinčiam susipažinti, kaip funkcionuoja miglotosios logikos kalkuliatorius.

Projekto keliami uždaviniai yra šie:

- Išsiaiškinti vartotojo norus ir pageidavimus projektuojamam modeliui.
- Išsiaiškinti projekto ypatumus.
- Išsiaiškinti pasirenkamos sistemos galimybes, privalumus bei trūkumus.
- Išsiaiškinti reikalavimus naudojamai programavimo kalbai.
- Apibrėžti vartotojo sąsają.
- Nustatyti sistemos testavimo būdus.
- Atlikti testavimą.
- Atsižvelgiant į anksčiau minėtus punktus, sudaryti projekto realizavimo grafiką.

Reikalavimai techniniai įrangai bei programinei įrangai:

- 486DX/66 MHz arba greitesnis mikroprocesorius (rekomenduojama Pentium)
- VGA 640x480 arba didesnės skiriamosios gebos darbinis ekrano režimas palaikomas Microsoft Windows.
- Microsoft Windows NT 4.0 arba vėlesnė versija, arba Microsoft Windows 95 arba vėlesnė versija.
- 24 MB RAM operacinei sistemai Windows 95/98, 32 MB - Windows NT.
- Naršyklė Microsoft Internet Explorer 4.01 versijos arba vėlesnės.

Reikalavimai kietajam diskui:

- Standard Edition: tipiniam įdiegimui 48 MB, pilnam įdiegimui 80 MB.

- Professional Edition: tipiniam įdiegimui 48 MB, pilnam įdiegimui 80 MB.
- Enterprise Edition: tipiniam įdiegimui 128 MB, pilnam įdiegimui 147 MB.
- Papildomi komponentai (jeigu reikalingi): MSDN (dokumentacija): 67 MB, Internet Explorer 4.x: apie 66 MB.

3.2. Reikalavimų planas

Reikalavimų išgavimui naudosime tokį planą:

- Vartotojų apklausa, bei panašaus pobūdžio programinės įrangos privalumų ir trūkumų analizė.
- Nefunkcinių reikalavimų nustatymas.
- Prototipų kūrimas.

3.2.1. Vartotojo reikalavimai

Programinė įranga yra skirta jos vartotojui (mūsų atvejų mokslo darbams), todėl pirmiausiai ir išsiaiškinsime jo poreikius.

Vartotojo poreikiai kuriamai programinei įrangai:

Vartotojas nori programos, kuri leistų jam atlikti matematinius veiksmus su miglotosios logikos elementais:

sudėti (+) ir atimti (-);

dauginti ($\cdot$) ir dalinti (:).

Programos pradiniai duomenys ir rezultatai:

Pradiniai programos duomenys tai reikšmės kalkuliatoriaus įėjimuose vykdymo laiko momentu. Pradiniai duomenys priklauso duomenų aibei {-X, X}.

'X' - loginis kintamasis kuris gali įgyti bet kuria iš reikšmių, bet tikros jo reikšmes mes nežinome.

Rezultatai - tai po vykdymo nusistovėjusios reikšmės kalkuliatoriaus išėjimuose būsena. Rezultatu reikšmės priklauso aibei {-X, X}.

Pradinių duomenų įvedimas ir rezultatų atvaizdavimas:

Pradiniai duomenys bus išrenkami pasirinkimo sąraše iš galimu reikšmių aibes spaudžiant kalkuliatoriaus atitinkamus klavišus. Visos kalkuliatoriaus įėjimų reikšmės yra griežtai apibrėžtos ir nekintančios. Vartotojas galės rinktis iš sąrašo reikšmes atitinkančias miglotosios logikos funkcijas, todėl niekada neįves blogų reikšmių. Sumažėja galimų

klaidingų situacijų kiekis programoje. Įvedus visas norimas reikšmes, operacijos vykdymui bus skirtas specialus mygtukas. Gauti rezultatai bus matomi kalkuliatoriaus lange skirtame rezultatų išvedimui. Reikšmių aibės pasirenkamos darbo metu .

Reikalavimai vartotojo sąsajai:

· suprantama;

· neperkrauta;

· lengvai valdoma;

· greitai įsisavinama;

· patogi;

· vieninga.

Numatomo projekto funkcijos ir jo našumas:

Programines įrangos paketas modeliuoja nurodyto kalkuliatoriaus darbą. Pradiniai duomenys kalkuliatoriuje turi būti lengvai įvedami vartotojo, jam leidžiant nustatyti kiekvieno įėjimo reikšmę. Reikalavimų našumui užsakovas nenurodė, todėl pagrindiniu savo tikslu laikau paprastą programos palaikomumą, bei programos teksto aiškumą.

Reikalavimai patikimumui:

Programa turi veikti taip, kad rezultatai sutaptų su realiais rezultatais, nesigilinant į vidinį kalkuliatoriaus veikimą, o struktūrinis modulis atliekamas žingsnis po žingsnio; duomenys perduodami struktūroms, atliekančioms loginius veiksmus, šios struktūros apskaičiuoja rezultatus, kurie perduodami, kaip duomenys sekančioms struktūroms ir t. t. kol gaunami galutiniai rezultatai; jei rezultatai sutampa kalkuliatorius veikia gerai; programa neturi sukelti vartotojo sistemos „lūžimo“.

3.2.2. Nefunkciniai reikalavimai

Kadangi reikalaujama, kad šitas modelis veiktų bet kokios architektūros kompiuteryje, su bet kokia operacine sistema, tai smarkiai susiaurina galimų programavimo kalbų pasirinkimą. Projektą galima realizuoti panaudojant Visual Basic 6,0 programavimo kalbą, nes programuoti ja kalkuliatorių yra patogu.

Išnagrinėjus programinę įrangą, bei jos reikalavimus sistemoms, planuojami programinės, bei aparatūrinės įrangos reikalavimai būtų tokie:

- 486DX/66 MHz arba greitesnis mikroprocesorius (rekomenduojama Pentium)
- VGA 640x480 arba didesnės skiriamosios gebos darbinis ekrano režimas palaikomas Microsoft Windows.

- Microsoft Windows NT 4.0 arba vėlesnė versija, arba Microsoft Windows 98 arba vėlesnė versija.
- 24 MB RAM operacinei sistemai Windows 98, 32 MB - Windows NT.
- Naršyklė Microsoft Internet Explorer 4.01 versijos arba vėlesnės.

Reikalavimai kietajam diskui:

- Standard Edition: tipiniam įdiegimui 48 MB, pilnam įdiegimui 80 MB.
- Professional Edition: tipiniam įdiegimui 48 MB, pilnam įdiegimui 80 MB.
- Enterprise Edition: tipiniam įdiegimui 128 MB, pilnam įdiegimui 147 MB.
- Papildomi komponentai (jeigu reikalingi): MSDN (dokumentacija): 67 MB, Internet Explorer 4.x: apie 66 MB.

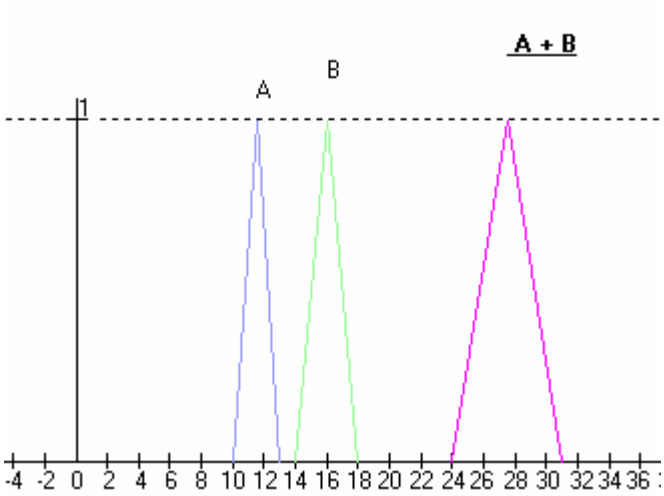
Programinė įranga:

- Programos greitis priklausys nuo turimos techninės įrangos.
- Planuojamas programos dydis: iki 2 Mb.
- OS: Windows 9x, NT, 2000, XP.
  Viena iš labiausiai paplitusių operacinių sistemų.
- Programavimo kalba: Visual Basic 6.0 .

  Viena populiariausių programavimo kalbų. Lanksti, palaiko objektinį programavimą. Be to, lengviau būtų, esant pareikalavimui, perrašyti programą kitai operacinei sistemai.
- Planuojama reikalinga techninė įranga: Min: P200, 800Mb HDD, 128 RAM, 16MB atmintis.

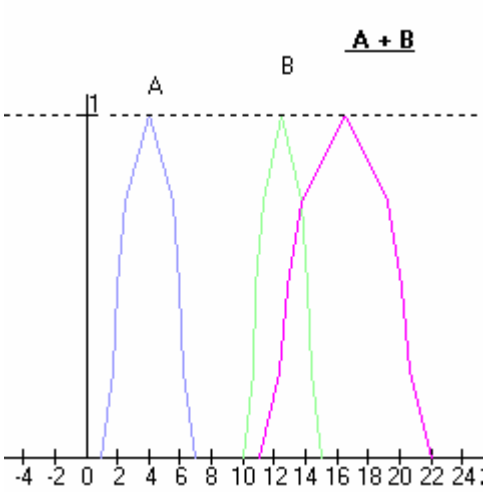
3.2.3. Prototipai

Programinės įrangos projektavimo procese labai svarbus yra prototipų vaidmuo. Buvo sukurti keli prototipai papildomų reikalavimų išgavimui. Kalkuliatoriuje miglotosios logikos veiksmai modeliuojami dviem budais. Pirmuoju atveju pasirinkta tiesinė trikampė funkcija, kurios viršūnė atitinka vieneto reikšmę. Viršūnė yra viduryje tarp pradinės ir galinės užduotos reikšmės. Paveiksle pateiktas sudėties pavyzdys.



8 pav. Miglotosios logikos sudėties pavyzdys

Antru atveju pasirinkta netiesinė varpo formos funkcija, kurios viršūnė yra viduryje užduotų pradinių reikšmių.



9 pav.Varpo formos funkcijos pavyzdys atliekant miglotosios logikos sudėties veiksmus.

3.3. Sistemos specifikacija

Formalios specifikacijos:

1. Reikalavimai, kurie gali būti suprasti dviprasmiškai:

Projekto realizavimo programinė kalba - reikalavimų specifikacijoje nėra konkrečiai nurodyta, todėl autoriai pasirenka ją patys. Šiuo atveju pasirinkta Visual Basic 6 programavimo kalba. Programos gyvavimo ciklas - netiksliai žinoma paskutinio gyvavimo

ciklo etapo (naudojimo ir priežiūros) trukmė. Nėra pakankamai specifikuoti reikalavimai testavimo algoritmui. Reikalavimai kokybei ir patikimumui: nors šie reikalavimai yra gana abstraktūs, programų kokybę ir patikimumą didele dalimi užtikrina naudojamos programinės kūrimo priemonės.

2. Formalių metodų naudojimas:

Aprašant reikalavimus formalūs metodai nenaudojami. Formalus aprašymas gerokai padidintų projekto realizavimo laiką ir sąnaudas. Projekto vykdytojams tektų papildomai mokytis formalių specifikacijų kalbų.

3. Duomenų aibės:

Įėjimo aibė {-X, X}. Čia X yra nežinoma (neapibrėžta) reikšmė . Dėl to laikome, kad paduodant X reikšmę į kurį nors elementą, išėjimo reikšmės taip pat X. Išėjimo aibė {-X, X}.

4. Naudojami miglotosios logikos veiksmai:

Sudėtis (+) ir atimtis (-).

Kai: $A=[a_1, a_2]$; $B=[b_1,b_2]$ ir $x \in [a_1, a_2]$; $y \in [b_1,b_2]$, tai $x+y \in [a_1+b_1, a_2+b_2]$.

$$A (+) B = [a_1, a_2] (+) [b_1,b_2] = [a_1+b_1, a_2+b_2]. \quad (20)$$

$$A (-) B = [a_1, a_2] (-) [b_1,b_2] = [a_1-b_2, a_2-b_1]. \quad (21)$$

Daugyba (·) ir dalyba (:).

Kai $A=[a_1, a_2]$; $B=[b_1,b_2]$, tai :

$$A (\cdot) B = [a_1, a_2] (\cdot) [b_1,b_2] = [a_1 \cdot b_1, a_2 \cdot b_2], \quad (22)$$

$$k (\cdot) A = [k, k] (\cdot) [a_1,a_2] = [ka_1, ka_2], \quad (23)$$

$$A (:) B = [a_1, a_2] (:) [b_1,b_2] = [a1:b2, a_2 :b_1], \quad (24)$$

jeigu $b_1=b_2= 0$, tai dalant gauname $+\infty$.

Atlikti max (v) ir min (^) operacijas

$$A (v) B = [a_1, a_2] (v) [b_1,b_2] = [a_1 v b_1, a_2 v b_2]. \quad (25)$$

$$A (\wedge) B = [a_1, a_2] (\wedge) [b_1,b_2] = [a_1 \wedge b_1, a_2 \wedge b_2]. \quad (26)$$

Sistemos specifikacija padės mums apibūdinti kuriamą projektą, bei nubrėžti jo realizavimo gaires. Toliau pateikiama:

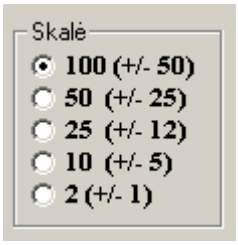
- Sistemos reikalavimų specifikacija.
- Sistemos struktūros specifikacija.
- Vartotojo sąsajos specifikacija.

3.3.1. Sistemos reikalavimų specifikacija

Įvertinus vartotojų reikalavimus bei įrangos analizės rezultatus, gaunami reikalavimai sistemai.

Reikalavimai sistemai:

- Leisti projektuoti miglotosios logikos elementus dvimatėje sistemoje.
- Leisti pasirinkti skalės ribas. Abiem kintamiesiems nurodoma ta pati skalė. Pavyzdys pateiktas paveiksliuke 10.

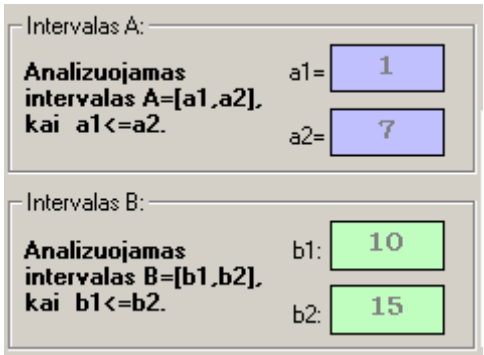


10 pav.Skalės pasirinkimo galimybės

- Leisti pasirinkti grafiko tipą. Pavaizduota 11 paveiksliuke.

11 pav. Grafiko tipo pasirinkimas

- Suteikti galimybę pasirinkti grafikų su kuriais atliekami veiksmai intervalą. Pateikta 12 pav.



12 pav.Intervalo pasirinkimas

- Leisti pasirinkti veiksmus su pasirinktais grafikais. Pateikta 13 pav.

13 pav.Operacijos parinkimas

Reikalavimai sistemos patikimumui ir kokybei:

- Programa turi pateikti kokybišką vaizdą.
- Kalkuliatoriaus atliekami veiksmai turi atitikti realius.
- Veiksmu brėžiniuose turi būti pateikiama detali informacija.

Projekto realizavimo būdai ir priemonės.

- Programavimo kalba: Visual Basic 6;
- Operacinė sistema: Win9x, Win2000;

3.3.2. Vartotojo sąsajos specifikacija

Reikalavimai vartotojo sąsajai:

· Suprantama;

· neperkrauta;

· lengvai valdoma;

· greitai įsisavinama;

· patogi;

· vieninga.

Toliau vartotojo sąsają numatoma tobulinti atsižvelgiant į vartotojų patarimus, norus vadovaujantis šiomis taisyklėmis:

1. Siekti nuoseklumo, kad informacija būtų išdėstyta nuosekliai, logiškai .
2. Paprastas klaidų apdorojimas.
3. Kad būtų lengva atšaukti veiksmus.
4. Ekrane saugoti tiktai svarbiausią ir reikalingiausią informaciją.
5. Parinkti malonias spalvų kombinacijas.
6. Turi būti galimybė pertraukti procesus išvengiant duomenų nuostolių.

Pradiniai programos duomenys, tai reikšmės kalkuliatoriaus įėjimuose vykdymo laiko momentu. Pradiniai duomenys priklauso duomenų aibei {-X, X}.'X' - loginis kintamasis, kuris gali įgyti bet kurią iš reikšmių, bet tikros jo reikšmės mes nežinome. Rezultatai - tai po vykdymo nusistovėjusios reikšmės kalkuliatoriaus išėjimuose būsena. Rezultatu reikšmės priklauso aibei {-X, X}. Pradiniai duomenys bus išrenkami pasirinkimo sąraše iš galimu reikšmių aibes spaudžiant kalkuliatoriaus atitinkamus klavišus. Visos kalkuliatoriaus įėjimų reikšmės yra griežtai apibrėžtos ir nekintančios. Vartotojas galės rinktis iš sąrašo reikšmes atitinkančias miglotosios logikos aibes, todėl niekada neįves blogu reikšmių. Sumažėja galimų klaidingų situacijų kiekis programoje. Įvedus visas norimas reikšmes, operacijos vykdymui bus skirtas specialus mygtukas. Gauti rezultatai bus matomi kalkuliatoriaus lange skirtame rezultatų išvedimui.

3.4. Projekto grafikų ir išlaidų planas

Projekto išlaidos:

Projekto išlaidos yra minimalios, kadangi projektas vykdomas kaip magistrinis – mokslinis darbas. Daugumoje projekto išlaidas sudaro laikas skirtas projektui kurti bei programai rašyti.

Projekto grafikas:

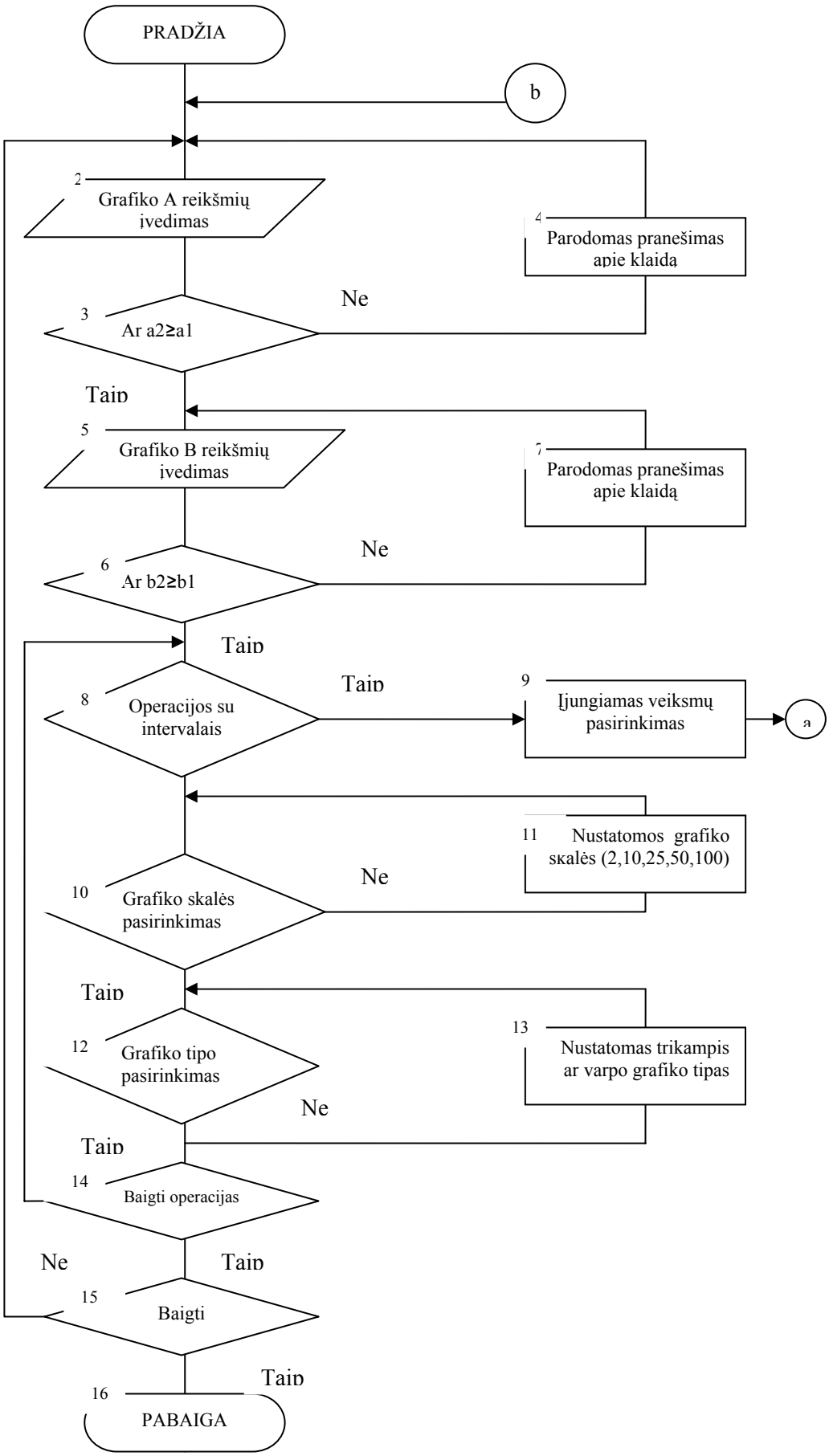
Projektas buvo kuriamas su atskiromis pertraukomis nuo 2001 metų:

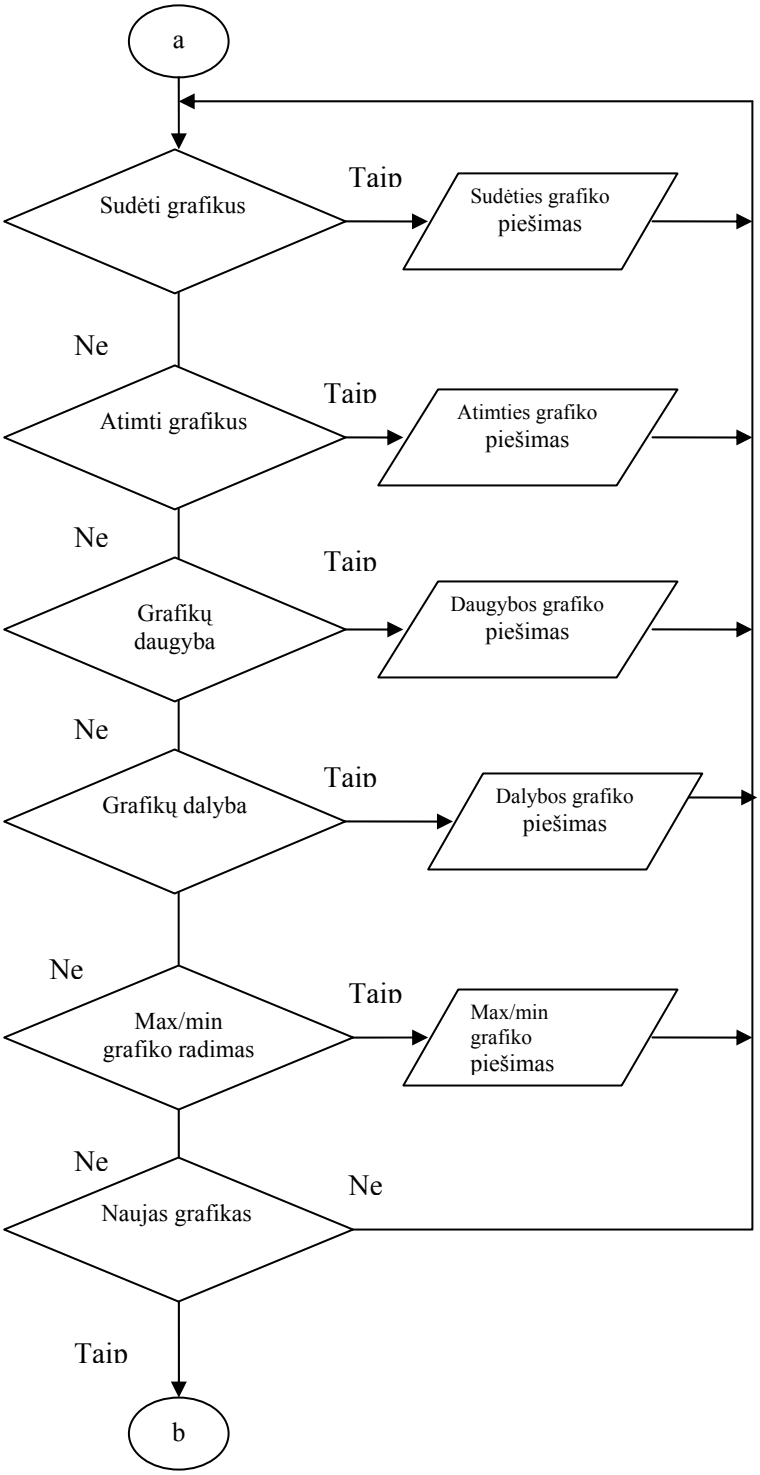
- *2001-11-01 – 2002-01-30*: sugalvota projekto idėja. Iškelti tikslai bei uždaviniai, nustatyti pradiniai reikalavimai.
- *2002-05-01 – 2002-06-30*: pradėti kurti pradiniai projekto prototipai. Toliau pildomi reikalavimai. Apibrėžta pradinė programos struktūra. Apgalvoti galimi objektai.
- *2002-09-01 – 2002-11-30*: galutinai nustatytas naudotinas objektinis modelis. Nagrinėjama ir kuriama kalkuliatoriaus koncepcija.
- *2002-12-01– 2003-08-31*: programuojami programos komponentai, programuojama vartotojo sąsaja.
- *2003-09-01 – 2003-12-01*: atliekamas testavimas, rašoma programos dokumentacija.

3.5.Programos kūrimas

Šiame projekte pasirinkta objektinė programavimo kalba Visual Basic 6.

Programos algoritmas pateiktas žemiau.


PRADŽIA
b
2
Grafiko A reikšmių įvedimas
4
Parodomas pranešimas apie klaidą
3
Ar a2≥a1
Ne
Taip
5
Grafiko B reikšmių įvedimas
7
Parodomas pranešimas apie klaidą
6
Ar b2≥b1
Ne
Taip
8
Operacijos su intervalais
Taip
9
Įjungiamas veiksmų pasirinkimas
a
11
Nustatomos grafiko skalės (2,10,25,50,100)
10
Grafiko skalės pasirinkimas
Ne
Taip
12
Grafiko tipo pasirinkimas
13
Nustatomas trikampis ar varpo grafiko tipas
Ne
Taip
14
Baigti operacijas
Ne
Taip
15
Baigti
Taip
16
PABAIGA


a
Sudėti grafikus
Taip
Sudėties grafiko piešimas
Ne
Atimti grafikus
Taip
Atimties grafiko piešimas
Ne
Grafikų daugyba
Taip
Daugybos grafiko piešimas
Ne
Grafikų dalyba
Taip
Dalybos grafiko piešimas
Ne
Max/min grafiko radimas
Taip
Max/min grafiko piešimas
Ne
Naujas grafikas
Ne
Taip
b

Pulto maketui sukurti atlikome šiuos veiksmus:

1. Sužadinome meniu komandas.
2. Norėdami sukurti grafikui skirtą vietą paspaudėme mygtuką **Picture** grafinio vaizdo objektui kurti.
3. Norėdami sukurti komandos mygtukus, operacijoms su intervalais ir naujų grafikų gavimui, pelyte paspaudėme mygtuką objektui **CommanButton** kurti ir perkėlėme pelytės žymę į tą pulto maketo **Form1** vietą, kur kūrėme mygtuką. Pelyte paspaudus mygtuką **Label1** indikatoriui kurti atlikome veiksmus kaip ketvirtame punkte.
4. Tokiu pat būdu sukūrėme visus likusius indikatorius ir išdėstėme prie mygtukų ir grafiniame fone (jais pavaizduoti skaičiai).
5. Pažymėję pultą **Form1** nustatome reikiamus pulto parametrus
6. Pulto makete nupiešėme teksto laukelius, kuriuose rašome pasirenkamus grafiko skaičius
7. Operacijoms pasirinkti (sudėties, atimties, daugybos, dalybos, max/min) sukūrėme veiksmų operacijos parinkimo lauką. Operacijų pavadinimams įrašyti naudojome indikatorius **Label.**
8. Dvigubai bakstelėję pelyte mygtukus į kodo langą įrašome programos kodą ir paprogramę. Pateikiu vieno mygtuko „Naujas grafikas“ programos kodą.

```
Private Sub Command1_Click()
Picture1.Cls
Command1.SetFocus
Text2.Text = ""
Text2.Enabled = True
Text1.Text = ""
Text1.Enabled = False
Text1.BackColor = &H808080
Text3.Text = ""
Text3.Enabled = False
Text3.BackColor = &H808080
Text4.Text = ""
Text4.Enabled = False
Text4.BackColor = &H808080
Frame1.Visible = False
Command2.Enabled = False
Label32.Visible = False
Label33.Visible = False
Label34.Visible = False
Label35.Visible = False
Label36.Visible = False
```

```
Label38.Visible = False
Label39.Visible = False
Label40.Visible = False
Label43.Visible = False
Label44.Visible = False
Label45.Visible = False
Label72.Visible = False
Label73.Visible = False
Option1.Value = False
Option2.Value = False
Option3.Value = False
Option4.Value = False
Option5.Value = False
Text2.SetFocus
Command3.Enabled = False
Frame5.Enabled = False
Frame6.Enabled = False
End Sub
```

9. Dvigubai bakstelėję pelyte darbo režimui pasirinkti skirtus mygtukus į kodo langą įrašysime programos kodą ir paprogramę:

Sudėties veiksmams atlikt Option1

```
Private Sub Option1_Click()
Picture1.Cls
Call Text1_LostFocus
Call Text4_LostFocus
Picture1.ForeColor = RGB(255, 0, 255)
If Option11.Value = True Then
S1 = (a1 + b1) - 50
s2 = (a2 + b2) - 50
Else
If Option6.Value = True Then
S1 = (a1 + b1) - 25
s2 = (a2 + b2) - 25
Else
If Option7.Value = True Then
S1 = (a1 + b1) - 12.5
s2 = (a2 + b2) - 12.5
Else
If Option8.Value = True Then
S1 = (a1 + b1) - 5
s2 = (a2 + b2) - 5
Else
If Option12.Value = True Then
S1 = (a1 + b1) - 1
s2 = (a2 + b2) - 1
End If
```

```
End If
End If
End If
End If
ds = (s2 - S1) / 2
Picture1.PSet (S1, 6.5)
ls = 5.5
For si = 1 To 4
hs = S1 + (ds / (5 - si))
Picture1.Line -(hs, ls)
ls = ls - 1
Next
Picture1.PSet (s2, 6.5)
ls = 5.5
For si = 1 To 4
hs = s2 - (ds / (5 - si))
Picture1.Line -(hs, ls)
ls = ls - 1
Next
Label34.Left = S1 + ds
Label34.Visible = True
Label35.Visible = False
Label36.Visible = True
Label37.Visible = False
Label38.Visible = False
Label39.Visible = False
Label40.Visible = False
Label43.Visible = False
Label44.Visible = False
Label45.Visible = False
Label72.Visible = False
Label73.Visible = False
End Sub
```

Atimties veiksmams atlikt Option2

```
Private Sub Option2_Click()
Picture1.Cls
Call Text1_LostFocus
Call Text4_LostFocus
Picture1.ForeColor = RGB(0, 255, 255)
If Option11.Value = True Then
sk1 = (a1 - b2) + 50
sk2 = (a2 - b1) + 50
Else
If Option6.Value = True Then
sk1 = (a1 - b2) + 25
sk2 = (a2 - b1) + 25
Else
If Option7.Value = True Then
```

```
sk1 = (a1 - b2) + 12.5
sk2 = (a2 - b1) + 12.5
Else
If Option8.Value = True Then
sk1 = (a1 - b2) + 5
sk2 = (a2 - b1) + 5
Else
If Option12.Value = True Then
sk1 = (a1 - b2) + 1
sk2 = (a2 - b1) + 1
End If
End If
End If
End If
End If
dsk = (sk2 - sk1) / 2
Picture1.PSet (sk1, 6.5)
lsk = 5.5
For ski = 1 To 4
hsk = sk1 + (dsk / (5 - ski))
Picture1.Line -(hsk, lsk)
lsk = lsk - 1
Next
Picture1.PSet (sk2, 6.5)
lsk = 5.5
For ski = 1 To 4
hsk = sk2 - (dsk / (5 - ski))
Picture1.Line -(hsk, lsk)
lsk = lsk - 1
Next
Label35.Left = sk1 + dsk
Label34.Visible = False
Label35.Visible = True
Label37.Visible = False
Label38.Visible = True
Label39.Visible = False
Label40.Visible = False
Label43.Visible = False
Label44.Visible = False
Label45.Visible = False
Label72.Visible = False
Label73.Visible = False
End Sub
```

Daugybos veiksmams atlikt Option3

```
Private Sub Option3_Click()
Picture1.Cls
Call Text1_LostFocus
Call Text4_LostFocus
```

```
Picture1.ForeColor = RGB(255, 153, 51)
If Option11.Value = True Then
sd1 = ((a1 - 50) * (b1 - 50)) + 50
sd2 = ((a2 - 50) * (b2 - 50)) + 50
End If
If Option6.Value = True Then
sd1 = ((a1 - 25) * (b1 - 25)) + 25
sd2 = ((a2 - 25) * (b2 - 25)) + 25
End If
If Option7.Value = True Then
sd1 = ((a1 - 12.5) * (b1 - 12.5)) + 12.5
sd2 = ((a2 - 12.5) * (b2 - 12.5)) + 12.5
End If
If Option8.Value = True Then
sd1 = ((a1 - 5) * (b1 - 5)) + 5
sd2 = ((a2 - 5) * (b2 - 5)) + 5
End If
If Option12.Value = True Then
sd1 = ((a1 - 1) * (b1 - 1)) + 1
sd2 = ((a2 - 1) * (b2 - 1)) + 1
End If
dsd = (sd2 - sd1) / 2
Picture1.PSet (sd1, 6.5)
lsd = 5.5
For sdi = 1 To 4
hsd = sd1 + (dsd / (5 - sdi))
Picture1.Line -(hsd, lsd)
lsd = lsd - 1
Next
Picture1.PSet (sd2, 6.5)
lsd = 5.5
For sdi = 1 To 4
hsd = sd2 - (dsd / (5 - sdi))
Picture1.Line -(hsd, lsd)
lsd = lsd - 1
Next
Label43.Left = sd1 + dsd
Label34.Visible = False
Label35.Visible = False
Label37.Visible = False
Label38.Visible = False
Label39.Visible = True
Label40.Visible = False
Label43.Visible = True
Label44.Visible = False
Label45.Visible = False
Label72.Visible = False
Label73.Visible = False
End Sub
```

Dalybos veiksmams atlikt Option4

```
Private Sub Option4_Click()
Picture1.Cls
Call Text1_LostFocus
Call Text4_LostFocus
Picture1.ForeColor = RGB(0, 0, 255)
If Option11.Value = True Then
dl1 = ((a1 - 50) / (b2 - 50)) + 50
dl2 = ((a2 - 50) / (b1 - 50)) + 50
Else
If Option6.Value = True Then
dl1 = ((a1 - 25) / (b2 - 25)) + 25
dl2 = ((a2 - 25) / (b1 - 25)) + 25
Else
If Option7.Value = True Then
dl1 = ((a1 - 12.5) / (b2 - 12.5)) + 12.5
dl2 = ((a2 - 12.5) / (b1 - 12.5)) + 12.5
Else
If Option8.Value = True Then
dl1 = ((a1 - 5) / (b2 - 5)) + 5
dl2 = ((a2 - 5) / (b1 - 5)) + 5
Else
If Option12.Value = True Then
dl1 = ((a1 - 1) / (b2 - 1)) + 1
dl2 = ((a2 - 1) / (b1 - 1)) + 1
End If
End If
End If
End If
End If
ddl = (dl2 - dl1) / 2
Picture1.PSet (dl1, 6.5)
ldl = 5.5
For dli = 1 To 4
hdl = dl1 + (ddl / (5 - dli))
Picture1.Line -(hdl, ldl)
ldl = ldl - 1
Next
Picture1.PSet (dl2, 6.5)
ldl = 5.5
For dli = 1 To 4
hdl = dl2 - (ddl / (5 - dli))
Picture1.Line -(hdl, ldl)
ldl = ldl - 1
Next
Label45.Left = dl1 + ddl
Label34.Visible = False
```

```
Label35.Visible = False
Label37.Visible = False
Label38.Visible = False
Label39.Visible = False
Label40.Visible = True
Label43.Visible = False
Label44.Visible = False
Label45.Visible = True
Label72.Visible = False
Label73.Visible = False
End Sub
```

Visas kalkuliatoriaus programos kodas pateiktas 2 priede.

3.6. Rizikos įvertinimo ir mažinimo planas

Projekto rizikos:

Reikalavimų pasikeitimas

Programos reikalavimų pasikeitimas gali turėti įtakos programos kūrimo procesui. Tačiau pagrindinė idėja nustatyta, todėl keistųsi tik atskiros detalės.

Techninės rizikos:

Kompiuterių gedimas

Nors šių dienų kompiuterinė technika ganėtinai patikima, tačiau tokia tikimybė išlieka.

Sprendimas: Kompiuterio komponenčių (išskyrus standųjį diską) gedimas labai didelės įtakos projekto eigai neturės, kadangi nėra sunku sugedusią komponentę pakeisti nauja, o projekto duomenims, pavyzdžiui, garso plokštės gedimas - nepakenktų.

Standžiojo disko gedimas

Tai labai įtakotų projekto eigą.

Sprendimas: Periodiškai susikurti projektui svarbios informacijos kopijas tiek į CD-R laikmenas, tiek ir į kitus standžiuosius diskus, tokiu atveju prarastos informacijos kiekis būtų nedidelis.

Darbuotojų komandos dydžio ir patirties rizikos:

Komandos dydis

Komanda pradinei projekto daliai atlikti yra pakankamo dydžio. Nors ją sudaro vienas narys, tačiau jo aplinkoje yra nemažai kolegų iš kurių, reikalui esant, jis gali gauti pagalbą. Projektui plečiantis komandoje turėtų būti daugiau narių.

3.7. Testavimas

Naudojami automatiniai testavimo įrankiai. Apskaičiuojami juodos ir baltos dėžės metodais rezultatai ir jie tarpusavyje palyginami. Pirmus testavimo darbus atliko Visual Basic 6.0 kompiliatorius, kuris, statiškai patikrino programos sintaksę bei patį kodą prieš pradedant vykdyti pačią programą. Toliau testuojami atskiri programiniai moduliai:

Juodai dėžei buvo panaudotas rankinis testavimas, įvedant pradines reikšmes ir lyginant rezultatus su iš anksto žinomais rezultatais.

Interfeiso testavime mes galime užfiksuoti kiekvieną programos padarytą veiksmą.

Struktūriniame testavime kalkuliatoriaus įėjimams priskiriami duomenys ir pagal juos apskaičiuoti rezultatai yra palyginami su etaloniniais rezultatais. Testavimas yra automatinis.

3.7.1. Kalkuliatoriaus rezultatų tikrinimas lyginant su realiais

Tikrinsime gautus rezultatus atliekant skaičiavimus naudojant kalkuliatorių su tikraisiais duomenimis. Skaičiavimams tikrinti pasirinksime konkrečius intervalus. Naudosimės miglotosios logikos formulėmis:

Sudėtis (+) ir atimtis (-). Kai: $A=[a_1, a_2]$; $B=[b_1, b_2]$

$$A (+) B = [a_1, a_2] (+) [b_1, b_2] = [a_1+b_1, a_2+b_2]. \quad (27)$$

$$A (-) B = [a_1, a_2] (-) [b_1, b_2] = [a_1-b_2, a_2-b_1]. \quad (28)$$

Daugyba (·) ir dalyba (:). Kai $A=[a_1, a_2]$; $B=[b_1, b_2]$, tai :

$$A (\cdot) B = [a_1, a_2] (\cdot) [b_1, b_2] = [a_1 \cdot b_1, a_2 \cdot b_2], \quad (29)$$

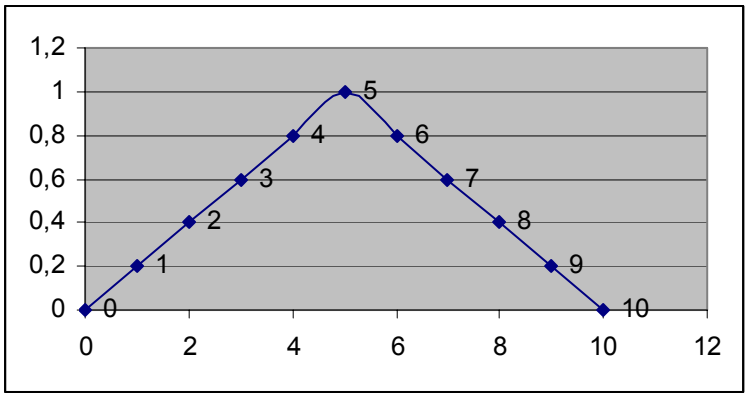
$$A (:) B = [a_1, a_2] (:) [b_1, b_2] = [a_1 : b_2, a_2 : b_1], \quad (30)$$

Max (v) ir min (^) operacijos :

$$A (v) B = [a_1, a_2] (v) [b_1, b_2] = [a_1 v b_1, a_2 v b_2]. \quad (31)$$

$$A (\wedge) B = [a_1, a_2] (\wedge) [b_1, b_2] = [a_1 \wedge b_1, a_2 \wedge b_2]. \quad (32)$$

Pav.: a_1 - b_1. Skaičiuojant rankiniu metodu parenkami vienodais tarpais penki lygiai. Kiekviename lygyje randami taškai paveiksle pažymėti skaičiais 1-10. Vieno duoto intervalo skaičių žymėjimas tame pat lygyje atitinka kito intervalo skaičių žymėjimą.



14 pav. Miglotosios logikos funkcijų lygių žymėjimas

Atliekami skaičiavimai ir gaunami rezultatai kiekviename funkcijų lygmenyje. Pavyzdžiui atliekant sudėties veiksmus atliekami tokie veiksmai :

A1 (+) B1 = $[a_{11}, a_{12}]$ (+) $[b_{11}, b_{12}]$ = $[a_{11}+b_{11}, a_{12}+b_{12}]$,

A2 (+) B2 = $[a_{21}, a_{22}]$ (+) $[b_{21}, b_{22}]$ = $[a_{21}+b_{21}, a_{22}+b_{22}]$, (33)

A3 (+) B3 = $[a_{31}, a_{32}]$ (+) $[b_{31}, b_{32}]$ = $[a_{31}+b_{31}, a_{32}+b_{32}]$, (34)

A4 (+) B4 = $[a_{41}, a_{42}]$ (+) $[b_{41}, b_{42}]$ = $[a_{41}+b_{41}, a_{42}+b_{42}]$, (35)

A5 (+) B5 = $[a_{51}, a_{52}]$ (+) $[b_{51}, b_{52}]$ = $[a_{51}+b_{51}, a_{52}+b_{52}]$, (36)

A6 (+) B6 = $[a_{61}, a_{62}]$ (+) $[b_{61}, b_{62}]$ = $[a_{61}+b_{61}, a_{62}+b_{62}]$. (37)

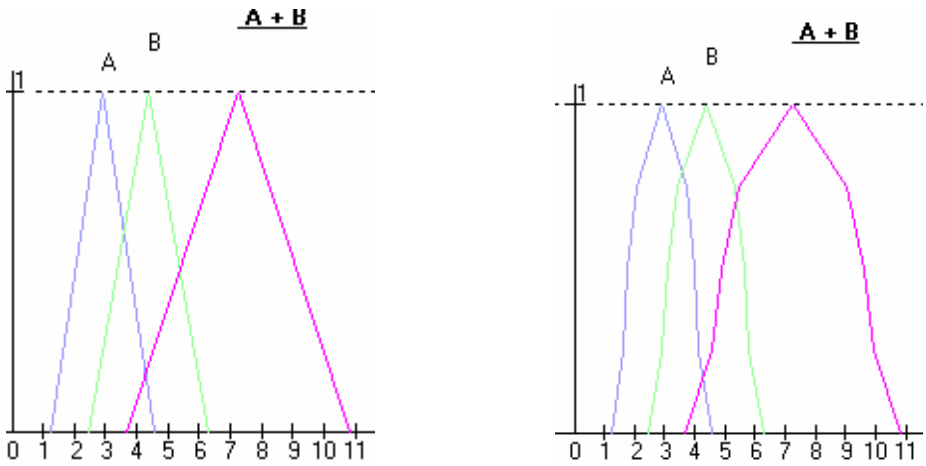
1. Pasirenkame intervalus: A = [1.23, 4.56], B = [2.45, 6.26].

Skaičiavimai pateikiami nuliniame lygyje .

A (+) B= [1.23, 4.56] (+) [2.45, 6.26] = [1.23+2.45, 4.56+6.26] = [3.68, 10.82]

Atlikus skaičiavimus su tais pačiais A, B intervalais kalkuliatoriumi gauname trikampi ir varpo formos grafiką.

15 pav. pavaizduota sumos skaičiavimas kalkuliatoriumi, kai duota A = [1.23, 4.56] ir B = [2.45, 6.26]



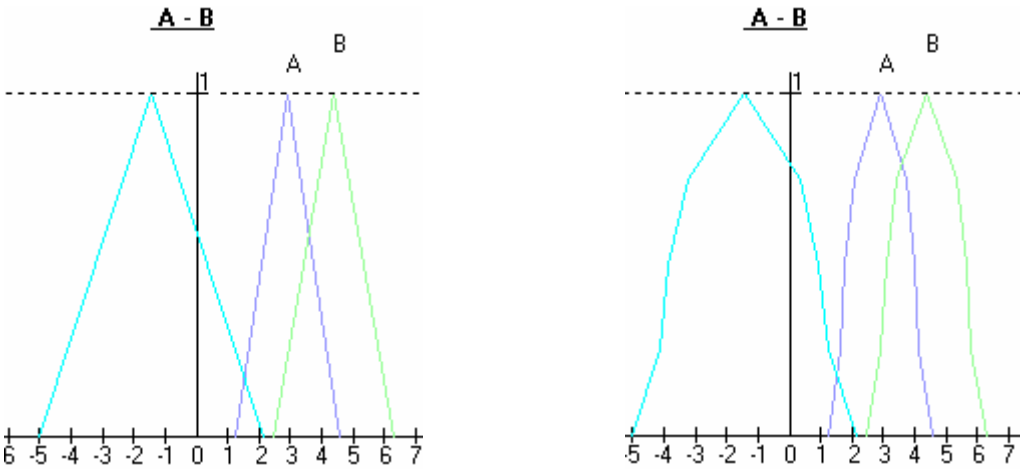
15 pav. Sumos skaičiavimas kalkuliatoriumi

Palyginus grafikus 15 pav. gauname vienodus rezultatus.

Skaičiavimus patikrinsime naudojant visus modeliuojamus veiksmus įvairiuose intervaluose. Toliau pateikiu miglotosios logikos atimties veiksmą, kuris skiriasi nuo mums įprastų skirtumų su intervalais veiksmų. Kad rezultatai būtų labai akivaizdūs naudosime skaičiavimams tuos pačius intervalus A = [1.23, 4.56] ir B = [2.45, 6.26]. Pateikiu skaičiavimą nuliniame lygyje .

A (-) B= [1.23, 4.56] (-)[2.45, 6.26] = [1.23-6.26, 4.56-2. 45] = [-5.03, 2.11].

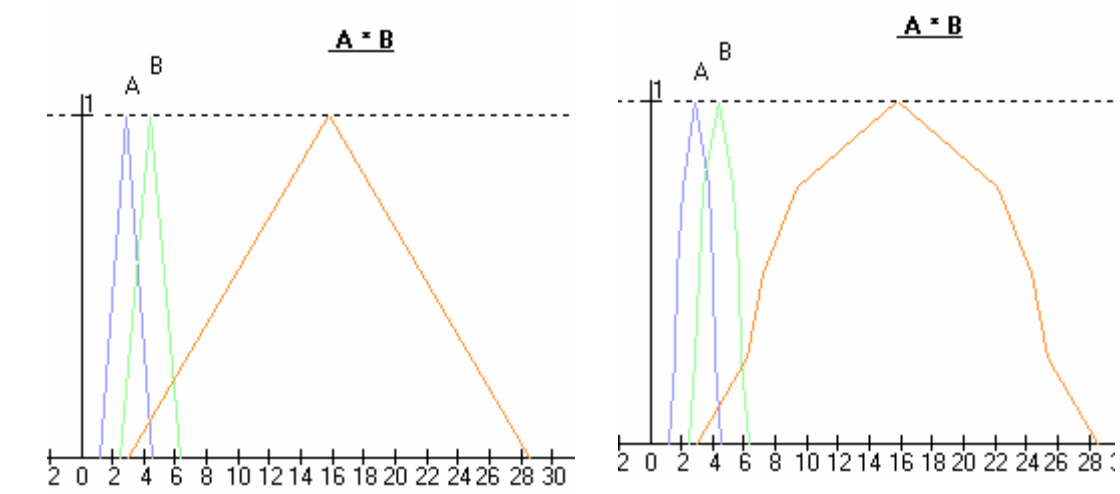
Toliau pateikiami skaičiavimai naudojant kalkuliatorių.



16 pav. Dviejų intervalų skirtumo skaičiavimas

Toliau tokia pat eiga, kaip anksčiau atliksime sandaugos skaičiavimus.

A (·) B = [1.23, 4.56] (·) [2.45,6.26] = [3.0135, 28.5456],

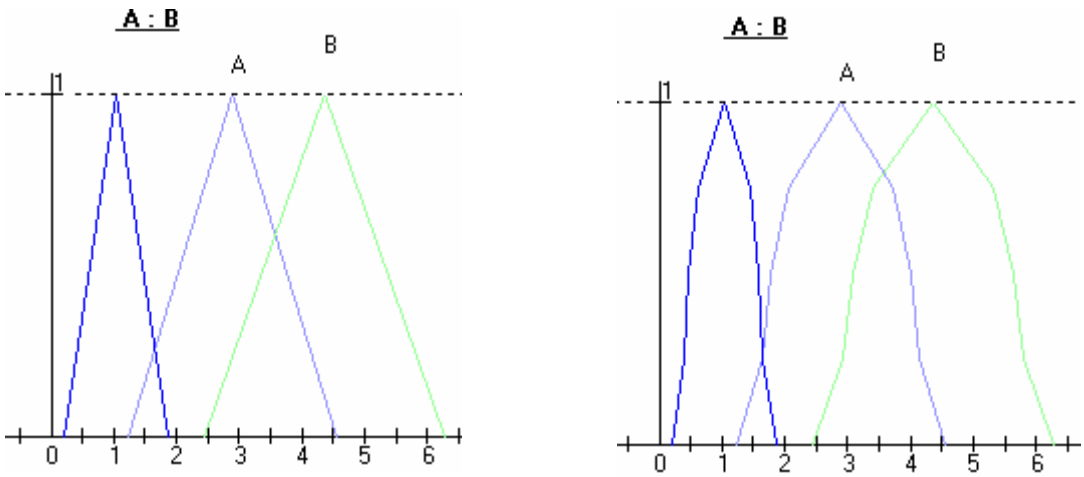


17 pav. Intervalų sandauga panaudojus kalkuliatorių

Atlikus sandaugos analize matome, kad kalkuliatoriumi gauti rezultatai atitinka realius. Modeliuojant kalkuliatorių buvo parinkta mažai skaičiavimo lygių. Šiuo atveju kalkuliatorių galima naudoti , kai pakanka žemiausio lygio rezultatų.

Toliau lyginsime dalybos rezultatus. Atliekame skaičiavimus su intervalais A,B.

A (:) B = [1.23, 4.56] (:) [2.45,6.26] = [1.23/6.26,4.56/2.45] = [0.1965, 1.8612],



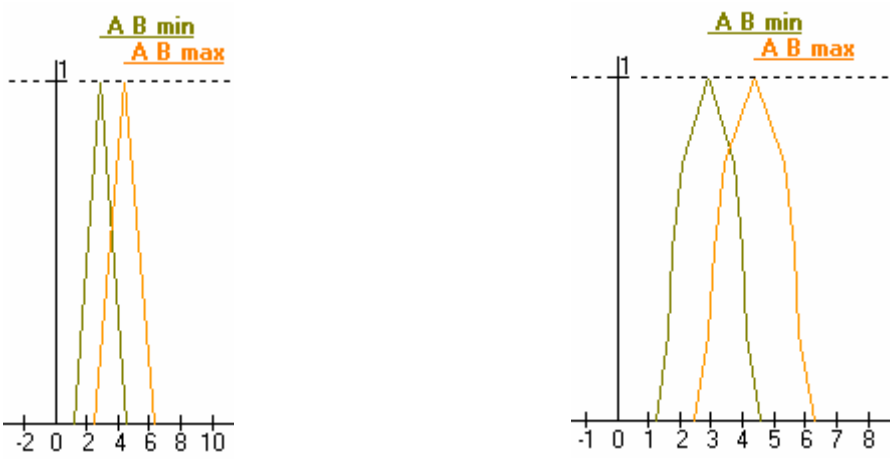
18 pav. Intervalų A = [1.23, 4.56] ir B = [2.45, 6.26] dalybos skaičiavimas kalkuliatoriumi

Gauname analogiškus rezultatus, kaip atliekant daugybos veiksmus. Priimsime išvadą, kad kalkuliatorių galima naudoti, kai pakanka žemiausio lygio rezultatų.

Kalkuliatoriuje pateiktas dar vienas galimas veiksmas, tai minimalaus ir maksimalaus intervalo ieškojimas. Naudojant anksčiau pateiktas formules galima įžvelgti, kad ne visada

parenkamas vienas iš pateiktų mažiausių intervalų. Gali susidaryti naujas intervalas. Žemiau pateikta skaičiavimas ir grafinis vaizdavimas su intervalais A = [1.23, 4.56] ir
B = [2.45, 6.26].

A (^) B = [1.23, 4.56] (^)[2.45, 6.26] = [1.23^2.45, 4.56 ^6.26] = [1.23,4.56].



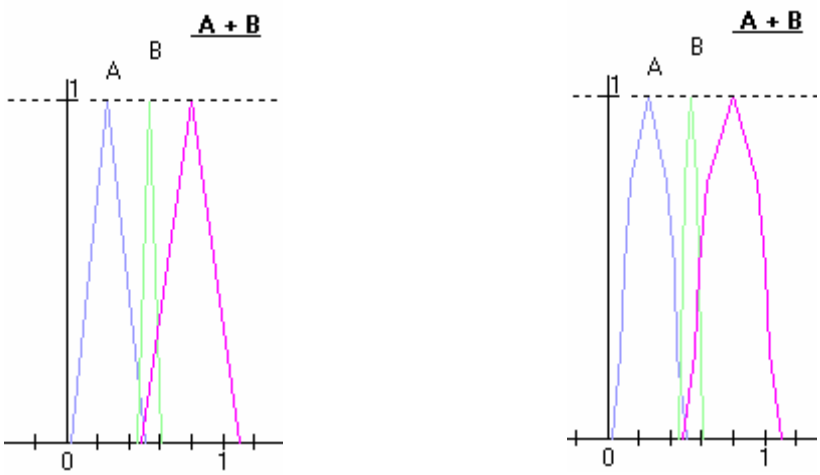
19 pav. Kalkuliatoriumi parenkamas mažiausias ir didžiausias intervalas.

Norint įsitikinti kalkuliatoriaus trukumais ir privalumais atliksime veiksmus su labai mažais intervalais. Pasirenkame intervalus: A = [0.03, 0.5], B = [0.45, 0.61].

Toliau atliksime visus anksčiau atliktus veiksmus ta pačia eiga.

Sumos skaičiavimai:

A (+) B= [0.03, 0.5] (+) [0.45, 0.61] = [0.03+0.45, 0.56+0.61] = [0.59,1.06],

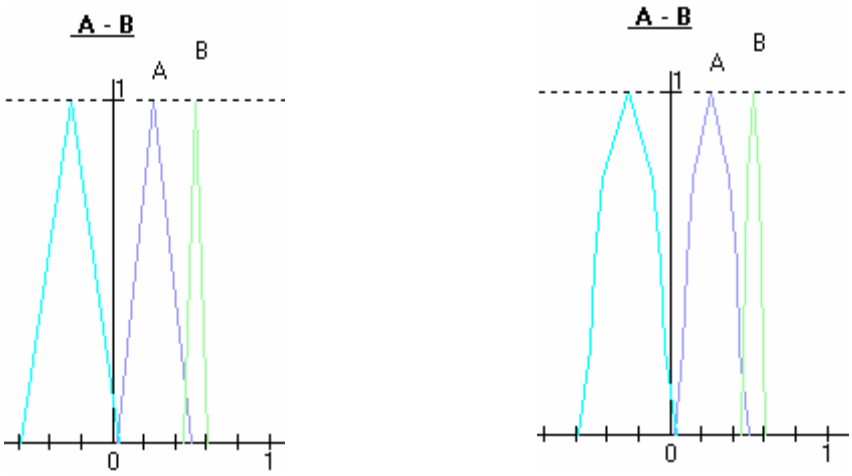


20 pav.Sumos skaičiavimas su A = [0.03, 0.5], B = [0.45, 0.61] intervalais kalkuliatoriumi

Rezultatai atliekant veiksmus skirtingais metodais vienodi.

Skirtumo skaičiavimas:

A (-) B= [0.03, 0.5] (-)[0.45, 0.61] = [0.03-0.61, 0.5-0.45] = [-0.58, 0.05],

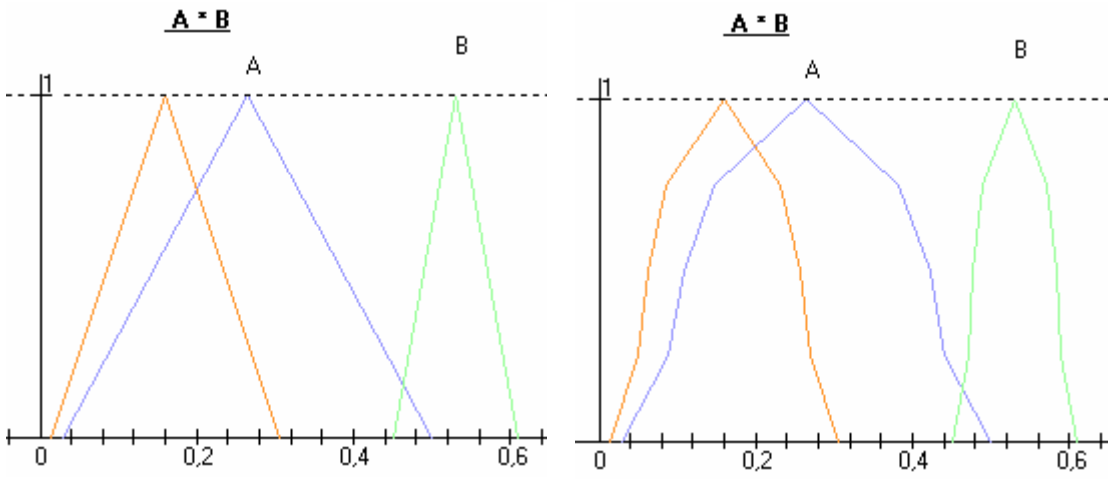


21 pav.Skirtumo skaičiavimas naudojant intervalus A = [0.03, 0.5], B = [0.45, 0.61] kalkuliatoriumi

Rezultatai atliekant veiksmus skirtingais metodais vienodi.

Sandaugos skaičiavimas:

A (·) B = [0.03, 0.5] (·) [0.45, 0.61] = [0.0135, 0.305],

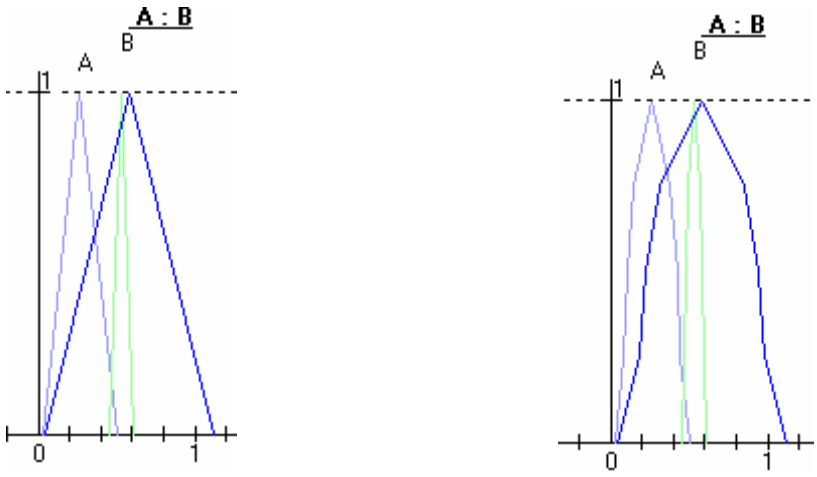


22 pav.Sandaugos skaičiavimas su A = [0.03, 0.5] ir B = [0.45, 0.61] intervalais kalkuliatoriumi

Rezultatai atliekant veiksmus skirtingais metodais vienodi.

Dalybos skaičiavimas:

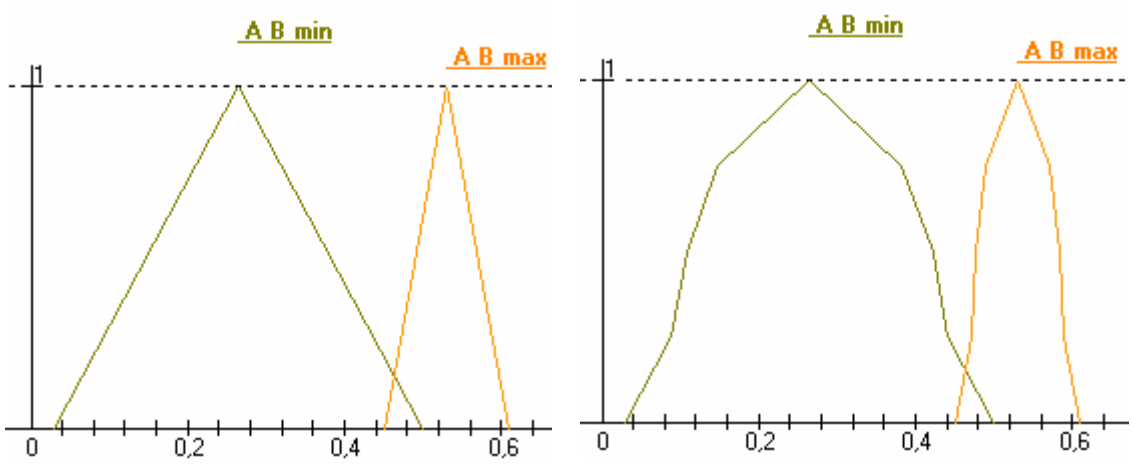
A (:) B = [0.03, 0.5] (:) [0.45, 0.61] = [0.03/0.61, 0.5/0.45] = [0.049, 1.111],



23 pav. Skirtumo skaičiavimas naudojant intervalus A = [0.03, 0.5], B = [0.45, 0.61] kalkuliatoriumi

Minimumo ieškojimas:

A (^) B = [0.03, 0.5] (^)[0.45, 0.61] = [0.03^0.45, 0.5 ^0.61] = [0.03, 0.5].



24pav. Kalkuliatoriumi parenkamas mažiausias ir didžiausias intervalas.

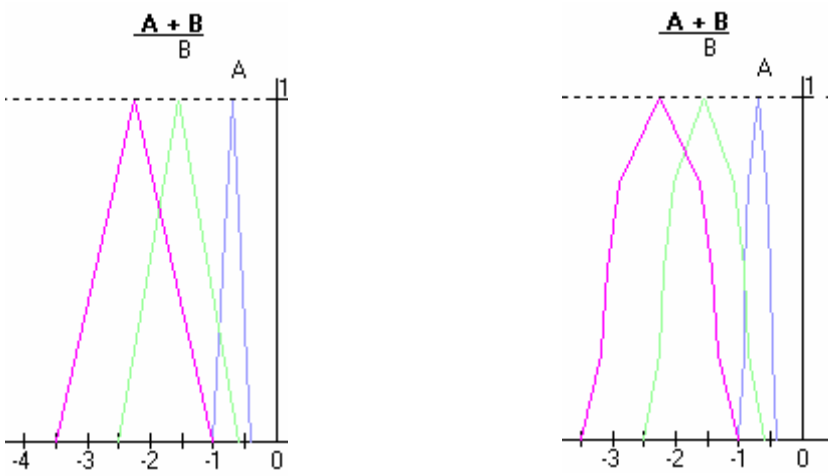
Atlikus įvairiais metodais miglotosios logikos veiksmus su intervalais, kurie mažesni už vienetą gavome žemiausiame lygyje vienodus rezultatus. Galime daryti išvada, kad kalkuliatorius tinkamas mažų intervalų skaičiavimui.

Kalkuliatoriumi galima atlikti skaičiavimus su įvairiais intervalais. Atliksime skaičiavimus su neigiamais intervalais ir palyginsime rezultatus su anksčiau gautais.

Pasirenkame neigiamus intervalus: A = [-1, -0.4], B = [-2.5, -0.6].

Sumos skaičiavimai:

A (+) B= [-1, -0.4] (+)[-2.5, -0.6] = [(-1)+(-2.5), (-0.4)+(-0.6)] = [-3.5, -1],

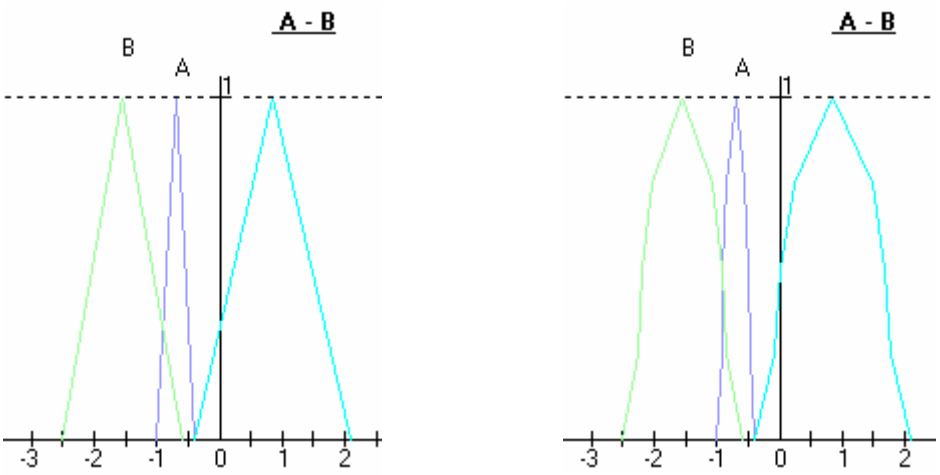


25 pav.Sumos skaičiavimas kalkuliatoriumi naudojant intervalus A = [-1, -0.4], B = [-2.5, -0.6].

Rezultatai atliekant veiksmus skirtingais metodais vienodi.

Skirtumo skaičiavimas:

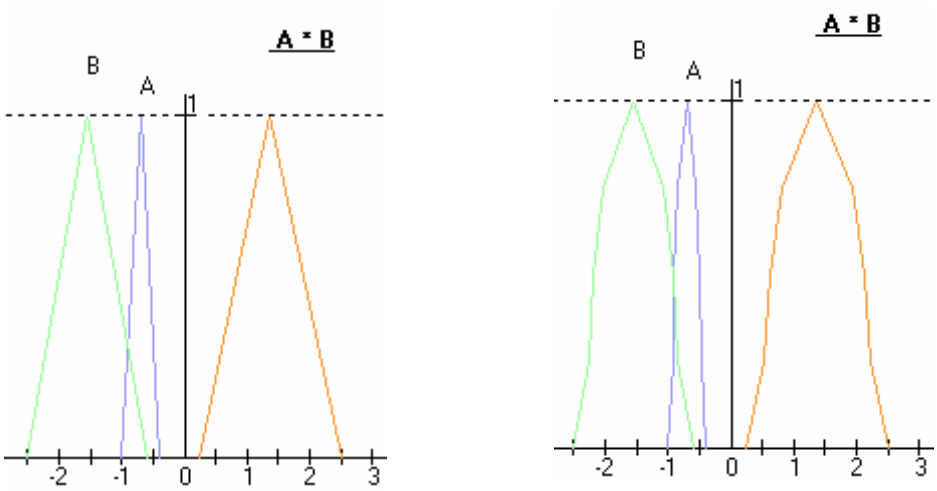
A (-) B=[-1, -0.4] (-)[-2.5, -0.6] = [(-1)-(-0.6), (-0.4)-(-2.5)] = [-0.04, 2.1],



26 pav.Skirtumo skaičiavimas kalkuliatoriumi naudojant intervalus A = [-1, -0.4], B = [-2.5, -0.6].

Sandaugos skaičiavimas:

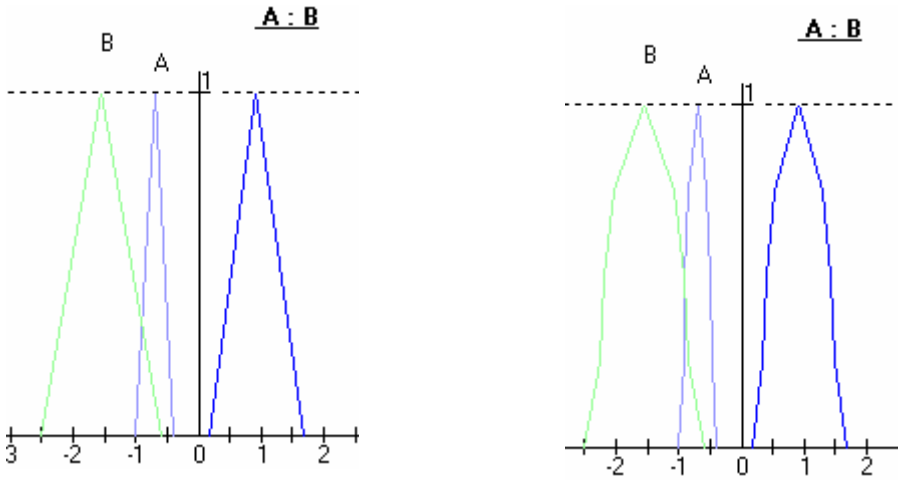
A (·) B = [-1, -0.4] (·)[-2.5, -0.6] = [2.5, 0.24],



27 pav.Sandaugos skaičiavimas kalkuliatoriumi naudojant intervalus A = [-1, -0.4], B = [-2.5, -0.6].

Dalybos skaičiavimas:

A (:) B =[-1, -0.4] (:)[-2.5, -0.6] = [(-1)/(-0.6),(-0.4)/(-2.5)] = [1.66, 0.16],



28 pav.Dalybos skaičiavimas kalkuliatoriumi naudojant intervalus A = [-1, -0.4], B = [-2.5, -0.6].

Atlikus veiksmų su neigiamais intervalais analizę matome vienodus rezultatus žemiausiame lygyje. Šiuo atveju nustatyta, kad kalkuliatorių galima naudoti, kai pakanka žemiausio lygio rezultatų.

3.8. Sistemos vystymo planas

Esant vartotojų poreikiui sistema toliau galima vystyti. Šiuo metu numatomas toks sistemos vystymo planas:

- Praplėsti miglotosios logikos grafike skaičiuojamų lygių skaičių .
- Papildyti sistemos funkcionalumą.

4. VARTOTOJO DOKUMENTACIJA

4.1.Sistemos funkcinis aprašymas

Miglotosios logikos kalkuliatoriaus paskirtis – grafiškai realizuoti, bei taikyti miglotosios logikos veiksmus.

Ši programa leidžia :

1) Atlikti miglotosios logikos sudėties veiksmus
2) Atlikti miglotosios logikos atimties veiksmus
3) Atlikti miglotosios logikos daugybos veiksmus
4) Atlikti miglotosios logikos dalybos veiksmus
5) Atlikti miglotosios logikos min/max paieškos veiksmus

4.2. Įžanginis vadovas

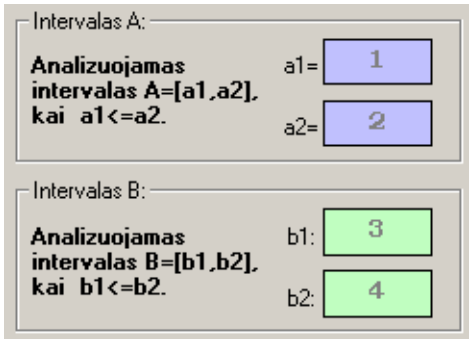
Naudodamiesi programos paleidimo failu 

paleidžiame miglotosios logikos kalkuliatoriaus programą. Atsiveria langas, kuris pateiktas 1 priede.

1. Programinės įrangos paslaugos: programa leidžia vartotojui nustatyti visas kalkuliatoriaus įėjimų reikšmes. Duomenys įvedami pasirinkimų sąrašo pagalba, pele nurodant pageidaujamą reikšmę. Programa pradeda skaičiuoti tik paspaudus komandinį mygtuką, tai užtikrina patogesnį programos valdymą. Paspaudus šį mygtuką atliekami veiksmai.

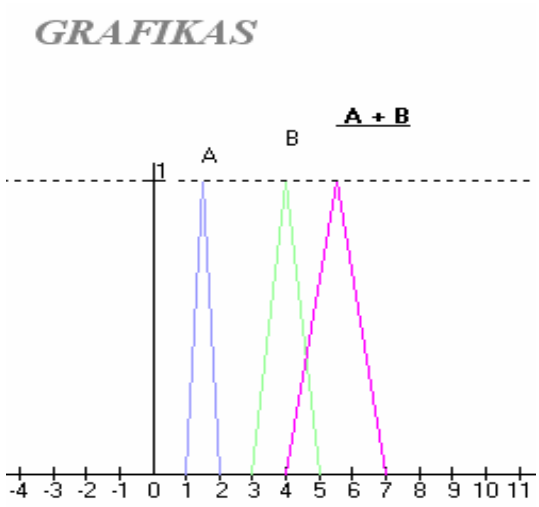
2.Pranešimai vartotojui: pranešimas apie skaičiavimų rezultatus. Šis pranešimas išvedamas tada, kai paspaudžiamas veiksmo komandos mygtukas. Lange iškarto matome įvestus pradinius duomenis ir skaičiavimų rezultatą. Norint gauti naujus rezultatus su tais pačiais intervalais pasirenkame pageidaujamas komandas „Sudėtis“, „Atimtis“, „Daugyba“, „Dalyba“. Norint pasirinkti naujus intervalus spaudžiame mygtuką „Naujas grafikas“.

3.Veiksmai dirbant su programa: dirbant kalkuliatoriumi, skaičiai ir komandos įvedami klaviatūros ir pelės pagalba. Pele pažymime įvedimo langelį, o klaviatūroje parenkame skaičius. Pirmasis intervalo skaičius turi būti didesnis už antrąjį pvz.



29 pav. Intervalų A,B duomenų įvedimas

Norint įvesti neigiamą skaičių klaviatūroje spaudžiame „-“. Įvesti skaičiai apibūdina grafinį vaizdą ir vaizduojami kalkuliatoriaus indikaciniame lauke grafine forma ir skaičiais 30 pav.



30 pav. Grafikas matomas ekrane

Suklydus, įvestą informaciją galima ištrinti. Norint pašalinti klaidingą informaciją spaudžiame mygtuką „Iš naujo“ 31pav.

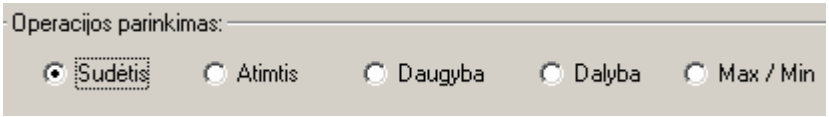
31 pav. Komanda klaidoms taisyti

Klaidingus duomenis galima keisti tik tol, kol nepaspaustas mygtukas su komanda „Operacijos su intervalais“ 32 pav.

32 pav. Mygtukas „Operacijos su intervalais“

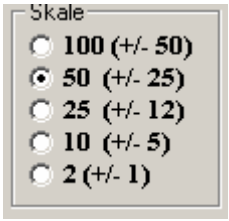
Kalkuliatorius turi daug mygtukų ir ekraną grafiniam vaizdavimui.

„Sudėtis“ mygtukas naudojamas aritmetiniai sumai apskaičiuoti, „Atimtis“ skaičių skirtumui rasti. „Daugyba“ mygtukas naudojamas sandaugos apskaičiuoti, o „Dalyba“–dalybos . Maksimumą, minimumą randame „Max/Min“ mygtuko pagalba.



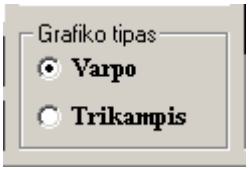
33 pav. Operacijų parinkimo mygtukai

Paspaudus operacijos mygtuką ekrane gauname grafinį vaizdą. Įvedus pasirinktus A ir B intervalus su jais galima atlikti visus 33 pav. pateiktus veiksmus. Jei užduoti A ir B intervalai yra dideli (maži) ir nematome pilno vaizdo, galima keisti skale 34 pav.



34 pav.Grafikų skalė

Kadangi atliekami skaičiavimai vaizduojami grafiškai, galime pasirinkti varpo ar trikampę vaizdavimo formą.



35 pav. Grafiko pasirinkimas

Norint atlikti naujus skaičiavimus pereiname prie mygtuko „Naujas grafikas“ 36 pav.

36 pav. Mygtukas „Naujas grafikas“

4.3.Sistemos instaliavimas

Darbui su paketu reikalinga tik Visual Basic 6.0 programa.

Reikalavimai ir apribojimai pateikiami kartu su Visual Basic 6 programa.

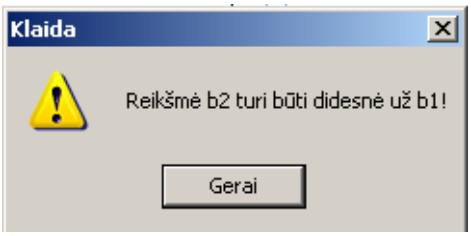
Dėl instaliavimo: jeigu jūs turite jau OS kurioje įdiegta Visual Basic 6.0 (ar aukštesnė versija) jums nereiks nieko daryti, jei ne tada jums reikės instaliuoti Visual Basic 6 (kaip tai padaryti aprašyta prie Visual Basic 6.0 programos).

PĮ nėra priklausoma nuo OS - programos bei dokumentacija be modifikacijų gali būti vykdomos bei peržiūrima įvairiose operacinėse sistemose: Windows NT ,Windows9x - t.y. tose OS, kurioms yra realizuota peržiūros programa palaikanti Visual Basic 6.0.

4.4.Sistemos administratoriaus vadovas

Miglotosios logikos kalkuliatoriaus sistema, priklausomai nuo susidariusios situacijos, gali generuoti įspėjimus:

Įspėjimai gaunami apie ketinimą įvesti didesnes intervalo pirmąsias reikšmes negu antrosios. Jei neteisingai surinkome skaičius atsiranda pranešimas „Klaida“. Uždarius langelį patvirtinimu „Gerai“, galime ištaisyti klaidą ir pereiti prie kitos operacijos.



37 pav. Pranešimas apie neteisingą įvestą reikšmę

5.MIGLOTOSIOS LOGIKOS KALKULIATORIAUS ĮVERTINIMAS

Bazinius miglotosios logikos teorijos ir taikymo principus paskelbė L.Zadeh 1965 metais. Miglotosios logikos metodai naudojami tiesinėse ir netiesinės valdymo sistemose, vaizdų atpažinimui, taikomi finansinėse sistemose, procesų tyrimuose o taip pat duomenų analizei [5]. Palaipsniui pripažįstama, kad miglotoji logika praverčia sprendžiant netiesinių sistemų, kurioms būdinga neapibrėžta bei nepatikima informacija apie sistemos elgesį, modeliavimo uždavinius.

Miglotosios logikos ekspertinių sistemų kūrimas apima kelis etapus [4] :

- Duomenų paruošimą
- Miglotosios logikos matematinių taisyklių pritaikymą
- Sistemos testavimą su naudojant realius duomenis
- Sistemos derinimą

Miglotosios logikos kalkuliatorių galima panaudoti įvairiose srityse.

5.1.Kalkuliatoriaus taikymas procesų tyrimui

Tirsime kaip galima pritaikyti miglotosios logikos kalkuliatorių procesų tyrimui.

Kalkuliatoriaus darbą tirsiu Panevėžio kolegijos elektroninių matavimų laboratorijoje. Jį taikysiu laboratorinių darbų duomenų analizei. Matuojant fizikinius dydžius, gaunami netikslus rezultatai, tai yra susidaro nuokrypos nuo tikrųjų verčių. Paklaidos dažniausiai gaunamos dėl metodo ir prietaiso netobulumo. Paklaidos, kurios matuojant gaunamos dėl prietaiso elementų netobulumo, yra vadinamos prietaiso paklaidomis. Prietaiso leistina paklaida išreikšta procentais vadinama tikslumo klase. Atliekant elektroninės grandinės matavimus keliais matavimo prietaisais gaunamos didesnės nuokrypos nuo tikrosios vertės negu atliekant vienu prietaisu. Gauname kelis išmatuotus neapibrėžtus skaičius. Atliksime grandinės matavimus ir skaičiavimus naudojantis miglotosios logikos kalkuliatoriumi.

Atliksime darbą „Nuoseklus ir lygiagretus rezistorių jungimas“.

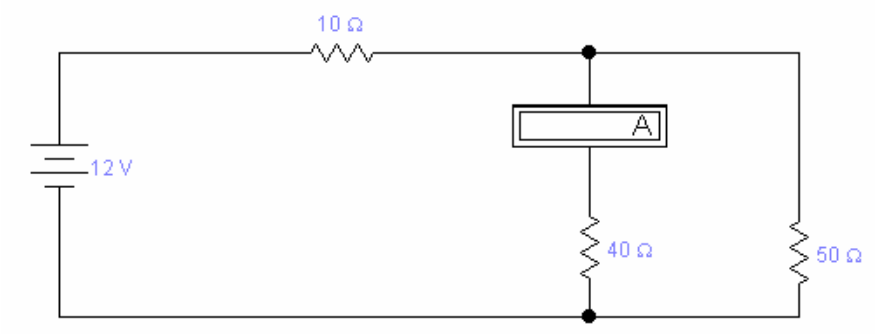
Darbo tikslas: bandymu keliu patikrinti nuoseklaus ir lygiagretaus rezistorių jungimo ypatumus.

Darbo eiga:

1.Sujungti schemą.

2.Išmatuoti sroves pratekančias rezistoriais, įtampas ant rezistorių, duomenis surašyti į lenteles.

3.Apskaičiuoti likusias reikšmes kuriu nematavome ir rezultatus pateikti lentelėje



38 pav. Mišriai sujungta grandinė

Duota: maitinimo įtampa U=12V, R1=10 Ω, R2=40 Ω, R3=50 Ω.

Apskaičiuojame bendrą grandinės varžą:

$$Rb=R1+(R2*R3)/(R2+R3) \quad (38)$$

Išmatuojame įtampas ant varžų 1,0 tikslumo klasės rodyklinių voltmetru. Išmatuotos reikšmės gali neatitikti tikrajai išmatuotai reikšmei ± 1%. Gauname neapibrėžtas išmatuotų reikšmių aibes. Sroves matuojame 2,5 tikslumo klasės rodykliniu ampermetru. Gautos išmatuotos srovės reikšmės gali neatitikti tikrųjų reikšmių ± 2,5 %. Rezultatus surašome į lentelę.

Schemos matavimo rezultatai

1 lentelė

Grandinė	U,V	I,A	P,W
R1=10 Ω	3,66-3,74	0,28-0,38	1,03-1,42
R2=40 Ω	8,13-8,31	0,21-0,22	1,71-1,83
R3=50 Ω	8,13-8,31	0,15-0,16	1,22-1,33
Rb=32,2 Ω	12	0,36-0,38	4,32-4,56

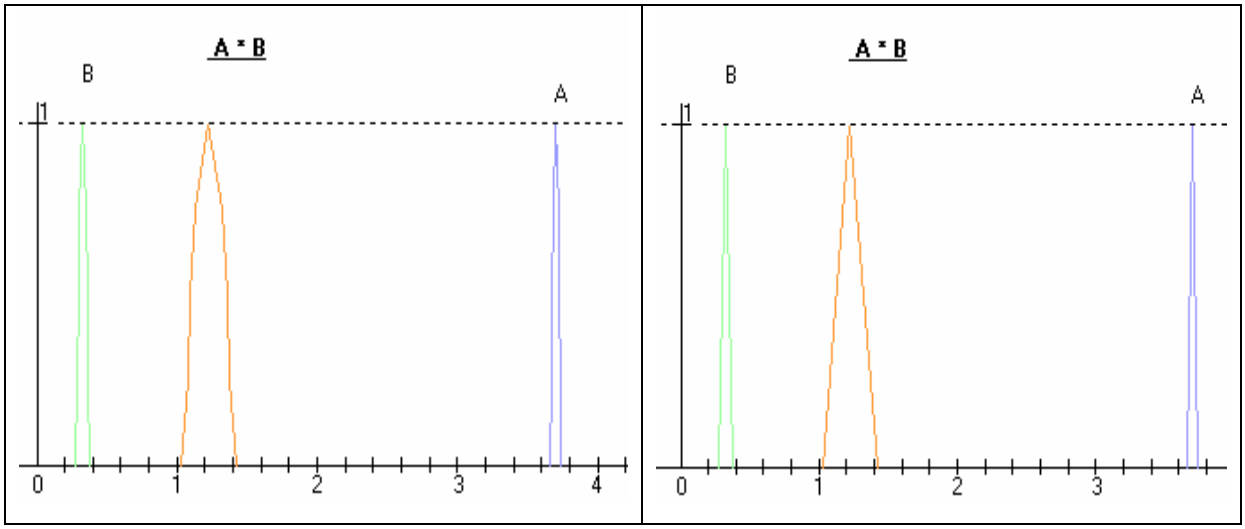
Galios skaičiavimus atliksime pasinaudodami miglotosios logikos kalkuliatoriumi.

Naudosime formulę:

$$P=I(\cdot)U \quad (39)$$

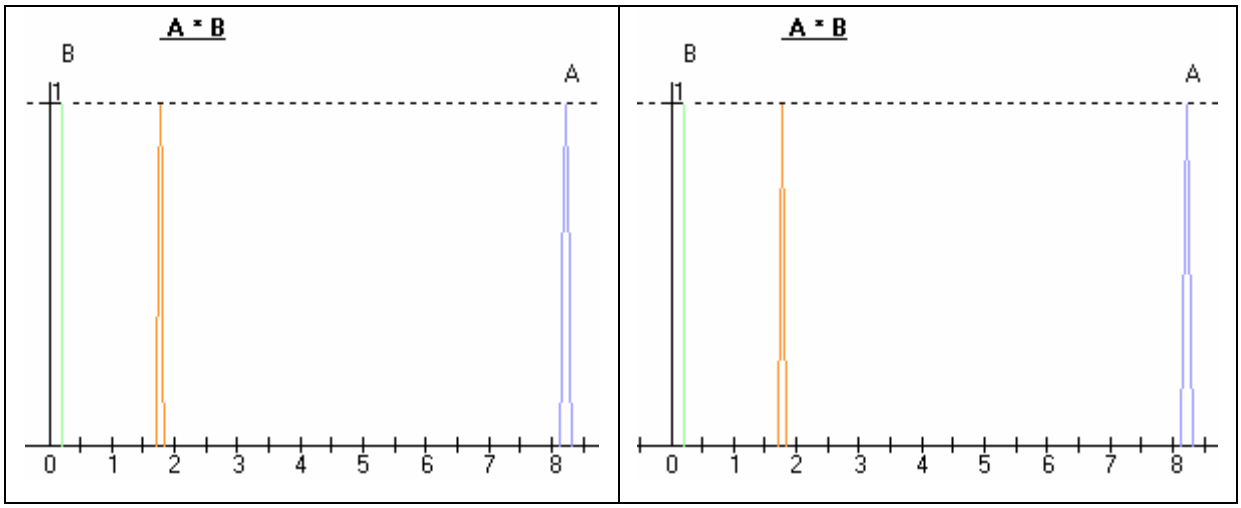
Atliekame veiksmus su neapibrėžtomis aibėmis:

$P=I(\cdot)U=[3.66, 3.74\ ]\ (\cdot)\ [0.28, 0.38\ ] = [1.03,\ 1.42],$



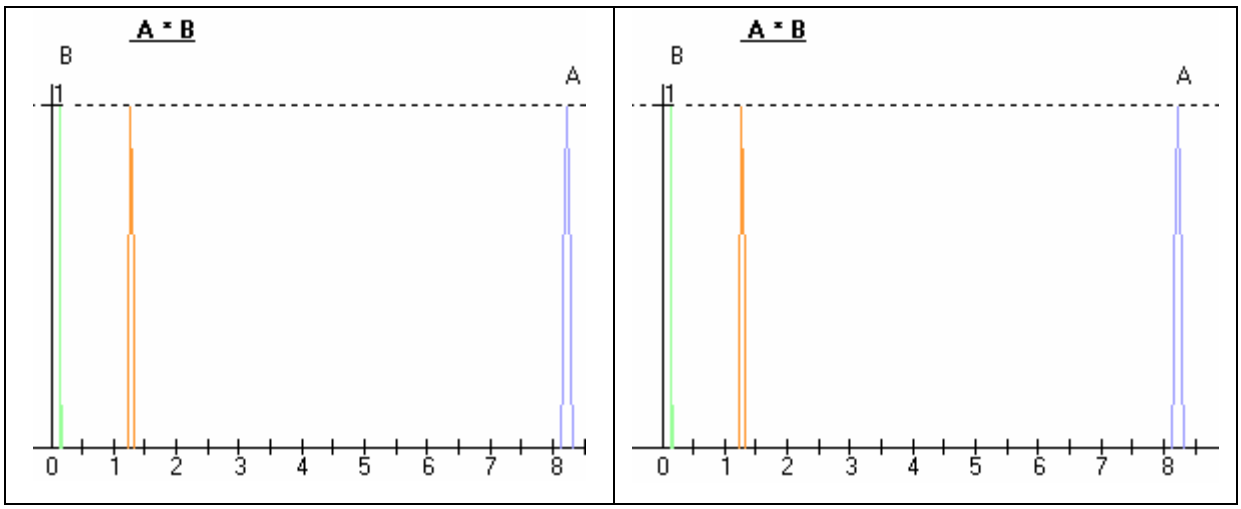
39 pav. Schemos galios P1 skaičiavimas panaudojant miglotosios logikos kalkuliatorių

$$P=I(\cdot)U=[8.13,\ 8.31\ ]\ (\cdot)\ [0.21,0.22\ ]=[1.71,\ 1.83],$$



40 pav. Schemos galios P2 skaičiavimas panaudojant miglotosios logikos kalkuliatorių

$$P=I(\cdot)U=[8.13,\ 8.31\ ]\ (\cdot)\ [0.15,0.16\ ]=[1.22,\ 1.33],$$



41 pav. Schemos galios P3 skaičiavimas panaudojant miglotosios logikos kalkuliatorių

Išvada: kalkuliatorius gali tiksliai atlikti veiksmus su neapibrėžtais skaičiais atliekant procesų tyrimą. Atliekant skaičiavimus varpo ir trikampės formos grafikais gaunami panašūs. Skaičiuojant nedidelius intervalus siūlau naudotis trikampės formos grafiku.

5.2. Kalkuliatoriaus taikymas duomenų analizei

Didėjant informacijos kiekiui, atliekant tyrimus informacijai apdoroti, analizuoti ir daryti prognozes ateičiai galima pasitelkti miglotosios logikos metodus. Kuo didesni informacijos srautai, tuo sudėtingiau arba net neįmanoma jos aprašyti klasikiniais matematiniais metodais ir tiksliai jos valdyti. Tam turi įtakos ir tai, jog sunku surinkti tikslią ir visa apimančią informaciją apie mus supančią aplinką bei sistemą. Sudėtinga aplinkos veiksnių poveikį įvertinti kiekybiškai. Kadangi prognozavimui ne visada galima panaudoti klasikinius metodus (daugiamatės tiesinės ir kreivinės regresijos, laiko eilučių ar kitus metodus), tenka pasitelkti euristinius, ekspertų nuomone grįstus metodus. Todėl vienu iš pagrindinių įrankių, atliekant prognozes tampa miglotosios logikos metodų panaudojimas.

Atliksiu analizę. Darant dėstytojo darbo savianalizę reikia atlikti tyrimą. Studentams keturiose paskaitose pateikiami skirtingo lygio klausymus.Tiriama kokio lygio informacija per tam tikrą laiką geriau įsisavinama. Tyrimą atlikau elektroninių matavimų paskaitoje pateikdama dvidešimt žinojimo, supratimo, taikymo, analizės klausymų. Pateikiama mažiausiai ir daugiausiai atsakytų klausimų skaičius. Surandamas greičiausiai ir ilgiausiai atliktų užduočių laikas. Duomenis pateikiami lentelėje. Naudojantis miglotosios logikos kalkuliatoriumi surandame kiekvieno uždavinių lygio atsakymų pateikimo greitį. Turėdami rezultatus galime spręsti apie studentų žinių įsisavinimą naudojant įvairius mokymo metodus.

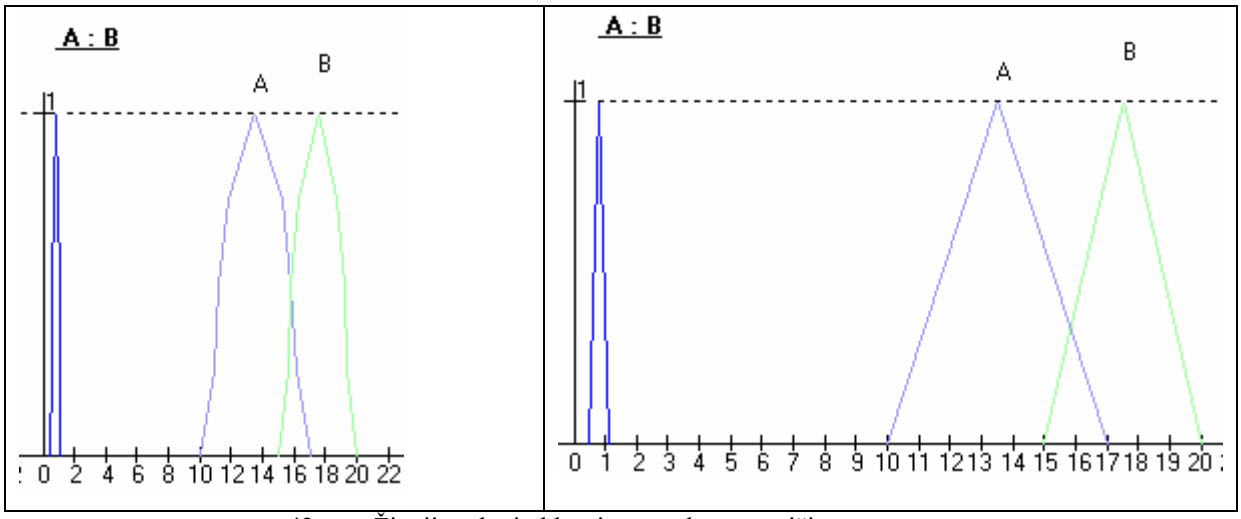
Studentų apklausos rezultatai

2 lentelė

Klausimų lygis	Atsakyta klausimų	Laikas per kuri atsakyta į klausimus,min.	Vieno klausimo atsakymo greitis
Žinojimo	10-17	15-20	0,5-1,33
Supratimo	14-19	17-20	0,7-1,12
Taikymas	13-18	13-20	0,65-1,39
Analizė	16-18	16-20	0,8-1,125

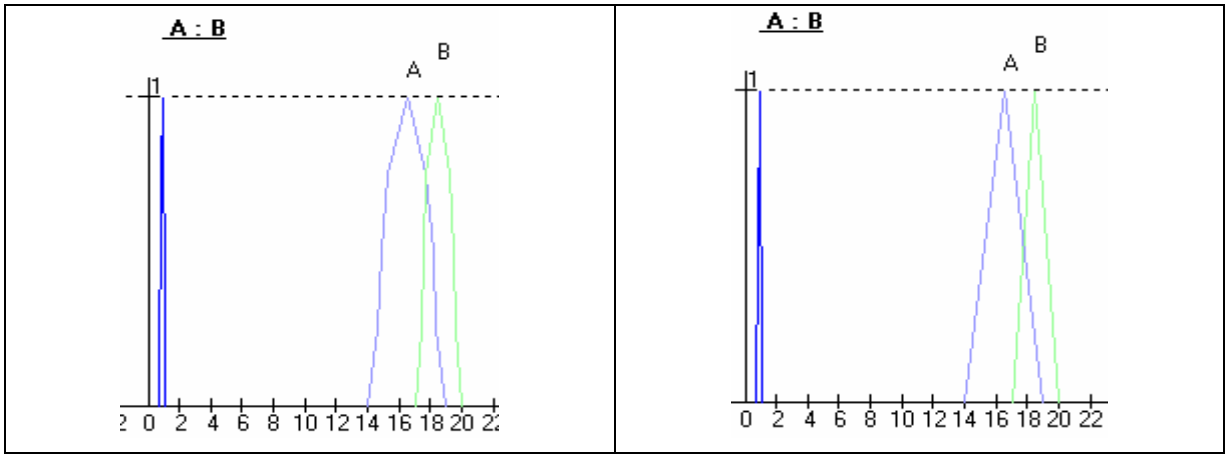
Skaičiavimams naudosimės miglotosios logikos dalybos formule:

$$A\ (:)\ B = [10, 17]\ (:)\ [15,20] = [0.5, 1.3],$$



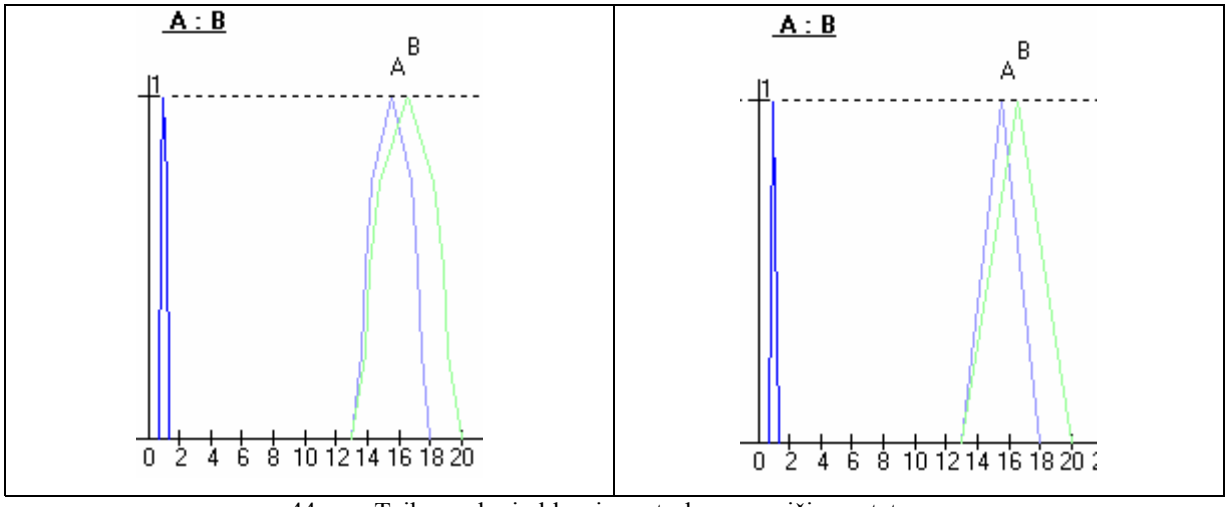
42 pav. Žinojimo lygio klausimo atsakymo greičio nustatymas

$$A\ (:)\ B = [14, 19]\ (:)\ [17,20] = [0.7, 1.12],$$



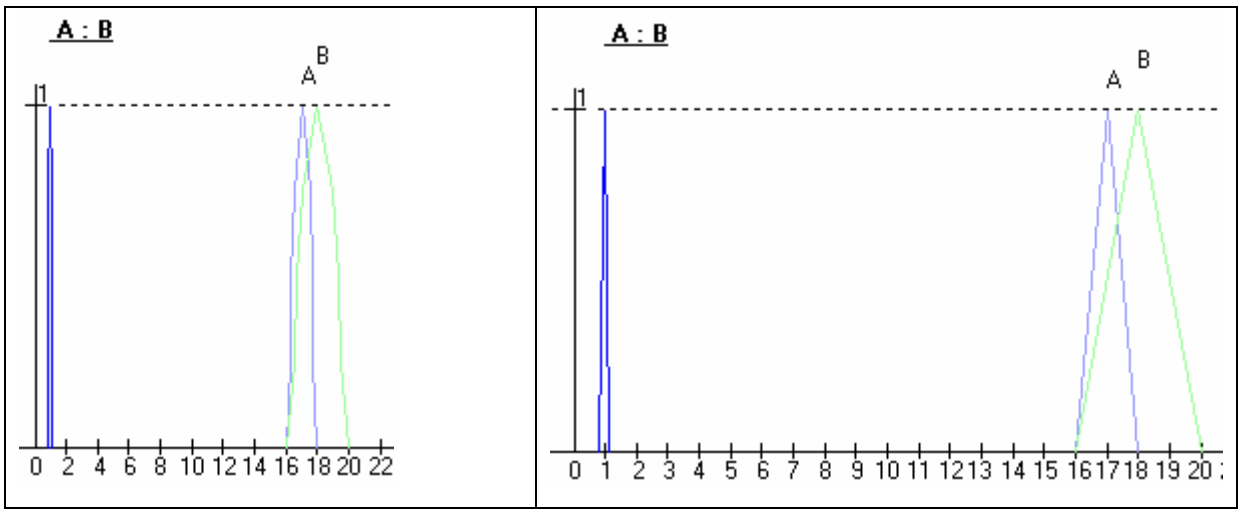
43 pav. Supratimo lygio klausimo atsakymo greičio nustatymas

$$A\ (:)\ B = [13, 18]\ (:)\ [13,20] = [0.65, 1.39],$$



44 pav. Taikymo lygio klausimo atsakymo greičio nustatymas

A (:) B = [16, 18] (:) [16,20] = [0.8, 1.13],



45 pav. Analizės lygio klausimo atsakymo greičio nustatymas

Galima teigti atliekant tyrimus informacijai apdoroti, analizuoti ir daryti prognozes ateičiai galima pasitelkti miglotosios logikos kalkuliatorių. Atliekant skaičiavimus gausime reikiamus rezultatus.

5.3. Kalkuliatoriaus taikymas valdymo sistemose

Laboratorijoje vykdomi fundamentalūs ir taikomieji tyrimai energetikos sistemų valdymo ir modeliavimo srityje. Mokslinių tyrimų tikslas – sukurti energetikos sistemų bei vartotojų matematinius modelius ir technines priemones, įgalinančias optimizuoti energijos generavimo, perdavimo ir vartojimo procesus, automatizuoti jų valdymą bei energiją vartoti efektyviau.

Energijos taupymo galimybių tyrimams sukurti tikslesni pastatų šildymo procesų dinaminiai modeliai. Pritaikius juos pastatų šildymo sistemų tyrimams nustatytos šildymo režimo, pastatų mikroklimato ir energijos suvartojimo priklausomybės, įvertintos šilumos taupymo galimybės mūsų klimato sąlygomis. Panaudojant miglotosios logikos kalkuliatorių galima skaičiuoti temperatūrų pokyčius.

6.IŠVADOS

- Darbe „Miglotosios logikos grandžių ir struktūrų modelių sudarymas“ sumodeliuotas kalkuliatorius atliekantis aritmetinius veiksmus su miglotosios logikos elementais. Ši programa leidžia atlikti tokius veiksmus su miglotosios logikos elementais : *sudėties (+) , atimties (-), daugybos ( ) , dalybos (:).*
- Miglotoji logikai įgalina pasiekti žymiai geresnių valdymo charakteristikų nei tradiciniais metodais.
- Darbe atlikta išsami miglotosios aritmetikos veiksmų analizė, išskirti jos veiksmai taikytini projektuojant kalkuliatorių.
- Darbo projektinėje dalyje išanalizuoti vartotojo reikalavimai bei išskirti funkciniai reikalavimai. Parengta sistemos specifikacija. Sudarytas projekto atlikimo grafikas. Pateiktas programos kūrimas. Nustatytos projekto rizikos ir jų išvengimo priemonės. Aprašytas atliktas testavimas. Pateiktas sistemos vystymo planas.
- Parašyta vartotojo dokumentacija. Atlikta išsami kalkuliatoriaus veikimo analizė, išskirti jos privalumai bei trūkumai, taikymo sritys.
- Sukurtą kalkuliatorių bus galima naudoti dėstant „Skaičiavimo metodai su neapibrėžtomis aibėmis“ kursą ir padės geriau studentams įsisavinti bei įsivaizduoti matematinius miglotosios logikos principus.
- Šio kalkuliatoriaus kūrimas leido geriau susipažinti su miglotosios logikos principais ir programavimo kalba Visual Basic 6.
- Esant vartotojų poreikiui sistema toliau galima vystyti papildant sistemos funkcionalumą.

7. LITERATŪRA

[1] Dimiter Driankov, Rainer Palm.Advances in Fyzzy Control.1998

[2] Cox, Earl. The Fuzzy Systems Handbook: A Practitioner´s Guide to Building, using, and Maintaining Fuzzy Systems. Academic Press, New York. 1994

[3] K.M.Passino, S.Yurkovich. Fuzzy Control. Addison Wesley. 1998

[4] Compact fuzzy models through complexity reduction and evolutionary optimization. FUZZ-IEEE 2000, Sant Antonio, Texas, USA, 2000 May 7-10

[5] Aleksander Elder. Traiding for a living.1993

8. SUMMARY

The topic of the work for Master's degree is: "The formation of fuzzy logic links and structural models".

The purpose of this work is to design a calculator, which would make arithmetic operations with the elements of fuzzy logic. This programme would allow to make such operations with fussy logic elements: addition (+), substraction (–), multiplication (×), division

(:).

Five main parts are presented in the work: analytical one, designed one, the documentation of the consumer, the evaluation of fuzzy logic calculator, the conclusions.

In the first part the analysis of fussy arithmetic operation is made, the operations which can be applied in designing are given.

In the second part the requirements of the consumer are analyzed and functional requirements are given.

The specification of the system is made. The schedule of the project performance is set. The measures of project hazard and prevention are set. The testing, which has been made is described. The plan of system developing is presented.

In the third part the documentation of consumer is written.

In the fourth part a thorough analysis of calculator working is made, its advantages and disadvantages, fields of application are singled out.

The fifth part is the conclusions. Fussy logic makes it possible to achieve much better management characteristics than by traditional methods. One may use the designed calculator by teaching the course of "Calculating methods with indefinite sets" and will help students to master and imagine the principles of fussy logic. If there is a consumer need, the system can be further developed by complementing system operation.

9. PRIEDAI

PRIEDAS 1

Miglotosios logikos kalkuliatoriaus langas