KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PRAKTINĖS INFORMATIKOS KATEDRA

Vilma Riškevičienė

GILJOTININIO SUPJAUSTYMO METODAS IR JO TYRIMAS

Magistro darbas

Vadovas
doc. dr. D.Rubliauskas

KAUNAS, 2005

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PRAKTINĖS INFORMATIKOS KATEDRA

TVIRTINU
Katedros vedėjas
doc. D.Rubliauskas
2005-01-10

GILJOTININIO SUPJAUSTYMO METODAS IR JO TYRIMAS

Informatikos magistro baigiamasis darbas

Recenzentas
Doc. K. Motiejūnas
2005-01-10

Vadovas
doc dr. Dalius Rubliauskas
2005-01-10

Atliko
IFN 2 gr. stud.
V.Riškevičienė
2005-01-10

KAUNAS, 2005

KVALIFIKACINĖ KOMISIJA

Pirmininkas: Raimundas Stulpinas, UAB „Strauja“ generalinis direktorius

Sekretorius: Antanas Lenkevičius, docentas

Nariai:

Rimantas Butleris, docentas

Valentinas Kiauleikis, docentas

Jonas Kazimieras Matickas, docentas

Bronius Paradauskas, docentas

Dalius Rubliauskas, docentas

Aleksandras Targamadzė, profesorius

SUMMARY

The combinatorial optimization problem considered in this paper is a special two dimensional cutting stock problem arising in the wood, metal and glass industry. Given a demand of non oriented small rectangles and a theoretically infinite set of large stock rectangles of given lengths and widths, the aim is to generate layouts with minimal waste, specifying how to cut the demand out of the stock rectangles. Special restrictions are chosen with respect to the production conditions of the flat glass industry.

Because of the increasing costs of raw material and the need to avoid industrial waste, solving cutting stock problems became of great interest in the area of operations research. Cutting stock problems belong to the class of combinatorial optimization problems, so a solution among all possible solutions has to be found, which optimizes a criterion function subject to a set of constraints.

An analysis of 2D object packing algorithms, created by foreign authors, and commercial two-dimensional stock packing software packages was performed in this work; also methods' advantages and disadvantages were pointed out. A model of two-dimensional rectangular object packing into stock was formed, and the software solving 2D object packing problems was created according to this model. The tests proved that the suggested method is effective and beneficial.

Turinys

Lentelių sąrašas

Paveikslėlių sąrašas

ĮŽANGA

Su dviejų dimensijų (2D) pakavimo problemomis susiduriama įvairiose pramonės srityse, kuriose yra būtinybė kuo geriau supakuoti tam tikrus objektus projektuojant juos į plokštę.

Dažniausiai su pakavimo problemomis susiduriama stiklo, medžio ar kitų medžiagų pjaustymo veiklos planavimo srityse. Specialūs apribojimai yra naudojami stiklo pramonėje. Norint, kad minėtose srityse būtų efektyviai sumažinti atliekų kiekiai, efektyviau panaudoti turimi resursai, yra **aktualu** nagrinėti 2D objektų pakavimo metodus, tirti jų ypatumus, pritaikymo galimybes ir principus. Pjaustymo uždaviniai priklauso optimizavimo uždavinių klasei, todėl įmanomi sprendimai bus randami, optimizuojant funkcijas.

Šiame darbe bus nagrinėjamas stačiakampių supakavimo plokštėje, uždavinys.

Tyrimo **tikslas** yra sudaryti 2D objektų giljotininio pjaustymo algoritmą, sukuriant techninį pakavimo projektavimo planą, kuris leistų su kuo mažesnėmis atliekomis supakuoti stačiakampius plokštėje, patenkindamas funkcinius ryšius tarp dalių rinkos sąlygomis, įskaitant kainą, saugumą, komfortą, įvertinant ekonominius ir aplinkos aspektus.

Projekto **uždaviniai**:

- Išanalizuoti užsienio autorių sukurtus dvimačio pakavimo algoritmus ir metodus.
- Išanalizuoti komercinę programinę įrangą, skirtą pjaustymo uždaviniams realizuoti.
- Sukurti programinės įrangos produktą, kuris leistų vartotojui pasirinkti optimalų pjaustomų stačiakampių pakavimo plokštėje, variantą.
- Suprojektuoti gerą, aiškų ir patogų vartotojui interfeisą.
- Testuoti programą, bandyti praktinį veikimą.
- Pateikti dokumentaciją.

Metodai: mokslinės literatūros analizė, programavimas.

1. GILJOTININIO SUPJAUSTYMO METODŲ TYRIMAS

1.1. METODŲ ANALIZĖ

Sprendžiant pakavimo problemas, buvo atlikta daugybė tyrimų ypač vienos arba dviejų dimensijų atvejams. Didelis dėmesys yra skiriamas 2D objektų pakavimo problemas sprendžiančių efektyvių modelių kūrimui ir vystymui. Darbe analizuojami įvairių autorių sukurti algoritmai, kuriuos apibendrinus išskiriami jų veikimo esminiai privalumai ir trūkumai.

Uždavinio formulavimas: duotas baigtinis skaičius n reikalingų stačiakampių, kurių ilgis l_i, ir plotis w_i, ir teoriškai begalinė aibė plokščių $\{R_1, R_2,...R_n\}$, kurių ilgis bei plotis žinomi. Reikia rasti pjaustomų stačiakampių supakavimą plokštėje, kad būtų sunaudotas minimalus plokščių skaičius. Leidžiami tik giljotininiai pjūviai, tai reiškia ,kad pjūvis turi prasidėti nuo vienos stačiakampio briaunos iki priešingos briaunos, tiesiniu pjūviu. Pjaustomus stačiakampius galima sukioti 90° kampu. Kai kurie papildomi apribojimai gali būti įvesti priklausomai nuo pramonės šakos specifikos.

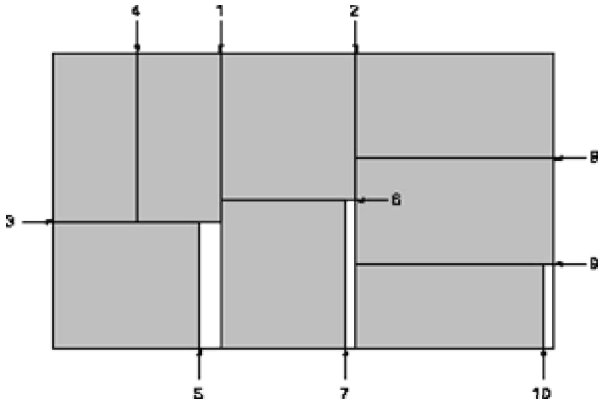
Tai yra NP sudėtingumo kombinatorinio optimizavimo uždavinys, kurio sprendimui yra sunku sukurti euristiką, paremtą godumo principu, kuri duotu mažą atliekų kiekį per trumpą skaičiavimo laiką. Ši problema yra sprendžiama daugelio mokslininkų, todėl pasaulyje yra paplitę daug įvairių sprendimo būdų. Paprastai yra naudojami euristiniai sprendimo būdai. Siekiant sumažinti skaičiavimo laiką pasitelkiamos genetinių algoritmų, dinaminio programavimo bei kitos technologijas. Šalia giljotininio supjaustymo neretai yra sprendžiama ir panaudotų, bet dar tinkamų pjaustymui plokščių panaudojimo problema. Gamybinio proceso metu kartais naudingiau yra pjaustomus stačiakampius supakuoti ant kelių plokščių nei ant vienos (susidaro mažiau atliekų), arba pjaustant plokštes lieka atraižų, tinkamų pakartotiniam supjaustymui. Tokiu būdu prie pjaustymo mašinos kaupiasi atraižos, kurios užima naudingą plotą. Todėl kai kurie mokslininkai siūlydami savo giljotininio supjaustymo metodo sprendimo būdus pateikia ir galimus šios šalutinės problemos sprendimo variantus. Pasitaiko, kad plokštės būna su paviršiaus defektais, todėl būna svarbu įvertinti ir defektinius plotus, iš kurių negali būti kerpami stačiakampiai.

1.2. UŽDAVINIO SPRENDIMO BŪDŲ ANALIZAVIMAS

Galimi tokie uždavinio sprendimo būdai:

1. Sprendimo būdai, kurie yra Stockmeyer pasiūlyto algoritmo modifikacijos, ar patobulinimai.
2. Sprendimo būdai, kuriuose pasitelkiama genetinių algoritmų technologija.
3. Sprendimo būdai, kur naudojamos kitos metodikos šios problemos sprendimui.

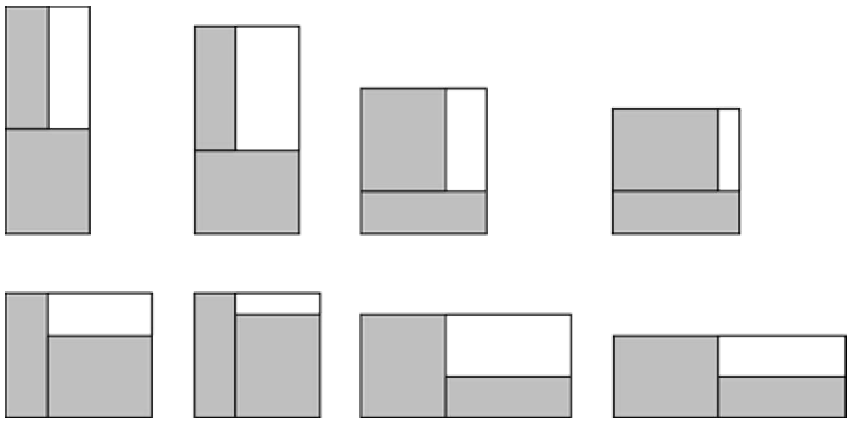
Plačiau paanalizuosime Stockmeyer siūlyto metodo algoritmą. Jo esmė yra ta, kad spręsti uždaviniui yra naudojamos formų funkcijos (shape function) ir pjaustymo medis(slicing tree).



1.1 pav. Galimi giljotininiai pjūviai

Sprendžiant šį uždavinį kiekviename žingsnyje pjaustomieji stačiakampiai yra poruojami pagal maksimalų svorių sutapimą. Tokios sutapusios poros yra traktuojamos kaip nauji stačiakampiai, kuriuos taip pat reikia supakuoti plokštėje. Naujai sudaryti stačiakampiai yra vadinami meta stačiakampiais. Tad sekančiuose žingsniuose meta stačiakampiai yra taip pat poruojami pagal maksimalų svorių sutapimą. Po to yra atliekama sutapimų paieška tol kol susidaro giljotininės struktūros. Paieška baigiama kuomet sudaryti meta stačiakampiai persikerta tarpusavyje. Kiekviename žingsnyje yra patikrinami visi galimi supakavimo variantai, pasitelkiant formų nustatymo funkcijas (shape functions}.

Meta stačiakampius galima supakuoti keletu būdų, priklausomai nuo vertikalaus ar horizontalaus pjūvio orientacijos. Pvz., jeigu turime du pjaunamus stačiakampius A ir B, tai rezultate galime gauti meta stačiakampį AB, kuris turi 8 visiškai skirtingus supakavimus: A iš kairės arba aukščiau už B, stačiakampiai gali būti pasukti. 1.2 pav parodyti 8 skirtingi meta stačiakampių, susidedančių iš dviejų pjaunamų stačiakampių, supakavimai plokštėje:

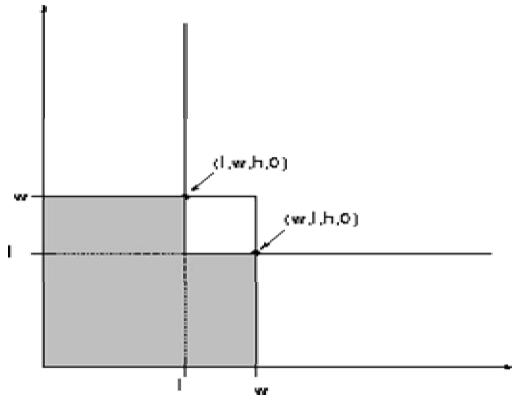


1.2 pav. Meta stačiakampių iš dviejų pjaunamų stačiakampių supakavimas plokštėje

Kiekvienas meta stačiakampio supakavimas yra aprašomas pjaustymo instrukcijoje T.y. keturių dimensijų vektorius (x, y, o, p), kur x ir $y$yra meta stačiakampio ilgis ir plotis, $o \in \{h, v\}$ yra orientacija vertikalaus ar horizontalaus pjūvio, kuris bus naudojamas padalinimui, p yra taškas nuo kurio turi prasidėti pjūvis.

Naudojant formų nustatymo funkcijas, galima nustatyti visas galimas meta stačiakampio supjaustymo kombinacijas.

Taigi, jei skaičiuojamas meta stačiakampis, turintis keletą pjaustomų stačiakampių, ir jei plokštės išmatavimai yra žinomi, tai maksimalus reikalingas dydis x, reikalingas meta stačiakampių pakavimui gali būti rastas įvedus plokštės plotį.



1.3 pav. Pjaustomo stačiakampio formų funkcija.

Tokios formų funkcijos yra pateikiamos kaip pjaustymo instrukcijų sąrašas. Pjaustomų stačiakampių, kurių ilgis yra l ir plotis $w > l$, formų funkcijos aprašomos kaip dvi pjaustymo instrukcijos $((l, w, h, 0), (w, l, h, 0))$. Pjūvio orientacija ir pradžios taškas kol kas nėra nurodytas.

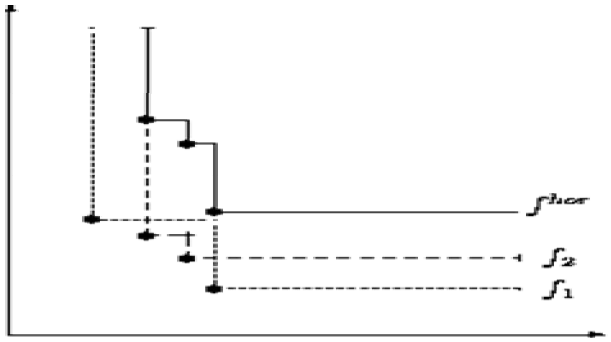
Meta stačiakampių formų funkcijos yra skaičiuojamos kompozicijos procedūros pagalba. Sakykime, kad f_1 ir f_2 yra dvi formų funkcijos ir $s^i = (x^i, y^i, o^i, p^i)$ yra pjaustymo instrukcijos, kai $f_i (i = 1, 2)$. Pjaustymo instrukcijos komponavimas:

Horizontaliam pjūviui : $\mathrm{comp}(\mathrm{hor}, s^1, s^2) := (\max(x^1, x^2), y^1 + y^2, h, \min(y^1, y^2))$

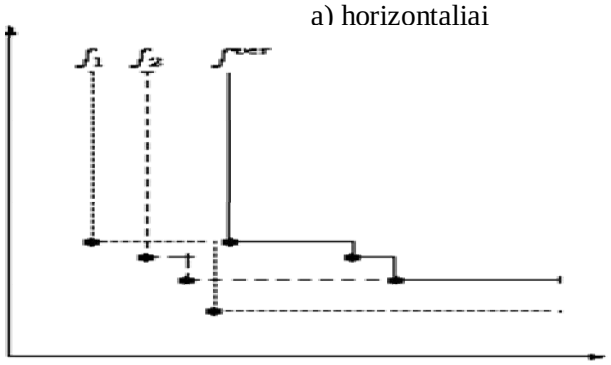
Vertikaliam pjūviui : $\mathrm{comp}(\mathrm{ver}, s^1, s^2) := (x^1 + x^2, \max(y^1, y^2), v, \min(x^1, x^2))$

Komponuojant visas pjaustymo instrukcijas f_1 ir f_2, dvi formų funkcijos f^{hor} ir f^{ver} yra skaičiuojamos visas įmanomas horizontalaus pakavimo vertikalias pjaustymo instrukcijas. 1.4 pav. matome horizontalią ir vertikalią formų funkciją, gautą komponuojant dviejų pjaustomų stačiakampių formų funkcijas. Pjūvio orientacijai buvo sukurtos trys pjaustymo instrukcijos.

1.5 pav. pavaizduotos vertikali ir horizontali formų funkcija, gautos komponuojant dviejų stačiakampių formų funkcijas:

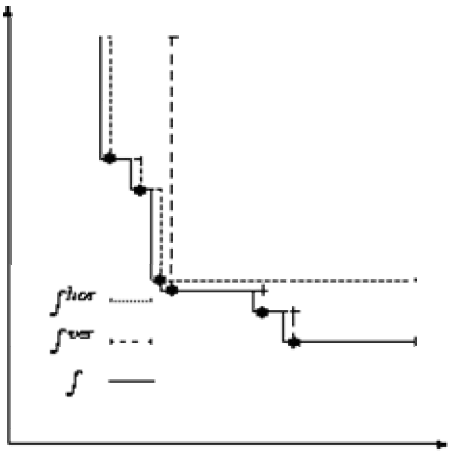


a) horizontaliai



b) vertikaliai

1.4 pav. Formų funkcijos su pjūvio orientacija



1.5 pav. Minimizuota formų funkcija.

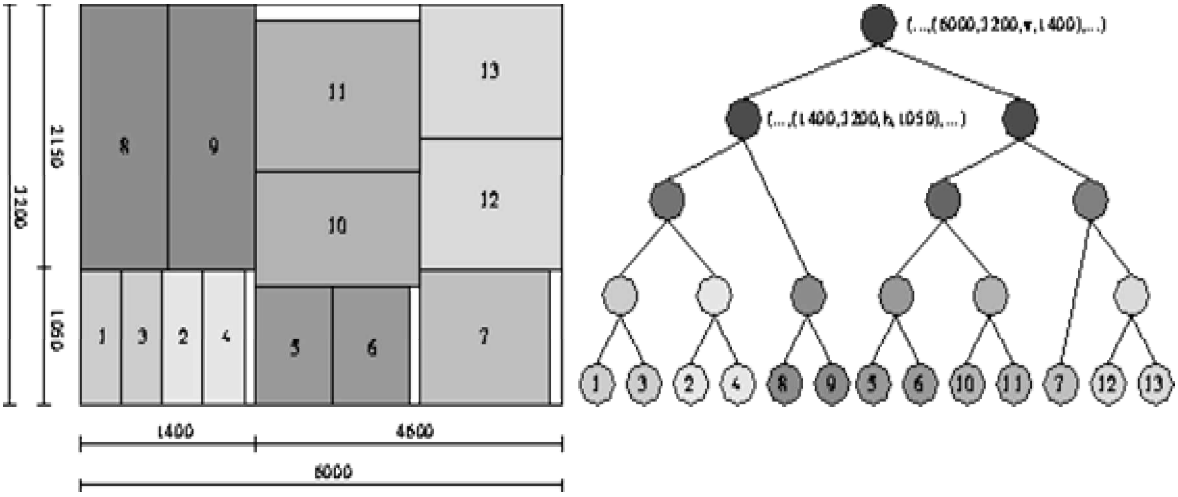
Skaičiuojant meta stačiakampiui formų funkciją f, imamas minimumas tarp f^{hor} ir f^{ver}, apjungiant šias dvi funkcijas ir išmetant dominuojančias pjaustymo instrukcijas. Pvz. jei pjaustymo instrukcija f^{ver} yra vyraujanti vienoje iš f^{hor}, jei $f^{hor}(w) < f^{ver}(w)$, su pločiu w . Jei dvi pjaustymo instrukcijos f^{hor} ir f^{ver} turi tokį patį plotį ir ilgį, tai pasirenkamas horizontalus pjūvis. Maksimalus pjaustymo instrukcijų skaičius, laikomas formų funkcijoje f yra skaičiuojamas, komponuojant dvi kitas formų funkcijas f_1 ir f_2 yra $2 \cdot (|f_1| + |f_2| - 1)$, kur $|f_i|$ nurodo funkcijos f_i pjaustymo instrukcijų skaičių.

Kiekviename iteraciniame žingsnyje, nustatyti poroms, gauti rezultatai yra saugomi pjaustymo medyje (slicing tree). Kiekvienas medžio lapas aprašo pjaunamą stačiakampį. Vidinis medžio mazgas nusako meta stačiakampį. Du mazgai turi tą patį tėvą, jei stačiakampiai, nusakomi šių mazgų yra poroje. Kiekviename mazge yra saugoma meta stačiakampį atitinkanti kreivės funkcija.

Pjaustymo medis yra formuojamas žemyn. Algoritmo pradžioje pjaustymo medyje yra tik vienas mazgas, aprašantis pjaunamą stačiakampį. Kiekviename iteracijos žingsnyje stačiakampiai yra poruojami, naujai sudaryti stačiakampiai yra vadinami meta stačiakampiais. Iteracijos baigiamos, kai daugiau nėra sutampančių porų. Visi mazgai turi formų funkciją ir kiekviena formų funkcija aprašo visus įmanomus meta stačiakampių supakavimus, pjaustymo medyje yra eksponentinis skaičius įmanomų plokštės supjaustymų.

Algoritmo pabaigoje galime sužinoti, kaip pjaustomi stačiakampiai gali būti išpjaunami iš plokštės. Iš kiekviename mazge esančios formų funkcijos galima sužinoti

kaip turi būti atliktas pirmas pjūvis ir surasti pirmą pjaustymo instrukciją, kurios reikšmė y yra mažesnė arba lygi plokštės pločiui. Priklausomai nuo aukščiau paminėtų savybių gauname pjaustymo instrukciją, kuri apibūdina porų išdėstymą su minimaliu ilgiu. Jei pirmas pjūvis bus atliktas, plokštė bus padalinta į dvi dalis. Šio medžio pagalba mes žinome atliekamų giljotininių pjūvių eiliškumą. Šis procesas parodytas 1.6 pav.:



1.6 pav. Stačiakampių plokštėje supjaustymas ir pjaustymo medis

Kiekvienam iteracijos žingsnyje suradus, kurie iš stačiakampių gali būti suporuoti pagal maksimalų svorių sutapimą, surašomi į grafą. Grafo mazgai vaizduoja meta stačiakampius, briaunos- įmanomas poras. Kuo didesnis svoris, tuo geresnė pora. Svoriai yra randami, naudojantis taip vadinamos kreivių funkcijos(benefit function) pagalba. Pvz. įmanoma dviejų stačiakampių kreivių funkciją apskaičiuoti visų galimų meta stačiakmpių supjaustymų atliekų vidurkį atėmus iš 100.

Šis uždavinys yra formuluojamas, kaip NP sudėtingumo uždavinys:

Maksimizuoti:

$$\sum_{i,j=1}^{n} c_{i,j} \cdot x_{i,j}$$

Į

$$\sum_{j=1}^{n} x_{i,j} \leq 1 \qquad \text{for} \qquad i = 1,\ldots,n$$

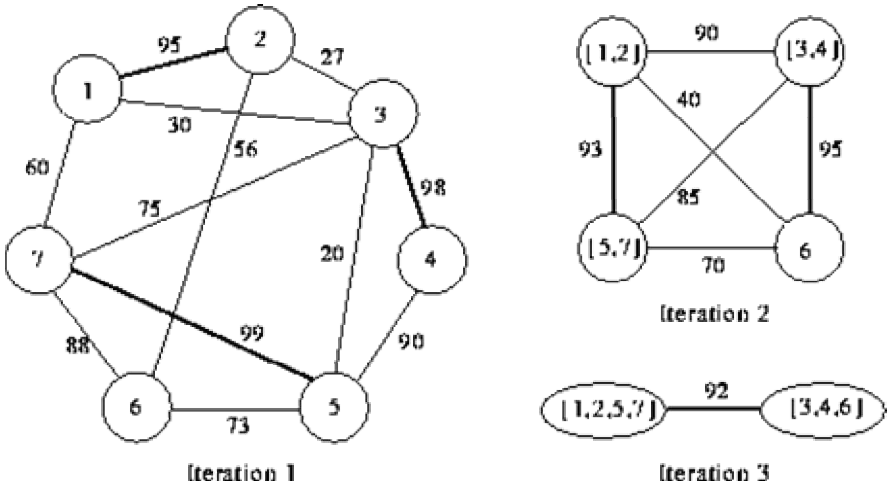
$$x_{i,i} = 0 \qquad \text{for} \qquad i = 1,\ldots,n$$

$$x_{i,j} = x_{j,i} \qquad \text{for} \qquad i,j = 1,\ldots,n$$

$$x_{i,j} \in \{0,1\} \qquad \text{for} \qquad 1 \leq i \leq j \leq n$$

Čia n yra meta stačiakampių kiekis, $\mathbf{c}$ -matrica. Kintamojo $x_{i,j}$ reikšmė yra 1, jeigu stačiakampiai i ir j sutampa, jei ne $x_{i,j} = 0$.

Algoritmas panaudotas šiame aprašyme, yra pasiūlytas mokslininko Edmonds, kuris parodė, kad maksimalių svorių sutapimo uždavinys gali būti išspręstas per laiką $O(n^3)$. 1.7 pav. parodytas septynių stačiakampių per tris iteracijos žingsnius sutapimo algoritmas(briaunos su nuliniu svoriu neparodytos):

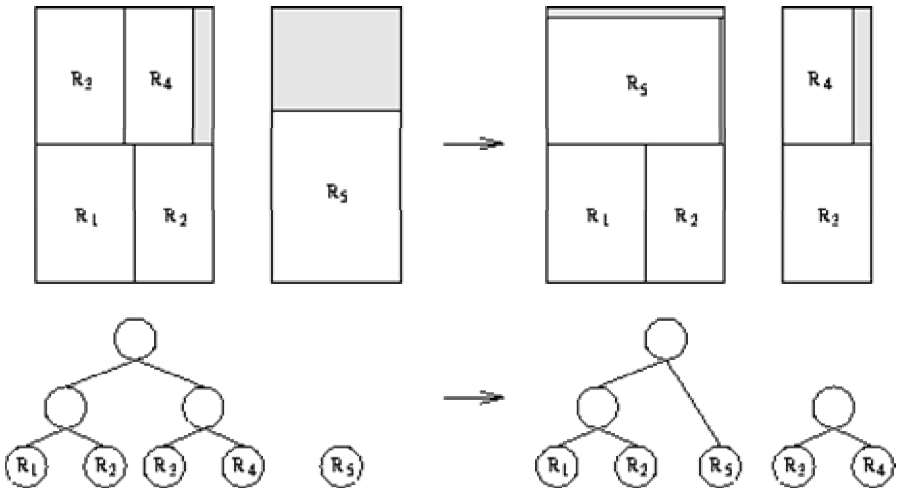


1.7 pav. Maksimalus svorių sutapimas per tris iteracijas

Paieška baigiama kuomet sudaryti meta stačiakampiai persikerta tarpusavyje.

Algoritmas susideda iš keturių etapų. Pirmuose dviejuose etapuose konstruojami persikirtimai. Trečiame etape jie pagerinami. Ketvirtame etape reikalinga pjūvius supakuoti į plokštę.

Pirmame etape universalus stačiakampis yra skaičiuojamas pagal maksimalų svorių sutapimą. Kad būtų galima pavadinti stačiakampį meta stačiakampiu, jis turi tenkinti dvi sąlygas: 1) Visų meta stačiakampių pjaustymo atliekų vidurkis turi būti mažesnis nei 5 procentai ir 2) Turi egzistuoti kitas meta stačiakampis, taip kad rezultate iš šių abiejų stačiakampių gavęsis meta stačiakampis persikirstų. Pjūvio atliekų plotas turi būti mažesnis nei 4 procentai. Pirmame etape, kiekviename iteracijos žingsnyje sutapimų grafas yra formuojamas iš turimų ne universalių stačiakampių. Pirmame etape kreivės funkcija suteikia porai aukščiausią reikšmę, jei visų meta stačiakampių supjaustymo atliekų vidurkis yra mažesnis nei 5 procentai. Pirmas etapas baigiasi kai nebegalima daugiau stačiakampių išdėlioti, nepažeidžiant apribojimų. Antra stadija susideda tik iš vieno iteracijos žingsnio. Joje n konstruojamų stačiakampių yra dalinami į $n/2$ pjūvių. Kuo didesnis užimamo ploto procentas, tuo geresni pjūviai. Trečioje stadijoje reikia pagerinti pjūvius, taip kad sumažėtų atliekų plotas. Tai įmanoma padaryti, kai pjaustomi stačiakampiai yra pakuojami ant mažesnių plokščių. 1.8 pav parodyta pjūvių patobulinimas:



1.8 pav. Pjūvių patobulinimas

Šio algoritmo privalumai:

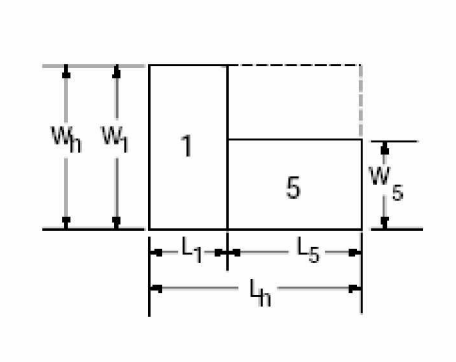
- *Tai efektyvus metodas, norint supakuoti didelį kiekį stačiakampių.*
- *Palyginti nedidelis skaičiavimų laikas.*
- *Mažas atliekų procentas*

Trūkumai:

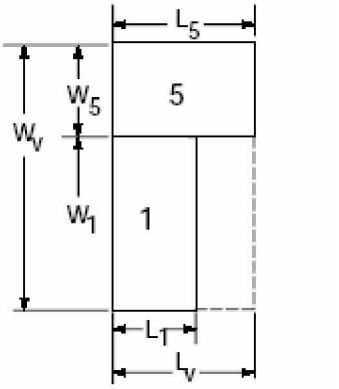
Didesnių trūkumų nepastebėta.

Kai kurie mokslininkai labiau linkę naudoti genetinių algoritmų galimybes, bandydami paspartinti skaičiavimus bei mažindami atliekų kiekį. Dažnai šie sprendimo būdai apjungia kelias technologijas. Toshihiko Ono ir Gen Watanabe pateiktame sprendimo variante yra naudojami genetinis algoritmas (GAs) ir giljotininio nustatymo algoritmas (GCLAs). Paanalizuosime plačiau šį algoritmą.

Algoritmas susideda iš dviejų dalių, horizontalios ir vertikalios funkcijos pakavimo, ir nuoseklaus dalies numerio priskyrimas visoms dalims. Išlyginimas vyks taip, visos stačiakampio dalys jungsis rekursyviai tol, kol pabaigoje plokštėje susidarys vienas didelis stačiakampis. Šis susijungimas vyks pagal sudarytus algoritmus: supakavimas ir išlyginimas bus perduodamas genetiniam algoritmui (GAs), ir Genetinis algoritmas perduos reikalingą plokštės ilgį. Sekantis veiksmas: Genetiniame algoritme tikslus supakavimas modifikuojamas genetinės operacijos, kuri sukuria mažiausiai tikėtiną reikalingo ilgio meta stačiakampį. Susijungus šioms dviems operacijoms, pavaizduojamas optimaliausias dalių supakavimas. 1.9 ir 1.10 pav. parodyti vertikalus ir horizontalus stačiakampių supakavimas:



1.9 pav. Horizontalus stačiakampių supakavimas



1.10 pav. Vertikalus stačiakampių supakavimas

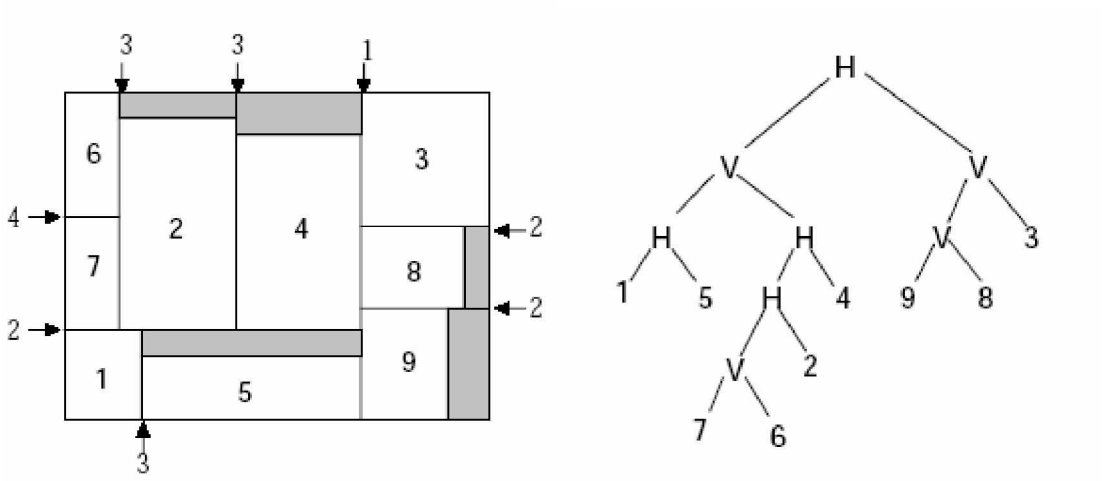
Sprendžiant optimalų stačiakampių pakavimą plokštėje, galimi du būdai: „iš viršaus– žemyn“ ir „iš apačios– aukštyn“. „Iš viršaus– žemyn“ supakavimas yra nustatomas remiantis dabartiniu pjaustymo veiksmu. Pradedamas nuo pirminio plokštės, pjovimo veiksmas tęsiasi tol, kol visos reikalingiausios stačiakampio dalys yra išgaunamos, iš horizontalaus ir vertikalaus giljotininio pjaustymo, išrenkamas tinkamas pjovimo metodas.

„Iš apačios – aukštyn“, prasideda nuo kiekvieno reikalingo stačiakampio, du ar daugiau stačiakampius jungiant rekursijos būdu horizontaliai ar vertikaliai, taip kad būtų gaunamas mažiausias stačiakampis, tol kol pagaliau susideda viena stačiakampio plokštuma. Tokiu būdu stačiakampiai sudėlioti plokštėje, gali būti pjaustomi į kelias dalis giljotininiu pjaustymu. Palyginus šiuos du būdus, buvo nuspręsta priimti pastarąjį „Iš apačios – aukštyn“ būdą, kadangi jis labiau atitinka uždavinį.

Kaip supakuoti visus stačiakampius plokštėje? Kai keletas stačiakampių jungiami į mažiausią stačiakampį, verčiant ir jungiant šiuos du stačiakampius horizontaliai ar

vertikaliai vieną šalia kito. Pritaikant šitą metodą visiems stačiakampiams ir stačiakampių susijungimui, pabaigoje gaunamas vienas didelis stačiakampis.

Atliekant genetinį stačiakampių dalių pavaizdavimą ir pakavimą plokštėje, turi nebūti jokių neaiškumų, todėl buvo naudojami dvinariai operatoriai: *H* ir *V*. *H*-operatorius, kaip argumentus paima du stačiakampius ir sukuria mažiausią stačiakampį, pakuojant juos horizontaliai. Pavyzdžiui, formulė „1 5 *H*“ reiškia kad stačiakampis Nr.5 yra iš dešinės pusės stačiakampiui Nr.1, kaip parodyta 1.11 pav. ir gražina identifikacinį šio integruoto stačiakampio numerį.



1.11 pav. Stačiakampių supakavimas plokštėje, naudojant giljotininius pjūvius ir medis

Taigi meta stačiakampio ilgis L_h ir plotis W_h apskaičiuojamas:

$$L_h = L_1 + L_5 \,,\; W_h = \max(W_1,\, W_5),$$

kur, L_1 ir L_5 yra šių dviejų mažų stačiakampių ilgis, W_1 ir W_5 yra pločiai. Funkcija max $(*, *)$ suskaičiuoja dviejų reikšmių maksimumą.

Operatorius *V* yra labai panašus į *H*- operatorių, išskyrus tai, kad du stačiakampiai yra supakuoti vertikaliai. Taigi formulė „1 5 *V*“ parodo, kad stačiakampis Nr.5 yra virš stačiakampio Nr.1, kaip parodyta 1.11 pav ir meta stačiakampio išmatavimai yra nustatomi taip:

$$L_v = \max(L_1, L_5) \,,\; W_v = W_1 + W_5 \,.$$

Panagrinėkime šių stačiakampių supakavimą, kurį galima užrašyti tokia formule:

1 5 *H* 7 6 *V* 2 *H* 4 *H V* 9 8 *V* 3 *V H*

Ši formulė rodo ruošinių supakavimą plokštėje, ir kaip matyti iš 1.11 pav. rodyklės rodo giljotininio pjūvio vietas. Iš 1.11 pav. taip pat galima nesunkiai suprasti, kad

nubraižytas medis rodo, kaip bus atliekamas giljotininis pjūvis. Formuojant medį, einantį žemyn nuo medžio šaknų iki lapų, galima gauti giljotininių pjūvių eilę, išpjaunant reikiamas stačiakampio dalis iš plokštės.

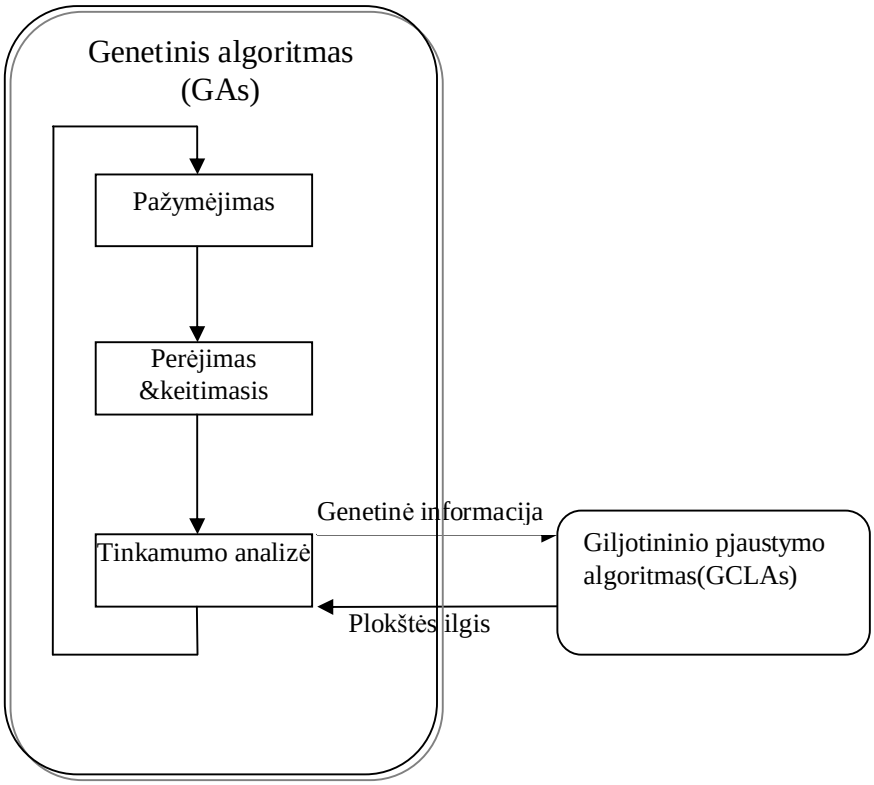
Išaiškinsime keletą šios lygties charakteristikų. Pirma, čia yra keletas apribojimų, kuriuos operatoriai naudoja plokštės pakavimui ir šių operatorių pozicija formulėse. Tegu stačiakampio dalių skaičius būna N_p, ir atitinkamai operatorių skaičius būna N_o, tada dvinarė lygtis yra:

$$N_o = N_p - 1 .$$

Antra, rašant formulę geriausia, kad nebūtų jokių skliaustelių, tada nekils jokių neaiškumų, ir formulės ilgio rezultatas bus aiškus, taigi N_g ilgis bus laikomas konstanta.

$$N_g = N_p + N_o = 2N_p - 1$$

Sistemos konfigūracija:



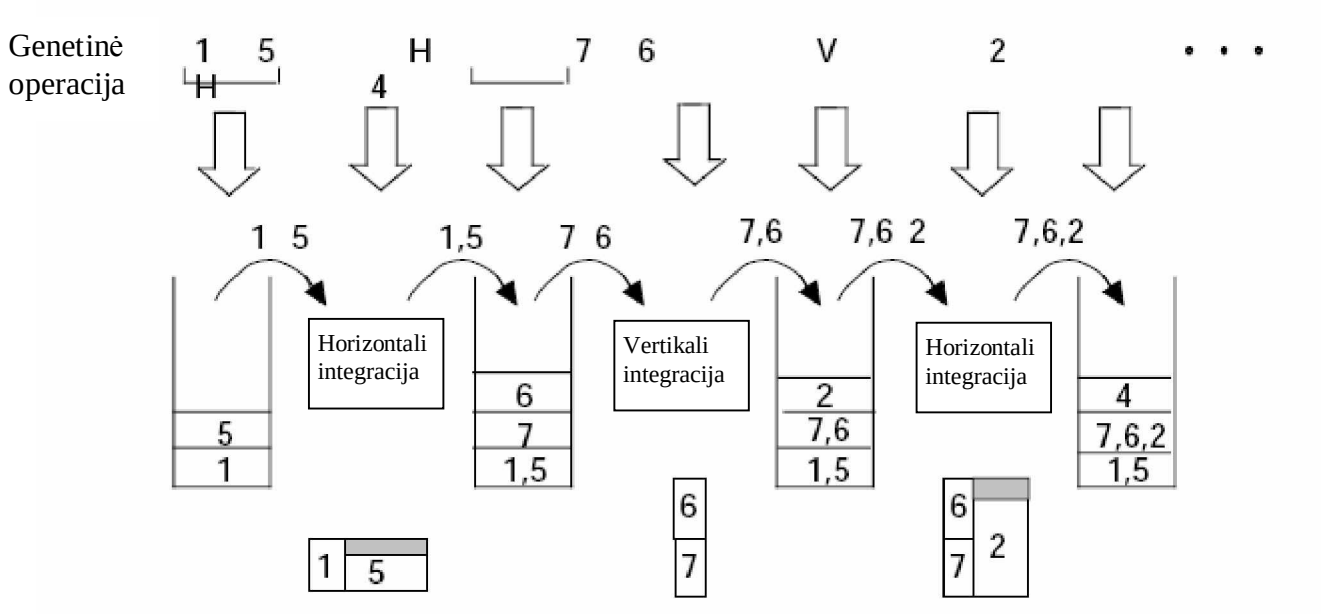
1.12 pav. Sistemos konfigūracija

Kitas svarstomas klausimas yra apribotas stačiakampio dalių ir operacijų skaičius. Palyginkime bet kurį vieną operatorių; tegul kairėje šio operatoriaus pusėje dalies numeris bus n_p ir operacijos numeris bus n_o, iš to seka ryšys:

$$1 \leq n_o \leq n_p - 1.$$

Sistema sudaryta iš dviejų dalių, Genetinio algoritmo ir Giljotininio pjaustymo algoritmo kaip pavaizduota 1.12 pav. Giljotininio pakavimo planavimo algoritmas pakuoja plokštėje stačiakampius, taip kaip jam nurodo Genetinis algoritmas (GAs), apskaičiuoja reikalingą ilgį plokštėje ir jį gražina GAs. Genetinis algoritmas (GAs) modifikuoja stačiakampių pakavimą plokštėje taip, kad reikiamos plokštės ilgis būtų minimalus.

Anksčiau užrašytą nelygybę, naudojantis šiuo algoritmu galima atvaizduoti 1.13 pav:



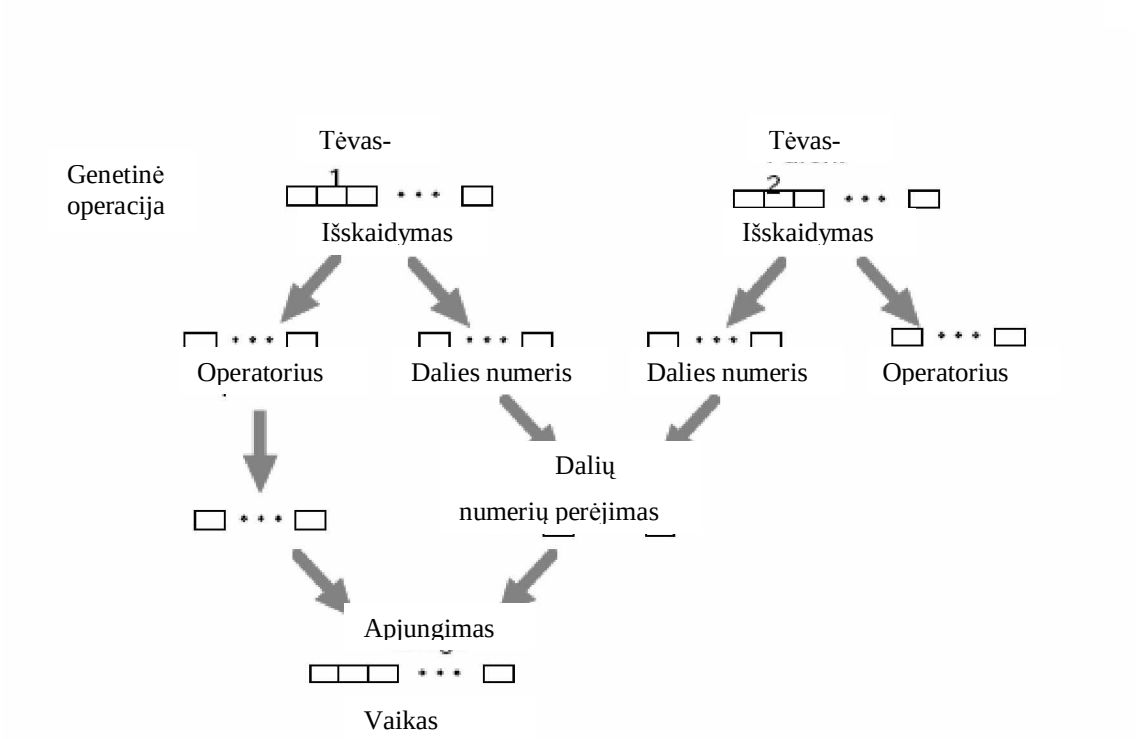
1.13 pav. Nelygybės interpretacija

1. Jei tai yra dalies numeris, tai įrašomas į sąrašą(steką).
2. Jeigu jis yra bet kuri *H* ar *V* operacija, tada iš keleto dalių imamos dvi dalys iš kurių gaunamas bendras stačiakampis ir šio naujai registruoto stačiakampio dalies numeris taip pat įrašomas į steką.
3. Kada lygties elementai interpretuoti, visi likučiai sudedami į vieną dalį ir dedami į galą, kai iš stačiakampių susidaro sujungtas stačiakampis, pakavimo operacija baigėsi. Horizontalus šio stačiakampio ilgis ir yra reikalingas plokštės ilgis.
4. Grįžtame prie 1 žingsnio.

Vykdant šias procedūras, jungiamo stačiakampio vertikalus ilgis gali būti didesnis negu plokštės ilgis. Kad tai neįvyktų, reikia ištaisyti įdiegtą algoritmą. Kai tai atsitinka, dabartinis lygties operatorius pakeičiamas į kitą tipą, tai reiškia kad *V* operatoriaus pakeičiamas į *H* operatorių ir šis gražinamas į Genetinį algoritmą(GAs). Jeigu

nepastebėta pasikeitimų, kada procedūrą vykdo *H* operatorius, paprasčiausiai tuo metu jokių sutrikimų nebuvo.

Genetinis algoritmas (GAs) vartojamas išreikšti ir sujungti stačiakampius ir operatorius, genetinis operatorius, skiriasi nuo paprasto operatoriaus. Stačiakampių dalių numeriai nagrinėjami kitokia tvarka, kol operatoriai nagrinėja kuris iš dviejų atliks veiksmą: *H* ar *V* operatorius. Šis operatorius sprendžia, ar pasirinkti sumaišytas genetines operacijas. Genetinis algoritmas suskirstytas į du veiksmus, tai: stačiakampio dalių veiksmas ir operacijų veiksmas, naudojant kaukę, kaip parodyta 1.14 pav. Po privalomos genetinės operacijos kiekvienas veiksmas yra atliekamas taip: du veiksmai sujungiami į vieną veiksmą, ir vėl iš naujo padeda vykti genetinės operacijos tol, kol visos operacijos sudėlioja stačiakampių dalis.



1.14 pav. Genetinis išskaidymas

Crossover (perėjimas.) Stačiakampio dalių grupės paverčiamos į kito tipo genetinį algoritmą (GAs) kol pritaikomos. Čia yra daugybė pavertimo būdų ir rūšių, po atlikto palyginimo apžvelgiami skirtingi būdai, tokie kaip PMX (dalinai – pažymėtas perėjimas), OX (sutvarkytas perėjimas) ir CX (ciklinis perėjimas) buvo sukurti, galiausiai pasirinktas buvo PMX pervertimas, nes jis geriausiai sudėliojo būdingas ypatybes.

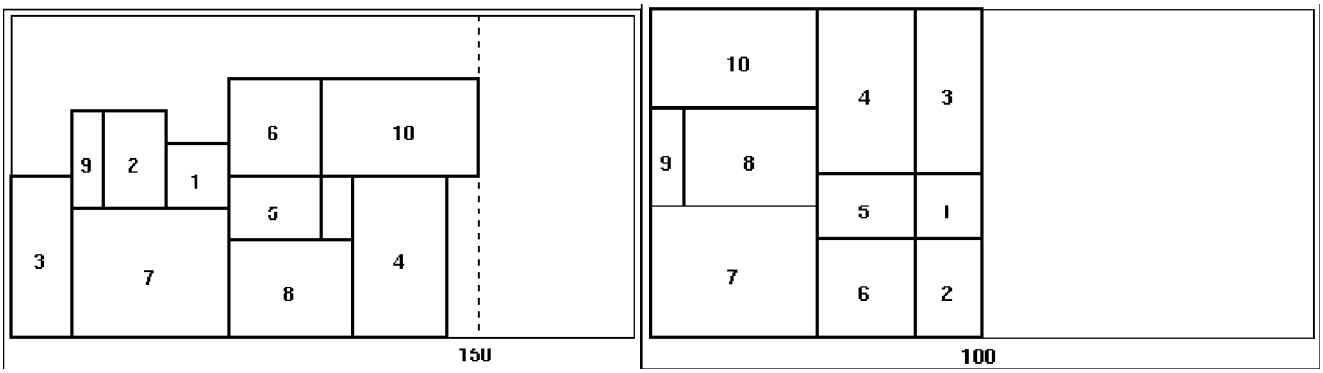
Mutation (keitimasis)**.** Keitimasis prasideda kai susidaro ir apsikeičia du elementai, įskaitant abu operatorius ir stačiakampio dalis kai jos apsikeičia skirtingo tipo operatoriais. Pasirinkime du atsitiktinius elementus ir pavadinkime juos atitinkamai p_1 ir p_2, kur p_1 randasi p_2 kairėje pusėje. Pagal tai kiek yra elementų rūšių, atitinkamai išrenkamos apsikeitimo tipo sekos. Kai abi p_1 ir p_2 yra stačiakampio dalyse, tai apsikeitimas tarp jų yra leidžiamas. Jeigu p_1 yra stačiakampio dalis ir p_2 operacija, apsikeitimas pasiseks tik tada, kai operacijos reikalavimo padėtis atitiks veiksmą. Po operacijos *p2* yra apsikeitusi su stačiakampio dalimi *p1*, tai nustatyti nelygybės apribojimai turi pasilikti, sekantis ryšys turi būti padengtas visoms operacijoms esamoms tarp p_1 ir p_2 .

$$n_0 \leq n_p - 3$$

Kitame pavyzdyje, kur p_1 yra operacija, nekyla jokių klausimų darant apsikeitimą nepaisant p_2 rūšies.

Kitokia procedūra, kai keičiasi H ir V operatoriai, tai reiškia kad išsijungimas tarp šių operatorių yra galimas.

Susivienijus perėjimo ir keitimosi operacijoms paaiškinamai visi įmanomi apsikeitimo būdai šablone.

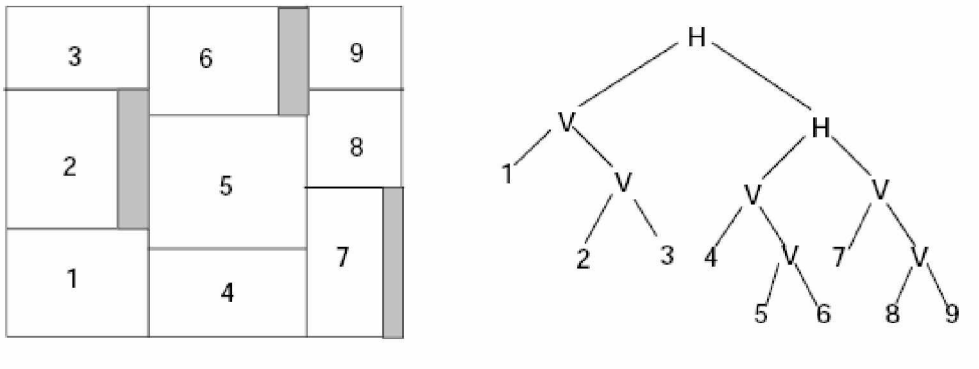


1.15 pav. Rezultatų atvaizdavimas: a) generacijos pradžioje; b) po 18-os generacijų

Anksčiau aprašyto metodo optimalaus uždavinio sprendimo suradimas yra sumodeliuotas ir parodytas 1.15 pav.

Giljotininio pjūvio ištęsimo stadija. Giljotininiame pjūvyje, kiekvieną kartą pjūvio kryptis keičiasi, pavyzdžiui iš horizontalaus į vertikalų ir atvirkščiai, pjūvis gali būti pasuktas 90 laipsnių. Tai yra papildomas darbas, norint sumažinti pasisukimų skaičių

kiek galima mažiau. Mažiausias apsisukimų skaičius yra 2, kiekvienam horizontaliam ir vertikaliam pjūviui, kaip parodyta 1.16 pav.:



1.16 pav. a) stačiakampių supakavimas, b) medis

1.16 pav. pavaizduotas pjūvio metodas, buvo išreikštas tokia formule:

123VV456VV789VVHH

Viena iš lygties padėčių išsidėstymo, po kurio galimi du giljotininiai pjūviai, kai visi *H* operatoriai nustatyti iš dešinės pusės visiems *V* operatoriams. Pridėjus neribotą genetinę operaciją, antra stadija būtų realizuoti giljotininį pjūvį.

Taigi šio algoritmo esmė yra ta, kad GAs ir GCLAs dirba paeiliui bendraudami tarpusavyje. GAs nustato šablonų eilę, kuri turi būti išdėliota plokštėje ir siunčia ją GCLAs. GCLAs fiksuoja kiekvieno šablono padėtį taip, kad šie šablonai tarpusavyje nepersidengtų bei nesudarytų laisvų plotų. Kuomet visų šablonų padėtis tampa fiksuota GCLAs siunčia reikalingą plokštės ilgį GAs. tam kad pastarasis atitinkamai išdėliotų šablonus. GAs turėdamas gautą ilgį nustato šablonų eiliškumo išdėstymą taip kad reikiamos plokštės ilgis būtų minimalus.

Šio algoritmo privalumai:

Mažas atliekų procentas;

Trūkumai:

Didelis skaičiavimo laikas.

Kai kurie mokslininkai bando ieškoti naujų euristinių būdų šio uždavinio sprendimui. Dažniausiai tai būna kelių metodikų kombinavimas. Vieną iš tokių sprendimų siūlo M.Ronnqvist ir E.Astrand savo straipsnyje. Problema dalinama į lygius, atvaizduojančius diskretizacijos elementus, reikalingus sprendimų priėmimui. Kiekviename lygyje gali būti keletas būsenų.. Būsenos suteikia informaciją, reikalingą optimalaus sprendimo priėmimui kiekviename lygyje. Sprendimas priimtas kiekviename lygyje nusako kaip esamo lygio būsena transformuosis į sekančio lygio būseną.

Sprendimas priimtas esamai būsenai yra nepriklausomas nuo ankstesnių būsenų ar priimtų sprendimų. Kiekviename lygyje priimti sprendimai yra įrašomi. Ryšys tarp lygių nusako galimą pjūvį.

Apibendrinant išnagrinėtus algoritmus, galime daryti išvadą, kad sprendžiant giljotininio supjaustymo uždavinį galima būtų pasinaudoti šiais sprendimo būdais:

1. *Stockmeyer pasiūlyto algoritmo modifikacijomis, ar patobulinimais.*
2. *Pasitelkiant genetinių algoritmų technologiją .*
3. *Kitos metodikos šios problemos sprendimui.*

Dėmesio verti yra sprendimo būdai, paremti Stockmeyer pasiūlytu algoritmu. Šio tipo algoritmai pasižymi mažu atliekų procentu, bei mažu skaičiavimo laiku, taipogi juos nesunkiai galima realizuoti. Mažu atliekų procentu pasižymi ir sprendimo būdai sudaryti genetinių algoritmų pagalba, tačiau jų skaičiavimo laikas yra daug didesnis nei Stockmeyer pasiūlyto algoritmo modifikacijų. Remiantis išanalizuotais metodais, darbe pateikiamas algoritmas, pasižymintis nedideliu atliekų procentu ir mažu skaičiavimo laiku.

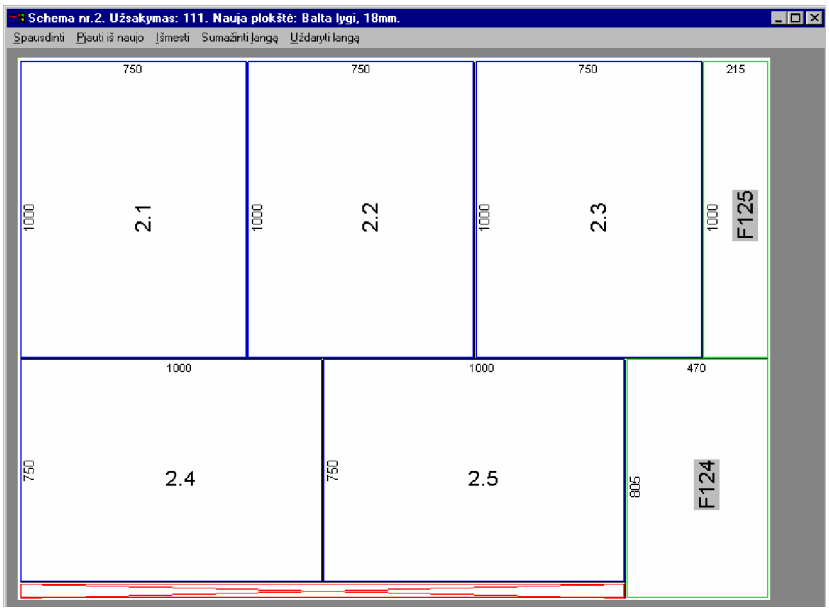
1.3. KOMERCINIŲ PJAUSTYMO PROGRAMŲ ANALIZAVIMAS

Šiuo metu rinkoje yra nemažas kiekis programinės įrangos, skirtos supjaustymo uždaviniui spręsti. Kiekviena jų skiriasi ne tik skaičiavimo laiko trukme, kaina ir kt. parametrais, bet, kas šiuo atveju ypač aktualu, ir efektyvumu(t.y. kuo tikslesniu ruošinių supakavimu plokštėje).

Optimizuoto pjaustymo programa -„ DALGIS“.

Programa **-„ DALGIS“** (1.17 pav.) skirta firmoms atliekančioms medžio (stiklo ir pan.) pjaustymo darbus. Jos paskirtis:

- § saugoti žaliavų bei jų likučių duomenis (duomenų failuose);
- § leisti suformuoti užsakymą (-us) tam tikros žaliavos (-ų), tam tikro kiekio gabalų išpjovimui;
- § saugoti jau įvestų užsakymų duomenis;
- § apskaičiuoti užsakymo (-ų) sąmatą, kad būtų galima, esant reikalui, pateikti klientui sąskaitą;
- § leisti supjaustyti įvestus užsakymus iš žaliavų naujų plokščių arba iš žaliavų likučių (nuopjovų);
- § supjaustytus užsakymų duomenis atvaizduoti šablonais, kuriuose grafiškai (Dekarto plokštumoje, milimetrų tikslumu) parodyta, kaip optimaliai supakuoti gabalus;
- § saugoti duomenis apie pjovimo metu atsirandančias nuopjovas tam, kad jas galima būtų panaudoti arba ne (priklausomai nuo vartotojo norų) sekančių pjovimų metu;
- § nustatyti nuopjovų numeravimo parametrus;
- § nustatyti pjovimo parametrus: išmetamų atliekų, apipjaunamų plokštės pakraščių, pjūklo pločio išmatavimus, supjovimo kainą ir kt.;
- § spausdinti sąmatą ir šablonus spausdintuvu.



1.17 pav. Programos „Dalgis“ ekrano vaizdas

Programos privalumai ir galimybės:

- *Pagrindinė ypatybė, išskirianti programą "Dalgis" iš įprastinių šios rūšies programų tarpo, yra analogo neturintys optimizavimo algoritmai Šie algoritmai leidžia optimaliai supakuoti pjaunamus gabalus žaliavų plokštėse.*
- *Galima panaudoti turimas žaliavų nuopjovas.*
- *Ypač programos privalumai išryškėja tada, kai vienu metu pjaustomas didelis užsakymų kiekis.*

Trūkumai:

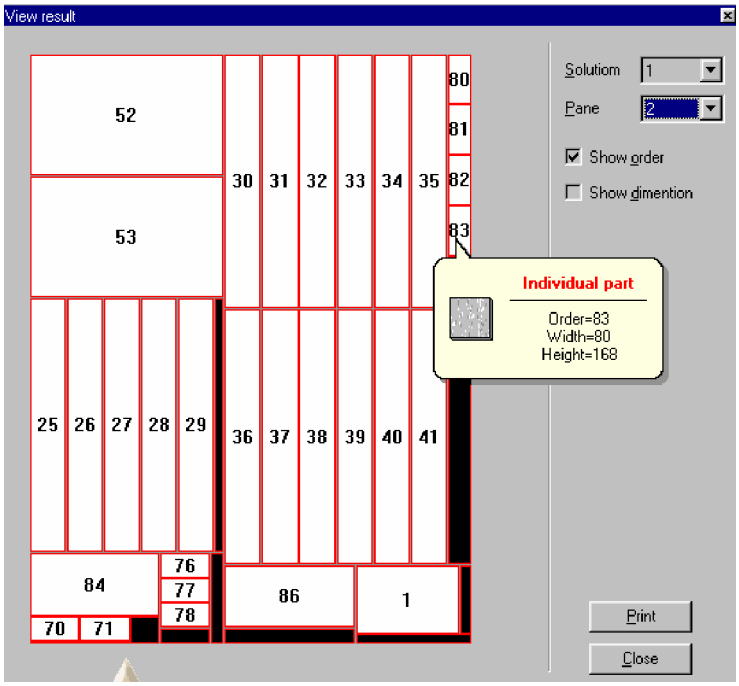
- *Sudėtinga sistema.*
- *Didelė kaina.*

Optimizuoto pjaustymo programa -„ PaneCutter“.

Programa **PaneCutter**(1.18 pav.) yra skirta įvairių plokščių (medžio, metalo, stiklo ir kt.) automatiniam išpjaustymui. Programa, naudodama rekursinį algoritmą, randa optimalų ruošinių pakavimo planą su kuo mažiausiomis atliekomis.

Pagrindiniame programos lange, vartotojas gali nustatyti tiek pagrindinės plokštės, tiek ir detalių plotį ir aukštį. Ši informacija gali būti išsaugoma ir panaudojama vėliau. Įvedus kiekvienos detalės išmatavus ir paspaudus skaičiavimo mygtuką, pradedamas

optimizavimo procesas. Jo trukmė priklauso nuo detalės numerio ir kompiuterio greičio.



1.18 pav. Programos „PaneCutter” ekrano vaizdas

Galima atsisiųsti programos bandomąją versiją iš interneto svetainės (http://www.digimpro.com/panecutter/).

Programos privalumai ir galimybės:

- *Vartotojas gali lengvai keisti detalių ir plokštės matmenis.*
- *Vartotojas gali pasirinkti jam patogius matavimo vienetus.*
- *Kiekvienas optimalus detalių pakavimo planas gali būti parodytas vaizdžiai kompiuterio ekrane ir atspausdintas.*
- *Vartotojas gali pats išsirinkti, kurį sprendinį parodyti vaizdžiai.*

Trūkumai:

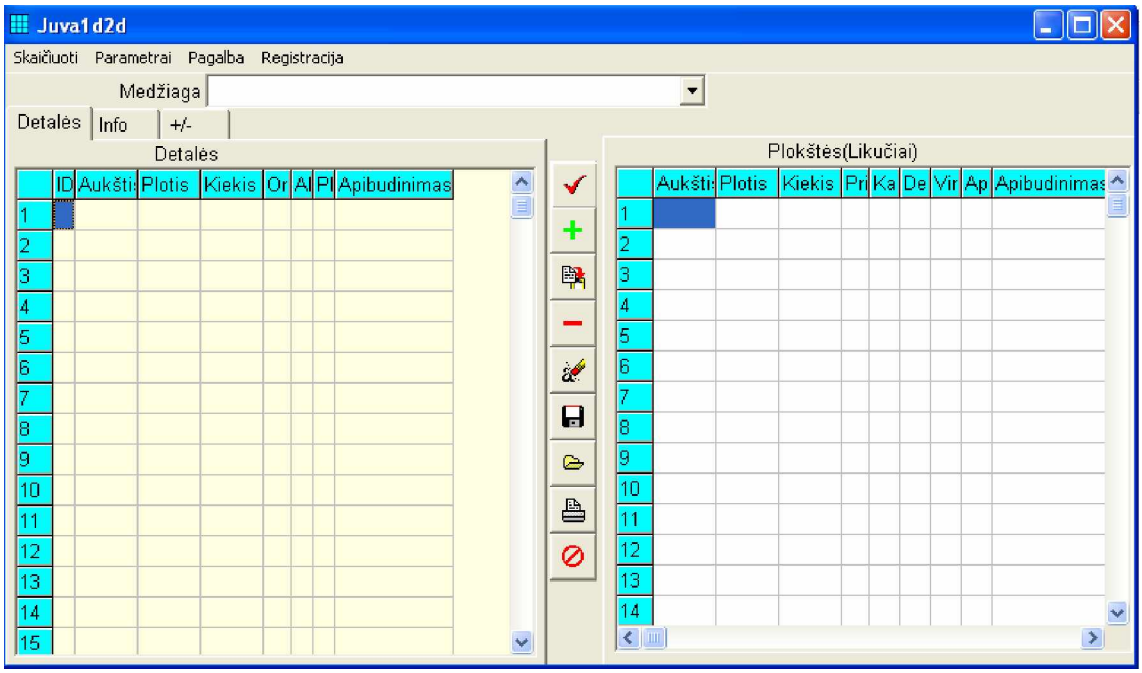
- *Mažai funkcionalumo.*

Plokščių (skersinių) pjaustymo programa "Juva1d2d"

Programa "Juva1d2d"(1.19 pav.) išdėsto stačiakampes detales (skersinius) į stačiakampes plokštes (skersinius) , minimizuojant atliekas. Taip pat skaičiuoja sandėlio medžiagų likučius , pjovimo išeigą, detalių laminavimo ilgį, kainą pagal pjovimo išeigą, kainą pagal detalių kvadratūrą. Programa atlieka tokias pagrindines funkcijas:

- Atliekų minimizavimas

- § Pjovimo išeigos , detalių kvadratūros skaičiavimas (kiekine ir pinigine išraiška)
- § Laminavimo ilgio skaičiavimas
- § Sandėlio likučių apskaita



1.19 pav. Programos "Juva1d2d" ekrano vaizdas

Programos bandomąją versiją galima atsisiųsti iš interneto svetainės (http://geocities.com/Kestas_lt/Juva/)

Programos privalumai ir galimybės:

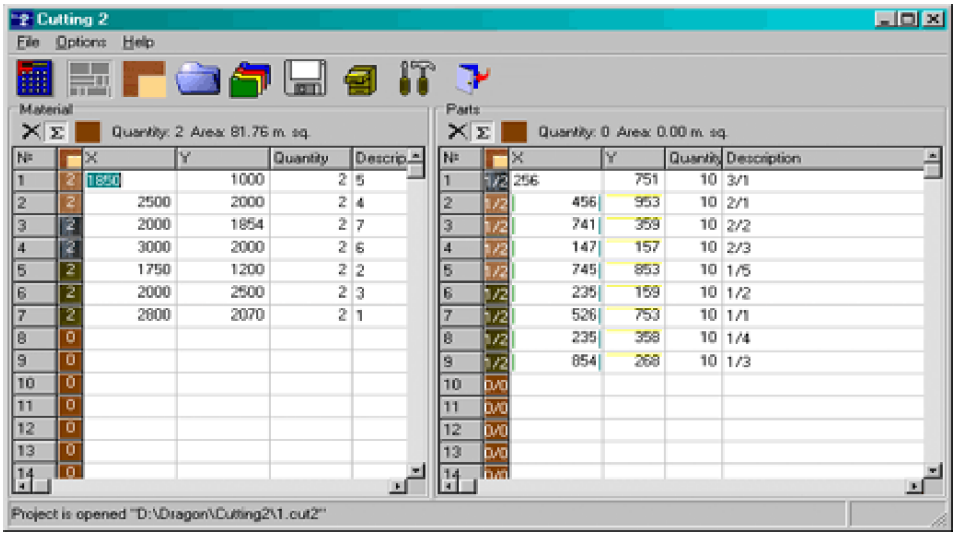
- § *Programoje galima surūšiuoti ir paruošti skaičiavimui plokštes pagal plotį*
- § *Atspausdinti plokščių likučių, ir išpjaustymo ataskaitas.*
- § *norimus duomenis papildomai išsaugoti tam , kad vėliau nereiktų iš naujo jų vesti, o užtektų tik iškviesti*
- § *Po kiekvieno paskaičiavimo programa užpildo apyvartų lentelę, kurioje yra nurašomos plokštės ir pajamuojamos atraižos*

Trūkumai:

- § *Sudėtinga sistema.*

Optimizuoto pjaustymo programa „Cutting 2“

Programa **„Cutting 2“**(1.20 pav.) yra skirta įvairių stačiakampių detalių pakavimui plokštėje. Ji gali būti naudojama medžio plokščių, stiklo ir metalo paviršių išpjaustymui. Naudoja unikalų greitaeigį algoritmą. Tai leidžia optimaliai išdėlioti ruošinius ant plokštės su kuo mažiausiomis atliekomis.



1.20 pav. Programos „Cutting 2” ekrano vaizdas

Programos bandomąją versiją galima atsisiųsti iš interneto svetainės (http://www.cuttinghome.com/).

Programos privalumai ir galimybės:

- *Pasirinkti pjaustomų detalių ir plokštės tipą;*
- *Saugoti likučius, likusius nuo pjaustymo, kad juos būtų galima dar kartą panaudoti;*
- *Rankiniu būdu pakeisti detalių išdėstymą, perkeliant detales iš vienos vietos į kitą;*
- *Peržiūrėti keletą ruošinių pakavimo būdų;*
- *Išsaugoti duomenis failuose;*
- *Išsaugoti pjaustymo rezultatus failuose;*
- *Lengvai pakeisti ar panaikinti detales iš ruošinio;*
- *Peržiūrėti ir išspausdinti rezultatus*
- *Viename lape spausdinti du skirtingus pakavimo variantus;*

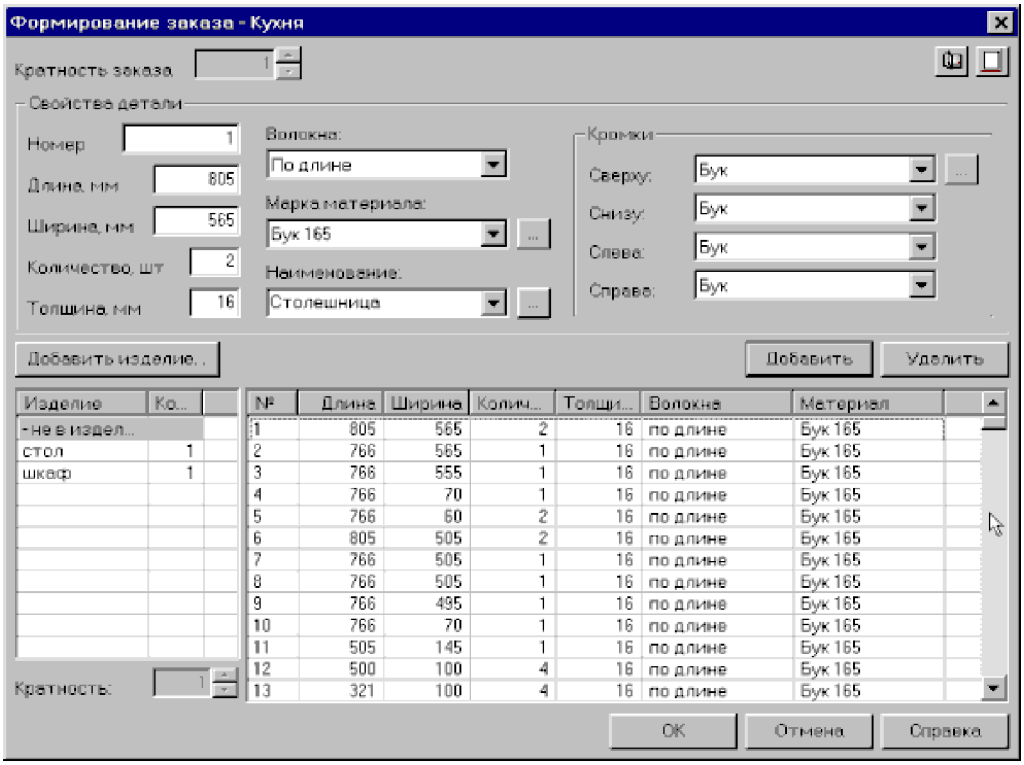
§ *Formuoti informaciją apie detales;*

Trūkumai:

§ *Paini vartotojo sąsaja*

Automatizuota plokščių pjaustymo sistema „ASTRA“

Skirta automatizuotam plokščių supjaustymui ir gali būti naudojama metalo, medžio plokščių, stiklo, plastmasės pjaustymui. Jos pagalba galima žymiai pakelti efektyvumą ir sumažinti laiką, skirtą pjaustymo ruošinių pakavimui. Programa leidžia aprašyti tipinių gaminių biblioteką, kurią galima vėliau panaudoti formuojant užsakymus.



1.21 pav. Programos „ASTRA“ ekrano vaizdas

Programos privalumai ir galimybės:

§ *Užsakymų detalėms galima nurodyti laminavimo juosteles.*

§ *Galima įvesti didelį kiekį laminavimo juostelių ir vėliau paskaičiuoti užsakymo kainą.*

§ *Automatinis pjaustymas yra atliekamas atsižvelgiant į technologinius ir organizacinius gamybos parametrus. T.y. leidžia pakeisti pjaustymo parametrus: pjūklo plotį, apipjaunamų kraštų plotį, mažiausios galimos nuopjovos ilgį, plotį, plotą, supjovimo kainą ir kitus.*

Trūkumai:

- *Sudėtinga sistema.*
- *Paini vartotojo sąsaja.*

Apibendrinant nagrinėtus programas, galima išskirti šiuos esminius jų privalumus:

- *Pasirinkti plokštės tipą.*
- *Pasirinkti pjaustomų stačiakampių išmatavimą.*
- *Peržiūrėti grafinį vaizdą, su išdėstytais stačiakampiais ekrane.*
- *Išsaugoti duomenis failuose.*
- *Peržiūrėti ir išspausdinti rezultatus.*

Išanalizavus 2D objektų pakavimo programas, apibendrinant galima išskirti šiuos sukurtų sistemų trūkumus:

- *Sudėtinga sistema.*
- *Didelė kaina.*
- *Suranda gerą, bet ne geriausią sprendinį.*
- *Paini vartotojo sąsaja.*

Analizė parodė, kad tikslinga sukurti specializuotą programinį produktą, skirtą pakuoti stačiakampius plokštėje, greitai pateikiantį sprendimą ir su nesudėtinga vartotojo sąsaja.

2. GILJOTININIO SUPJAUSTYMO UŽDAVINIO PROJEKTAVIMAS

2.1. REIKALAVIMŲ DOKUMENTAS

2.1.1. VARTOTOJAI

Potencialiausi programinio projekto vartotojai yra medienos apdirbimo įmonės, gaminančios įvairios paskirties baldus.

Programine įranga gali naudotis kiekvienas, net ir mažai darbo su kompiuteriu įgūdžių turintis firmos darbuotojas. Tačiau darbas bus efektyvesnis, jei vartotojas turės bent minimalias su Microsoft Windows operacine sistema darbo žinias ir įgūdžius.

2.1.2. PROJEKTO TIKSLAS

- Sukurti nesunkiai įsisavinamą, nebrangią programinę įrangą, leidžiančią vartotojui gauti pakankamai gerą rezultatą, pakuojant stačiakampius plokštėje.
- Sukurti mažai darbo patirties su kompiuteriu turintiems vartotojams patogią ir lengvai suprantamą sistemą.
- Sudaryti greitai ir vaizdžiai pateikiančią rezultatus programinę įrangą, kuri leidžia vartotojui pateikti programos apdorojimo duomenis Microsoft Windows ir Microsoft Office dažniausiai naudojamų standartinio paketo programų *.txt formato faile.

2.1.3. PAGRINDINĖS PRODUKTO FUNKCIJOS

- Pagal duotas stačiakampių ir plokštės dimensijas pateikti geriausią stačiakampių pakavimo plokštėje sprendimo būdą.
- Pateikti vartotojui geriausią stačiakampių pakavimo plokštėje sprendinį vizualine išraiška.

2.1.4. FUNKCINIAI VARTOTOJO REIKALAVIMAI

Produktas tiesiogiai sąveikauja su vartotoju, kadangi visų plokščių bei stačiakampių dimensijų duomenys nuskaitomi iš vartotojo apibrėžto failo arba tiesiai iš ekrano.

Vartotojas turi suvesti įėjimo reikšmes: plokštės ir stačiakampių išmatavimus. Programa išveda geriausią stačiakampių pakavimo sprendinį, paskaičiuodama jį pagal užduotas įėjimo reikšmes. Sprendinio duomenys turėtų būti išvedami kuo greičiau.

Duomenų įvedimas ir pateikiami rezultatai turi būti aiškūs ir suprantami.

Programinė įranga yra skirta jos vartotojui, todėl norint išsiaiškinti vartotojo poreikius, reikėtų iškelti keletą klausimų:

- Ko iš kuriamos programos nori vartotojas?

Vartotojas tikisi, kad programa užims mažai vietos kompiuterio atmintyje ir bus paprastai įdiegiama. Vartotojas tikisi, kad programa pateiks tikslų ir aiškų stačiakampių pakavimo plokštėje rezultatą.

- Kaip įsivaizduojamas programinės įrangos veikimo procesas?
  - § Sukuriamas duomenų įvesties failas.
  - § Vykdoma programa, kuri sugeneruoja rezultatus.
  - § Parodomas stačiakampių supakuotų plokštėje grafinis vaizdas.
- Kokie yra vartotojo keliami uždaviniai 2D objektų pakavimo sistemai?
  - § Leisti pasirinkti bet kokius plokštės matmenis.
  - § Leisti pasirinkti skirtingus stačiakampių matmenis.
  - § Leisti patiems nepriklausomai nuo programinės įrangos sukurti įvesties failus.
  - § Leisti koreguoti faile netinkamai įvestus stačiakampių matmenis.
  - § Leisti perskaičiuoti gautą rezultatą, įvesties faile pakeitus plokštės matmenis.
  - § Greitai gauti rezultatus.

2.1.5. NEFUNKCINIAI VARTOTOJO REIKALAVIMAI.

Reikalavimai patikimumui ir kokybei:

- Sukurtas modelis, realizuotas Java programine kalba, turi realiai funkcionuoti ir išvesti teisingus rezultatus.
- Programa turi veikti taip, kaip vartotojas tikisi ir turi atitikti aprašymo reikalavimus.
- Turi būti suteikta galimybė pasirinkti galimas įėjimo reikšmes.
- Rezultatai privalo būti gaunami prie bet kurių pasirinktų reikšmių.
- Turi būti vengiama kritinių situacijų.
- Turi būti patogi ir neerzinanti vartotojo sąsaja.
- Dokumentacija turi būti išsami, ir padėti naudotis vartotojui neturinčiam įgūdžių šioje srityje.

Programinės bei aparatūrinės įrangos reikalavimai:

- Aparatūrinė įranga:

IBM PC.

- Programinė įranga:
  - § Programos greitis priklausys nuo turimos techninės įrangos.
  - § Planuojamas programos dydis: apie 1 Mb.
  - § OS: Microsoft Windows bet kuri terpė (viena iš labiausiai paplitusių operacinių sistemų vietinėse rinkose.)
  - § Programinei įrangai kurti turi būti naudojamos objektinio projektavimo ir programavimo technologijos.

2.1.6. SISTEMOS ARCHITEKTŪRA

Projektuojant siekiama sukurti kiek galima lankstesnę sistemą, kurią būtų galima reikalui esant nesunkiai išplėsti papildant naujomis funkcionalumo ar vizualizacijos galimybėmis. Siekiant anksčiau paminėtų tikslų didelis dėmesys bus skiriamas tam, kad kuriama sistema būtų patikima, paprasta naudotis vartotojams.

Programinės įrangos sukūrimui ir testavimui bus vartojami standartiniai IBM šeimos kompiuteriai. Turi būti naudojamos objektinio programavimo technologijos, kurios bus realizuotos Java programavimo kalba.

2.1.7. REIKALAVIMAI VARTOTOJO SĄSAJAI

Vartotojo grafinė sąsaja padeda verifikavimui, galimų klaidų tikrinimui ir prezentaciniam geriausio sprendinio pateikimui. Sugeneravus rezultatą vartotojo grafinėje sąsajoje turėtų būti parodomas aiškus supakuotų stačiakampių vaizdas.

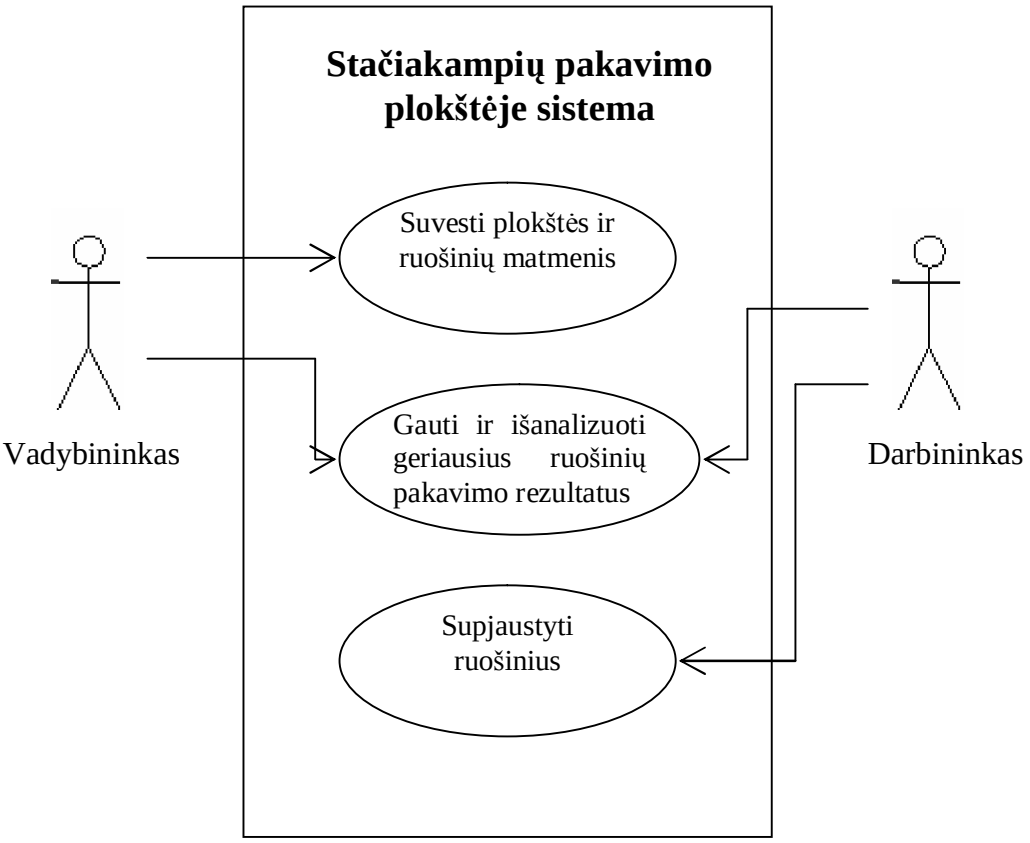
2.1.8. REIKALAVIMAI PROGRAMINĖS ĮRANGOS PATIKIMUMUI, SAUGUMUI, MOBILUMUI

- Programa turi teisingai veikti su visomis įmanomomis įėjimo reikšmėmis.
- Turi būti išvengta kritinių situacijų.
- Duomenų įvedimo ir išvedimo procedūros turi būti aiškios ir paprastai naudojamos.
- Pateikti pjaustymo rezultatai turi būti tikslūs.
- Programa turėtų veikti su naujausiomis Windows versijomis: Windows 2000, Windows ME, Windows XP.

2.2. 2D STAČIAKAMPIŲ PJAUSTYMO PLOKŠTĖJE SISTEMOS MODELIAI

2.2.1. SISTEMOS PANAUDOJIMO ATVEJŲ MODELIS

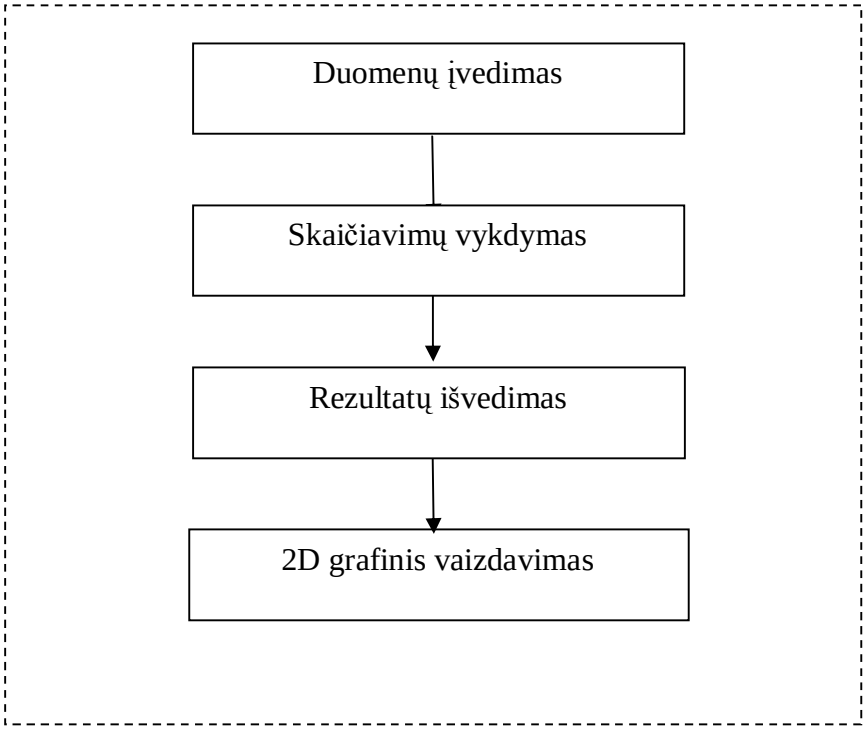
Sistemos panaudojimo atvejų modelyje (2.1 pav.) veiklos dalyvis „vadybininkas“ dalyvauja tokiuose veiksmuose: dėstomų stačiakampių ir plokščių matmenų suvedimas, geriausio pakavimo rezultato gavimas ir analizavimas. „Darbuotojas“ taip pat analizuoja programos gautus geriausio pakavimo rezultatus ir pagal juos atlieka pjaustymą.



2.1 pav. Stačiakampių pakavimo plokštėje sistemos panaudojimo atvejų modelis.

2.2.2. SISTEMOS STRUKTŪROS MODELIS

Abstrakčią kuriamos sistemos struktūrą apibrėšime sistemos struktūros modeliu (2.2 pav.).



2.2 pav. Sistemos struktūros modelis.

- Duomenų įvedimas – tai vartotojo apibrėžto failo sukūrimas, kuriame suvedami plokštės ir stačiakampių matmenys.
- Skaičiavimų vykdymas – tai vartotojo įvestų duomenų generavimas ir geriausio sprendinio suradimas.
- Rezultatų išvedimas – tai skaičiavimo rezultatų pateikimas ataskaitoje.
- Grafinis vaizdavimas – tai skaičiavimo rezultatų pateikimas grafiniame pavidale.

2.3. PROGRAMINĖS ĮRANGOS PROJEKTAVIMAS

2.3.1. PROGRAMINĖS ĮRANGOS BENDRA CHARAKTERISTIKA

Stačiakampių ruošinių supjaustymo plokštėje problemoms spręsti sukurtas modelis, kuris realizuotas Java programine kalba. Atlikus suderinimą, verifikavimą bei ratifikavimą, buvo sukurtas programinis paketas, kuris atlieka tam tikro skaičiaus

stačiakampių supakavimą plokštėje. Sukurtas modelis duotoje plokštėje supakuoja tiek stačiakampių kiek yra duota(suvesta) arba kiek telpa plokštėje. Ši programa per trumpą laiką suranda geriausią ruošinių supakavimo plokštėje sprendinį.

2.3.2. GILJOTINIO SUPJAUSTYMO UŽDAVINIO SPRENDIMO ALGORITMAS

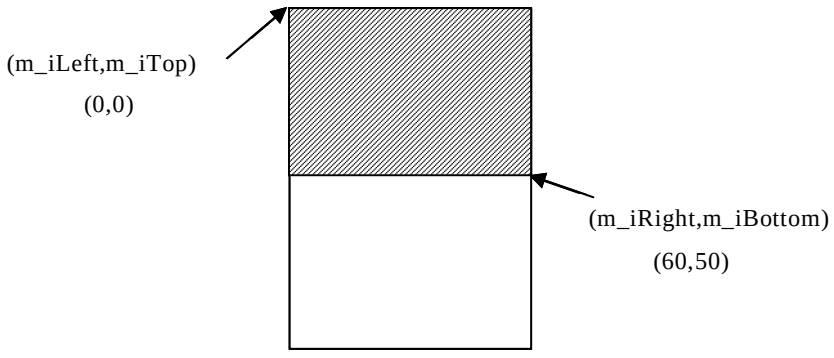
Programinis produktas buvo kuriamas pagal anksčiau aprašytą metodą. Programos algoritmą galima aprašyti taip:

Įvedami duomenys:

m_iBottom	m_iRight
50	60
30	50
10	50
10	60
10	20

Plokštės išmatavimai 60x100.

Pasibaigus duomenų įvesties procesui, suformuojamas sąrašas **m_pPaterns**, kuriame talpinami stačiakampių duomenys. Stačiakampiai, kurių kraštinės yra didesnės nei plokštės kraštinės į sąrašą neįtraukiami. Darbo pradžioje stačiakampių duomenys funkcijos *Sort()* pagalba yra surūšiuojami ir funkcijos *findPattern()* pagalba išrenkamas stačiakampis, turintis didžiausią kraštinę. Šis stačiakampis yra talpinamas plokštės viršutiniame kairiajame kampe ir pažymimas, kaip panaudotas:

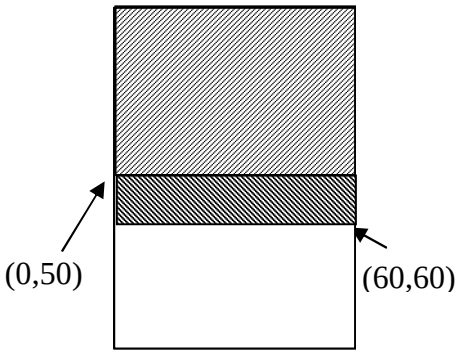


Pjūvio padėtis(horizontali ar vertikali) yra apskaičiuojama pagal formulę:

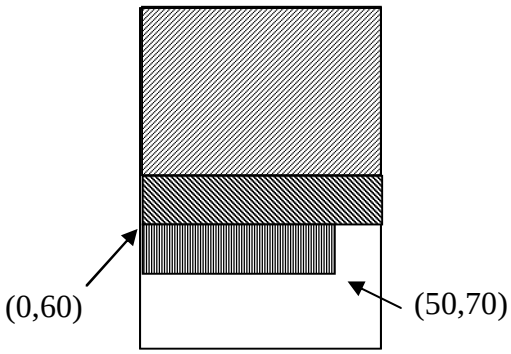
- jei iš koordinatės *m_iRight* atėmus *m_iLeft* rezultatas bus didesnis arba lygus *m_iWidth* ir iš *m_iBottom* atėmus *m_iTop* rezultatas bus didesnis arba lygus koordinatei *m_iHeight***,** tuomet priskiriamas elementas, kurio koordinatės (*m_iLeft,m_iTop,m_iWidth,m_iHeight*);

§ jei iš koordinatės *m_iRight* atėmus *m_iLeft* rezultatas bus didesnis arba lygus *m_iHeight***,** ir iš *m_iBottom* atėmus *m_iTop* rezultatas bus didesnis arba lygus koordinatei *m_iWidth* tuomet priskiriamas elementas, kurio koordinatės (*m_iLeft,m_iTop,m_iHeight,m_iWidth*), t.y. plotis sukeičiamas vietomis su aukščiu;

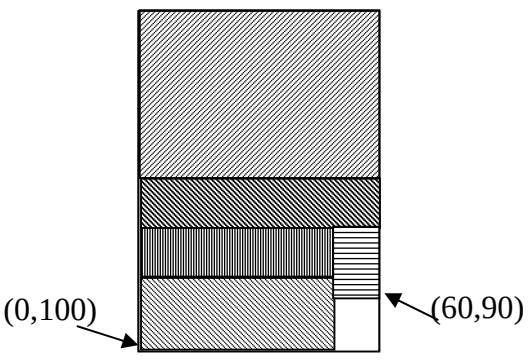
Ieškomas kitas stačiakampis, turintis didžiausią kraštinę, kuris dar yra nepanaudotas. Surastas stačiakampis talpinamas prie anksčiau rasto taip, kad jie abu sudarytų kuo mažiausią galimą plotą, t.y. sukamas 90^o kampu ir tikrinamas abiejų stačiakampių užimamas plotas. Iš visų variantų išrinkus stačiakampių pjūvį su užimamu mažiausiu plotu, jis pažymimas kaip meta stačiakampis ir traktuojamas kaip meta stačiakampis:



Vėl ieškomas kitas stačiakampis ir pakuojamas prie šio meta stačiakampio pagal tą pačią anksčiau aprašytą taisyklę:



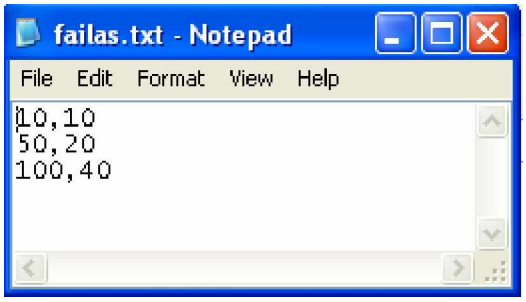
Tai vykdoma tol, kol susidaro giljotininis pjūvis. Giljotininis pjūvis susidaro tuomet, kai horizontaliame ar vertikaliame išdėstyme nebeįmanoma supakuoti nė vieno nepanaudoto ruošinio taip, kad sumažėtų atliekų plotas:



Kai susidaro tokia situacija, tai laikoma, kad kitas pjūvis bus naujas ir viskas kartojama iš naujo.

2.3.3. DUOMENŲ STRUKTŪRA

Pradiniai duomenys ir jų įvedimas. Visų stačiakampių dimensijų duomenys nuskaitomi iš vartotojo apibrėžto failo. Vartotojo apibrėžtas failas turi būti pateikiamas Microsoft Windows ir Microsoft Office standartinio paketo programų *.txt formato faile. Sukurtas failas privalo būti tame pačiame disko aplanke, kur ir yra vykdoma programa. Įvesti duomenis į failą vartotojas privalo apibrėžtu formatu:

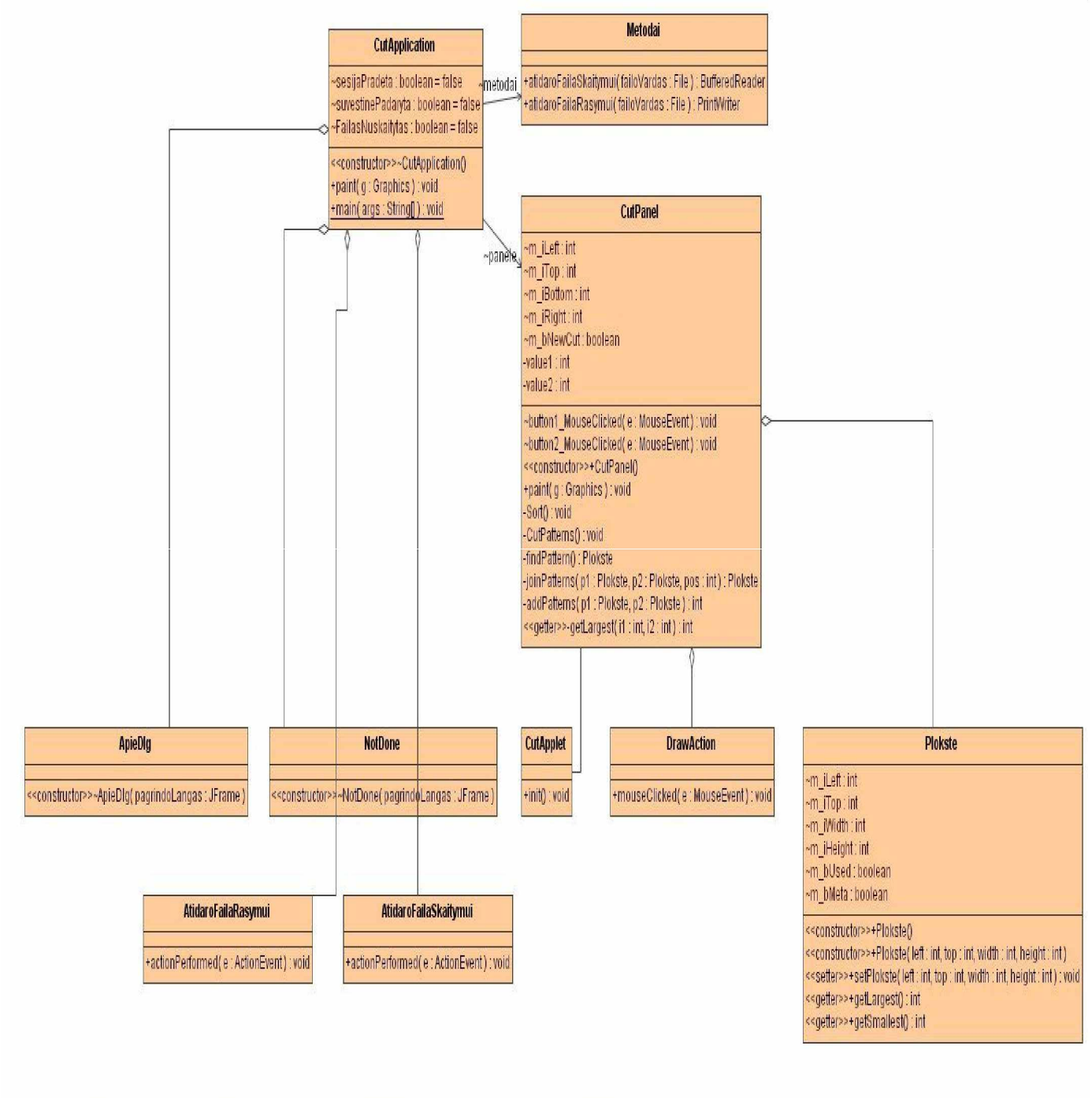


2.3 pav. Įvedamų duomenų failo pavyzdys

Šiame faile eilutėmis surašomi stačiakampių, kurie bus dėstomi plokštėje išmatavimai, t.y. plotis ir aukštis, tarp išmatavimų dedami kableliai.

Duomenų struktūra. Sukurtame projekte yra svarbu, kad duomenys būtų gaunami kuo greičiau. Tam tikslui yra naudojamas sąrašas **m_pPaterns,** kuriame yra talpinami stačiakampių duomenys.

2.3.4. PROJEKTUOJAMOS PROGRAMOS DETALI KLASIŲ DIAGRAMA



2.4 pav. Detali klasių diagrama

Aprašymas

CutApplication- pagrindinė klasė.

Plokste- plokščių duomenų klasė.

CutPanel- abstraktus programos interfeisas, realizuotas Java JPanel komponente.

DrawAction- pagalbine klasė, realizuojanti plokščių pjaustymą panelėje.

CutApplet- aplretas.

AtidaroFailaRasymui- pagalbinė klasė, skirta įvestiems duomenis įrašyti.

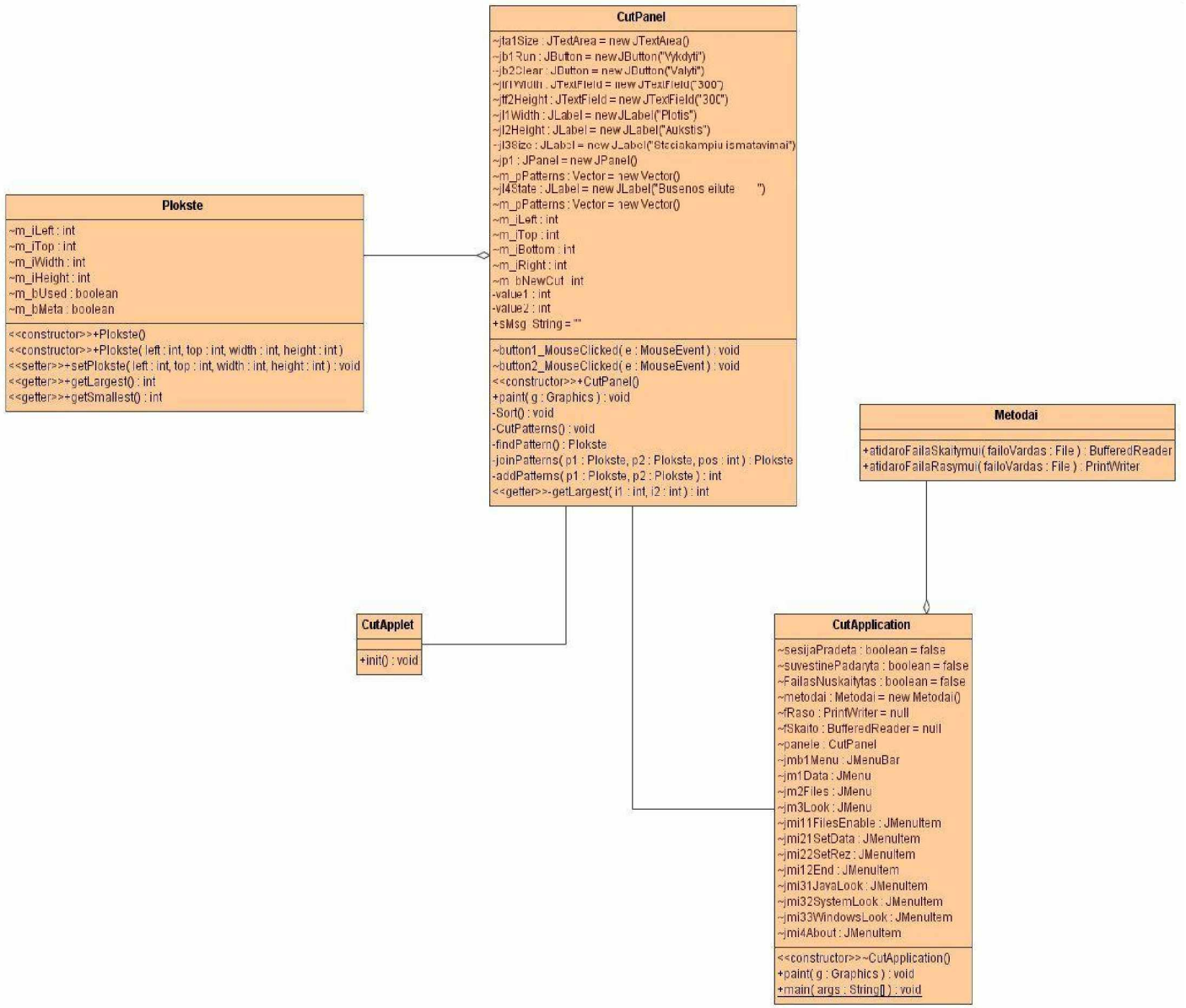
AtidaroFailaSkaitymui- pagalbinė klasė, skirta atidaryti duomenų failą.

ApieDlg –pagalbinė klasė, informaciniam dialogui išvesti.

NotDone –pagalbinė klasė, informaciniam dialogui išvesti.

Metodai- klasė, atliekanti plokščių formavimo veiksmus

2.3.5. PROJEKTUOJAMOS PROGRAMOS KLASIŲ DIAGRAMA



2.5 pav. Klasių diagrama

Aprašymas

CutPanel- abstraktus programos interfeisas realizuotas Java JPanel komponente. Tiesiogiai komponentas negali būti paleistas, kaip programos langas, tačiau jis panaudojamas CutApplication Java programos lange ir CutApplet Java apleto lange. CutPanel klasė sudaryta is Java komponentų: mygtukų-button (jb1Run, jb2Clear), žymių - labels (jl1Width, Jl2Height, jl1Size, jl4State), tekstinių laukų- (jt1Width, jtf2Height) . Šioje klasėje suformuojamas

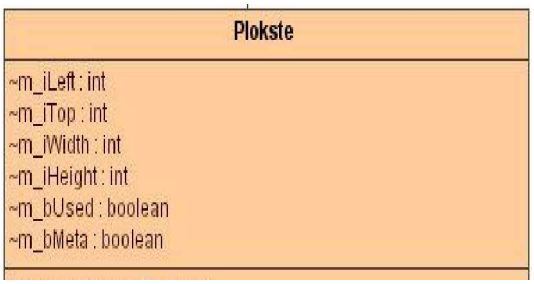
pagrindinis lango interfeisas, matomas tiek iš apleto, tiek iš programos lango. Taip pat čia yra metodai atliekantys plokščių formavimo veiksmus.

CutApplet klasė ypatingų komponentu neturi, ji tik paveldi JApplet Java klasę ir panaudoja CutPanel komponentą.

CutApplication klasėje realizuotas interfeisas nebudingas apletui: meniu ir meniu punktuose realizuoti veiksmai. Kadangi apletai meniu neturi, tai papildomas funkcionalumas realizuotas programos lange. Iš meniu numatyta galimybė skaityti/rašyti duomenis i failus, todėl čia sukurtos papildomos klasės failų rašymo ir skaitymo veiksmams (atidaroFailaSkaitymui, atidaroFailaRasymui). Taip pat informacinių dialogų langai (ApieDlg, NotDone).

Plokste- yra plokščių duomenų klasė, ji naudojama klasėje CutPanel.

Duomenų struktūrą galima atvaizduoti duomenų struktūros klasės diagrama:



2.6 pav. Duomenų struktūra

m_iLeft- stačiakampio kairė koordinatė.

m_iTop- stačiakampio viršaus koordinatė.

m_iWidth- stačiakampio plotis.

m_iHeight- stačiakampio aukštis.

m_bUsed- pažymėtas, kaip panaudotas stačiakampis.

m_bMeta- pažymėtas, kaip meta stačiakampis.

2.4. RIZIKOS ĮVERTINIMO IR MAŽINIMO PLANAS

Projekto rizikos:

Ar galimi reikalavimų pasikeitimai?

Kuriant programą bus laikomasi reikalavimų dokumento nurodymų. Reikalavimai gali keistis tik tuo atveju, jeigu programos kūrimo proceso metu bus pasiūlyta patobulinti keletą programos detalių arba vartotojui atsirastų poreikis įtraukti į programą tam tikrus papildomus reikalavimus.

Su vartotoju susijusios rizikos:

Ar gali vartotojas atsisakyti produkto?

Vartotojas gali atsisakyti produkto. Bet projektas yra kuriamas ir moksliniais tikslais.

Spendimas: Kuriant programą siekiama sukurti kiek galima lankstesnę sistemą, kurią būtų galima reikalui esant nesunkiai išplėsti papildant naujomis funkcionalumo ar vizualizacijos galimybėmis. Todėl sukurtą programos universalų paketą, galima lengvai pritaikyti pagal kiekvieno vartotojo poreikius.

Proceso rizika:

Ar gali atsirasti kiti, papildomi, darbai?

Gali atsirasti papildomi darbai, kurie apsunkintų savalaikį projekto užbaigimą.

Sprendimas: Paaiškinti vartotojui apie galimą projekto uždelsimą, uždelsimo priežastis ir galimas pasekmes. Derėtis su vartotoju, pabrėžiant, kad projekto kūrimo uždelsimas gali duoti tik teigiamus rezultatus, nes per ilgesnį laiką bus sukurtas efektyvesnis ir tobulesnis produktas.

Priešingu atveju, vartotojui nesutikus dėl projekto laiko uždelsimo, dirbant teikti prioritetus svarbiausioms projekto dalims, jog vartotojui atrodytų, kad praktiškai visas funkcionalumas egzistuoja ir projektas pristatytas laiku. Pilnas produkto funkcionalumas įdiegiamas vėliau.

Techninės rizikos:

Ar gali sugesti kompiuteriai?

Kompiuteriai gali sugesti. Nors šių dienų kompiuterinė technika ganėtinai patikima, tačiau tokia tikimybė išlieka. Sugedus kietajam diskui galima prarasti visą informaciją. Tai įtakotų projekto kūrimo eigą.

Sprendimas: Kuriamos programos projekto kopija periodiškai yra įrašoma į kompaktinį diską CD-R, taip pat į kitų kompiuterių kietuosius diskus. Tokiu atveju prarastos informacijos kiekis būtų nedidelis.

Darbo priemonių rizikos:

Ar yra tikimybė, kad kokia nors projektui reikalinga priemone nebus galima pasinaudoti?

Tikimybė yra labai maža, kadangi šiuo metu visos priemonės yra turimos, o atsiradus poreikiui naujoms priemonėms, jas būtų nesunku gauti universitete.

Darbuotojų komandos dydžio ir patirties rizikos:

Ar komanda pakankamo dydžio?

Komanda pradinei projekto daliai atlikti yra pakankamo dydžio. Nors ją realiai sudaro vienas narys, tačiau jo aplinkoje yra nemažai kolegų, kurie gali suteikti profesionalią pagalbą

Ar komandos nariai pakankamai kompetentingi?

Komandos narys yra pakankamai kompetentingas projekto vykdymui. Jis turi pakankamai teorinių ir praktinių žinių 2Dobjektų pakavimo problemoms spręsti.

2.1 lentelė

Rizikos vertinimo lentelė

Rizika	Tikimybė (%)	Įtaka
Kompiuterių gedimas	20	3
Pavėluotas pristatymas	30	1
Reikalavimų pasikeitimai	5	3
Kompetencijos trūkumas	10	3
Kietojo disko gedimas	20	2
Vartotojo atmetimas	30	3
Papildomi darbai	50	1

Įtakos vertinimo skalė:

1 – Tragiška

2 – Labai blogai

3 – Pakenčiama

4 – Neturi įtakos

3. GILJOTININIO SUPJAUSTYMO METODO REALIZACIJA

3.1. PROGRAMINĖS ĮRANGOS PASKIRTIS IR GALIMYBĖS

Plokščių pjaustymo programinė įranga skirta stačiakampių supjaustymo plokštėje uždaviniui spręsti. Programinė įranga pagal duotus plokštės ir stačiakampių išmatavimus suranda geriausią stačiakampių pakavimo plokštėje sprendimo būdą. Programa skirta naudotis įmonėms, kurioms yra aktualu, kuo optimaliau pakuoti ruošinius tam tikroje plokštėje, kad liktų kuo mažiau ir mažesnių atliekų. Kitaip tariant, kuo daugiau ir efektyviau išnaudoti plokštės plotą.

Programinės įrangos paketas užima nedaug vietos kompiuterio atmintyje ir yra paprastai įdiegiamas. Vartotojas pats paruošia programai įvedimo duomenis, kuriuos programa apdoroja ir išveda stačiakampių supakuotų plokštėje rezultatą grafinį vaizdą. Naudojant šią programą galima efektyviau išnaudoti plokštės erdvę, taupyti laiko ir pinigų sąnaudas, gauti lūkesčius tenkinančius darbo rezultatus.

Programinės įrangos galimybės:

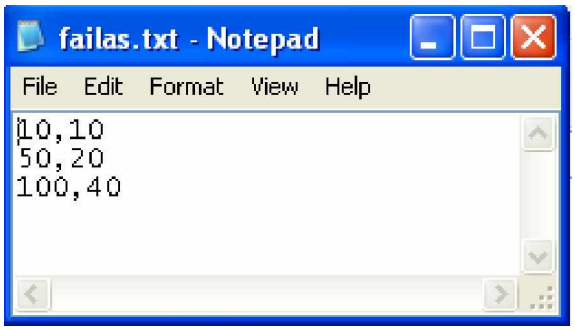
- Programa parodo, kaip reikėtų efektyviai supakuoti ruošinius plokštėje.
- Programa per trumpą laiko tarpą suranda geriausią stačiakampių pakavimo plokštėje sprendimą.
- Darbo rezultatai pateikiami stačiakampių supakuotų plokštėje grafiniu vaizdu.

3.2. NAUDOJIMOSI PROGRAMA INSTRUKCIJA

Vartotojas pradėdamas naudotis 2D objektų pakavimo programa pirmiausia turi paruošti įvedimo duomenis programai.

- Informaciją vartotojas turi suvesti į failą pavadinimu ,,duomenys.txt“. Šį failą galima sukurti naudojantis Microsoft Windows ir Microsoft Office standartinio paketo programomis ir daugelį kitų sukuriančių *.txt formato failą. Galima naudoti programas: Notepad, WordPad, Microsoft Word ir kt.

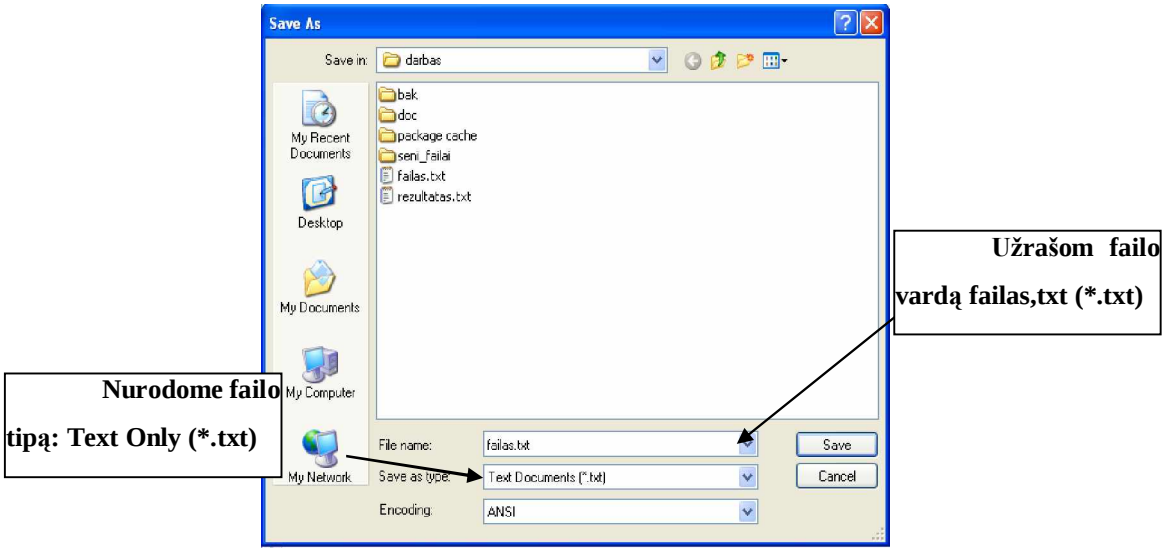
- Įvesti duomenis į failą vartotojas privalo apibrėžtu formatu (3.1 pav.):



3.1 pav. Įvedamų duomenų failo pavyzdys

Šiame faile eilutėmis surašomi stačiakampių, kurie bus pakuojami plokštėje išmatavimai, t.y. plotis ir aukštis, tarp išmatavimų dedami kableliai.

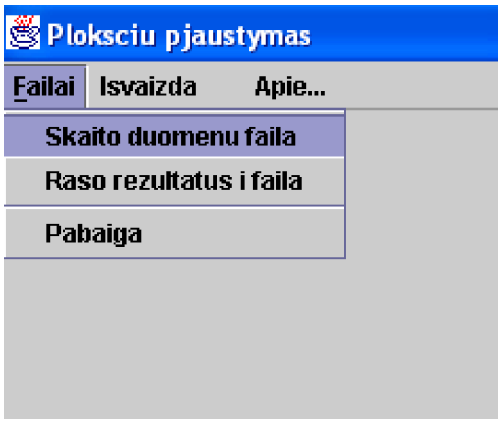
- Suvedus duomenis į failą atliekami sekantys veiksmai: meniu pasirinkimo grafoje spaudžiama ,,File, Save As“ ir atsidariusiame lange ,,Save As“ (pav.) užrašome failo vardą ir nurodome failo tipą:



3.2 pav. Įvesties failo išsaugojimo procedūra.

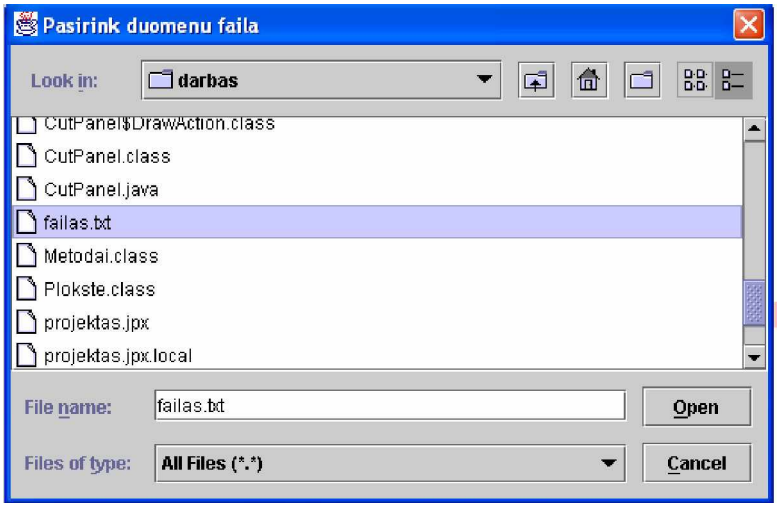
- Sukurtą failą reikia įkelti į tą patį disko aplanką, kuriame yra vykdoma programa.

- Iškviesti duomenų failą galima įvykdžius komandą **„Failai à Skaito duomenų faila“**:



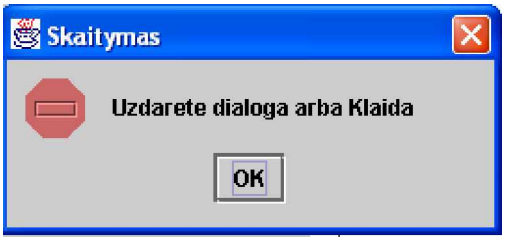
3.3 pav. Duomenų failo iškvietimas

Atsivėrusiame dialogo lange reikia pasirinkti duomenų failą **„failas.txt“**:



3.4 pav. Duomenų failo pasirinkimas

Nepasirinkus duomenų failo bus išvestas sisteminis pranešimas:



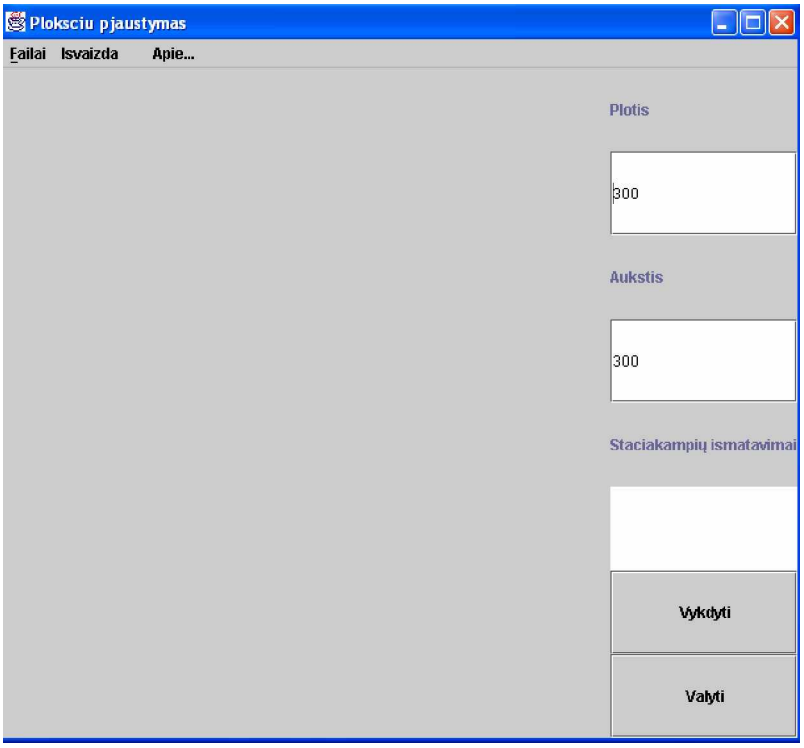
3.5 pav. Sisteminis pranešimas apie klaidą

- suvesti duomenis galima ir tiesiogiai į laukus „Stačiakampių išmatavimai“, čia duomenys yra vedami analogiškai, t.y. eilutėmis

surašomi stačiakampių, kurie bus pakuojami plokštėje išmatavimai, t.y. plotis ir aukštis, tarp išmatavimų dedami kableliai;

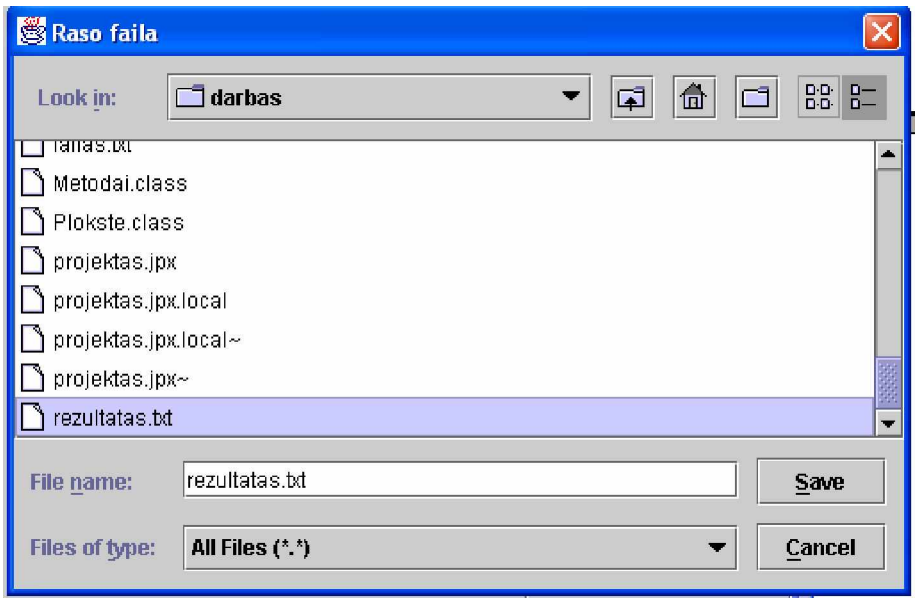
- jei duomenis apie pjaunamus stačiakampius bus neįvesti, bus išvestas pranešimas *"Nera duomenu. Iveskite staciakampių ismatavimus";*
- Į lauką „**Plotis**“- vedamas plokštės, kurioje bus atliekamas supakavimas, plotis(pagal nutylėjimą reikšmė bus 300);

- Į lauką „**Aukstis**“- vedamas plokštės, kurioje bus atliekamas supakavimas, aukštis(pagal nutylėjimą reikšmė bus 300):



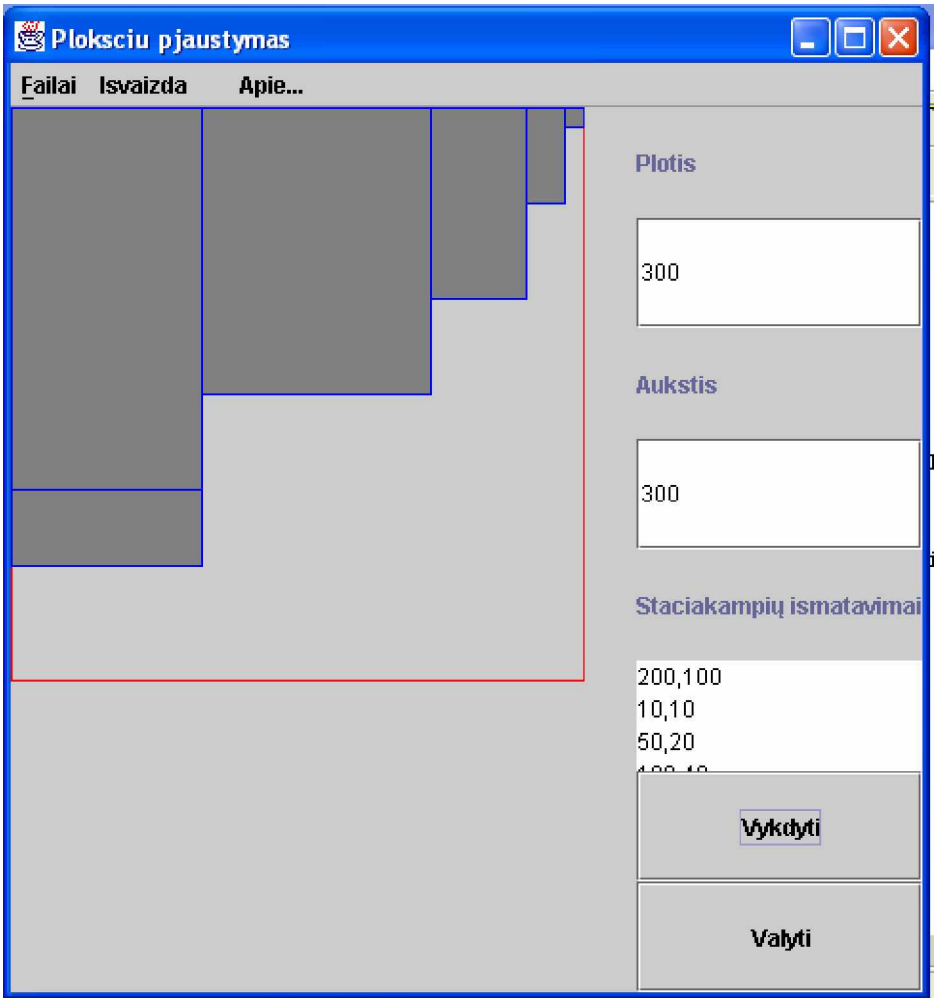
3.6 pav. Programos darbo langas

- mygtukas „**Vykdyti**“ yra skirtas pradėti skaičiuoti programos rezultatus.
- mygtukas „**Valyti**“ yra skirtas išvalyti darbo lauką.
- Įvestus duomenis galima užsaugoti išsaugoti, įvykdžius komandą **„Failai** à **Raso rezultatus i faila“,** failo vardas **rezultatas.txt:**



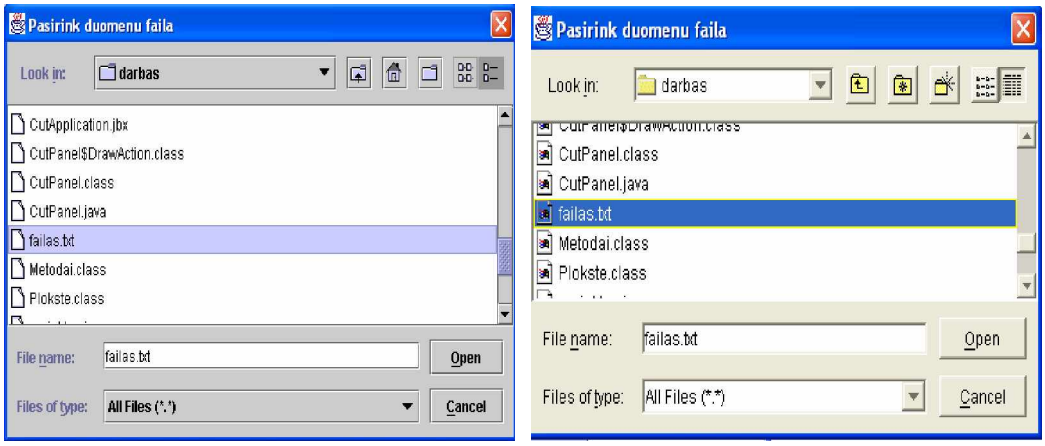
3.7 pav. Duomenų išsaugojimas

- Stačiakampių supakuotų plokštėje grafinis vaizdas pateikiamas iškart programai suskaičiavus rezultatus:

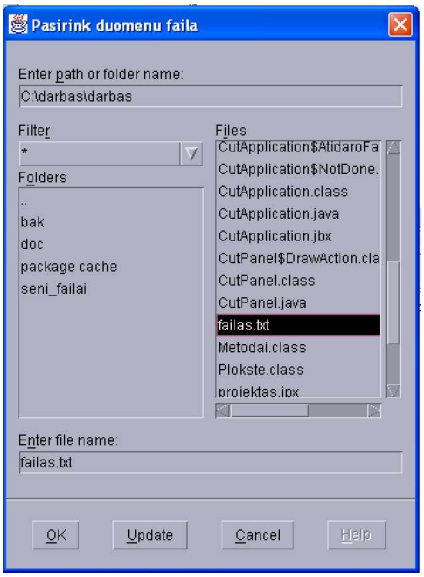


3.8 pav. Stačiakampių, supjaustytų plokštėje, grafinis vaizdas

Naudodamasis meniu punktu **Išvaizda,** vartotojas gali pasirinkti jam priimtiną sisteminių dialogo langų išvaizdą.(Java, Sistemos, Windows).



3.9 pav. „Java“ ir „Sistemos“ išvaizdos dialogo langai



3.10 pav. „Windows“ išvaizdos dialogo langas

3.3. PROGRAMINĖS ĮRANGOS ĮDIEGIMO INSTRUKCIJA

Programinė įranga įdiegiama sekančiais etapais:

- Programinė įranga įrašoma į kompiuterį. Programinį paketą sudaro „**progr.jar**“ – programos archyvas, paleidimo failas- „**start.bat**“, „**build.bat**“- sukuria paleidimo failą „**start.bat**“.
- Programa veiks, jei kompiuteryje bus suinstaliuota Java versija. Jei Java paketas bus instaliuotas kitoje disko vietoje, faile „**build.bat**“ reikia pakeisti komandinę eilutę: „*C:\JBuilder7\jdk1.3.1\bin\jar*“.

4. PRODUKTO KOKYBĖS ĮVERTINIMAS

4.1 lentelė

Sukurtos programinės įrangos kokybė vertinama pagal tokius kriterijus

Kriterijai	Vertinimas prioritetais
Naudojamumas	3
Saugumas	3
Efektyvumas	3
Teisingumas	4
Patikimumas	4
Palaikomumas	4
Testuojamumas	2
Lankstumas	3
Suprantamumas	4
Pakartotinis panaudojamumas	4
Pernešamumas	2
Įsiliejamumas	3

Kokybė vertinama skalėje nuo 1 (žemiausia kategorija) – 5 (aukščiausia kategorija).

5. PROGRAMINĖS ĮRANGOS TESTAVIMAS

Sukurta programinė įranga buvo testuojama pagal šias testavimo metodikas:

- Testavimui naudosime struktūrinį testavimą, dar vadinamą ,,baltos dėžės“ testavimu, kadangi žinoma programos struktūra ir jos veikimas.
- Pavienių objektų testavimą.
- Integruotos sistemos testavimą.

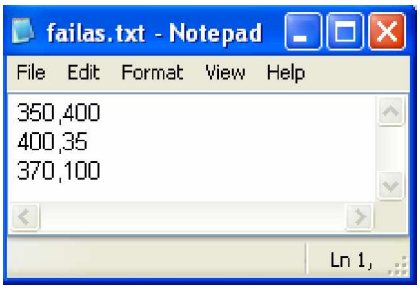
Vartotojo sąsajos testavimas verifikuoja vartotojo sąveiką su programine įranga. Vartotojo sąsajos testavimas reikalingas tam, kad užtikrinti, jog vartotojo sąsaja suteikia vartotojui galimybę pasiekti testuojamo objekto funkcijas. Taip pat šis testavimas užtikrina, jog objektai vartotojo sąsajos funkcijoje veikia kaip tikimasi ir atitinka bendrus ar sukurtus standartus.

Vartotojo funkcionalumo strategija identifikuoja klaidas, susijusias su teisingu programos funkcijų atlikimu, t.y. ar programa atlieka funkcijas teisingai, pagal reikalavimus. Šio testo tikslas yra verifikuoti tinkamą duomenų priėmimą, apdorojimą ir išvedimą bei tinkamą verslo taisyklių įdiegimą. Šio tipo testavimas yra paremtas juodos dėžės principu.

Testavimas buvo vykdomas sekančiais etapais:

- Suprojektuoti testavimo atvejai.
- Paruošti testavimo duomenys.
- Paleidžiama programa su testavimo duomenimis.
- Palyginami rezultatai su testavimo atvejais.

Sukurta programinė įranga buvo testuojama paruošiant klaidingus įvedimo duomenis. Įvesties faile buvo suvesti stačiakampių išmatavimai, didesni nei plokštės išmatavimai. Tokiu atveju, programa stačiakampių nepakavo. Failas su klaidingais duomenimis, kai plokštės išmatavimai –plotis 300, aukštis 300:



5.1 pav. Failas su klaidingais duomenimis

Testuojant programą su atsitiktinai parinktais duomenis, klaidų nepastebėta. Lyginant rezultatus su testavimo atveju, programoje klaidų neaptikta.

IŠVADOS

1. Išanalizuoti siūlomi supjaustymo uždavinio sprendimo algoritmai.
2. Atlikta supjaustymo uždavinio programinių paketų analizė.
3. Pagal atliktą analizę, nustatyti dviejų dimensijų pakavimo metodų privalumai ir trūkumai.
4. Atsižvelgiant į analizės rezultatus, sukurtas stačiakampių pakavimo plokštėje, naudojant giljotininį pjaustymą, algoritmas.
5. Sukurta programinė įranga dviejų dimensijų stačiakampių pakavimo plokštėje uždaviniams spręsti.

LITERATŪRA

1. Pane Cutter, Inc Cutting optimization software [interaktyvus]. [žiūrėta 2004-10-14]. Prieiga per Internetą: <http://www.digimpro.com/panecutter/>.
2. Cutting 2, Inc Home of Cutting [interaktyvus]. [žiūrėta 2004-10-16]. Prieiga per Internetą: http://www.cuttinghome.com/.
3. Plokščių-skersinių pjaustymo optimizavimo programa Juva1d2d, [interaktyvus]. [žiūrėta 2004-09-27]. Prieiga per Internetą: <http://geocities.com/Kestas_lt/Juva/>.
4. Optimizuoto pjaustymo programa -„ DALGIS“, Programos aprašymas ir vartotojo vadovas. KTU – 2000.
5. Астра Раскрой, Автоматизированные Технологические системы, [interaktyvus]. [žiūrėta 2004-10-17]. Prieiga per Internetą: <http://www.techno-sys.com/rus/html/astra.htm>.
6. Genetic algorithms for optimal cutting, [interaktyvus]. [žiūrėta 2004-02-05]. Prieiga per Internetą:< http://www.fit.ac. ip/~ono/abst-e.html>.
7. Ronnqvist M., Astrand E. Integrated defect detection and optimization for cross cutting of wooden boards, European Journal of Operational Research, 1998, Nr. 108, p. 490 – 508.
8. A.M.C.Almeida,. Optimal cutting directions and rectangle orientation algorithm - European Journal of Operational Research. - 1998, Nr. 109, p.660 – 671.
9. Miro Gradišar, Gortan Resinovič, Miroljub Klajijc. A hybrid approach for optimization of one-dimensional cutting, European Journal of Operational Research - 1999, Nr.. 119, p. 719 - 728.
10. Cutting Stock by Iterated Matching: [interaktyvus]. [žiūrėta 2003-12-11]. Prieiga per Internetą: <http://www2.informatik.uos.de/papers_html/or_94/nodel.html.>
11. Optimal Cutting Problem Using Java, examples of discrete optimization using approximate algorithm, [interaktyvus]. [žiūrėta 2004-01-23]. Prieiga per Internetą: <http://soften.ktu.lt/~mockus/>
12. Java 2 Platform, Standard Edition: [interaktyvus]. [žiūrėta 2003-12-11]. Prieiga per Internetą: <http://java.sun.com/>

TERMINŲ IR SANTRUMPŲ ŽODYNAS

Shape function- formų funkcijos.

Slicing tree- pjaustymo medis.

GAs- genetinis algoritmas.

GCLAs-giljotininio pjaustymo algoritmas.

Manifestas-specialus XML failas.

Mutation -keitimasis.

Crossover- perėjimas.

Jar- Java archyvavimo programa.

PMX- dalinai–pažymėtas perėjimas.

OX- sutvarkytas perėjimas.

CX- ciklinis perėjimas.