ՀՀ ԳԱԱ ԻՆՖՈՐՄԱՏԻԿԱՅԻ ԵՎ ԱՎՏՈՄԱՏԱՑՄԱՆ ՊՐՈԲԼԵՄՆԵՐԻ ԻՆՍՏԻՏՈՒՏ

ՔԵՅԱՆ ԱՐԹՈՒՐ ԺՈՐԱՅԻ

ԿԱՄԱՅԱԿԱՆ ՈՒՐՎԱԳԾՈՎ ԵՐԿՉԱՓ ՄԱՐՄԻՆՆԵՐԻ ՁԱՄԱՆ ԽՆԴՐԻ ՀԵՏԱԶՈՏՈՒԹՅՈՒՆ և ԾՐԱԳՐԱՅԻՆ ՀԱՄԱԿԱՐԳԻ ՄՇԱԿՈՒՄ

ԱՏԵՆԱԽՈՍՈՒԹՅՈՒՆ

Ե.13.05 – «Մաթեմատիկական մոդելավորում, թվային մեթոդներ և ծրագրերի համալիրներ» մասնագիտությամբ

տեխնիկական գիտությունների թեկնածուի գիտական աստիճանի հայցման ատենախոսության

Գիտական ղեկավար՝

ֆիզմաթ գիտությունների դոկտոր, պրոֆեսոր

Մ.Ե. Հարությունյան

Երևան 2016

ԲՈՎԱՆԴԱԿՈՒԹՅՈՒՆ

ՆԵՐԱԾՈՒԹՅՈՒՆ

Թեմայի արդիականությունը:

Այսօր արտադրության ոլորտում բնորոշ է ինֆորմացիոն տեխնոլոգիաների կիրառությունը՝ հումքի կորուստները առավելագույնս նվազեցնելու նպատակով, ինչը բավական կարևոր է շուկայական հարաբերություններում: Արտադրող տարբեր սուբյեկների առաջ խնդիր է դրված առավելություն ձեռք բերել մրցակցային պայքարում նոր տեխնոլոգիաների կիրառության շնորհիվ նյութական ծախսերի մակարդակը նվազեցնելու միջոցով: Դրան կարելի է հասնել ներդնելով արդյունավետ մաթեմատիկական մեթոդներ և ծրագրային միջոցներ:

Իրական կյանքի արտադրության պայմաններում կոմբինատոր օպտիմիզացիայի խնդիրների մի դաս են կազմում ձևման և փաթեթավորման խնդիրները, որոնք համակարգչային գիտության ոլորտում դասական և դժվարին խնդիրների ցանկում են [3, 7, 99]: Այս խնդիրները ուսումնասիրվել են տասնյակ տարիներ, և այսօր աշխատանքները շարունակվում են գտնելու նոր մոտեցումներ և ավելի լավ արդյունքներ: Վերջին տարիներին այս խնդիրների նկատմամբ աճող հետաքրքրությունը պայմանավորված է դրանց տեսական բարդությամբ և կիրառությունների բազմազանությամբ:

Այս տիպի խնդիրները պատկանում են NP լրիվ դասին, այսինքն հնարավոր չէ գտնել ալգորիթմ, որը որոշի օպտիմալ լուծումը բազմանդամային ժամանակում: Հաջողվում է միայն գտնել մոտարկող ալգորիթմներ, որոնք օպտիմալին ամենամոտ հնարավոր լուծումն են առաջարկում:

Կան խնդրի բազմապիսի տարբերակներ կախված կիրառություններից, ինչպես նաև բազմապիսի մոտեցումներ դրանց լուծման համար: Այդ խնդիրները բաժանվում են ենթադասերի ըստ չափողականության (միաչափ, երկչափ և եռաչափ) կախված

կիրառական մեկնաբանությունից: Ալգորիթմները դասակարգվում են նաև որպես առցանց (online) և արտացանց (offline) խմբերի: Առցանց տարբերակում դետալները հարկավոր է փաթեթավորել կամ ձևել ըստ դրանց հայտվելու հերթականության, չունենալով ինֆորմացիա հաջորդների մասին, ի տարբերություն արտացանց դեպքին, երբ դետալների ամբողջ բազմությունը նախորոք հայտնի է, և կարելի է դրանց հերթականությունը փոխել:

Այս աշխատանքը նվիրված է արտացանց երկչափ ձևման խնդիրների ուսումնասիրությանը, երբ հարկավոր է սահմանափակ հումքից, օրինակ, փայտից, կաշվից, մետաղից կամ թղթից, ձևել տարբեր ավելի փոքր չափի դետալներ, որոնց ցուցակը հայտնի է: Խնդրի նպատակն է հումքի մաքսիմալ օգտագործումը կամ կորուստների նվազեցումը: Հումքի օպտիմալ օգտագործման խնդրի լուծմանը վերաբերվող բազմաթիվ հոդվածներ են հրապարակվել [14, 68, 78]: Գծային ծրագրավորման դասական մեթոդները, որոնք հաջողությամբ կիրառվում են մասսայական արտադրության դեպքում, դժվար կիրառելի են անհատական արտադրության պայմաններում, երբ լինում են անհատական պատվերներ, և որպես կանոն օգտագործվում է թանկարժեք հումք: Ճշգրիտ օպտիմալ լուծումը հնարավոր է միայն հատարկման եղանակով, ինչը պրակտիկայում հնարավոր չէ կիրառել ժամանակի մեծ կորուստների պատճառով: Այդ իրավիճակում մեծ ուշադրություն է հատկացվում էվրիստիկ ալգորիթմների մշակմանը և հետազոտմանը: Դրանց արդյունավետությունը ստուգվում է փորձարկումների միջոցով:

Հաշվողական ժամանակի էական ծախսի և տեխնոլոգիական սահմանափակումների պայմաններում սովորաբար կիրառվում են ծրագրային համակարգեր: Սակայն երկչափ ձևում իրականացնող գոյություն ունեցող հասանելի համակարգերը աշխատում են միայն ուղղանկյունների հետ, մինչդեռ շատ կիրառություններում, օրինակ, կաշվի իրերի արտադրությունում, առկա են կամայական ուրվագծով երկչափ մարմիններ:

Այս իրավիճակում արդիական էր մշակել նոր էվրիստիկ ալգորիթմ և ծրագրային համակարգ, որը կիրականացներ կամայական ուրվագծով երկչափ մարմինների արդյունավետ ձևումը:

Հետազոտության նպատակը:

Աշխատանքի նպատակն է մշակել կամայական ուրվագծով երկչափ մարմինների արդյունավետ ձևումը իրականացնող ծրագրային համակարգ: Այդ նպատակին հասնելու համար մշակվել են մաթեմատիկական մոդելներ, նոր էվրիստիկ ալգորիթմ և ծրագրային համակարգ կամայական ուրվագծով մարմինների ձևման խնդրի լուծման համար:

Հետազոտման օբյեկտը:

Հետազոտման օբյեկտ են հանդիսանում ձևման խնդրի հետ կապված արտադրական տեխնոլոգիական գործընթացները: Հետազոտության առարկա են ձևման և փաթեթավորման խնդիրների մաթեմատիկական մոդելները և լուծման մեթոդները:

Հետազոտման մեթոդները:

Ալգորիթմների և ծրագրային համակարգի մշակման համար կիրառվել են ալգորիթմների տեսությունը, ծրագրային համակարգերի նախագծման ժամանակակից տեխնոլոգիաները: Ծրագրային համակարգերի նախագծման համար օգտագործվել են .NET պլատֆորմի C#, բազաների հետ աշխատելու համար SQL և ինտերֆեյսի նկարագրման համար HTML լեզուները և Visual Studio ծրագրային միջավայրերը: Ստացված արդյունքների արդյունավետությունը գնահատելու համար կիրառվել են թվային փորձարկումների արդյունքների գնահատման ստանդարտ եղանակները իրական արտադրության պայմաններում:

Հետազոտության գիտական նորույթը:

Կամայական ուրվագծով երկչափ մարմինների ձևման խնդրի լուծման ալգորիթմ գրականության մեջ չկար, նման լուծում առաջարկվում է առաջին անգամ:

Աշխատանքում առաջարկված են նոր մոտեցումներ, ինչպես նաև հայտնի մեթոդների մոդիֆիկացիաներ օբյեկտների մոդելավորման և դիսկրետացման համար:

Առաջարկված է կամայական ուրվագծով երկչափ մարմինների ձևման նոր էվրիստիկ ալգորիթմ:

Մշակվել է matrix.dll մատրիցների հետ աշխատելու ֆունկցիաների գրադարան:

Մշակվել է կամայական ուրվագծով երկչափ մարմինների ձևումը իրականացնող ծրագիր, ի տարբերություն բոլոր երկչափ ձևում իրականացնող ծրագրերի, որոնք աշխատում են միայն ուղղանկյունների հետ:

Ստացված արդյունքների կիրառական նշանակությունը:

Մշակված պարզ և մատչելի ինտերֆեյսով MIC Host համակարգը հնարավորություն է տալիս մոդելավորել իրական մարմինները, ներմուծել հումքի մոդելը վնասված մասերով: Այն իրականացնում է ձևումը, հաշվի առնելով հումքի վնասված հատվածները: Ատենախոսությունում դիտարկված մոդելները և առաջարկված լուծումները օժտված են ունիվերսալությամբ և կարող են կիրառվել արտադրության տարբեր ոլորտներում:

Ներդրումներ:

MIC Host Ծրագրային համակարգը փորձարկվել և ներդրվել է երկու փայտամշակման մասնավոր արդյունաբերության արտադրամասերում: Համակարգի կիրառման շնորհիվ բարձրացել է արտադրության արդյունավետությունը, կրճատվել են ծախսերը ավելի քան 20 տոկոսով:

Մշակված matrix.dll գրադարանը ներդրվել է github ազատ օգտագործման գրադարանների ռեսուրսում:

Պաշտպանության ներկայացվող հիմնական դրույթները:

- Մշակվել է ծրագրային միջավայր կամայական ուրվագծով երկչափ օբյեկտները մոդելավորելու համար: Այն ներառում է գործիքներ պարզ երկրաչափական օբյեկտների ստեղծման համար, դրանց հետ փոփոխություններ կատարելու, ինչպես նաև բարդ ուրվագծեր ստեղծելու համար:
- Առաջարկվել է եղանակ երկրաչափական մոդելների դիսկրետացման համար:
- Առաջարկվել է էվրիստիկ արտացանց ալգորիթմ կամայական ուրվագծով երկչափ մարմինների ձևման խնդրի արդյունավետ լուծման համար:
- Մշակվել է matrix.dll մատրիցների հետ աշխատելու ֆունկցիաների գրադարան:

- Մշակվել է ծրագիր, որը իրականացնում է կամայական ուրվագծով երկչափ մարմինների ձևումը տված հումքի պայմաններում: Այն ունի պարզ և մատչելի ինտերֆեյս:

Ապրոբացիա:

Ատենախոսության արդյունքները զեկուցվել են.

- Երկրորդ միջազգային ուսանողական սիմպոզիումում “Математика и информационные технологии в приложениях”, Сочи, 2016.
- ՀՀ ԳԱԱ ԻԱՊԻ ընդհանուր սեմինարում:

ԳԼՈՒԽ 1.

ՁԱՄԱՆ և ՓԱԹԵԹԱՎՈՐՄԱՆ ՀԻՄՆԱԿԱՆ ԽՆԴԻՐՆԵՐԸ, ԱԼԳՈՐԻԹՄՆԵՐԸ

Այս գլխում ձևակերպվում և դասակարգվում են ձևման և փաթեթավորման խնդիրների դասը, ինչպես նաև վերլուծվում են հայտնի խնդիրները և ալգորիթմները:

1.1. Խնդրի դրվածքը

Օբյեկտների ձևման և փաթեթավորման խնդիրը հանդիսանում է համակարգչային գիտության դասական և բարդ խնդիրներից մեկը:

Ձևման խնդիրն առաջին անգամ ձևակերպվել է 1939թ. Կանտորովիչի կողմից [6] և ուսումնասիրվել է տասնյակ տարիներ, որպեսզի հայտնաբերվեն լավագույն արդյունքներ: Այսօր ձևման և փաթեթավորման խնդրի համար ստացված նոր արդյուքներն ու հատուկ տարբերակներն ունեն կարևոր կիրառական նշանակություն տեղեկատվական տեխնոլոգիաների ոլորտում: Բացի այդ, դիտարկված խնդիրն ունի բազմաթիվ պոտենցիալ կիրառություններ իրական աշխարհի տարբեր ճյուղերում (օրինակ, տրանսպորտային, լոգիստիկա, տեղեկատվական տեխնոլոգիաներ և այլն): Որոշ կիրառություններից են հեռուստատեսային ծրագրերի պալանավորումը, օպտիմալ կտրման խնդիրը, ամպային հաշվարկները, բեռնատարների բեռնման խնդիրը և այլն:

Գոյություն ունեն նշված խնդրի մի շարք տարբերակներ և նրանց լուծման մի քանի մոտեցումների տարբեր օրինակներ, բազմաթիվ հրատարկված հոդվածներ [14, 39, 77, 80, 100]: Քանի որ փաթեթավորման խնդիրը բարդ խնդիրների NP լրիվ դասին է պատկանում, հետևաբար դժվար է ստեղծել ժամանակի բազմանդամային ալգորիթմ, որը կլուծի օպտիմալ արդյունքի ստացման խնդիրը: Հետևաբար, արդյունքում,

մոտարկող ալգորիթմները ներկայացված են գտնելու օպտիմալին հնարավորինս ամենամոտ լուծումը:

Եռաչափ և երկչափ փաթեթավորման խնդիրները հանդիսանում են միաչափ փաթեթավորման խնդրի ընդհանրացումները և նույնպես պատկանում են բարդ խնդիրների Np լրիվ դասին: Երկչափ փաթեթավորման խնդրի պոտենցիալ օգտագործումը իրական ժամանակի և արդյունաբերական բազմաթիվ կիրառություններում հանդիսանում է շարժառիթային փաստ նշված խնդրի ուսումնասիրման գործընթացում: Երկչափ փաթեթավորման խնդիրը օգտագործվում է այնպիսի ոլորտներում, ինչպիսիք են առարկաների փաթեթավորումը պահեստներում և բեռնատար մեքենաներում, ֆոնդային կտրման խնդիրներում (ուղղանկյունների կտրում տրված չափի թերթերից) և այլն: Ընդհանուր առմամբ այս ալգորիթմները կարող են դասակարգվել երկու խմբի.

- առցանց (online) ալգորիթմներ,
- արտացանց (offline) ալգորիթմներ:

Առցանց ալգորիթմները չունեն տեղեկություն մուտքային հաջորդականության հաջորդական տարրերի մասին, ի տարբերություն նրանց, արտացանց ալգորիթմներն ունեն տեղեկություն բոլոր տարրերի մասին, և հնարավոր է նրանց տեղավորել որոշակի կարգով՝ մինչ դետալի փաթեթավորումը:

Այս աշխատանքում դիտարկվում են ձևման խնդրի **երկչափ արտացանց տարբերակը**: Այդ ոլորտում հայտնի են որոշ արդյունքներ ֆիքսված չափի ուղղանկյունների համար: Մասնավորապես, 1982 թ. Չանգը [32] տվեց մոտարկման ալգորիթմ, լավագույններից մեկը տրվեց 2002 թ. Կապրարայի կողմից [25]: 2009 թ. Բանսալը ներկայացրեց հավանականային ալգորիթմը, որը լավացրեց նախորդը [13]: Բացի այդ նա ցույց տվեց, որ երկչափ ձևման խնդիրը թույլ չի տալիս ասիմտոտիկ բազմանդամային ժամանակով մոտարկման սխեմա: 2011 թ. լուծվել է երկչափ ձևման խնդիրը փոփոխական չափերի հումքով՝ պատահական ադապտիվ փնտրման

պրոցեդուրայով և փոփոխական շրջակայքի փնտրումով [33]: 2011 թ. ներկայացվեց հոդված «Փաթեթավորում չհամնընկնող տեղադրումներով» [41]: Այս ոլորտի հայտնի աշխատանքներից են նաև [10, 11, 21, 37, 72]-ը: Խնդրի պատմական զարգացումը կարելի է գտնել [16, 36]-ում: Մոտ խնդիրներով զբաղվում են նաև Հայաստանում [1, 2]:

Առավել մեծ հետաքրքրություն է ներկայացնում կամայական կամ ոչ կանոնավոր ուրվագծով օբյեկտների ձևման և փաթեթավորման խնդիրը, որի լուծման մոտեցումներ մինչ այս աշխատանքը հասանելի գրականության մեջ չեն հանդիպել:

Ներկայացնենք նշված տեսությանը վերաբերող մի քանի հայտնի խնդիրների և ալգորիթմների նկարագրությունները:

1.2. Միանման միաչափ բեռնարկղերի փաթեթավորման խնդիրը

Փաթեթավորումը որպես գծային ծրագրավորման խնդիր

Ենթադրենք տրված է V ծավալով բեռնարկղերի բազմությունը և $a_1, \cdots, a_n$ չափերի n առարկաների բազմությունը: Պետք է գտնել բեռնարկղերի B ամբողջ թիվը և $\{1, \cdots, n\}$ բազմության բաժանումը B ենթաբազմությունների՝ $S_1 \cup \cdots \cup S_B$ այնպիսիք, որ $\sum_{i \in S_k} a_i \leq V$ բոլոր $k = 1, \cdots, B$: Լուծումն անվանում են օպտիմալ, եթե B-ն մինիմալ է: Հետագայում մինիմալ B-ն նշանակվում է OPT:

Բեռնարկղերում փաթեթավորման խնդիրը կարող է ձևակերպվել որպես գծային ծրագրավորման խնդիր հետևյալ կերպ.

Մինիմալացնել

$$B = \sum_{i=1}^{n} y_i$$

ֆունկցիան, հետևյալ սահմանափակումների դեպքում՝

$$\sum_{j=1}^{n} a_j x_{ij} \leq V y_i, \qquad \forall i \in \{1, \cdots, n\}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad \forall j \in \{1, \cdots, n\}$$

$$y_i \in \{0,1\}, \qquad \forall i \in \{1, \cdots, n\}$$

$$x_{ij} \in \{0,1\}, \ \forall i \in \{1, \cdots, n\} \ \forall j \in \{1, \cdots, n\},$$

որտեղ $y_i = 1$, եթե i-րդ բեռնարկղն օգտագործվում է և $x_{ij} = 1$, եթե j-րդ առարկան տեղադրված է i-րդ բեռնարկղում:

1.3. Բազմանդամային մոտարկման ալգորիթմներ

Փաթեթավորման ամենապարզ բազամանդամային ալգորիթմներից են հանդիսանում՝ *Best Fit Decreasing — BFD* (լավագույն համապատասխանողն ըստ նվազման) *և First Fit Decreasing — FFD* (առաջին համապատասխանողն ըստ նվազման): Առարկաները դասավորում են ըստ չափերի նվազման և հաջորադական ձևով փաթեթավորում են բեռնարկղերում, որում փաթեթավորումից հետո մնում է ամենափոքր ազատ տեղ՝ BFD, կամ առաջին բեռնարկղում, որտեղ այն տեղավորվում է՝ FFD: Ապացուցված է, որ այս ալգորիթմներն օգտագործում են ոչ ավելի, քան $\frac{11}{9}\mathrm{OPT} + 1$ բեռնարկղեր []:

Սակայն փաթեթավորման խնդրի համար գոյություն ունեն նաև ասիմպտոտիկ ε-օպտիմալ բազմանդամային ալգորիթմներ: Խնդրի պնդումը հետևյալն է՝ OPT-ն հավասար է 2-ի կամ 3-ի հանդիսանում է NP-բարդ: Դրա համար կամայական $\varepsilon > 0$ –ի համար, դժվար է փաթեթավորել առարկաները $(3/2 - \varepsilon)\mathrm{OPT}$ բեռնարկղերում (եթե այդպիսի բազմանդամային ալգորիթմ գոյություն ունի, ապա բազմանդամային ժամանակում կարելի է որոշել բաժանվում է արդյոք n ոչ բացասական թվերը երկու բազմության, որոնք ունեն նույն գումարով տարրեր: Հայտնի է, որ այս խնդիրը NP-բարդ է: Հետևաբար, եթե P դասը չի համընկնում NP դասի հետ, ապա բեռնարկղերում փաթեթավորման խնդրի չկա բազմանդամային ժամանակով մոտարկման սխեմայի ալգորիթմ (PTAS): Մյուս կողմից, կամայական $\varepsilon > 0$ –ի համար, կարելի է գտնել լուծում

բազմանդամային ժամանակում ոչ ավելի քան $(1+\varepsilon)\mathrm{OPT}+1$ բեռնարկղերով: Այսպիսի ալգորիթմները դասվում են ասիմպտոտիկ PTAS դասին [25]: Բայց, քանի որ այդ դասի ալգորիթմների an^b բարդության գնահատականում երկու հաստատունները կամայականորեն կախված են ε-ից, նման ալգորիթմները ի տարբերություն FFD և BFD-ից կարող են պրակտիկորեն լինել ոչ օգտակար:

Փաթեթավորվող առարկաների հավանական բաշխումների որևէ դասի համար, ներառելով ուռուցիկ վեր և վար բաշխման ֆունկցիաներ, առարկաների թվի անսահմանափակ աճի դեպքում գոյություն ունի ասիմպտոտիկ օպտիմալ փաթեթավորման պրակտիկ բազմանդամային ալգորիթմ: Այդ դասին չպատկանող բաշխումների համար կարող են կառուցվել անհատական բազմանդամային ասիմպտոտիկ օպտիմալ ալգորիթմներ [8]:

1.4. Փաթեթավորման խնդրի դրվածքը և լուծման ալգորիթմները

Տրված է $L=\{1,\cdots,n\}$ առարկաների բազմությունը և նրանց $w_i \in (0,1), i \in L$ կշիռները: Գտնել L բազմության տրոհումը մինիմալ m քանակով այնպիսի $\{B_1, B_2,\cdots,B_m\}$ ենթաբազմությունների, որ

$$\sum_{i\in B_j} w_i \le 1$$

բոլոր $1 \le j \le m$: B_j բազմությունն անվանում են բեռնարկղեր: Պահանջվում է փաթեթավորել առարկաները մինիմալ թվով բեռնարկղերում [4, 5]:

«Համապատասխան հաջորդը» ալգորիթմը (NF)

Կամայական հաջորդականությամբ փաթեթավորում ենք առարկաները հետևյալ օրենքով: Առաջին առարկան տեղավորում ենք առաջին բեռնարկղում: k-րդ քայլում փորձում ենք k-րդ առարկան տեղավորել ընթացիկ բեռնարկղում: Եթե առարկան

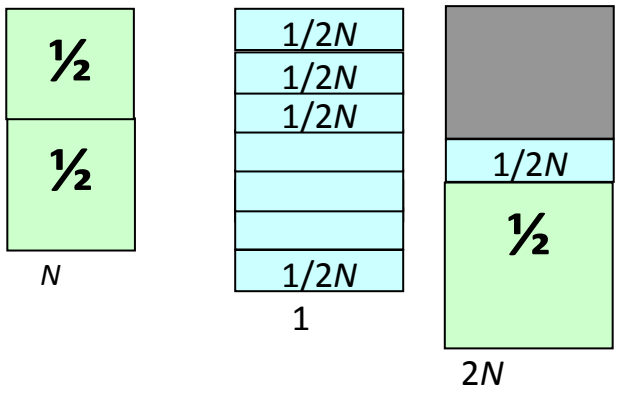
մտնում է, ապա տեղավորում ենք այն և անցնում հետևյալ քայլին, այլապես առարկան տեղավորում ենք նոր բեռնարկղում:

Թեորեմ. $NF(L) \leq 2OPT(L)$:

Ապացույց: Ենթադրենք $W = \sum_{i \in L} w_i$: Քանի որ կամայական երկու հաջորդական բեռնարկղեր պարունակում են առարկաներ, որոնց գումարային կշիռը փոքր չէ միավորից, ապա $NF(L) \leq 2[W]$: Բացի դրանից $OPT(L) \geq [W]$, որից էլ հետևում է պահանջվածը [4, 5]:

Օրինակ.

$L = \left\{\frac{1}{2}, \frac{1}{2N}, \frac{1}{2}, \frac{1}{2N}, \cdots, \frac{1}{2}, \frac{1}{2N}\right\}$, ընդամենը $4N$ առարկաներ:



$$OPT(L) = N + 1,\ NF(L) = 2N:$$

Դիտողություն. $NF(L) \leq 2OPT(L)$:

Ենթադրենք A ալգորիթմը L բազմության համար ծնում է $A(L)$ բեռնարկղեր և

$$R_A(L) \equiv {A(L)}/{OPT(L)}:$$

Մինիմումի խնդրի համար R_A երաշխավորված հարաբերական ճշտությունը A ալգորիթմի համար որոշվում է հետևյալ կերպ.

$$R_A \equiv inf\{r \geq 1 |\ R_A(L) \leq r, \text{բոլոր L} - \text{երի համար}\}:$$

Սահմանում. R_A^∞ ասիմպտոտիկ երաշխավորված հարաբերական ճշտությունը A ալգորիթմի համար որոշվում է հետևյալ կերպ [4, 5].

$$R_A^\infty \equiv inf\{r \geq 1 |\ \exists N > 0 \text{ այնպես, որ } R_A(L) \leq r, \text{բոլոր L} - \text{երի համար } OPT(L) \geq N\ \}:$$

«Համապատասխան առաջինը» ալգորիթմը (FF)

Կամայական հերթականությամբ փաթեթավորում ենք առարկաները հետևյալ կանոնով: Առաջին առարկան տեղադրում ենք առաջին բեռնարկղում: k-րդ քայլում գտնում ենք փոքրագույն համարով բեռնարկղը, որտեղ տեղավորվում է k-րդ առարկան և տեղադրում ենք այն: Եթե այդպիսի բեռնարկղ չկա, ապա վերցնում ենք նոր դատարկ բեռնարկղ և այնտեղ տեղադրում առարկան [4, 5]:

Թեորեմ. (առանց ապացույցի) Բոլոր L –երի համար

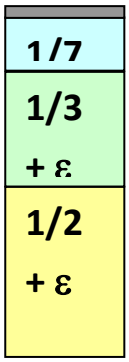
$$FF(L) \leq \left[\frac{17}{10} OPT(L) + 1\right]$$

և գոյություն ունեն օրինակներ որքան հնարավոր է OPT-ի մեծ արժեքներով, որոնց համար՝

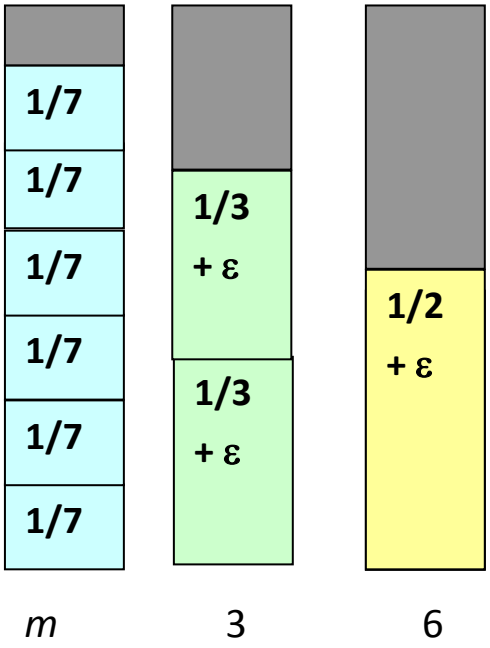
$$FF(L) \geq \left[\tfrac{17}{10} OPT(L) - 1\right]:$$

Օրինակ.

$$\mathrm{L} = \{1, \cdots, 18\mathrm{m}\},\quad w_i = \begin{cases} \frac{1}{7} + \varepsilon, & 1 \le i \le 6m \\ \frac{1}{3} + \varepsilon, & 6m < i \le 12m \\ \frac{1}{2} + \varepsilon, & 12m < i \le 18m \end{cases}$$



OPT(*L*) = 6*m*



FF(*L*) = 10*m*

$$\frac{FF(L)}{OPT(L)} = \frac{5}{3}:$$

«Համապատասխան լավագույնը» ալգորիթմը (BF)

Կամայական հերթականությամբ փաթեթավորում ենք առարկաները հետևյալ կանոնով: Առաջին առարկան տեղադրում ենք առաջին բեռնարկղում: k-րդ քայլում տեղադրում ենք k-րդ առարկան: Գտնում ենք մասնավորապես լցված բեռնարկղերը, որտեղ նրա համար բավարար է տեղը և նրանց մեջ ընտրում ենք առավել լցվածները:

Եթե այդպիսիք չկան, ապա վերցնում ենք նոր դատարկ բեռնարկղ և տեղադրում ենք այնտեղ k-րդ առարկան:

Թեորեմ. (առանց ապացույցի) [5] $\mathrm{R_{BF}} = R_{FF}$, $R_{BF}^{\infty} = R_{FF}^{\infty}$ և գոյություն ունեն ինչքան հնարավոր է մեծ արժեքներով $OPT(L)$, որոնց համար՝

$$BF(L) = \frac{4}{3}FF(L) \text{ և } FF(L) = \frac{3}{2}BF(L):$$

1.5. Առցանց (On-line) տիպի ալգորիթմներ

Առարկաները գալիս են անկանխատեսելի կարգով: Պահանջվում է նրանց փաթեթավորել մինիմալ քանակի բեռնարկղերում: Փաթեթավորված առարկան չի կարելի տեղափոխել այլ բեռնարկղ: Առարկաների համար նախապես պահեստավորող տեղը բացակայում է: NF, FF, BF ալգորիթմները հանդիսանում են On-line ալգորիթմներ [4, 5]:

Թեորեմ. (առանց ապացույցի) Կամայական On-line A ալգորիթմի համար ճիշտ է հետևյալ անհավասարությունը.

$$R_A^{\infty} > 1.5:$$

Սահմանափակ թույլտվությամբ բեռնարկղերի ալգորիթմը

On-line ալգորիթմն անվանում են սահմանափակ թույլտվությամբ բեռնարկղերի ալգորիթմ, եթե յուրաքանչյուր քայլում ալգորիթմն ունի հնարավորություն տեղադրել առարկաները միայն K բեռնարկղերից որևէ մեկում ($K = const$): Դրանց անվանում են բաց բեռնարկղեր: Եթե բեռնարկղը փակել են, ապա այն էլ չի բացվի (օրինակ, ուղարկվում է սպառողին): Նախքան բաց դատարկ բեռնարկղ ավելացնելը, պետք է փակել K բաց բեռնարկղերից մեկը:

Ալգորիթմ NF, օրինակ $K = 1$*-ի համար:*

Բեռնարկղերի ընտրության կանոնները.

1. փակել փոքրագույն համարով բեռնարկղը,
2. փակել ամենալցված բեռնարկղը:

Սահմանափակ թույլտվությամբ ալգորիթմների օրինակներ [4, 5].

FF_1- 1-ին կանոնով FF ալգորիթմը

FF_2-2-րդ կանոնով FF ալգորիթմը

BF_1- 1-ին կանոնով BF ալգորիթմը

BF_2- 2-րդ կանոնով BF ալգորիթմը

Թեորեմ. $\forall \mathrm{K} \geq 2$

1) $R_{FF_1}^{\infty} = R_{FF_2}^{\infty} = 1{,}7 + \frac{3}{10(K-1)}$

2) $R_{BF_1}^{\infty} = 1{,}7 + \frac{3}{10K}$

3) $R_{BF_2}^{\infty} = 1{,}7$

4) Կամայական սահմանափակ թույլտվությամբ բեռնարկղերի ալգորիթմի համար

$$R_A^{\infty} \geq 1{,}69103\cdots$$

«Ըստ դասակարգման առաջին համապատասխանողը» ալգորիթմը (FFD)

- Առարկաները դասակարգենք ըստ կշիռների ոչ աճման կարգով

$$w_1 \geq w_2 \geq \cdots \geq w_n$$

- Կիրառենք FF (BF) ալգորիթմը:

Թեորեմ. (առանց ապացույցի) [28,29] $\mathrm{FFD(L)} \leq \frac{11}{9}\mathrm{OPT(L)} + 4$, բոլոր L-երի համար և գոյություն ունեն օրինակներ ինչքան հնարավոր է $\mathrm{OPT(L)}$–ի մեծ արժեքներով, որոնց համար՝

$$\mathrm{FFD(L)} \geq \frac{11}{9}\mathrm{OPT(L)},$$

բացի այդ $R_{FFD}^{\infty} = R_{BFD}^{\infty} = \frac{11}{9} \approx 1.22$:

Ասիմպտոտիկ երաշխավորված ճշտության գնահատականները

Ալգորիթմ	T_A^*	R_A^{∞}	$R_A^{\infty}(1/2)$	$R_A^{\infty}(1/3)$	$R_A^{\infty}(1/4)$
NF	*O*(*n*)	2.000	2.000	1.500	1.333
FF	*O*(*n* log *n*)	1.700	1.500	1.333	1.250
BF	*O*(*n* log *n*)	1.700	1.500	1.333	1.250
NFD	*O*(*n* log *n*)	1.691	1.424	1.302	1.234
FFD	*O*(*n* log *n*)	1.222	1.183	1.183	1.150
BFD	*O*(*n* log *n*)	1.222	1.183	1.183	1.150

$R_A^{\infty}(\alpha)$-ասիմպտոտիկ ճշտությունը $w_i \leq \alpha, \text{բոլոր i} \in \mathrm{L}$ կշիռներով առարկաների օրինակների համար:

Թեորեմ. (առանց ապացույցի) [4, 5] $\forall \varepsilon \in (0,1]$ գոյություն ունի A_ε ալգորիթմը, որը գտնում է ոչ ավելի $(1+2\varepsilon)OPT+1$ բեռնարկղերի քանակով փաթեթը: A_ε-ի բարդությունը բազմանդամորեն կախված է n-ից:

A_ε ալգորիթմը.

1. Հեռացնել ε –ից ոչ պակաս կշռով առարկաները:
2. Դասավորել մնացած առարկաները և տրոհել նրանց $K = \left\lceil 1/\varepsilon^2 \right\rceil$ խմբերի:
3. Յուրաքանչյուր խմբում առարկաների կշիռներն ավելացնել մինչև խմբում եղած մաքսիմալ կշիռը:
4. Գտնել ε –ից ոչ պակաս K տարբեր կշիռներ ունեցող առարկաների օպտիմալ փաթեթավորումը:

5. Վերադարձնել առարկաների սկզբնական կշիռները և կիրառել FF ալգորիթմը ε –ից ոչ պակաս կշռով առարկաների համար:

Բացասական արդյունք

Թեորեմ. [4, 5] $\forall\, \varepsilon > 0$-ի համար $R_A = \frac{3}{2} - \varepsilon$ երաշխավորված ճշտությամբ մոտարկող բազմանդամային A ալգորիթմի գոյությունը կբերի $P = NP$:

Ապացույց. Ենթադրենք այդպիսի A ալգորիթմ գոյություն ունի: Ցույց տանք,ինչպես նրա օգնությամբ ճշգրտորեն լուծել NP –լրիվ դասի խնդիրներից մեկը, այսինքն տրոհման խնդիրը: Տրված է n ոչ բացասական $\{a_1, \cdots, a_n\}$ թվերը: Կարելի է արդյոք նրանց բաժանել երկու ենթաբազմությունների այնպես, որ ամեն մի ենթաբազմությունում թվերի գումարը հավասար լինի՝

$$C = \frac{1}{2}\sum_{i=1}^{n} a_i:$$

Դիտարկենք $w_i = a_i/C,\ i = 1, \cdots, n$ կշիռներով առարկաների բեռնարկղերում փաթեթավորման խնդիրը: Եթե դրանք կարելի է փաթեթավորել երկու բեռնարկղում, ապա տրոհման խնդրի պատասխանն «այո» է: Կիրառենք A ալգորիթմը բեռնարկղերի խնդրի համար: Եթե $OPT = 2$, ապա A ալգորիթմը տալիս է 2, այլապես $R_A \geq \frac{3}{2}$, այսինքն A ալգորիթմը ճիշտ է:

Ստորին գնահատականները

Խնդրի փոփոխականները.

$$y_j = \begin{cases} 1, & j - \textit{րդ բեռնարկղն օգտագործվում է} \\ 0, & \textit{հակառակ դեպքում} \end{cases}$$

$$x_{ij} = \begin{cases} 1, \textit{եթե } i - \textit{րդ } \textit{առարկան } j - \textit{րդ } \textit{բեռնարկղում է տեղավորված} \\ 0, \qquad\qquad\qquad\qquad\qquad\qquad \textit{հակառակ դեպքում} \end{cases}$$

Մաթեմատիկական մոդելը

$$min \sum_{j=1}^{n} y_j,$$

հետևյալ սահմանափակումների դեպքում՝

$$\sum_{i=1}^{m} w_i \, x_{ij} \leq y_j, j = 1, \cdots, n,$$

$$\sum_{j=1}^{n} x_{ij} = 1, i = 1, \cdots, n,$$

$$y_j, x_{ij} \in \{0,1\}, \qquad i, j = 1, \cdots, n:$$

Գծային ծրագրավորման պայմանների թուլացում

Փոփոխականների բուլյան պայմանները փոխարինենք հետևյալ պայմաններով [4, 5].

$$0 \leq y_j \leq 1, \qquad j = 1, \cdots, n$$

$$0 \leq x_{ij} \leq 1, \quad i, j = 1, \cdots, n :$$

Այդ դեպքում օպտիմալ լուծումներից մեկն ունի հետևյալ տեսքը.

$$x_{ij}^* = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}, \; y_j^* = w_j,$$

որը տալիս է ստորին գնահատականը՝

$$H_0 = \left\lceil \sum_{i=1}^{n} w_i \right\rceil:$$

(առարկաները կարելի է կտրել կամայական ձևով):

Martello & Toth-ի գնահատականները

Օրինակի համար $\mathrm{L} = \{1, \cdots, \mathrm{n}\}$, $0 \leq \mathrm{w_i} < 1$ և $\forall\ 0 \leq \alpha \leq 1/2$, տեղադրենք՝

$L_1 = \{i \in L | w_i > 1 - \alpha\}$-խոշոր առարկաները

$L_2 = \{i \in L | 1 - \alpha \geq w_i > 1/2\}$-միջին առարկաները

$L_3 = \{i \in L |\ 1/2 \geq w_i \geq \alpha\}$-փոքր առարկաները:

Թեորեմ. [4, 5] $\forall\ 0 \leq \alpha \leq 1/2$

$$\mathrm{H_1}(\alpha) = |L_1| + |L_2| + max\left(0, \left\lceil \sum_{i \in L_3} w_i - \left(|L_2| - \sum_{i \in L_2} w_i\right)\right\rceil\right)$$

մեծությունը հանդիսանում է *OPT*(*L*)-ի համար ստորին գնահատական:

Ապացույց. $L_1 \cup L_2$ բազմության յուրաքանչյուր առարկա պահանջում է առանձին բեռնարկղ: Դրա համար ցանկացած թույլատրելի լուծման մեջ կա ոչ պակաս $|L_1| + |L_2|$ բեռնարկղ: L_3 բազմության առարկաները L_1-ի առարկաների հետ տեղադրված չեն: Դա նշանակում է, այդ առարկաները կամ L_2-ի առարկաների հետ են տեղադրված կամ առանձին բեռնարկղերում:

L_2-ի բեռնարկղերում մնացել է $S = \left(|L_2| - \sum_{i \in L_2} w_i\right)$ ազատ տեղ: Հետևաբար, L_3 բազմության առարկաների համար պահանջվում է մինիմումը $\left\lceil \sum_{i \in L_3} w_i - S \right\rceil$ առանձին բեռնարկղեր:

Թեորեմ. [4, 5] $\forall\ 0 \leq \alpha \leq 1/2$

$$\mathrm{H_2}(\alpha) = |L_1| + |L_2| + max\left\{0, \left\lceil \frac{|L_3| - \sum_{i \in L_2} \left\lfloor \frac{1 - w_i}{\alpha} \right\rfloor}{\left\lfloor \frac{1}{\alpha} \right\rfloor} \right\rceil\right\}$$

մեծությունը հանդիսանում է *OPT*(*L*)-ի համար ստորին գնահատական:

Ապացույց. L_3 բազմության յուրաքանչյուր առարկայի կշիռը փոխարինենք α-ով: Այդ դեպքում մեկ բեռնարկղում կտեղավորվի $\left[\frac{1}{\alpha}\right]$ առարկաներ, և L_3 բազմության համար կպահանջվի $\left[{|L_3|}/{\left[\frac{1}{\alpha}\right]}\right]$ լրացուցիչ բեռնարկղեր: Սակայն L_3-ի առարկաների մի մասը կարելի է տեղադրել L_2-ի բեռնարկղերում: Նրանցից յուրաքանչյուրն ունի $1-w_i, i \in L_2$ ազատ տեղ, որտեղ կտեղավորվի $\left[\frac{1-w_i}{\alpha}\right]$ առարկաներ L_3-ից:

Հետևություն 1. $H = max\{H_1(\alpha),\ H_2(\alpha),\ \ 0 \le \alpha \le 0{,}5\}$ մեծությունը հանդիսանում է *OPT*(*L*)-ի ստորին գնահատական:

Հետևություն 2. $H \ge H_0 \equiv [\sum_{i \in L} w_i]$:

Ապացույց. $\alpha = 0$-ի դեպքում կստանանք՝

$$H \ge H_1(0) = max\{|L_2|, H_0\} \ge H_0:$$

Ինչպես գտնել H-ը՝ չտեսակավորելով α-ի բոլոր արժեքները:

Հետևություն 3. Ենթադրենք V-ն բազմություն է բոլոր $w_i \le 0{,}5$ տարբեր արժեքներից: Այդ դեպքում՝

$$H = \begin{cases} n, & \text{եթե } V = \emptyset \\ max\{H_1(\alpha), H_2(\alpha),\ \ \alpha \epsilon V - \text{ի համար}\}, & \text{եթե } V = \emptyset \end{cases},$$

այսինքն առարկաների դասակարգումից հետո, կստանանք՝

$$T_H = O(n + nlogn):$$

1.6. Հիլմորի և Հոմորիի ստորին գնահատականները

Ենթադրենք տրված են անսահմանափակ թվով բեռնարկղեր՝ միավոր տարողությամբ: Յուրաքանչյուր $i \in L$ նախապատրաստուկի համար տրված է

երկարությունը՝ $0 < w_i < 1$ և նրանց քանակը՝ $n_i \geq 1$: Պահանջվում է փաթեթավորել նախապատրաստուկները մինիմալ քանակով բեռնարկղերում [4, 5]:

Դիտարկենք մեկ բեռնարկղում փաթեթավորման տարբեր եղանակներ:

Ենթադրենք (a_{ij}) մատրիցը տալիս է i-րդ տիպի նախապատրաստուկի քանակը j-րդ եղանակով:

Խնդրի փոփոխականները.

$x_j \geq 0$, բեռնարկղերի ամբողջ թիվն է՝ փաթեթավորված j-րդ եղանակով:

$$min \sum_{j \in J} x_j$$

հետևյալ պայմանների դեպքում՝

$$\sum_{j \in J} a_{ij} x_j \geq n_i\,, \qquad i \in L,$$

$$x_j \geq 0, \text{ ամբողջ են, } j \in J:$$

J բազմությունն ունի էքսպոնենցիալ հզորություն:

Ստորին գնահատականները

$$H = min \sum_{j \in J} x_j$$

$$\sum_{j \in J} a_{ij} x_j \geq n_i\,, \qquad i \in L,$$

$$x_j \geq 0,\ j \in J:$$

Լուծելով գծային ծրագրավորման խնդիրը՝ կստանանք ստորին H գնահատականը: Սակայն խնդիրն ունի ահռելի չափողականություն:

1.7. Սյուների գեներացման խնդիրը

Ընտրենք $J' \subset J$ ենթաբազմությունն այնպես, որ հետևյալ $ԳԾ(J')$ ենթախնդիրն ունենա լուծում՝

$$min \sum_{j \in J'} x_j$$

$$\sum_{j \in J'} a_{ij} x_j \geq n_i\,, \qquad i \in L,$$

$$x_j \geq 0,\ j \in J':$$

Սա հեշտ է կատարել կամայական *NF*, *FF*, *BF*,... հիերիստիկ ալգորիթմների միջոցով:

Ենթադրենք x_j^*-ը օպտիմալ լուծում է J'-ի համար և λ_i^*-ը օպտիմալ լուծում է համապիտատասխան J' երկակի խնդրի համար [4, 5]:

Դիտարկենք երկակի խնդիրը J' բազմության համար.

$$max \sum_{i \epsilon L} n_i \lambda_i$$

հետևյալ սահմանափակումների դեպքում՝

$$\sum_{i \epsilon L} a_{ij} \lambda_i \leq 1, \qquad j \in J',$$

$$\lambda_i \geq 0,\ i \in L:$$

Եթե բոլոր $j \in J$ ճիշտ է

$$\sum_{i \epsilon L} a_{ij} \lambda_i^* \leq 1 \qquad (1.1)$$

ապա

$$\overline{x_J} = \begin{cases} x_{j,}^*\ j \in J' \\ 0,\ \ j \in J \setminus J' \end{cases}$$

վեկտորը գծային ծրագրավորման (ԳԾ) օպտիմալ լուծումն է ամբողջ j բազմության համար և

$$H = \sum_{j \in J'} x_j^*:$$

Խնդիր: Ինչպես ստուգել (1.1)-ը՝ չդիտարկելով ամբողջ j բազմությունը, և ինչ անել, եթե պայմանը չի կատարվում:

Դիտարկենք **ուսապարկի** խնդիրը՝

$$\alpha = max \sum_{i \epsilon L} \lambda_i^* y_i,$$

Հետևյալ սահմանափակումների դեպքում՝

$$\sum_{i \in L} w_i y_i \leq 1, \qquad \text{բեռնարկղերի տարողունակությունը:}$$

$$y_j \geq 0, \text{ամբողջ են, i} \in \text{L:}$$

Այս խնդրի օպտիմալ լուծումը տալիս է բեռնարկղի փաթեթավորման նոր եղանակ:

Եթե $\alpha \leq 1$, ապա $(*)$-ը կատարված է:

Եթե $\alpha > 1$, ապա կստանանք փաթեթավորման եղանակ, որը պետք է ավելացնել J' բազմությանը (այսինքն սիմպլեքս աղյուսակում գտել ենք առաջատար սյունը):

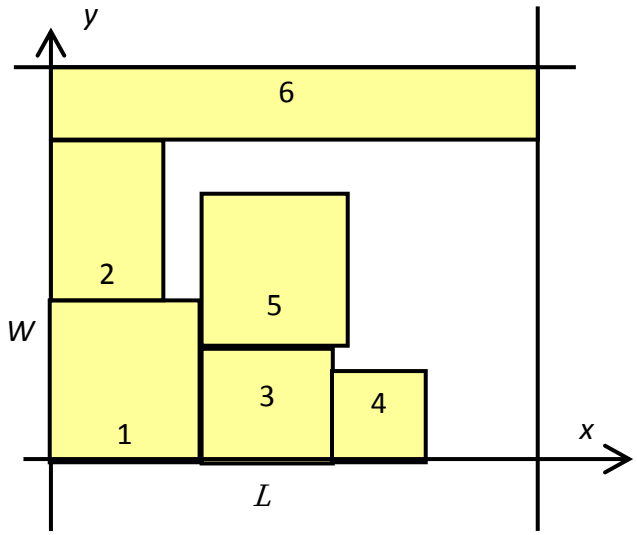
Մեթոդի ընդհանուր սխեման.

1. Ընտրել $J' \subset J$ ենթաբազմությունը
2. Լուծել $ԳԾ(J')$ խնդիրը, ստանալ x_j^* , λ_j^*
3. Լուծել ուսապարկի խնդիրը, ստանալ α-ն
4. Եթե $\alpha \leq 1$, ապա $H = \sum_{j \in J'} x_j^*$, STOP
5. J'-ում ավելացնել փաթեթավորման նոր եղանակ և վերադառնալ քայլ 2-ին:

$H = \sum_{j \in J'} x_j^*$ գնահատականը ժամանակատար է, բայց գերիշխում է մյուսներին ըստ ճշտության: $x_j = [x_j^*], j \in J$ լուծումը հաճախ հանդիսանում է օպտիմալ:

1.8. Երկչափ փաթեթավորման խնդիրը

Տրված են n ուղղանկյունաձև առարկաներ $w_i \times l_i, i = 1, \cdots, n$ չափերով: Պետք է գտնել դրական կիսահարթությունում նվազագույն մակերեսով եզրագծող ուղղանկյան փաթեթավորումը (Նկ.1.1):



Նկ.1.1. Դրական կիսահարթությունում նվազագույն մակերեսով եզրագծող ուղղանկյան փաթեթավորումը:

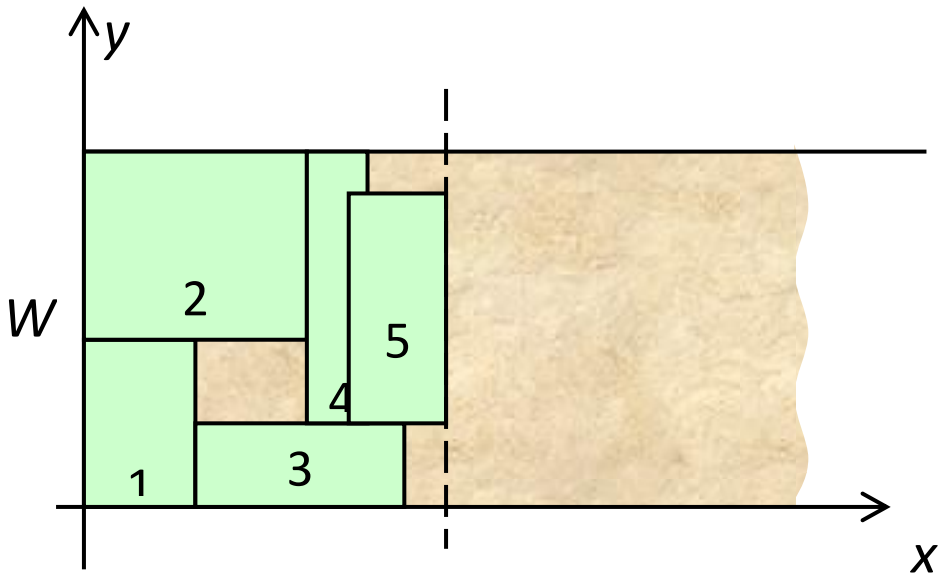
Ոլորաններն արգելված են: Առարկաների կողմերը զուգահեռ են կոորդինատների առանցքներին:

$$\boldsymbol{L \times M \rightarrow min:}$$

Խնդիրը NP լրիվ դասին է պատկանում [4, 5, 78]:

Շերտավոր փաթեթավորման խնդիրը

Տրված են n ուղղանկյունաձև առարկաներ $w_i \times l_i, i = 1, \cdots, n$ չափերով և W լայնությամբ կիսաանվերջ շերտ: Պետք է գտնել նվազագույն երկարություն զբաղեցնող առարկաների փաթեթավորումը (Նկ.1.2):

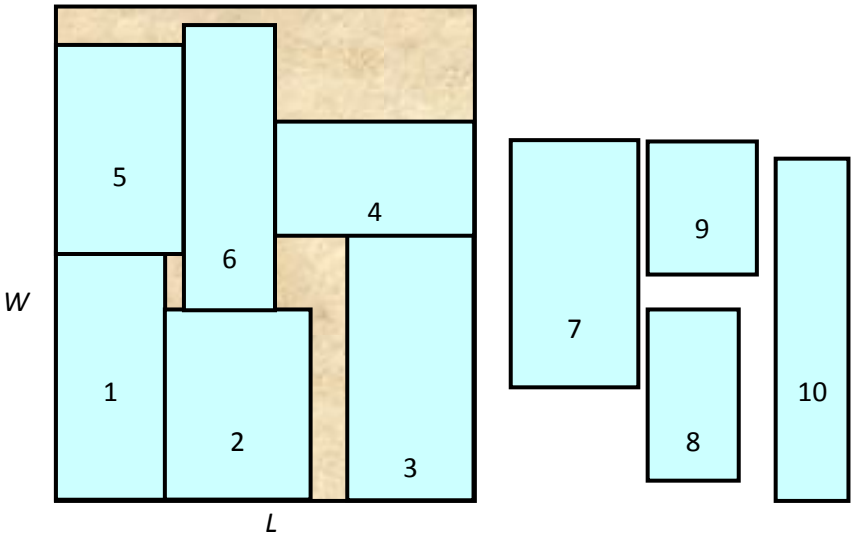


Նկ. 1.2. Շերտավոր փաթեթավորում:

$l_i = 1, i = 1, \cdots, n$, կստանանք բեռնարկղերում միաչափ փաթեթավորման խնդիրը [4, 5]:

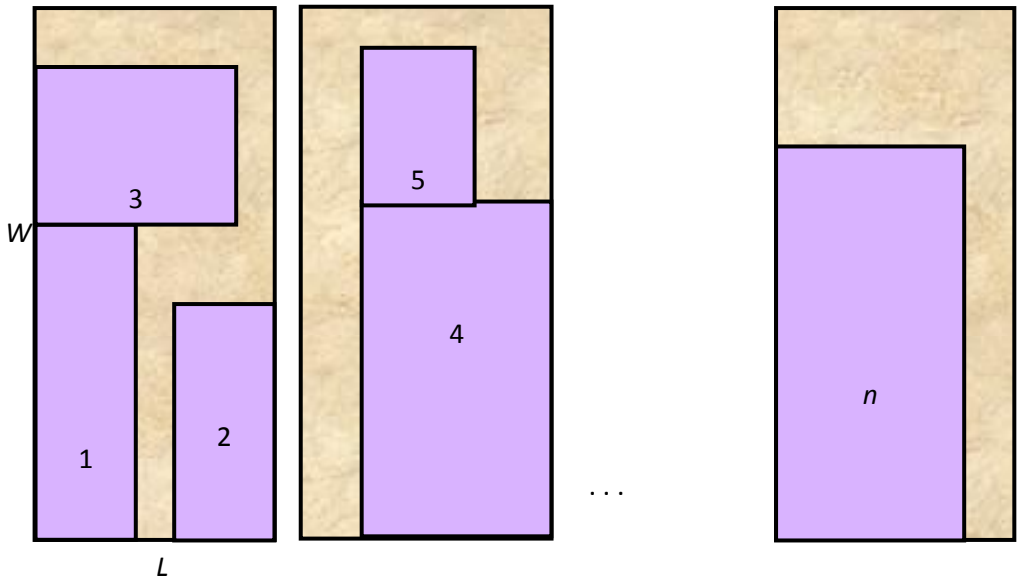
Ուսապարկի խնդիրն ուղղանկյունների համար

Տրված են n ուղղանկյունաձև առարկաներ $w_i \times l_i$ չափերով և c_i կշիռներով, $i = 1, \cdots, n,$ ու մեծ ուղղանկյուն՝ $W \times L$: Գտնել առավելագույն կշռով առարկաների ենթաբազմություն, որոնք կարելի է կտրել մեծ ուղղանկյունից, $l_i = L, i = 1, \cdots, n,$ ստանում ենք ուսապարկի հայտնի խնդիրը (Նկ.1.3) [78]:



Նկ. 1.3. Ուսապարկի խնդիրը:

Բեռնարկղերում փաթեթավորման խնդիրը ուղղանկյունների համար

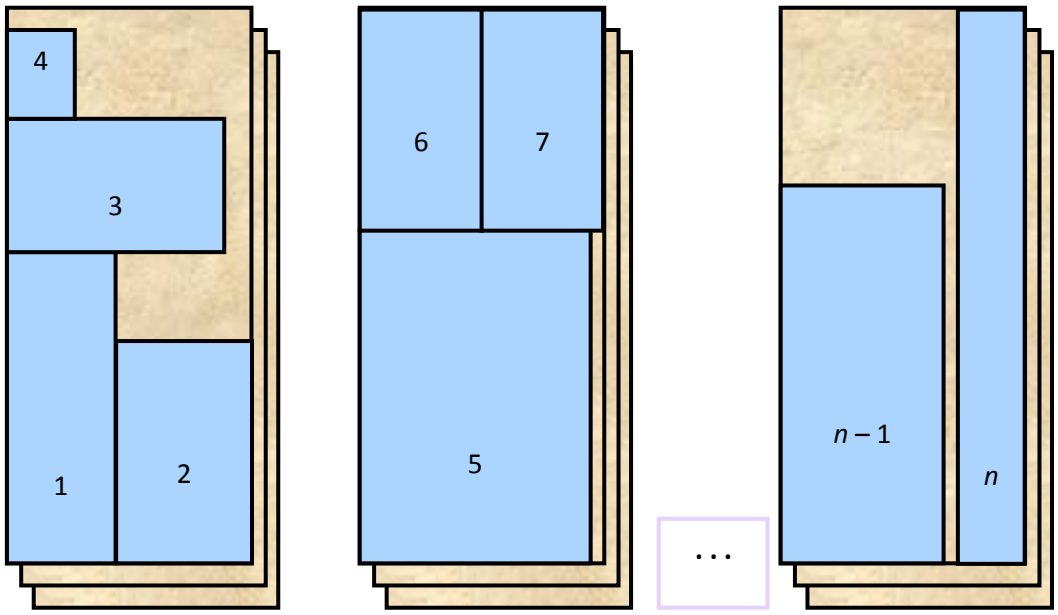


Նկ. 1.4. Բեռնարկղերում փաթեթավորման խնդիրը ուղղանկյունների համար:

Տրված են n ուղղանկյունաձև առարկաներ $w_i \times l_i$ չափերով, $i = 1, \cdots, n,$ ու անսահմանափակ թվով միանման նախապատրաստուկներ՝ $W \times L$ չափով: Գտնել առարկաների փաթեթավորումը օգտագործվող նախապատրաստուկների նվազագույն քանակով (Նկ.1.4) [78]:

Ուղղանկյունաձև կտրման խնդիրը հաջորդական արտադրության համար

Տրված են n ուղղանկյունաձև առարկաներ $w_i \times l_i$ չափերով և k_i, $i = 1, \cdots, n$ քանակի պահանջարկով: Հայտնի են նախապատրաստուկների չափերը՝ $W \times L$: Գտնել նախապատրաստուկների կտրման պլանը, որը թույլ կտա ստանալ պահանջված քանակով այն բոլոր առարկաները, որոնք օգտագործում են նվազագույն քանակով նախապատրաստուկներ(Նկ.1.5) [78]:



Նկ.1.5. Ուղղանկյունաձև կտրման խնդիրը հաջորդական արտադրության համար

1.9. Եռաչափ փաթեթավորման խնդիրը և նրա լուծման մեթոդները

Եռաչափ փաթեթավորման խնդիրն իրենից ներկայացնում է միաչափ և երկչափ դասական խնդիրների բնական էվոլյուցիան: Նշված խնդրի առավել հաճախ օգտագործվող պրակտիկ կիրառությունը բեռնափոխադրումն է, որը պետք է փաթեթավորվի բեռնարկղերում կամ տրանսպորտային միջոցների թափքերում կամ բեռի փաթեթավորումը պալետների վրա:

Արկղերի փաթեթավորումը բեռնարկղում, տրանսպորտային միջոցների թափքերում կամ պալետների վրա հանդիսանում է փաթեթավորման բարդագույն խնդիրներից մեկը՝ սահմանափակումների պատճառով, որոնք առաջանում են իրական աշխարհում: Օրինակ, կազմակերպելով օպտիմալ փաթեթավորում առավել արդյունավետ կերպով օգտագործելով ունեցած ռեսուրսները, փոխադրումների համար անհրաժեշտ բեռնարկղերի քանակության չգնահատելը կարող է թանկ նստել լրացուցիչ բեռնարկղերի ուշացման և փոխադրման պատճառով, քանի որ ոչ լրիվությամբ են օգտագործվում՝ վերածվելով ապարդյուն ծախսված ռեսուրսի: Հաշվի առնելով արկղերի չափերը՝ գոյություն ունեն բազմաթիվ սահմանափակումներ՝ կապված կշռի բաշխման, հարթեցման, փաթեթավորման կայունության հետ: Բացի այդ հաճախորդները կարող են սահմանափակել բեռները, որոնք անհրաժեշտ են տեղափոխել միասին կամ, օրինակ, տրանսպորտի բեռնաթափումը կարող է տեղի ունենալ մի քանի տեղերում [24]:

1.10. Էվրիստիկական ալգորիթմը երկչափ մեկ մեծ ուղղանկյան փաթեթավորման խնդրի համար

Վ. Մ. Կոտովը և Դ. Ցաոն առաջարկել են [69] էվրիստիկ ալգորիթմ երկչափ մեկ մեծ ուղղանկյան փաթեթավորման խնդրի համար (2D-SLBPP-two-dimensional single

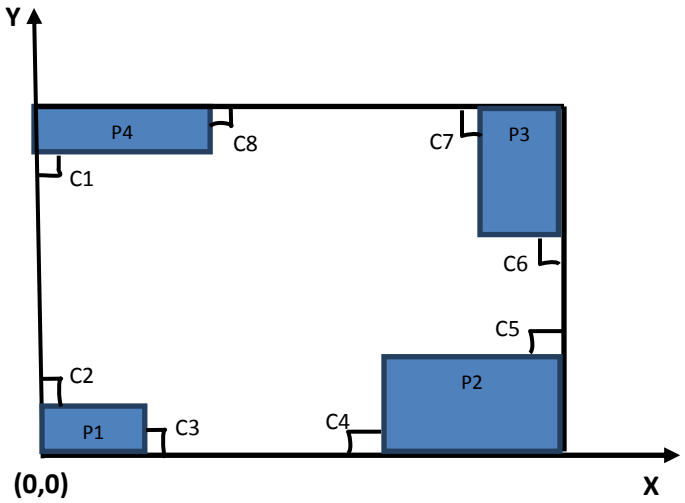
large bin packing problem)՝ հիմնված գոգավոր անկյան սկզբունքի վրա (BCC-based on concave corner): Փորձերը ցույց են տվել, որ առաջարկված ալգորիթմը հանդիսանում է բավականին մրցունակ և կարող է դիտարկվել որպես այլընտրանքնային մեթոդ 2D-SLBPP-ի համար: Ալգորիթմը, հատուկ դեպքերում, կարող է ստանալ բավարար արդյունքներ ավելի արագ, քան համապատասխան մյուս մոտեցումները: Փաթեթավորման խնդիրը իր մեջ ներառում է բազմաթիվ արդյունաբերական կիրառություններ: Օրինակ, փայտամշակություն, նավաշինություն, տեքստիլ արդյունաբերություն, կաշվեգործություն և այլն: Բոլոր այս հավելվածները ձևակերպված են որպես փաթեթավորման խնդիրներ [78, 39, 100]: Այս ալգորիթմը հիմնաքար հանդիսացավ նոր ալգորիթմի ստեղծման համար, այդ պատճառով արժանացավ հատուկ ուշադրության:

Խնդիրն այս դեպքում կարող է նկարագրվել հետևյալ կերպ. տրված է մեծ ուղղանկյուն ֆիքսված չափերով և փոքր ուղղանկյունների բազմություն: 2D-SLBPP-ի հետազոտության նպատակն է՝ ինչպես փաթեթավորել ուղղանկյուն կտորները օրթոգոնալ ձևով, բացի այդ վատագույն դեպքում փորձել փոքրացնել կտորներն այնպես, որ ոչ մի երկու կտոր չհամընկնեն:

Գոգավոր անկյան (BCC) սկզբունքի վրա հիմնված տեղադրման ռազմավարությունը

Նախքան ալգորիթմի նկարագրությունը, ենթադրենք, որ ուղղանկյան լայնությունը և բարձրությունը W և H են: Բոլոր պարամետրերն ընդունվում են որպես ամբողջ: Կտորները պետք է փաթեթավորվեն կողերով, որոնք զուգահեռ են արկղի եզրերին և չեն կարող թեքվել 90^0-ով: Էվրիստիկ ալգորիթմի կառուցման համար կատարվում են հետևյալ նշանակումները:

Ենթադրենք C_i գոգավոր անկյուն է (Նկ. 1.6), որը կազմված է երկու կողերից, անկյան չափը 90^0 է, բացի այդ գոգավոր անկյունը չի պատկանում որևէ i- րդ ($\mathrm{P_i}$) մասին:



Նկ.1.6. Գոգավոր անկյան օրինակ:

Սահմանվում է U-ն հետևյալ կերպ.

$$U = \{U_1, U_2, \cdots, U_k\};\ U_i \cap U_j = \emptyset, i \neq j,$$

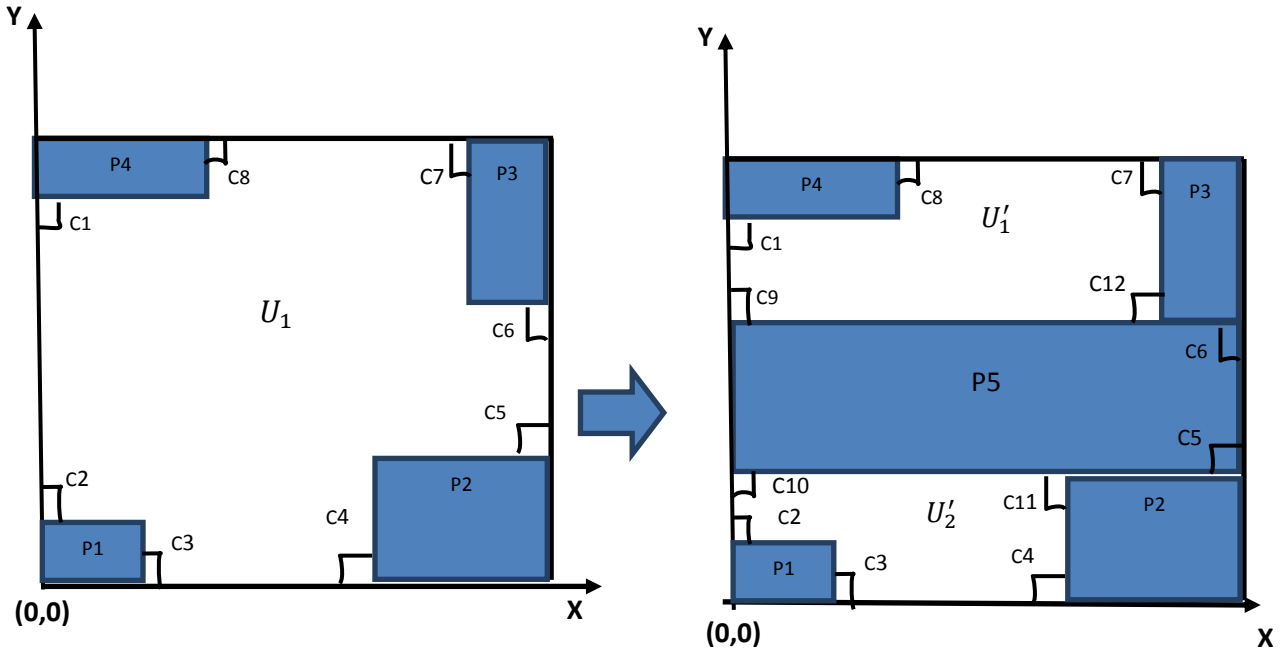
որտեղ U_i –ն իրենից ներկայացնում է գոգավոր անկյունների հավաքածու, իսկ k-ն արկղի վրա իրարից չկախված դոմենների քանակը՝ նախքան արկղի վրա P_5-ի փաթեթավորումը: Օրինակ, (Նկ.1.7) $U = U_1$, $U_1 = \{C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8\}$, P_5 –ի փաթեթավորումից հետո, U_1-ը պետք է բաժանվի երկու մասի, այդ դեպքում ունենք՝

$$U = \{U_1', U_2'\},\ U_1' = \{C_1, C_7, C_8, C_9, C_{12}\},\ U_2' = \{C_2, C_3, C_4, C_{10}, C_{11}\}:$$

Ակնհայտ է, որ մինչ արկղի վրա կամայական մասի փաթեթավորումը, գոյություն ունեն 4 գոգավոր անկյուններ և $k = 1$:

Նախքան տախտակի վրա կամայական մասի փաթեթավորումը, l_U_1-ով նշանակվում է U_1-ի ձախ կողը, իսկ $r\_U_1 = W$-ով՝ աջ կողը: Նման ձևով սահմանվում է t_U_1 և b_U_1, որպեսզի նշանակվի U_1-ի վերին և ստորին կողմերը, այնպես, որ եթե U_i –ն

բաժանված է U'_e և U'_f-ի, պետք է այդ պարամետրերը թարմացնել (1.2 – 1.5) բանաձևերի օգնությամբ:



Նկ.1.7. U_1-ի բաժանումը U'_1 և U'_2-ի:

$l\_U'_e = Min\{x'_j\}, 1 \le j \le s, C'_j \subset U'_e$, x'_j-ը հանդիսանում է C'_j-ի x կոորդինատը, (1.2)

$r\_U'_e = Max\{x'_j\}, 1 \le j \le s, C'_j \subset U'_e$, x'_j -ը հանդիսանում է C'_j-ի x կոորդինատը, (1.3)

$b\_U'_e = Min\{x'_j\}, 1 \le j \le s, C'_j \subset U'_e$, y'_j -ը հանդիսանում է C'_j-ի y-ը կոորդինատը, (1.4)

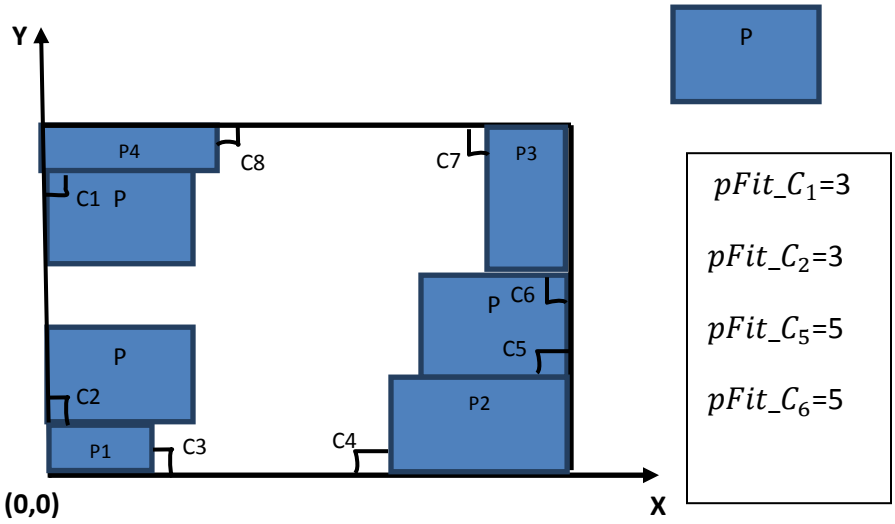
$t\_U'_e = Max\{x'_j\}, 1 \le j \le s, C'_j \subset U'_e$, y'_j -ը հանդիսանում է C'_j-ի y-ը կոորդինատը, (1.5)

որտեղ s-ը U'_e-ում գոգավոր անկյունների քանակն է: Երբ նոր մասը կփաթեթավորվի, եթե U_i-ն բաժանված չէ, U_i-ի կողմերը չպետք է փոփոխվեն:

Երբ P_i նոր կտորը փաթեթավորված է, թող s-ը լինի եզրերի թիվը, որը շոշափվել է որևէ մի C_k դիրքի որևէ P_h փաթեթավորված մասի հետ, եթե P_i-ն արդեն փաթեթավորվել է, ապա $pFit\_C_k$ պարամետրը հաշվվում է ըստ (1.6) բանաձևի.

$$pFit_{C_k} = \sum_{j=1}^{s} p_j \ : \qquad (1.6)$$

Եթե P_i-ն փաթեթավորվել է C_k-ի վրա, կտորի անկյան հետ միասին (յուրաքանչյուր U_m հարցում է), մեկ կողմով P_i-ն հպվում է U_m-ին ($p_j = 2$) և մեկ կողմով P_i-ն հպվում է այլ փաթեթավորված կտորի ($p_j = 1$), (Նկ.1.8):



Նկ. 1.8. $pFit\_C_i$-ի հաշվումը յուրաքանչյուր U_k-ում

Որոշել հեռավորությունը C_1 եզրից.

$$ed\_C_k = Min\{\ C_k \text{ գագաթի և } U_e \text{ եզրի միջև հեռավորությունը}\}, \qquad (1.7)$$

եթե $C_k \subset U_e$.

Փաթեթավորման կարգը. Երբ P_i-ն փաթեթավորված է ուղղանկյան վրա, յուրաքանչյուր C_k –ի համար հաշվել $pFit\_C_k$ և ընտրել փաթեթավորման դիրքը մեծագույն $pFit\_C_k$-ով, եթե $pFit\_C_k$ միմյանց հավասար են, հետո ընտրել ed_C_k-ով կարճագույն դիրքը, եթե ed_C_k նույնն է, ապա դիրքն ընտրել պատահական ձևով: Փաթեթավորման կանոնների

և սահմանումների կատարումից հետո, կառուցվում էվրիստիկ ալգորիթմը, որը հիմնված է գոգավոր անկյան սկզբունքի վրա՝ ըստ ալգորիթմ 1-ի:

Ալգորիթմ 1: Էվրիստիկ փաթեթավորումը (փաթեթավորման հաջորդականությունը)

$s \Leftarrow 0$;

$i \Leftarrow 0$;

while փաթեթավորման հաջորդականությունը 0-ական չէ **do**

ստանալ P_i-ն փաթեթավորման հաջորդականությունից;

օգտագործել վերը նշված փաթեթավորման կանոնը, որպեսզի ստանալ P_i-ի համար լավ դիրք;

if գոյություն ունի լավ դիրք **then**

փաթեթավորել P_i-ն լավ դիրքում;

փաթեթավորման ենթակա հաջորդականությունից հեռացնել P_i-ն;

$s \Leftarrow s + P_i$ տիրույթը;

continue;

end if

$i \Leftarrow i + 1$;

end while

return s;

Պատահական փնտրում

Քանի որ գոգավոր անկյան էվրիստիկ փաթեթավորումը կախված է փաթեթավորման հաջորդականությունից, դրա համար ներմուծվում է պատահական փնտրումը՝ լուծման որակը բարձրացնելու համար: Այն նկարագրվում է հետևյալ կերպ.

Ալգորիթմ 2: Միջին էվրիստիկա (բոլոր մասերի օրիգինալ տվյալները, maxcall)

ստեղծել փաթեթավորման հաջորդականությունը, կախված բոլոր մասերի մակերեսներից, մեծից փոքր;

$best \Leftarrow 0$;

$area \Leftarrow 0$;

(տեղադրման սահմանը) swapLimit⇐կտորների քանակը×1/3;

for i = 0 to maxcall **do**

$area =$ էվրիստիկ փաթեթավորում (փաթեթավորման հաջորդականություն);

if area = բոլոր մասերի գումարային մակերեսին **then**

break;

end if

if area > best **then**

$best \Leftarrow area$;

end if

for j = 0 to swapLimit **do**

փաթեթավորման հաջորդականության մասերի կարգը փոխել պատահական ձևով;

end for

end for

1.11. Եռաչափ փաթեթավորման խնդիների դասակարգումը

Որպես բեռներ կդիտարկենք զուգահեռանիստաձև եռաչափ օբյեկտները: Բեռները կանվանենք արկղներ, իսկ տարածքը, որտեղ նրանք փաթեթավորվում են՝ բեռնարկղ: Տարբերում են երկու տիպի արկղերի հավաքածու.

- *Համասեռ*, երբ հավաքածուում քիչ են տարբեր տիպի արկղերը, իսկ արկղերի քանակը մեծ է,
- *Տարասեռ,* երբ հավաքածուում շատ են տարբեր տիպի արկղերը, իսկ արկղերի քանակը քիչ է:

Եթե բեռնարկղերը մի քանիսն են, ապա նրանք կարող են լինել *միանման*, *համասեռ կամ տարասեռ*: Սկզբում դիտարկենք այն խնդիրները, որոնց նպատակն է բոլոր արկղերը փաթեթավորել հնարավորինս նվազագույն քանակով բեռնարկղերում:

Այսպիսով, համադրելով տարբեր տիպի արկղերը և բեռնարկղերը, կառանձնացնենք փաթեթավորման հիմնական 6 տիպի խնդիրները՝ համասեռ կամ տարասեռ:

1. SSSCSP (Single stock-size cutting stock problem) –եթե բեռնարկղերը միանման են, իսկ արկղերը՝ համասեռ,
2. SBSBPP (Single bin-size bin packing problem)- եթե բեռնարկղերը միանման են, իսկ արկղերը՝ տարասեռ,
3. MSSCSP(Multiple stock-size cutting stock problem) – եթե բեռնարկղերը և արկղերը համասեռ են,
4. MBSBPP (Multiple bin-size bin packing problem)- եթե բեռնարկղերը համասեռ են, իսկ արկղերը՝ տարասեռ,
5. RCSP (Residual cutting stock problem) –եթե բեռնարկղերը տարասեռ են, իսկ արկղերը՝ համասեռ,
6. RBPP (Residual bin packing problem) – եթե բեռնարկղերը և արկղերը տարասեռ են:

Երբ անհրաժեշտ է բոլոր արկղերով լցնել ըստ ներկայացված բեռնարկղերի մեծագույն ծավալով հավաքածուն, տարբերում են 7 տարբեր տիպի խնդիրներ.

1. IIPP (dentical item packing problem) –եթե բեռնարկղը միակն է, և արկղերը միանման են,
2. SLOPP (Single large object placement problem) – եթե բեռնարկղը միակն է, և արկղերը համասեռ են,
3. SKP (Single knapsack problem) – եթե բեռնարկղը միակն է, և արկղերը տարասեռ են,
4. MILOPP (Multiple identical large object placement problem) –եթե միանման բեռնարկղերը մի քանիսն են, և արկղերը համասեռ են,
5. MHLOPP (Multiple heterogeneous large object placement problem) –եթե բեռնարկղերը համասեռ են կամ տարասեռ, իսկ արկղերը համասեռ,
6. MIKP (Multiple identical knapsack problem) –եթե միանման բեռնարկղերը մի քանիսն են, իսկ արկղերը տարասեռ,

7. MHKP (Multiple heterogeneous knapsack problem) –եթե բեռնարկղերը համասեռ կամ տարասեռ են, իսկ արկղերը տարասեռ:

Ենթադրվում է, որ բոլոր թվարկված խնդիրներում բեռնարկղերի չափերը ֆիքսված են: Սակայն, եթե բեռնարկղի երկու չափերը ֆիքսված են, իսկ երկարությունը և բարձրությունը փոփոխվում են, ապա այդ խնդիրը SCIMP (single container input minimization problem)-ն է:

Խնդրի սահմանափակումները

Բոլոր դիտարկված խնդիրներում առկա են ստանդարտ, ընդհանուր սահմանափակումներ.

- արկղերը կարող են տեղադրվել միայն այնպիսի եղանակով, որպեսզի նրանց եզրերը լինեն զուգահեռ բեռնարկղի պատերին,
- բոլոր արկղերը պետք է լինեն փաթեթավորված բեռնարկղի ներսում,
- արկղերը չպետք է հատվեն միմյանց հետ:

Որոշ խնդիրներում արկղերը պետք է սերտորեն հպվեն իրար, իսկ մյուսներում՝ պետք է թույլ տրվեն ոչ մեծ արանքներ: Եթե անհրաժեշտ է հաշվի առնել օբյեկտների ծանրության կենտրոնը, ապա ենթադրվում է, որ այն համընկնում է օբյեկտի երկրաչափական կենտրոնի հետ:

Տարածության մեջ հնարավոր է արկղերի 6 տարբեր դիրքեր կոորդինատային առանցքների նկատմամբ [40, 49, 98]: Հաճախ խնդիրներ ունեն արգելքներ արկղերի որոշակի դիրքերի կամ տրված առանցքների նկատմամբ պտույտի համար [55, 28]: Որոշ խնդիրներում արկղերի կամայական պտույտ արգելված է [94, 85]:

Փաթեթավորման կայունությունը հանդիսանում է կարևոր պայման խնդրի հաջող լուծման համար: Ոչ կայուն փաթեթավորումը կարող է հանգեցնել բեռի վնասման կամ դժվարությունների բեռնման կամ բեռնաթափման ժամանակ: Փաթեթավորման

կայունության ամենահարմար և հուսալի եղանակն այն է, որ բեռնարկղի լցման ժամանակ յուրաքանչյուր արկղ լրիվությամբ գտնվի հարթ մակերևույթի վրա [49]: Այլ հեղինակներ առաջարկում են թուլացնել նշված սահմանափակումները և թույլ են տալիս մասնակի արանքներ թույլատրելի մեծագույն արժեքով [89, 90]: Որոշ աշխատանքներում օգտագործում են մոտեցում, համաձայն որի յուրաքանչյուր փաթեթավորված արկղի գոնե մեկ կողմը պետք է հպվի մյուս արկղերի հետ [19]: Հարթեցման սահմանափակումը ենթադրում է, թե ինչպես պետք է արկղերը դրվեն մեկը մյուսի վրա՝ հաշվի առնելով նրանց կրելու կարողությունը: Յուրաքանչյուր արկղ ունի մաքսիմալ կշիռ, որին այն կարող է դիմակայել՝ կախված արկղի դիրքից:

Բեռների կշիռն առանձին և կշիռների տեղաբաշխումը բեռնարկղում հանդիսանում են կարևոր սահմանափակումներ արդեն փաթեթավորված բեռնարկղում բեռնման կամ տեղաշարժման դեպքում: Կշռի իդեալական բաշխումն այն դեպքում է, երբ բեռնարկղի ծանրության կենտրոնը համընկնում է հատակի երկրաչափական կենտրոնի հետ: Տրանսպորտի բեռնման ժամանակ պետք է հաշվի առնել կշռային բեռնվածքը տրանսպորտային միջոցի առանցքի վրա և բեռնարկղի մաքսիմալ թույլատրելի գումարային կշիռը [19, 71, 93]:

Բոլոր դիտարկվածխ սահմանափակումները, որոնք անմիջականորեն ազդում են բեռնարկղի փաթեթավորման վրա, պետք է միավորվեն դրված խնդրի առավել արդյունավետ լուծման էվրիստիկայով: [42]-ում ներմուծված են ևս երկու սահմանափակումներ, որոնք ազդում են տարբեր բեռնարկղերում արկղերի փաթեթավորման վրա.

- բեռների առանձնացում - երբ անհրաժեշտ է տեղափոխել մեկ անգամ, օրինակ, սնունդը և օծանելիքը, դրա համար նրանց պետք է առանձնացնել տարբեր բեռնարկղերում,
- փաթեթավորման լրիվությունը-երբ փաթեթավորման մեջ պետք է ներառվեն նաև բեռներ, որոնք պատկանում են որոշակի ենթախմբի:

1.12. Դիրքի էվրիստիկան

Դրված խնդրի կամայական մոտարկված լուծման համար պետք է տրվի մեխանիզմ, որը թույլ կտա որոշել ինչպես տեղաբաշխել արկղերը բեռնարկղերում, անհրաժեշտ է արդյոք գեներացնել սկզբնական մոտարկումը կամ ընթացիկ դիրքը արդյոք օպտիմալ լուծման մոտարկումն է: Նշված մեխանիզմն անվանում են դիրքի էվրիստիկա: Բեռնարկղի բեռնման համար գոյություն ունի նմանատիպ բազմաթիվ էվրիստիկաներ: Եթե արկղերի հավաքածուն համասեռ է, ապա հաճախ օգտագործում են «պատի կառուցումը» կամ «շերտի կառուցումը»: Բեռներով տարասեռ արկղերի դեպքում նպատակահարմար է արկղերը տեղադրել մեկ առ մեկ:

Պատի կառուցումը

Նշված էվրիստիկայի սկզբնական տարբերակը լցնում է բեռնարկղը արկղերի աշտարակների հավաքածուով (պատերով) բեռնարկղի ամբողջ խորությամբ: Աշտարակները կառուցվում են հաջորդաբար, դրա համար էլ կշռի բաշխման խնդիրները լուծվում են փաթեթավորման փոխանակմամբ, հարթագայությամբ կամ հայելային արտապատկերմամբ, որպեսզի արկղերը չհատվեն միմյանց հետ [52, 18, 86, 30, 91, 50]:

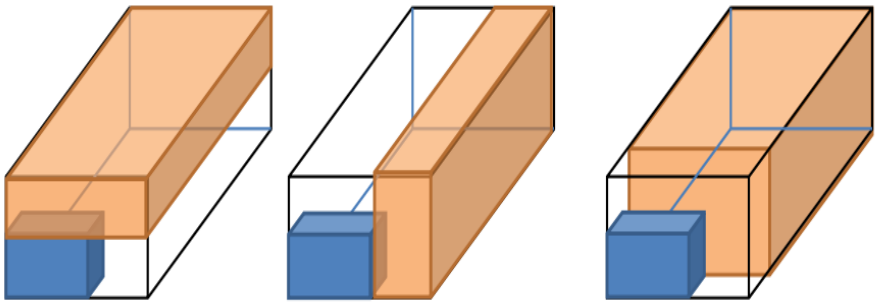
- Կիրառվում են արկղի խորությամբ խորանարդաձև մոդելներ, որոնք առավել արդյունավետ են, քան պատերը: Առաջին հերթին, խորանարդը կազմված է միանման արկղերի բազմությունից և կարող են միանալ ըստ բարձրության կամ լայնության: Երկրորդ հերթին, երկու այդպիսի խորանարդները տեղադրվում են մեկը մեկի հետևից կամ մեկը մյուսի դիմաց: Բոլոր ազատ տեղերը պահպանվում են ցուցակում, իսկ նվազագույն ծավալով տարածությունը անընդհատ լցվում է առաջինը [22, 23]: Բացի

այդ օգտագործելով երկու այլընտրանքային կանոններ փաթեթավորման տարածքում տեղային դիրքի ստացման համար.

- տեղադրված արկղերի ընդհանուր ծավալը պետք է լինի մաքսիմալ,
- երկակի չափանիշների օգտագործումը՝ հնարավոր ծավալի նվազագույն կորուստ և հնարավոր արդյունավետ մաքսիմալ ծավալ:

Շերտի կառուցումը

Շերտի կառուցման էվրիստիկան հանդիսանում է տեղադրման բազային էվրիստիկաներից մեկը, սակայն որոշ հետազոտողներ կատարելագործել են այս մոտեցումը: Փաթեթավորումը տեղի է ունենում հետևյալ կերպ. արկղերն ըստ հերթականության տեղադրվում են բեռնարկղի հատակին՝ ստեղծելով բազային շերտ: Հենց այդ շերտը լցվում է, նոր շերտը ձևավորվում է բազային շերտի վրա, և նոր շերտերի ստեղծումը կատարվում է այնքան ժամանակ, քանի դեռ բեռնարկղի մնացած ազատ բարձրությունում արդեն ոչ մի շերտ չի տեղավորվի (Նկ.1.9):



Նկ.1.9. Շերտի կառուցումը:

Կան աշխատանքներ, որոնցում քննարկվում են հետևյալ հարցերը՝

- Ներմուծվում է որոշակի գործակից, որը յուրաքանչյուր իտերացիայից հետո գնահատում է ստացված փաթեթավորման «անհարմարությունները», որի արդյունքում «անհարմար» արկղերը ավելի վաղ կփաթեթավորվեն [74]:
- Համասեռ արկղերը խմբավորվում են ըստ բարձրության, որից հետո խմբերը դասակարգվում են ըստ բարձրության նվազման, յուրաքանչյուր խմբի ներսում արկղերը դասավորվում են ըստ հիմքի մակերեսի նվազման: Արկղերը փաթեթավորվում են բեռնարկղի վերջից՝ ըստ հերթականության, սկսելով նրանցից, որոնք ավելի մեծ տարածք են զբաղեցնում [92, 81, 79]:

Դիրքի այլ էվրիստիկաներ

Ենթադրենք տրված են համասեռ բլոկներ, որոնք կազմված են որոշակի տիպի արկղերից: Բեռնարկղի փաթեթավորումը այդպիսի բլոկներով ապահովում է մի քանի առավելություններ.

- Այսպիսի սխեման հեշտ է կազմակերպել և այսպիսի բեռնման համար պահանջվում է քիչ ժամանակ:
- Բեռնումից հետո բեռի վերադասավորման անհրաժեշտություն չկա, քանի որ միանման արկղերը փաթեթավորվում են միմյանց շատ մոտ:
- Արկղերի բեռնատարողության խնդիրն այստեղ ավելի քիչ բարդ է, քանի որ տեղադրվում են միանման արկղեր:
- Բլոկային կառուցվածքը ապահովում է փաթեթավորման լրացուցիչ կայունություն, շնորհիվ այն բանի, որ միանման արկղերը, որոնք տեղադրված են ուղղանկյունաձև, չեն կարող հեշտությամբ սահել:

Հայտնի է նաև հիբրիդային մոտեցումը, որի դեպքում բեռների տիպերը դասավորվում են ըստ ծավալի նվազման՝ սկզբնական լուծման ստացման համար: Միատիպ արկղերից

կազմում են բլոկներ այնպես, որ առավել արդյունավետ լցնեն փաթեթավորման համար հասանելի տեղը [76, 93]: Կիրառվում է նաև բլոկի ընտրության բազմաշերտային մոտեցումը [103], նաև արկղերի դասավորման «գոգավոր» անկյան մոտեցումը և մի շարք այլ նմանատիպ մեթոդներ [12, 17, 29, 31, 34, 43, 51, 54, 56-58, 61, 62, 70, 73, 75, 83, 87, 105, 106]:

1.13. Փաթեթավորման հաջորդականությունը

Արկղերի դասակարգումը, նախքան փաթեթավորումը, մեծապես ազդում է փաթեթավորման այս կամ այն էվրիստիկայի արդյունավետության վրա: Փաթեթավորման դասակարգումը բաժանվում է երկու խմբի՝

- Ստատիկ,
- Դինամիկ:

Ստատիկ դասակարգումը ենթադրում է արկղերի կամ արկղերի տիպերի ֆիքսված դասակարգում մինչ փաթեթավորումը: Դինամիկ դասակարգման ժամանակ օգտագործվում են չափանիշներ, որոնք որոշում են հաջորդ արկղը կամ փաթեթավորման տեղը փաթեթավորման գործընթացի ժամանակ [102]:

1.14. Էվրիստիկաների լավացումը

Էվրիստիկաների օգտագործումը թույլ է տալիս արագ և որակյալ լուծման ստացում, բացի այդ նրանց լավացումը բերում է նշանակալի առավելության: Ընդհանուր առմամբ, լավացումը բերում է մի քանի լրիվ լուծումների փնտրման գործընթացին, կատարվում են հարևան քայլեր օպտիմալ լուծման փնտրման համար: Նշված գործընթացը փոփոխական է և բարդ՝ անցում կատարելով պարզ կառուցվածքների և չափանիշների ընտրությունից մինչև բարդ տարածքները և չափանիշները, որոնց առկայության դեպքում միայն լավացվում են լուծմանը հասնելու

քայլերը՝ թույլ տալով գտնել բազմաթիվ լոկալ օպտիմումներ՝ ներառելով ստանդարտ մետաէվրիստիկաների շատ իրականացումներ: Այդպիսի ալգորիթմներից են՝ գենետիկական ալգորիթմը (GA), TS և SA երեք մետաէվրիստիկաների ստանդարտ հավաքածուները, GLS, GRASP (greedy randomized adaptive search procedure) մեթոդները և այլն [9, 20, 26, 27, 35, 38, 44-48, 53, 59, 63, 64, 82, 84, 88, 95, 96, 101, 104,]:

1.15. Գործող համակարգերի վերլուծություն

Այս ատենախոսության շրջանակներում կատարվել է գոյություն ունեցող հասանելի ծրագրային համակարգերի վերլուծություն: Մշակված են բավականին մեծ քանակի կտրման օպտիմալացման ծրագրեր: Ստորև թվարկված է դրանց մի մասը.

Раскрой - Ծրագիրը նախատեսված է հարթ և գալարաձև նյութերի կտրման օպտիմալ քարտեզի կազմման համար: Թույլ է տալիս արդյունավետ և արագ ստանալ կտրման քարտեզներ, որոնք ապահովում են արտադրանքի մեծ տոկոս:

Астра Раскрой - Ծրագիրը նախատեսված է հարթ նյութերի կտրման օպտիմալացման համար: Աստրա Կտրումն ապահովում է նյութերի և պատվերների մասին մուտքային արագ ինֆորմացիա, կտրման քարտեզի ավտոմատ և մեխանիկական ձևավորում, մնացորդային ծավալների և հաջորդ պատվերներում նրանց կտրման լրիվ հաշվարկ, կտրման քարտեզների և առանձնահատկությունների տպում:

Базис-Раскрой 2.0- ավտոմատացված ծրագիր է հարթ կտրման քարտեզների ստեղծման համար, որն իր մեջ ներառում է տրված չափումներով սկզբնական նյութի ուղղանկյունաձև դետալների ուրվագծերի տեղադրման բարձր արագության հաշվման օպտիմալություն: Այն իրենից ներկայացնում է Базис-Конструктор-Мебельщик համալիրի մասերից մեկը:

Быстрый линейный раскрой - Այս ծրագիրը նախատեսված է գծային նախապատրաստուկներից կազմված դետալների կտրման օպտիմալացման համար: Կիրառման ոլորտը. արտադրության համար պահանջվող քանակությամբ դետալների արտադրության համար անհրաժեշտ քանակությամբ նախապատրաստուկների հաշվարկը: Սկզբնական տվյալներ. նախապատրաստուկների չափը, տեխնոլոգիական պարամետրերը, անհրաժեշտ դետալների չափը և քանակը: Աշխատանքի արդյունքը. Նախապատրաստուկների անհրաժեշտ քանակը, նախապատրաստուկների վրա դետալների տեղադրման քարտեզի օպտիմալության մաքսիմալ մոտարկումը:

Оптимальный раскрой линейных листов – Ծրագիրը նախատեսված է ուղղանկյուն դետալի վրա ուղղանկյուն շերտերի կտրման համար: Ծրագիրը կարող է օգտագործվել փայտամշակման և կահույքի արտադրության, մետաղական հատումների, ապակու կտրման համար և այլն: Ծրագրի հիմքում դրված է բացառիկ, բարձր արագագործությամբ ալգորիթմ, որը թույլ է տալիս անցկացնել արագ կտրում՝ նվազագույն թափոններով:

Оптимальный раскрой линейных отрезков - Ծրագիրը նախատեսված է տարբեր երկարությամբ գծային հատվածների վրա գծային նախապատրաստուկների օպտիմալ կտրման համար և կարող է օգտագործվել փայտամշակման և թղթի արդյունաբերության, մետաղական վերամշակման, կարի արտադրության և այլնի համար:

PaneCut - Ծրագիրը նախատեսված է հարթ հումքի կտրման օպտիմալացման համար, որը թույլ է տալիս նվազեցնել օգտագործվող նյութերի թափոնների տոկոսը:

Cutting – Ծրագիրը նախատեսված է ուղղանկյուն հումքից ուղղանկյուն դետալների ձևման համար: Այն կարող է օգտագործվել փայտամշակման, ապակու կտրման, մետաղի և այլ արտադրություններում: Ծրագրի հիմքում դրված է արագագործ ալգորիթմ, որը իրականացնում է ձևումը փոքրագույն թափոններով:

Look - Ծրագիրը նախատեսված է հարթ հումքի կտրման քարտեզների ստեղծման և օպտիմալացման համար: Ծրագրում ապահովվում է պատվերների և նյութերի մասին ինֆորմացիայի արագ մուտք, կտրման քարտեզի ավտոմատ և մեխանիկական ձևավորում, բոլոր կտրման պարամետրերի ավտոմատ հաշվում, կտրման քարտեզի և առանձնահատկությունների տպում:

Optimum – Հումքի օպտիմալ կտրման ծրագիր է ուղղանկյուն դետալների:

БАЗИС-Мебельщик –Փոխկապակցված ծրագրերի համալիր է, որը թույլ է տալիս կամայական ձեռնարկությունում արդյունավետ ձևով կազմակերպել կահույքի անհատական կամ զանգվածային արտադրություն՝ սկսած կահույքի հսկաներից մինչև անհատական ձեռնարկատերերը: Համակարգի կազմությունն է՝ Базис-Мебельщик, Базис-Раскрой, Базис-Смета, Базис-Интерьер.

SDCC – սալիկների կտրման օպտիմալացում - սալիկների կտրման քարտեզների նախագծման ավտոմատ համակարգը նախատեսված է կահույքի, պատուհանների և այլ արտադրանքների արտադրությամբ զբաղվող ձեռնարկություններում կտրման քարտեզների ստեղծման գործընթացի ավտոմատացման համար, որոնցում կիրառվում են ուղղանկյուն դետալներ շերտավոր նյութերից:

SDCC Lite – սալիկների կտրման քարտեզների ավտոմատ նախագծման համակարգ է, որը նախատեսված է կահույքի, պատուհանների և այլ արտադրանքների արտադրությամբ զբաղվող ձեռնարկություններում կտրման քարտեզների ստեղծման գործընթացի ավտոմատացման համար, որոնցում կիրառվում են ուղղանկյուն դետալներ շերտավոր նյութերից:

Cutting Optimization PRO – համակարգչային ծրագիր է, որն օգտագործվում է դետալների օպտիմալ տեղադրման ստացման համար՝ միաչափ (1D) և երկչափ (2D) կտրման դեպքում: Ծրագիրը նաև թույլ է տալիս որոշել և օգտագործել սեփական օբյեկտները, ինչպես առանձին դետալներ, այնպես էլ պատրաստի արտադրանքներ.

սեղաններ, խոհանոցային պահարաններ և այլն: Կարող է նաև օգտագործվել փայտից, ապակուց, մետաղից կամ այլ հումքից ուղղանկյունների կտրման համար, որոնք օգտագործվում են արտադրությունում: Բացի այդ կարող է օգտագործվել միաչափ հումքի ձևման համար, ինչպիսիք են՝ ձողերը, խողովակները, մալուխները, պողպատե սալիկները, մետաղական պրոֆիլները, տախտակները և այլն:

Այս ցանկը դեռ կարելի է շարունակել, տարբեր համակարգեր ստեղծված են տարբեր նպատակների համար, կիրառելի են տարբեր արտադրություններում, սակայն կամայական ուրվագծով դետալների հետ աշխատող համակարգեր գտնել չհաջողվեց:

1.16. Եզրակացություններ

Այս գլխում փորձ կատարվեց վերլուծել ձևման և փաթեթավորման խնդրի առավել հայտնի մոդելները, դասակարգել դրանք: Իհարկե, բոլոր հնարավոր մասնավոր դեպքերին հնարավոր չէ անդրադառնալ, սակայն նույնիսկ այս կարճ վերլուծությունը ցույց է տալիս խնդիրների բազմազանությունը, դրանց կիրառական նշանակությունը և առաջարկվող լուծումների բարդությունը: Չնայած գիտական հրապարակումների մեծ թվին, գոյություն ունեցող համակարգերին, մնում են մի շարք բաց խնդիրներ, որոնցից մեկը կամայական ուրվագծով երկչափ մարմինների ձևման խնդիրն է: Այս աշխատանքի հաջորդ գլուխները նվիրված են այս պրոբլեմի հետազոտմանը:

ԳԼՈՒԽ 2.

ԿԱՄԱՅԱԿԱՆ ՈՒՐՎԱԳԾՈՎ ՕԲՅԵԿՏՆԵՐԻ ԵՐԿՉԱՓ ՄՈԴԵԼԱՎՈՐՈՒՄԸ

Այս գլխում նկարագրվում է կամայական ուրվագծով երկչափ օբյեկտների մոդելավորման համար առաջարկված մոտեցումը և մշակված ծրագիրը: Այն ներառում է գործիքներ պարզ երկրաչափական օբյեկտների ստեղծման համար, դրանց հետ տարբեր գործողություններ կատարելու համար, ինչպես նաև բարդ ուրվագծեր ստեղծելու համար:

2.1. Մոդելավորման նպատակը

Երկչափ մոդելավորման խնդիրը հանդիսանում է տրված մակերեսի վրա կամայական ուրվագծով դետալների օպտիմալ տեղաբաշխման հասանելու առաջին քայլը: Ըստ էության, գոյություն ունեն որոշ ծրագրեր, որոնք լուծում են նշված պրոբլեմը [3], բայց նրանք չեն աշխատում կամայական ուրվագծով օբյեկտների հետ: Նրանցից որոշներն աշխատում են միայն ուղղանկյունների հետ [15, 97], մյուսները մասնավոր են կամ շատ թանկ արժեն:

Օբյեկտների նկարագրությունը, որպես օրենք, սկսվում է թվային մոդելի մշակմամբ: Այդ նպատակի իրականացման համար մշակվել է ծրագրային միջավայր երկչափ թվային մոդելներ ստեղծելու հնարավորությամբ:

Նշված գործիքի ինտերֆեյսը նման է արդեն գոյություն ունեցող եռաչափ խմբագրիչներին, ինչպիսիք են՝ AutoCAD, 3ds Max և այլն: Սակայն գոյություն ունեցող այժմյան միջոցները նախատեսված են եռաչափ օբյեկտների համար և պարունակում են չկիրառվող գործիքների մեծ հավաքածու, ինչը դժվարեցնում է օգտագործումը: Միևնույն ժամանակ երկչափ խմբագրիչի ստեղծման գաղափարը թույլ կտա պարզեցնել

և արագացնել այդ գործընթացը մի քանի անգամ և կպահանջի ավելի պակաս արտադրական պոտենցիալ:

Կամայական ուրվագծով օբյեկտների ստեղծման հիմնական սկզբունքը տարրական երկրաչափական պատկերների հետ գործողություններ կատարելն է, քանի որ յուրաքանչյուր բարդ պատկեր բաղկացած է տարրական պատկերների որևէ քանակից: Այդ պատկերները կարող են մշակվել ձեռքով, մկնիկի կամ կոորդինատների օգնությամբ:

Նշված գործընթացը նկարագրված է հետևյալ ձևով.

- տարրական երկրաչափական պատկերների մոդելավորման նկարագրությունը,
- ինչպես ղեկավարվել գոյություն ունեցող պատկերներով,
- տարրական պատկերներից բարդ պատկերի ստեղծման նկարագրությունը:

2.2. Տարրական երկրաչափական մարմինների մոդելավորման նկարագրությունը

Երկրաչափական պատկերներ կամ մարմիններ ստեղծելու համար ծրագրային միջավայրում մշակվել են հատուկ գործիքներ, որոնց միջոցով կարելի է կառուցել հիմնական երկրաչափական մարմինները՝ ուղղանկյուն, էլիպս, բեկյալ, ընդ որում նշված 3 տարրերից հնարավոր է պատրաստել մնացյալ բոլոր մարմինները՝ քառակուսի, շրջան և այլն:

Տարրական պատկերներից ավելի բարդ մարմիններ ստանալու համար պետք է նրանց ձևափոխել, այսինքն տեղաշարժել, պտտել, փոփոխել չափսերը: Բարդ օբյեկտներ կառուցելու համար ծրագիրը պետք է հիշի բոլոր այն պարզ երկրաչափական օբյեկտները, որոնցից բաղկացած է բարդ օբյեկտը, ձևափոխի դրանք տված ճանապարհով:

Վերը նշված տարրերից յուրաքանչյուրի համար առանձնացված են գործիքամիջոցներ, նրանցից յուրաքանչյուրն ունի աշխատանքի մի քանի ռեժիմ, դրանք կօգնեն, որպեսզի ծրագիրը լինի առավել ճկուն, իսկ մարմինների կառուցումը լինի ավելի արագ և առանց խոչընդոտների ու մոդելավորման խորը գիտելիքների տիրապետման: Ստորև տրված են տվյալ գործիքների և նրանց աշխատանքային ռեժիմների ցուցակը:

- **Ուղղանկյուն (Rectangle)**

- *Տալ անկյունագծային կետերը*: Դա կարելի է անել կամ մկնիկի միջոցով կամ տալ այդ երկու կետերի x և y կոորդինատները՝ հատուկ դրա համար առանձնացված դաշտերում, հնարավոր է նաև կոմբինացված տարբերակ՝ այսինքն առաջին կետը տալ մկնիկով, իսկ երկրորդը ներմուծել ստեղնաշարից:
- *Տալ ուղղանկյան անկյուններից մեկի կետը և ուղղանկյան բարձրությունն ու լայնությունը*: Սա նույնպես հնարավոր է կատարել ինչպես մկնիկի, այնպես էլ ստեղնաշարի միջոցով:

- **Էլիպս (Ellipse)**

- *Տալ էլիպսին արտագծած ուղղանկյան երկու հակադիր կետերը*: նման է ուղղանկյան առաջին կետին:
- *Տալ շրջանագծի կենտրոնը և շառավիղը*: Հասկանալի է, որ այս դեպքում ունենում ենք էլիպսի մասնավոր դեպքը՝ շրջանագիծը: Այստեղ նույնպես հնարավոր է տվյալների ներմուծման երկու եղանակ:

- **Բեկյալ (Polyline)**

- *Տալ անկյունային կետերի կոորդինատները*: Վերջին կետի կոորդինատները տալուց հետո ծրագիրն ավտոմատ փակում է շղթան՝ կապելով իրար հետ առաջին և վերջին կետերը:

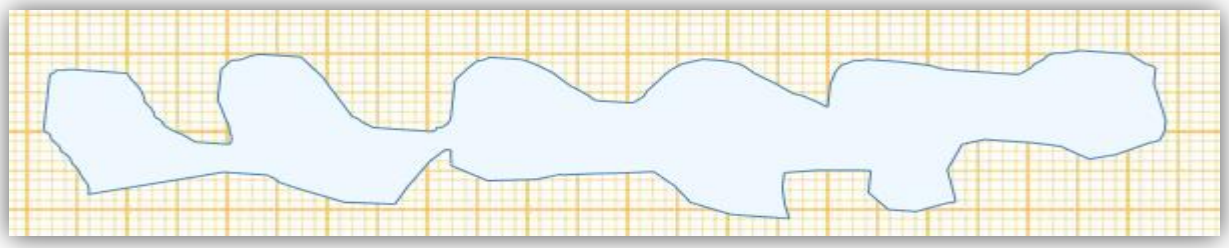
- **Ուրվագիծ (Conture)**

- Սա գործնականում իրենից ներկայացնում է բեկյալի մասնավոր դեպք, սակայն այն տարբերությամբ, որ այստեղ կետերը գտնվում են իրար շատ մոտ: Այս դեպքը ներմուծված է *անկանոն գծերի և կորերի* հետ աշխատանքը մատչելի դարձնելու համար: Հարմար է տալ մկնիկի միջոցով, այն անընդհատ ընդունում է կետեր և ստեղծում է շատ անկյուններով բեկյալներ: Նշված գործիքը հարմար է օգտագործել, եթե շատ բարձր կարգի ճշտության անհարժեշտություն չկա (տես Նկ. 2.1.):

- **Կետ (Point)**

- Մկնիկով ընտրել այն կետը, որը հարկավոր է առանձնացնել, կամ համապատասխան դաշտերում տալ կետի երկու կոորդինատները: Նշվածն անկախ օբյեկտ չէ, այն կօգնի օգտագործել վերը նշված բոլոր գործիքները:

Նկ. 2.1. Ծրագրի միջոցով ստեղծված օբյեկտ



2.3. Գործողություններ ստեղծված մարմինների հետ

Օբյեկտների **շարժումը** տեղի է ունենում երկրաչափական կենտրոնի նկատմամբ, իսկ այդ կենտրոնը որոշվում է հետևյալ կերպ՝ ուղղակի նրան արտագծած ուղղանկյան վերին ձախ անկյունի կոորդինատներին գումարում ենք նրա բարձրության և լայնության կեսը, և քանի որ մեր դեպքում որպես սկզբնակետ ընտրված է վերին ձախ անկյունը, ապա կունենանք հետևյալ տեսքը՝

$$x = TopLeft.x + Width/2\,;$$

$$y = TopLeft.y + Height/2\,; \tag{2.1}$$

որտեղ $TopLeft.x$ և $TopLeft.y$ ուղղանկյան վերին ձախ անկյան x և y կոորդինատներն են: Տեղաշարժելուց հետո նոր կետի կոորդինատները կորոշվեն հետևյալ բանաձևով՝

$$x' = x + d_x;$$

$$y' = y + d_y;$$

(2.2)

որտեղ x' -ը և y' -ը կետի նոր կոորդինատներն են, իսկ d_x և d_y տեղաշարժման արժեքներն են, որոնք որոշվում են մկնիկի ընթացիկ կոորդինատներից հանելով սկզբնական կոորդինատները:

Պտտման ժամանակ որպես առանցք ընտրվում է կենտրոնը, կամ այն կետը, որի կոորդինատները մուտքագրվել են օգտագործողի կողմից: Պտտումը կատարվում է կամ նախապես տրված անկյան չափով կամ մկնիկի անիվի օգնությամբ կամ տրված քանակով: Նոր կոորդինատները որոշվում են հետևյալ բանաձևերով՝

$$x' = x + \tan(A) - (y - Ck_x);$$

$$y' = y + \tan(A) - (x - Ck_y); \tag{2.3}$$

որտեղ A-ն պտտման անկյունն է, Ck_x և Ck_y համապատասխանաբար կենտրոնական կետի x և y կոորդինատներն են:

Հնարավոր է նաև ներմուծել **չափի փոփոխություն**, դրա համար կրկին անհրաժեշտ են կենտրոնական կետի կոորդինատները: Չևափոխությունը կկատարվի հետևյալ բանաձևերի օգնությամբ.

$$x' = S_x * x + (CS_x - S_x * CS_x);$$

$$y' = S_y * y + (CS_x - S_y * CS_y); \quad (2.4)$$

որտեղ S_x և S_y հորիզոնական և ուղղահայաց մասշտաբայնության գործակիցներն են, CS_x և CS_y կենտրոնական կետի կոորդինատներն են:

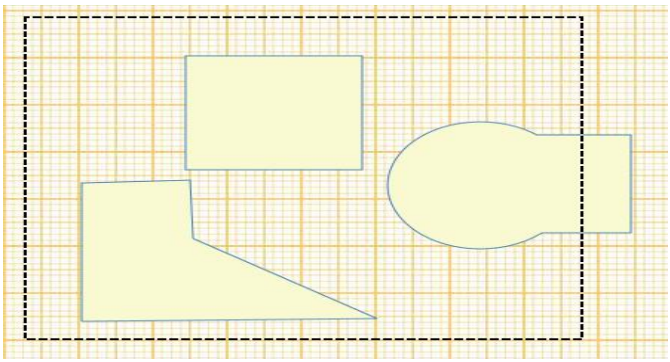
2.4. Գործողությունների գործիքամիջոցները

Պատկերների հետ գործողություններ իրականացնելու նպատակով առկա են հետևյալ գործիքամիջոցները՝ իրենց ռեժիմներով:

- **Տեղաշարժ** (**Move**)

- Մարմինը տեղաշարժելու համար ընտրում ենք այն՝ աջ հարված անելով նրա վրա: Երբ մարմինն ընտրված է, այն ստանում է ավելի մուգ գույն, այնուհետև, եթե ցանկանում ենք այն տեղաշարժել, ապա աջ սեղմակը սեղմած պահելով մկնիկը սահեցնում ենք: Երբ պատկերը հասնում է նախապես պլանավորած դիրքին, կարելի է թողնել աջ սեղմակը և պատկերը կվերադառնա իր նախկին գույնին:
- Վերը նշված կարգով ընտրում ենք մարմինը և տալիս ենք այն վեկտորը, որով այն պետք է տեղաշարժվի, այսինքն պետք է տալ միայն այդ վեկտորի վերջնական կոորդինատները: Օրինակ, եթե տրվի 10 և 15, ապա պատկերը 10 միավորով կշարժվի աջ և 15 միավորով ներքև: Կարելի է տալ միայն այդ թվերից մեկը, լռելյայն վերցվում է այն արժեքը, որը հավասար չէ 0-ի, այսինքն պատկերը կտեղաշարժվի միայն մի ուղղությամբ՝ ուղիղ գծով:
- Այս երկու տարբերակներն էլ կարելի է իրագործել ինչպես յուրաքանչյուր պատկերի համար առանձին – առանձին, այնպես էլ մի քանիսի հետ միանգամից, դրա համար անհրաժեշտ է սեղմել **Multi** կոճակի վրա, դրանից հետո կարելի է շրջանակի մեջ առնել

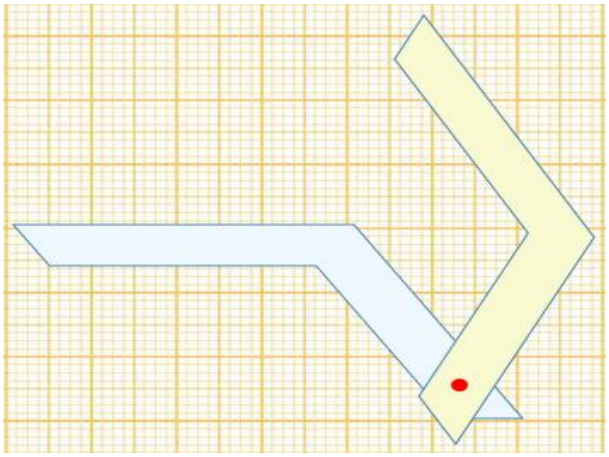
միանգամից մի խումբ պատկերներ և տեղաշարժել դրանք վերը նկարագրված տարբերակներից մեկով (Նկ. 2.2):



Նկ 2.2. Մի խումբ պատկերնեի տեղաշարժ՝ կետագծերով ներկայացված է ընդգրկող շրջանակը:

- **Պտտում** (**Rotate**)

- Առաջին և պրակտիկ լուծումը կայանում է հետևյալում՝ նախ աջ սեղմակով ընտրում ենք պատկերը, այն մգանում է, մյուս քայլով տալիս ենք այն կետը, որի նկատմամբ պետք է պտտվի պատկերը՝ այսինքն պտտման առանցքը, դա պատկերվում է որպես կարմիր կետ, այնուհետև պտտելով մկնիկի անիվը, պատկերը սկսում է պտտվել 1 աստիճանով: Եթե ցանկանում ենք պտտել ավելի արագ, ապա նախ սեղմում ենք այդ անիվը, որից հետո միայն պտտում այն:
- Պտտման համար անհրաժեշտ տվյալները ներմուծում ենք ստեղնաշարից: Դրանք են՝ պտտման առանցքի կոորդինատները և պտտման անկյունը: Դրական անկյունները պտտում են ժամսլաքի ուղղությամբ, իսկ բացասականները՝ ընդհակառակը (Նկ 2.3):

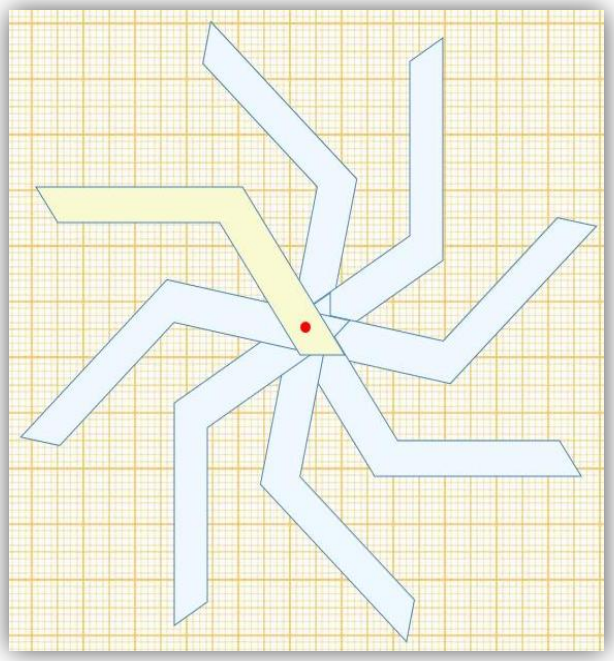


Նկ. 2.3. Անկյունների պտտումը ժամսլաքի և հակառակ ուղղություններով:

- Կարելի է օգտագործել կոմբինացված եղանակ, այսինքն տվյալների մի մասը տալ ստեղնաշարից, իսկ մի մասն էլ մկնիկի օգնությամբ: Կարելի է նաև օգտվել «որպես առանցք օգտագործել կենտրոնը» կոճակից:
- Հնարավոր է նաև դեպք, երբ օրինակ անհրաժեշտ է ստեղծել պատկերի պատճենները, որոնք կլինեն հավասարաչափ պտտված ինչ որ առանցքի շուրջը: Դրա համար գոյություն ունի «ստեղծել օբյեկտի պատճենները, հավասարաչափ պտտելով առանցքի շուրջ» կոճակը: Այս դեպքում բացի պտտման առանցքից պետք է տալ նաև պատճենների քանակը, իսկ նրանց պտտման անկյունը ավտոմատ կորոշվի (2.5) բանաձևով,

$$\alpha = \frac{i * (359 + c)}{\mathrm{c}} \qquad (2.5)$$

որտեղ α – ն պտտման անկյունն է յուրաքանչյուր պատճենի համար, i – ն ընթացիկ պատճենի համարն է, իսկ c–ն ընդհանուր պատճենների քանակը (Նկ. 2.4):

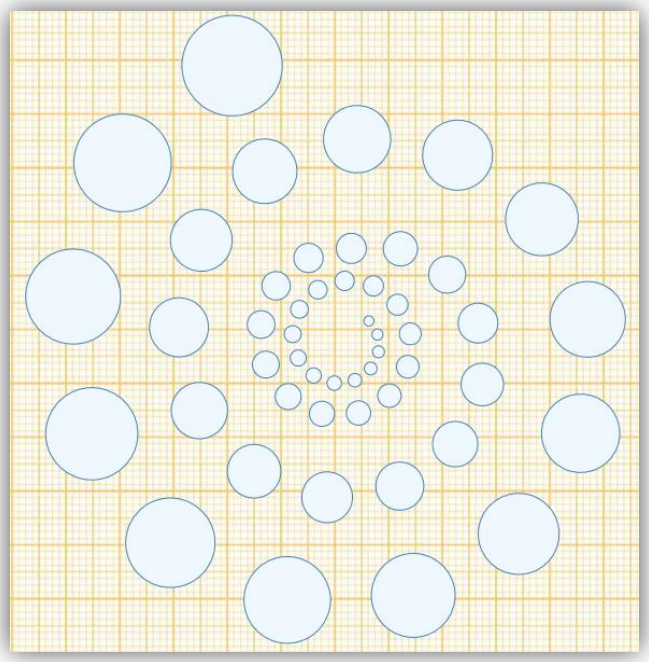


Նկ. 2.4. Հավասարաչափ պտտում

- Հնարավոր է նաև պտտմանը զուգահեռ մեծացնել պատկերի պատճենների չափսերը՝ այսինքն ստանալ պարուրաձև պատկեր: Դրա համար անհրաժեշտ է սեղմել «աստիճանաբար մեծացնել օբյեկտի չափսերը և պտտել գալարաձև»: Սա ստացվում է (2.6) բանաձևով,

$$\alpha = \frac{i * \alpha t}{c} \qquad L_2 = L_1 * \beta \qquad H_2 = H_1 * \beta \qquad (2.6)$$

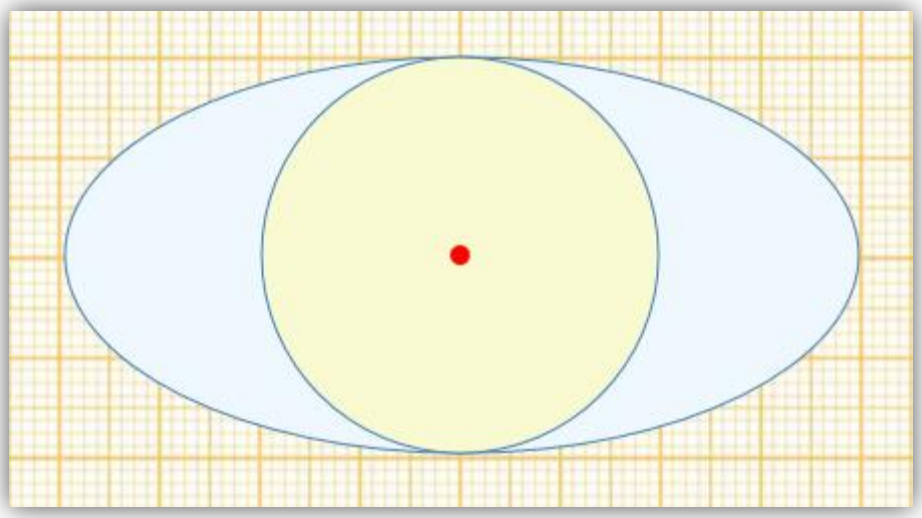
որտեղ L – ը դա պատկերի լայնությունն է, իսկ H – ը բարձրությունը, իսկ β - ն խոշորացման գործակիցը (Նկ 2.5):



Նկ. 2.5. Պարուրաձև պտտում:

- **Խոշորացում** (**Zoom**)

- Ինչպես պտտման դեպքում, այստեղ նույնպես պետք է տրվի խոշորացման առանցքը և այն գործակիցները, որոնցով պետք է մեծացնել պատկերը: Գոյություն ունեն երկու գործակիցներ՝ հորիզոնական և ուղղահայաց խոշորացման համար: Աշխատանքային ռեժիմներից մեկում կարելի է տալ միայն մի գործակիցը, մյուսը կհամարվի նրան հավասար և մեծացված պատկերը կլինի իր սկզբնաղբյուրին նման: Այստեղ նույնպես հանարավոր է տվյալներ ներմուծել մկնիկի օգնությամբ, ստեղնաշարից և կոմբինացված եղանակներով:
- Կարելի է տալ ինչպես հորիզոնական, այնպես էլ ուղղահայաց խոշորացման գործակիցները, այս դեպքում պատկերը «կաղավաղվի» և կխոշորանա ոչ հավասարաչափ, այսպես կարելի է օրինակ շրջանից ստանալ օվալ (Նկ 2.6):

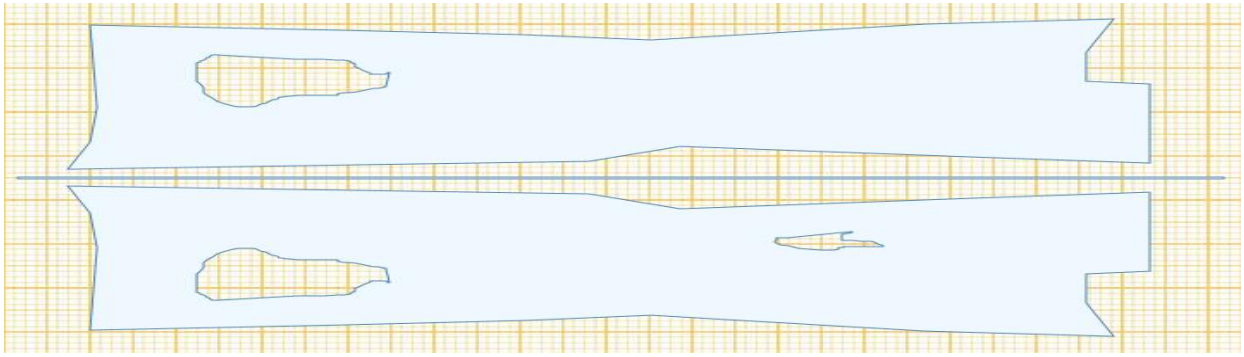


Նկ. 2.6. Շրջանից ստանալ օվալ:

- **Պատճենում** (**Copy**)

- Ստեղծում է օբյեկտի պատճենը, դրա համար անհրաժեշտ է աջ սեղմակով սեղմել այն օբյեկտի վրա, որի պատճենը պահանջվում է:
- Mirror right (Աջ հայելային պատճեն): Ստեղծում է պատկերի հայելային պատճենն այնպես, կարծես հայելին տեղադրված է նրա աջ կողմում:
- Mirror bottom (Հորիզոնական հայելային պատճեն): Ստեղծում է պատկերի հայելային պատճենն այնպես, կարծես հայելին տեղադրված է նրա ներքևի մասում:

Վերը նշվածները հիմնականում օգտագործվում են այն դեպքում, երբ հումքը ներմուծում են ծրագիր, և հումքի մի կողմում առկա թերությունները մյուս կողմում չեն երևում, այդ դեպքում, հումքը պտտելուն համապատասխան արդյունք է տալիս հայելային պատճենը, և արդեն կարելի է ներմուծել հումքը՝ հաշվի առնելով երկու կողմի թերությունները (Նկ. 2.7):



Նկ. 2.7. Հորիզոնական հայելային պատճեն:

- **Հեռացում** (**Delete**)

  Հեռացնում է պատկերը, անհրաժեշտ է աջ սեղմակով սեղմել պատկերի վրա, կամ նշել պատկերների խումբ և սեղմել «Հեռացում» կոճակը:

- **Հիշել** (**Save**)

  Հիշում է պատկերը, կամ պատկերների խումբը:

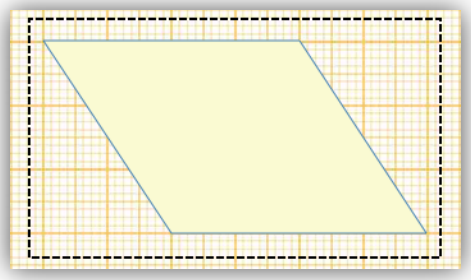
- **Բացել** (**Open**)

  Բացում է արդեն իսկ հիշած պատկերը:

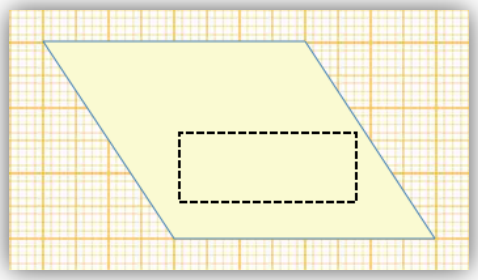
- **Խմբակային ընտրում** (**Multi Group**)

  Բոլոր վերը նշված գործընթացները կարելի է կատարել ինչպես մեկ, այնպես էլ մի քանի տարրերի հետ, որոնք նախապես նշվել են՝ այդ նպատակով տեղադրված կոճակի միջոցով: Պատկերների ընտրումը կատարվում է, եթե պատկերը և ընդգրկող շրջանակն իրար նկատմամբ գտնվում են հետևյալ դասավորություններից մեկում՝

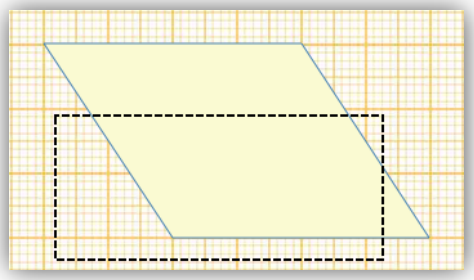
1. եթե շրջանակը ամբողջությամբ գտնվում է պատկերի ներսում (Նկ. 2.8),
2. եթե պատկերն է ամբողջությամբ գտնվում շրջանակի ներսում (Նկ. 2.9),
3. եթե նրանք խաչվում են (Նկ. 2.10):



Նկ. 2.8. Շրջանակը պատկերի ներսում:



Նկ. 2.9. Պատկերը շրջանակի ներսում:



Նկ. 2.10. Պատկերները խաչվում են:

2.5. Տարրական երկրաչափական մարմիններից բարդ մարմինների ստացում

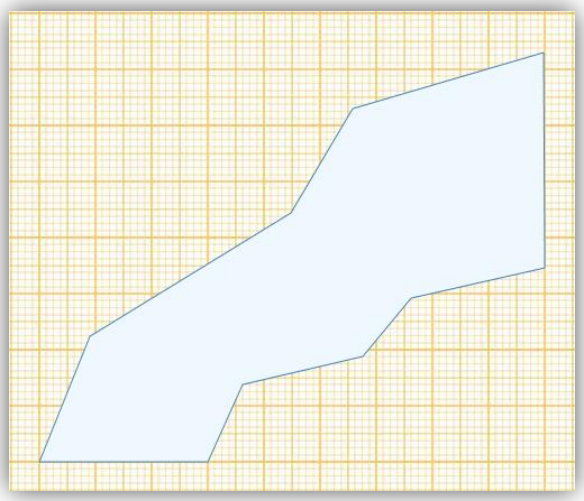
Ցանկացած երկրաչափական մարմին, անկախ նրանից, թե այն ինչ բարդություն ունի, կարելի է ներկայացնել տարրական երկրաչափական մարմիններից բաղկացած պատկերի տեսքով, այսինքն նրանք բոլորն իրենցից ներկայացնում են տարրական երկրաչափական մարմինների համադրություն, և կախված այդ վերջնական նպատակային օբյեկտի բարդությունից փոփոխվում են այն տարրական օբյեկտների քանակը, որոնցից նա բաղկացած է, հիմնականում այդ օբյեկտներն են՝ ուղղանկյուն

կամ մասնավոր դեպքում քառակուսի, էլիպս կամ մասնավոր դեպքում շրջանագիծ, և բեկյալ, որն իրենից ներկայացնում է բազմանկյուն, որը չի ենթարկվում ինչ որ որոշակի օրինաչափության, կարող է ունենալ ցանկացած ձև ու ցանկացած քանակով կողեր:

Հիմնականում, եթե ավելի խիստ դատենք, ապա բեկյալի օգտագործման պարագայում կարող ենք խուսափել նաև ուղղանկյան օգտագործումից: Սակայն, քանի որ ուղղանկյունը նման դեպքերում բավականին շատ է օգտագործվում, ապա կարելի է նրան նույնպես դասակարգել որպես տարրական մարմին:

Բեկյալները հիմնականում օգտագործվում են, որպեսզի տրվի մարմնի հիմնական ուրվագծերը, իսկ նրա հետագա մշակումը կատարվում է նրանից հեռացնելով, կամ նրան կցելով այլ բեկյալներ, կամ, օրինակ, շրջաններ:

Դիտարկենք որևէ միջին բարդության պատկերի ստեղծման քայլերը:

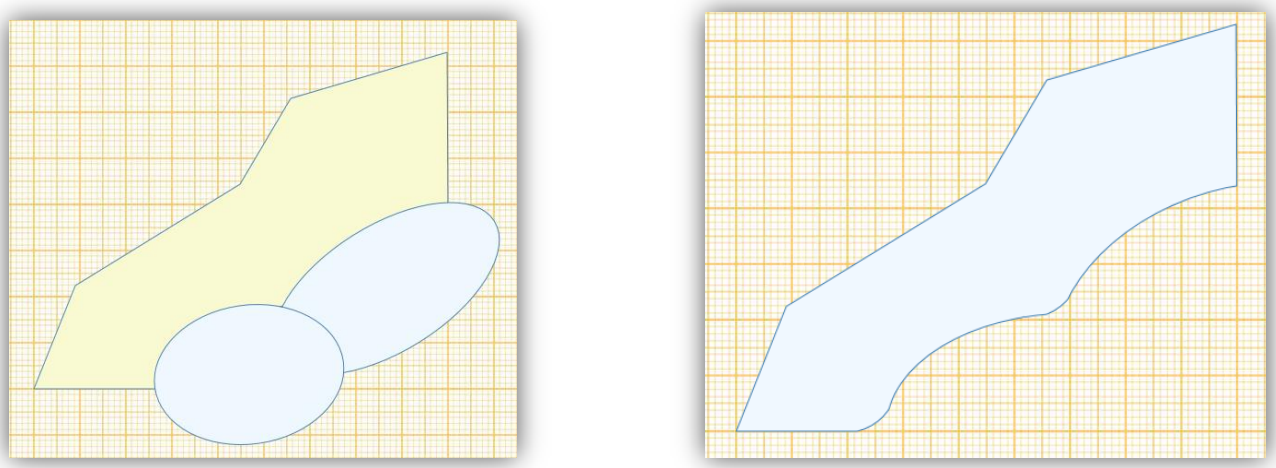


Նկ. 2.11. Պատկերի ուրվագիծը:

Բեկյալի կամ ուղղանկյունների միջոցով տալ ակնկալվող պատկերի ուրվագիծը, ինչպես ցույց է տրված նկ. 2.11 – ում:

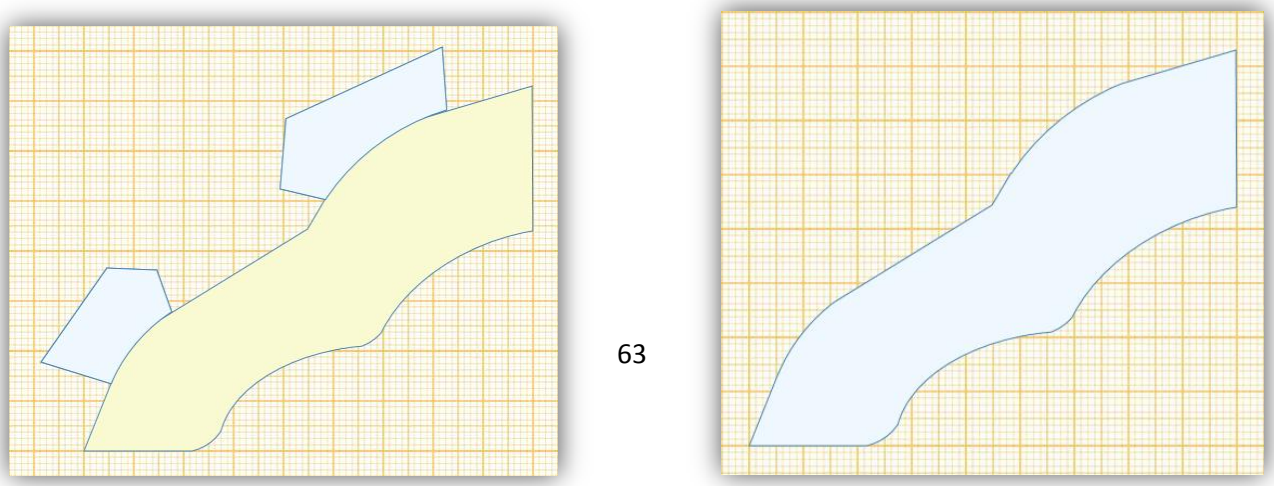
Եթե ուրվագիծը բաղկացած է մի քանի բաղադրիչներից, ապա պետք է նրանց խմբավորել՝ միավորել, հանել, կամ վերցնել ընդհանուր մասերը:

Միավորման եղանակներից օգտվելով՝ պետք է հղկել պատկերի կողերը՝ էլիպսների միջոցով տալ կորությունները, ինչպես ցույց է տրված նկ. 2.12 – ում:



Նկ. 2.12. Անկյունների գոգավորում:

Եթե խոսքը գնում է ոչ թե գոգավորության, այլ ուռուցիկացման մասին, ապա այս դեպքում առաջանում է ևս մի քայլ՝ դա է կաղապարի պատրաստումը, որն իրենից ներկայացնում է գոգավորված մարմին, որը հանելով պատկերի ցանկացած անկյունային մասից, կարելի է ստանալ պահանջվող կորությունը (Նկ. 2.13):



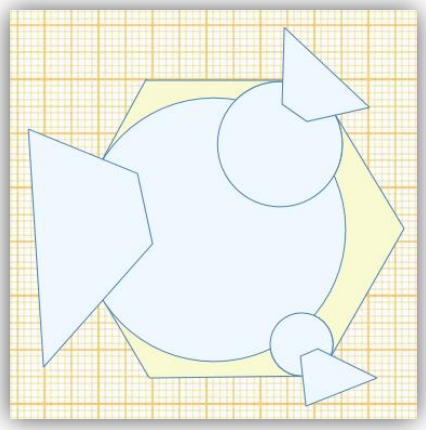
Նկ. 2.13. Անկյունների ուռուցիկացում:

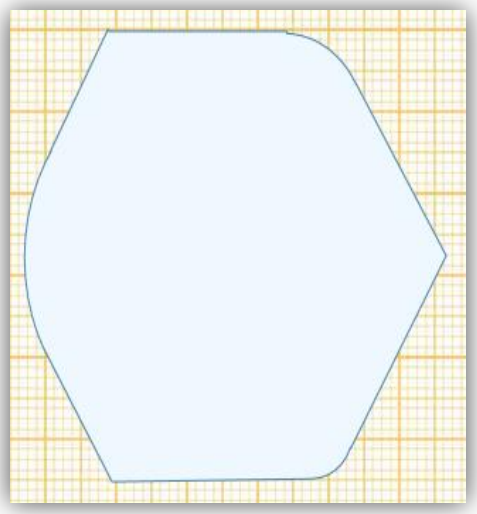
Մարմնի անկյունների ուռուցիկացման համար, պետք է կատարել հետևյալ 3 քայլերը.

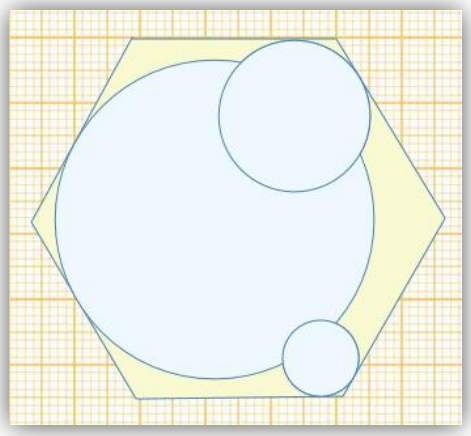
- գծել այն շրջանագիծը, որով հարկավոր է կորացնել անկյունը,
- գծել այնպիսի բեկյալ, որից կարելի է հանել այդ շրջանագիծը և ստանալ կաղապարը,

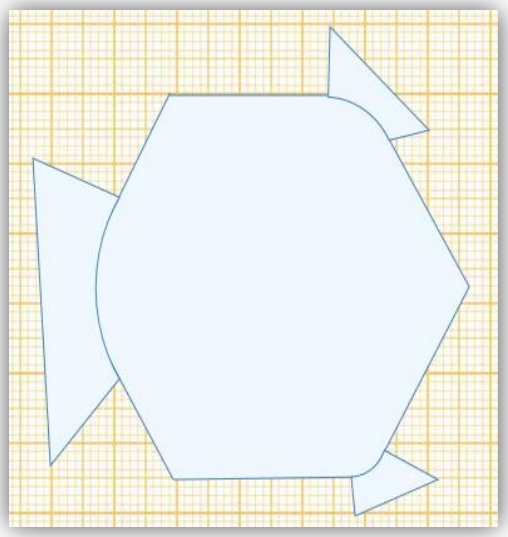
կաղապարը տեղադրել կորացմանը ենթակա անկյան մոտ և այդ պատկերից հանել կաղապարը (Նկ. 2.14):

Վերը նկարագրված եղանակով կառուցվում են տարրական երկրաչափական տեսք ունեցող տարրեր, սակայն հիմնականում գործնականում հանդիպող դետալներն ունեն ավելի բարդ տեսք և բաղկացած են մի քանի տարրական երկրաչափական մարմիններից, այսինքն անհրաժեշտ է միջոց, որով հնարավոր կլինի մոդելավորել ավելի բարդ տեսք ունեցող դետալներ, այս խնդրի լուծման համար ստեղծված են միջոցներ, որոնց օգնությամբ կարելի է այդ բարդ պատկերները ստանալ մի քանի տարրական տարրերի համադրությամբ:







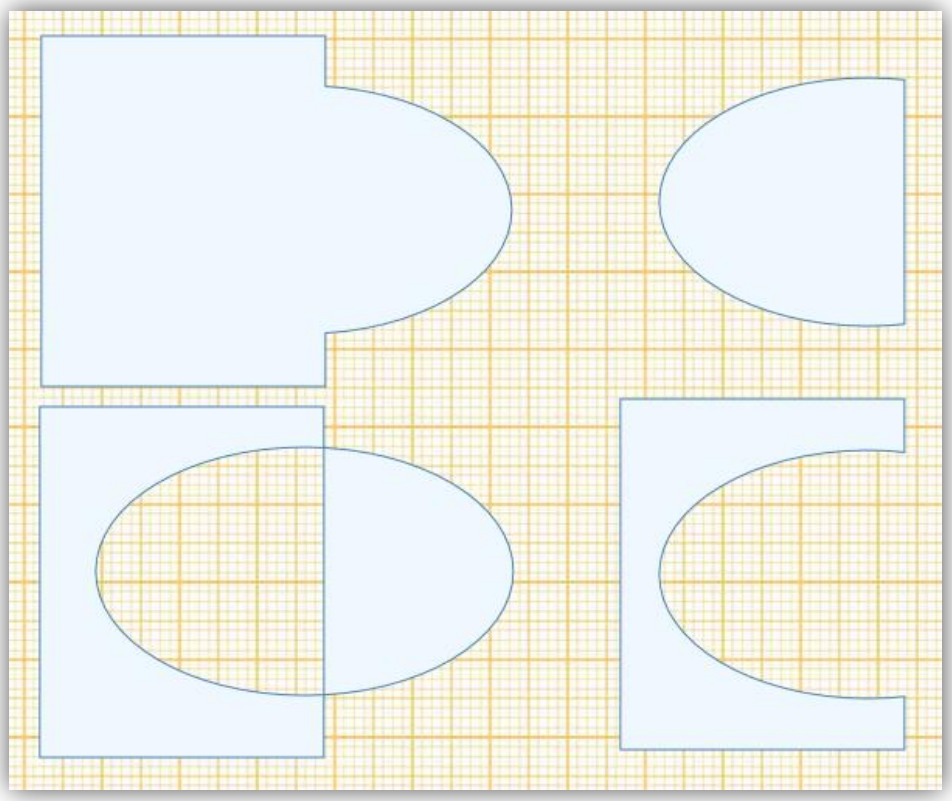


Նկ. 2.14. Պատկերի անկյունների կլորացում:

Գոյություն ունեն վերը նշված համադրությունների 4 տեսակ.

- **Union** ստեղծում է տարր, որն իր մեջ ներառում է 2 բաղկացուցիչ տարրերի բոլոր կետերը,
- **Intersect** ստեղծում է տարր, որն իր մեջ ներառում է 2 բաղկացուցիչ տարրերի ընդհանուր կետերը,
- **Xor** ստեղծում է տարր, որն իր մեջ ներառում է 2 բաղկացուցիչ տարրերի բոլոր այն կետերը, որոնք պատկանում են կամ 1-ին կամ 2-րդ բաղկացուցիչ տարրին, բայց ոչ այդ երկուսին միանգամից: Սա կարելի է բնութագրել նաև հետևյալ կերպ՝ 2 բաղկացուցիչ տարրերը միավորվում են, այնուհետեւ դրանից հեռացվում է նրանց 2-ի ընդհանուր կետերը:
- **Exclude** ստեղծում է տարրը, որն իր մեջ ներառում է 1-ին բաղկացուցիչ տարրի բոլոր կետերը, բացառությամբ այն կետերի, որոնք ընդհանուր են 2-րդ բաղկացուցիչ տարրի հետ:

Վերը նկարագրված միջոցների գրաֆիկական պատկերը տրված է ստորև (Նկ. 2.15):



Նկ. 2.15. Համադրությունների 4 տեսակները:

Թվում է այս միջոցի կիրառմամբ կարելի է հեշտությամբ լուծել 2.2 -ում առաջադրված խնդիրը, սակայն այս միջոցով կարելի է խնդիրը լուծել միայն այն դեպքում, երբ պատկերը կազմված է 2 տարրից, իսկ դրանից ավելի տարրերի դեպքում կան որոշակի բարդություններ, սակայն այդ խնդիրը բավականին հեշտ լուծում է ստանում, երբ օգտագործվում է ռեկուրսիա, այսինքն նույն գործողությունը կատարվում է այնքան անգամ մինչև հասնի վերջնական լուծմանը: Դա կատարվում է հետևյալ կերպ. ենթադրենք դետալը բաղկացած է n մասերից, և այդ բոլոր մասերը նախապես կառուցվել են մոդելավորման մասում: Նախ վերցվում են առաջին 2 տարրերը և միավորվում, այնուհետև արդեն նոր ձևավորված տարրը միավորվում է 3-րդ տարրին և այդպես շարունակ, մինչև առաջին n-1 տարրերն արդեն միավորված կլինեն և կմիավորվեն n-րդ տարրի հետ:

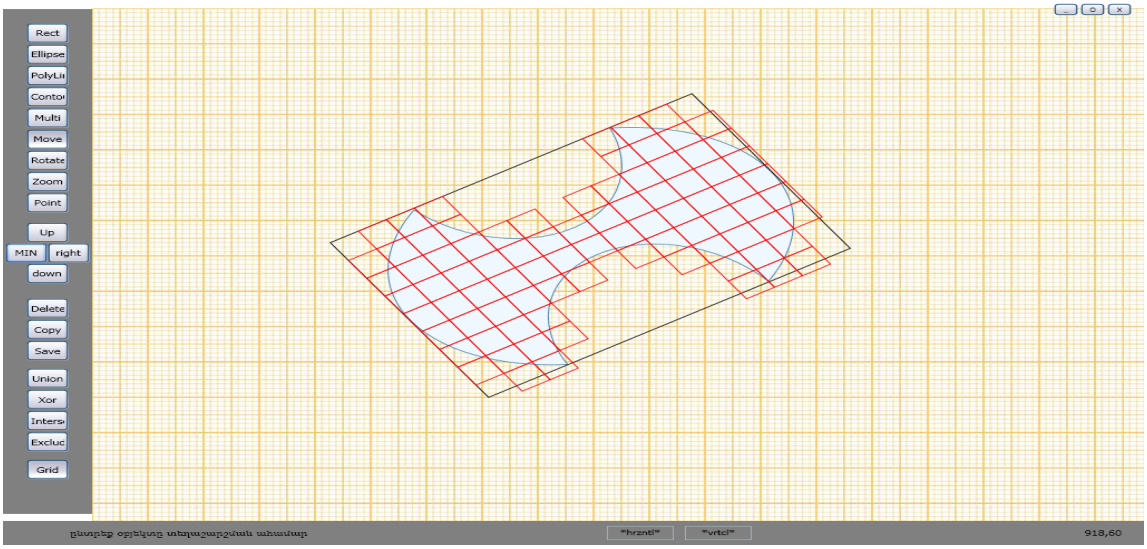
2.6. Եզրակացություն

Այսպիսով, մշակվել է եղանակ և ստեղծվել է ծրագրային միջավայր [65], որը իրականացնում է կամայական ուրվագծով երկչափ օբյեկտների մոդելավորում: Այն ներառում է մի շարք գործիքներ տարբեր բարդության օբյեկտների մոդելավորման և դրանց հետ գործողություններ կատարելու համար: Ծրագրի առավելություններից են արագագործությունը և պարզությունը, ինչը շատ կարևոր է լայն ոլորտի օգտվողների համար:

ԳԼՈՒԽ 3.

ԵՐԿՐԱՉԱՓԱԿԱՆ ՄՈԴԵԼՆԵՐԻ ԴԻՍԿՐԵՏԱՑՈՒՄԸ

Մակերեսի վրա մարմինների օպտիմալ բաշխման խնդրի լուծման նպատակով առաջարկվել է մոդելների դիսկրետացման մոտեցումը, որը նկարագրված է այս գլխում: Տրված խնդիրը լուծվում է մի քանի քայլերով՝ նախ պետք է գտնել դետալներին արտագծված մինիմալ մակերեսով ուղղանկյունը, որն անհրաժեշտ է, որպեսզի խնդրի լուծումը բերվի արդեն իսկ լուծված խնդրի, որն է՝ ուղղանկյուների օպտիմալ բաշխումն ուղղանկյան վրա: Հետո այն բաժանվում է ավելի մանր խորանարդների, որից հետո հեռացվում են ավելորդ խորանարդները, դետալը մոտավոր ընդունում է հետևյալ տեսքը (Նկ. 3.1), որից հետո նրանք դասավորվում են հումքի վրա:

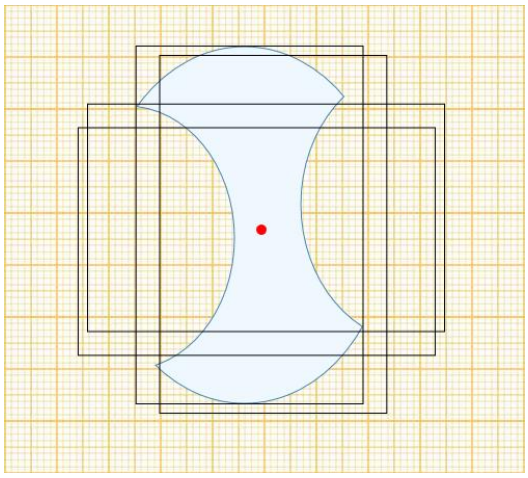


Նկ. 3.1. Մոդելավորված դետալի տեսքը ինտերֆեյսում:

3.1. Մինիմալ մակերեսով արտագծած ուղղանկյան որոնումը

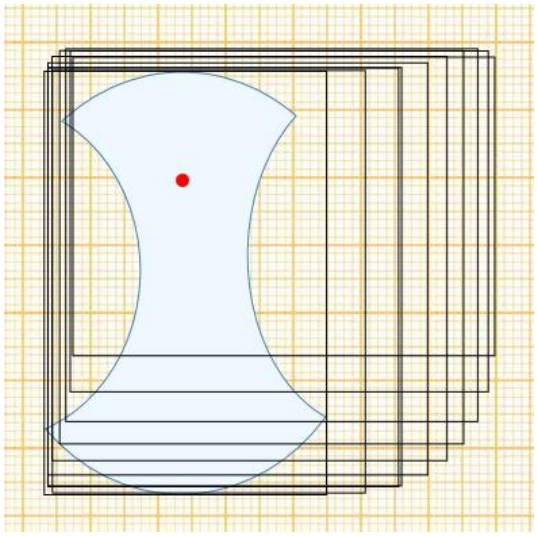
Օպտիմալ բաշխման խնդրի լուծման ճանապարհին առաջ եկավ նշված խնդիրը, որը լուծելու համար պետք է գտնել մի մեթոդ, որով կարելի է տարրին հնարավոր արտագծած ուղղանկյուններից գտնել մինիմալ մակերեսովը: Լուծման համար որոշվեց արտագծած ուղղանկյունները կառուցել անյպես, որ նրանց կողմերը լինեն կոորդինատների առանցքներին զուգահեռ, և տարրը պտտելով նրա կենտրոնի շուրջը, նույն ալգորիթմով նրան արտագծել ուղղանկյունը, որոշել թե բոլոր արտագծած ուղղանկյուններից որն ունի մինիմալ մակերես:

Տեսականորեն, պետք է տարրը պտտել բոլոր հնարավոր անկյուններով և կառուցել նրան արտագծած ուղղանկյունը, այնուհետեւ հաշվել նրանց բոլորի մակերեսները և գտնել մինիմալ մակերեսով ուղղանկյունը: Սակայն նշված գործընթացն որոշակի առումով աշխատատար է և պահանջում է օպտիմալացում, որին կարելի է հասնել հետևյալ եղանակով՝ պտտել ոչ թե 0 – 360, այլ 0 – 90 աստիճաններով, քանի որ ամեն 90 աստիճանից հետո ուղղանկյան մակերեսը նույնությամբ կրկնվում է (Նկ. 3.2):

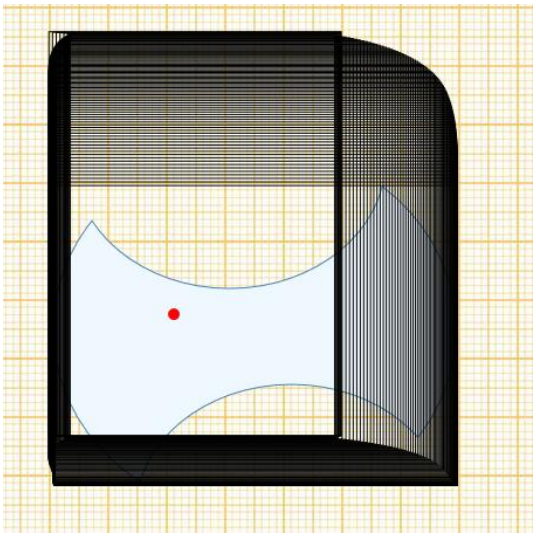


Նկ. 3.2. Պտտում 0 – 90 աստիճաններով:

Բոլոր այն ուղղանկյունները, որոնց մեջ իմաստ ունի փնտրել մինիմալ մակերես ունեցողը, պատկերված են նկ. 3.3 և 3.4-ում: Նշված գործընթացը կատարվում է հերթականությամբ համեմատելու եղանակով՝ այսինքն սկզբից ենթադրվում է, որ մինիմալն է այն ուղղանկյունը, որը ստացվում է տարրին արտագծելով, երբ նրան ընդհանարապես չենք պտտել, այնուհետև տարրը պտտում ենք 1 աստիճանով և համեմատում նախորդ դիրքում արտագծած ուղղանկյան հետ: Եթե այն ստացվում է ավելի փոքր, ապա որպես մինիմալ ուղղանկյուն ընդունվում է դա, հակառակ դեպքում ուղղակի անցնում ենք առաջ, մինչև 90 աստիճան:



Նկ. 3.3 Պտույտի մեծ քայլ



Նկ. 3.4 Պտույտի փոքր քայլ

Երբ ուղղանկյունը գտնվում է, դետալին պետք է պտտել հետ այն անկյունով, որի ժամանակ արտագծած ուղղանկյունը ստացվել է մինիմալ:

Այս գործընթացի ճշտությունը կախված է պտույտի քայլից, այսինքն ինչքան փոքր է պտույտի անկյունը, այնքան ճշգրիտ են, բայց ավելի շատ հաշվարկները (նկ. 3.3, 3.4):

Այն կիրառական խնդիրներում, որտեղ հումքը համասեռ չէ, այսինքն կարևոր է դետալի ուղղությունը, օրինակ, կտոր, փայտ և այլն, ընտրվում է այն ուղղանկյունը, որը արտագծած է դետալին անհրաժեշտ ուղղությամբ՝ առանց պտույտների:

3.2. Դիսկրետացում

Նշված խնդրում դիսկրետացում ասելով հասկանում ենք, որ արտագծած ուղղանկյունը պետք է բաժանվի ավելի փոքր վանդակների: Տվյալ գործընթացը կատարվում է հետևյալ մի քանի պատճառներով՝

1. որպեսզի հումքն օգտագործվի օպտիմալ՝ թափոնների չափերը լինի հնարավորինս քիչ,
2. որպեսզի վերջնական բաշխման ժամանակ խնայենք ժամանակը:

Նշված գործընթացն անհրաժեշտ է, քանի որ հումքն իր բոլոր հնարավոր դիրքերով պտտելու և տեղադրելու փոխարեն հնարավոր է այն ունենա պտտման կարիք միայն 90 աստիճանով:

Դիսկրետացումը կատարվում է ըստ այն ճշտության, որով պետք է լուծվի խնդիրը, այսինքն որոշվում է ինչ չափով է կատարվում ուղղանկյունների բաժանումը:

Ընդհանրապես, տվյալ դեպքում օգտագործողը պետք է լուծի կոմպրոմիսային խնդիր՝ կախված գործի ճշտությունից և աշխատանքի կատարման արագությունից՝ այսինքն ինչքան փոքր լինեն քառակուսիները, որոնցով պետք է կատարվի դիսկրետացումը, այնքան ավելի դանդաղ կկատարվի այդ գործընթացը, սակայն ճշտությունը կլինի ավելի բարձր: Եթե շատ մեծ ճշտության կարիք չկա, ապա կարելի է քառակուսիների չափը տալ ավելի մեծ, այդ դեպքում արդեն դիսկրետացումը կկատարվի անհամեմատ ավելի արագ: Սա կարելի է կատարել նաև այն դեպքում, երբ օգտագործում ենք առավել հարթ կողերով մարմիններ, այսինքն կարիք չկա սուզվելու

մանրամասնությունների մեջ: Լռելայն դիսկրետացման գործակից է վերցվում 30 պիքսելը:

Քանի որ դիսկրետացման գործակիցը չի կարող լինել ոչ ամբողջ թիվ, այդ պատճառով պետք է որոշել, թե ինչ քանակով քառակուսիների է բաժանվելու պատկերին արտագծած ուղղանկյունը: Դրա համար կա երկու մոտեցում՝

1. կլորացնել քառակուսիների քանակը ի վնաս դետալի, այսինքն եթե պահանջվում է 22.3 քառակուսի, ապա կարելի է վերցնել 22 քառակուսի, սակայն սա ոչ ճիշտ է արդյունաբերական առումով, քանի որ չի կարելի թույլ տալ, որ դետալն աղավաղվի,
2. կլորացնել քառակուսիների քանակը ի վնաս հումքի. այստեղ ունենում ենք առավել շատ կորուստներ, այսինքն նույն քանակով դետալները կտեղավորվեն ավելի մեծ մակերեսով հումքի վրա: Այս դեպքում, եթե ստացվում է նույնիսկ 20.01 հատ քառակուսի, վերցվում է 21-ը: Սրան հասնում ենք 3.1 բանաձևով: Այստեղ N_h և N_vը համապատասխանաբար հորիզոնական և ուղղահայաց քառակուսիների քանակներն են, L և H-ը համապատասխանապար մինիմալ մակերեսով արտագծած ուղղանկյան լայնությունն ու բարձրությունն են, իսկ l - ը դա դիսկրետացման գործակիցն է՝ քառակուսու չափսը:

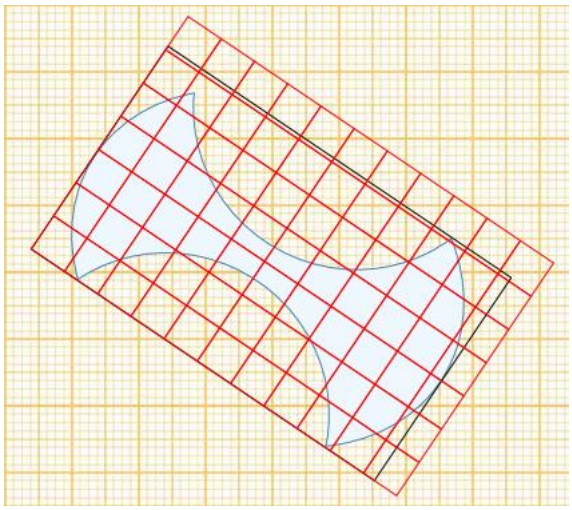
$$N_h = \frac{L + (l-1)}{l}$$

$$N_v = \frac{H+(l-1)}{l}$$

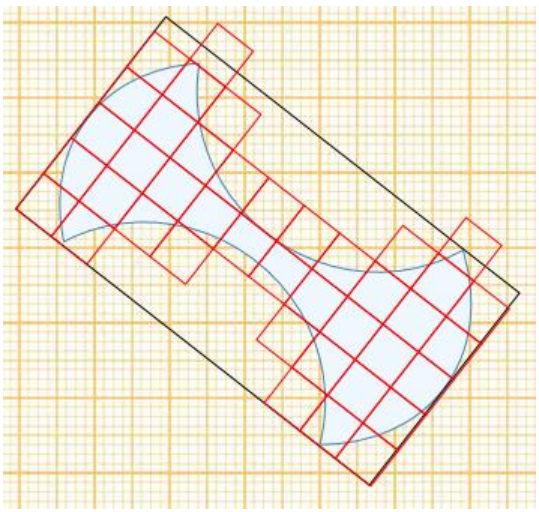
(3.1)

Այսպիսով, նախ արտագծված ուղղանկյունը բաժանվում է վանդակների (նկ 3.5), այնուհետև հեռացվում են այն վանդակները, որոնք չեն ընկնում պատկերի

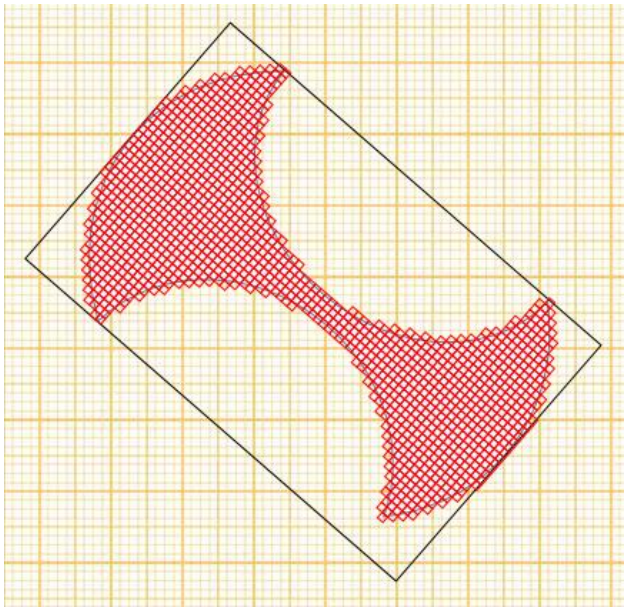
մակերևույթի մեջ (նկ 3.6): Նույնը պատկերված է նկ. 3.7- ում, այն բացառությամբ, որ այստեղ դիսկրետացման գործակիցը 5 պիքսել է: Այնուհետև անցնում ենք պատկերների օպտիմալ բաշխմանը:



Նկ. 3.5 Վանդակի ստեղծում



Նկ. 3.6 Ավելորդ վանդակների Հեռացում

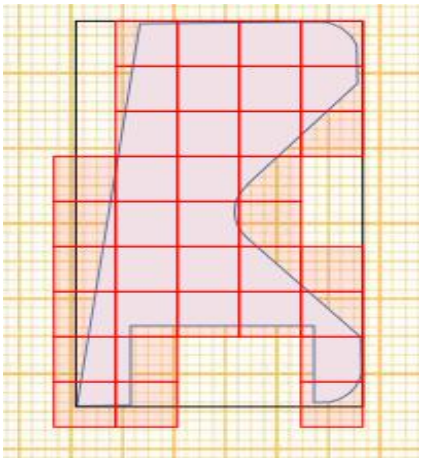


Նկ. 3.7 Այլ դիսկրետացման գործակից

3.3. **Դիսկրետացման վերջին փուլը**

Նկարագրենք դիսկրետացման վերջին փուլը, երբ յուրաքանչյուր ուղղանկյանը համապատասխանության մեջ է դրվում երկուական մատրից: Վերցնում ենք n և m չափսերով երկչափ մատրից, որտեղ n–ը համապատասխանում է հորիզոնական վանդակների թվին, իսկ m-ը դա ուղղահայաց վանդակների թիվն է: Դրանից հետո այն բոլոր դիրքերում, որտեղ առկա է դետալ, դնում ենք 1, իսկ որտեղ վանդակները հեռացված են, դնում ենք 0, այսպիսով նկ. 3.8-ում պատկերված տարրը ստանում է նկ 3.9-ում նկարագրված մաթեմատիկական տեսքը: Սակայն իրականում այս մատրիցները ոչ թե երկչափ, այլ եռաչափ են, երրորդ չափումը ցույց է տալիս դետալի համարը՝ որպեսզի հնարավոր լինի այդ մատրիցը հետ բերել և վերականգնել պատկերը հետագա օգտագործման համար:

Դրանից հետո ծրագրային մշակումն անհամեմատ պարզ է դառնում, և եթե հարկավոր է օրինակ, պտտել պատկերը 90 աստիճանով, ապա իր հետ պտտում ենք նաև նրան կից մատրիցը՝ նկ. 3.10 և նկ. 3.11:



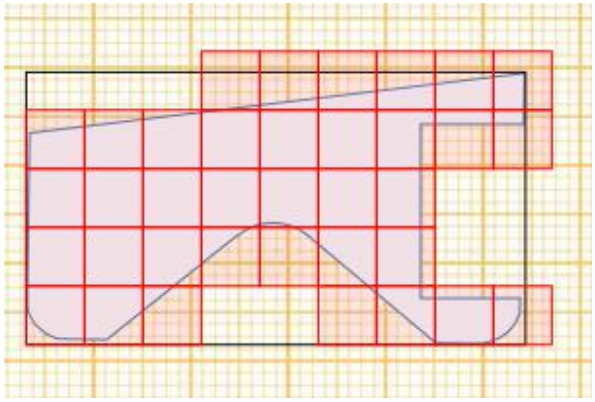
Նկ. 3.8 Դետալը

$$\begin{matrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{matrix}$$

$$\begin{matrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{matrix}$$

$$\begin{matrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \end{matrix}$$

Նկ.3.9. Համապատասխան մատրիցը



Նկ. 3.10. Պտույտ 90 աստիճանով

0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	0

1	1	1	1	1	1	1	0	0
1	1	1	0	0	1	1	1	1

Նկ. 3.11. Համապատասխան մատրիցը

Այսպիսով, նշված մատրիցների հիման վրա տեղի է ունենում բաշխումը, իսկ արդեն բաշխումից հետո անցնում ենք մատրիցների փոխարինմանը իրենց տարրերով, որը տեղի է ունենում ըստ իրենց 3 չափման՝ տարրի համարի:

3.4. Եզրակացություն

Այսպիսով, ներկայացվեց կամայական ուրվագծով երկչափ երկրաչափական մոդելների դիսկրետացման առաջարկված եղանակը [66]: Այն կիրառվում է ինչպես հումքի, այնպես էլ ձևելու ենթակա դետալների մոդելների վրա: Նախ գտնում ենք երկրաչափական մոդելին արտագծած ուղղանկյուն՝ նվազագույն մակերեսով: Հաջորդ քայլում արտագծած ուղղանկյունը բաժանվում է վանդակների տված ճշտությամբ, և դատարկ վանդակները հեռացվում են: Վերջին փուլում յուրաքանչյուր մոդելին համապատասխանության մեջ է դրվում երկուական մատրից: Այս տեսքը հետագայում մասնակցում է առաջարկված էվրիստիկ ալգորիթմում, որը նկարագրված է հաջորդ գլխում: Յուրաքանչյուր մոդելի համարը պահվում է, ձևելուց հետո դետալը վերականգնելու համար:

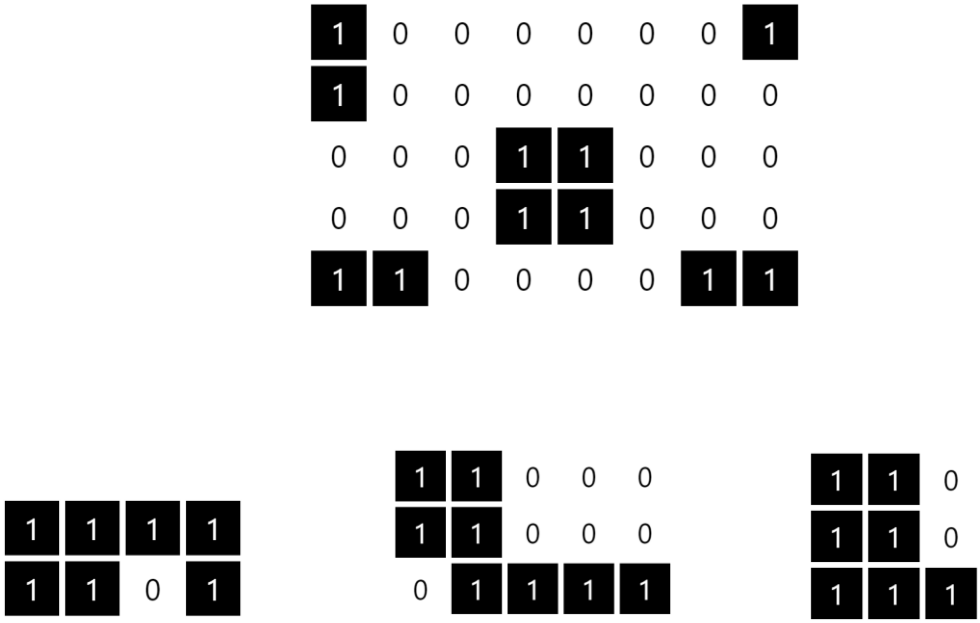
ԳԼՈՒԽ 4.

ԿԱՄԱՅԱԿԱՆ ՈՒՐՎԱԳԾՈՎ ԵՐԿՉԱՓ ՄԱՐՄԻՆՆԵՐԻ ՁԱՄԱՆ ԱԼԳՈՐԻԹՄԸ և ԾՐԱԳՐԱՅԻՆ ՀԱՄԱԿԱՐԳԸ

Այս գլխում նկարագրված է առաջարկված էվրիստիկ ալգորիթմը և մշակված ծրագրային համակարգը՝ հիմնված այդ ալգորիթմի վրա: Բերված են ծրագրային համակարգի տեխնիկական նկարագրությունը և առաջարկված լուծումները:

4.1. Նոր էվրիստիկ ալգորիթմի նկարագրությունը

Խնդրի լուծման այս կետում հումքը և դետալներից յուրաքանչյուրը իրենցից ներկայացնում են երկուական մատրից: Նկ 4.1-ում տրված են մոդելավորված հումքի ՝ իր ազատ և օգտագործված տարացքներով և երեք դետալի օրինակ :



Նկ. 4.1. Հումքի և դտալների մոդելավորված և դիսկրետացված տեսքը

Հարմարավետության համար բոլոր 0 էլեմենտները՝ որոնք արտահայտում են հումքի վրայի ազատ տարածքները, արտապատկերված են սպիտակ, իսկ 1 էլեմենտները՝ որոնք ներկայացնում են հումքի կամ դետալի արդեն օգտագործված կամ դեֆեկտային տարածքները, ներկայացված են սև գույնով:

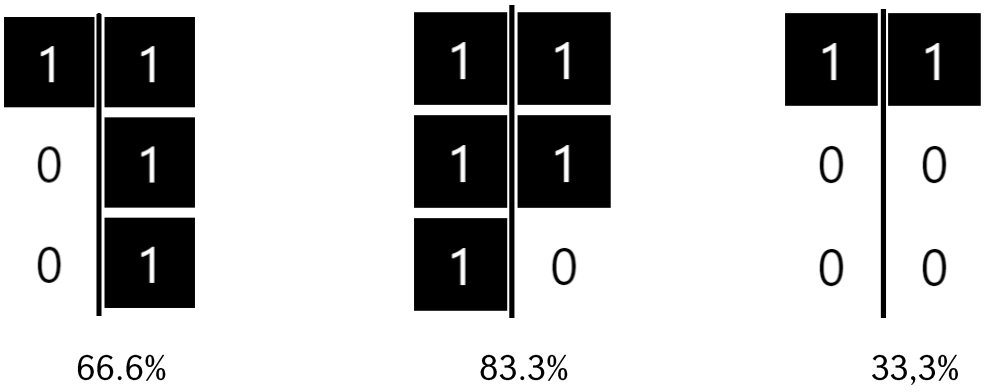
Այժմ մտցնենք մաքսիմալ համապատասխանություն (fit) գաղափարը: Խնդիրն է՝ ապահովել դետալների միմյանց առավելագույն հպումը, ինչը նշանակում է, որ մի դետալի եզրային 1 էլեմենտին անմիջապես պետք է հաջորդի մյուս դետալի եզրային 1 էլեմենտը:

Սահմանում:

Ֆիթ է կոչվում երկու հարևան դետալների ընդհանուր շփման գծի վրայի էլեմենտների հարաբերությունը նույն շփմա գծի վրա գտնվող 1-երի քանակին արտահայտված տոկոսներով:

$$fit = n_{մեկ} / n_{ընդհանուր} * 100\%$$

Նկ. 4.2 –ում պատկերված են օրինակներ հաշվարկված ֆիթերով:



Նկ. 4.2 Ֆիթի հաշվարկման օրինակ

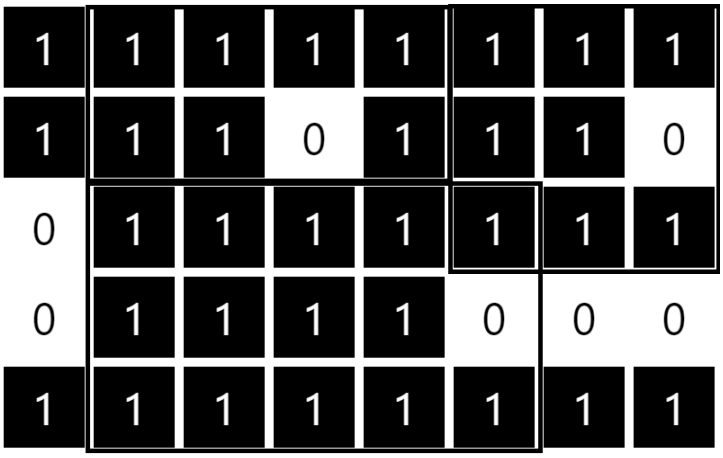
Ալգորիթմի կառուցման ժամանակ օգտվում ենք 1.10 բաժնում նկարագրված գոգավոր անկյունների գաղափարից: Դետալների բաշխումը հումքի վրա սկսում ենք ամենամեծ կտորից, կամ այլ կերպ ասած ամենաշատ 1-եր ունեցողից: Ալգորիթմը ունի հետևյալ քայլերը.

1. գտնել ամենաշատ 1-եր ունեցող ձևման ենթակա դետալը;
2. յուրաքանչյուր գոգավոր անկյան համար հաշվել ֆիթը;
3. տեղադրել դետալը այն անկյունում, որի ֆիթը մեծագույնն է;
4. հումքի այն մասը, որտեղ տեղադրվել է դետալը, փոխել 0-երը 1-երով;
5. հանել դետալը ձևման ենթակա հաջորդականությունից:

Ալգորիթմը շարունակում է իր աշխատանքը մինչև տեղի ունենա հետևյալ պայմաններից որևէ մեկը՝

1. այլևս ոչ մի դետալ չտեղավորվի հումքի դատարկ մասերում, այսինքն հումքի սահմանակից զրոների քանակը փոքր լինի ամենափոքր դետալի 1-երի քանակից;

2. խնդիրը լուծվի ամբողջությամբ՝ դետալները ամբողջությամբ բաշխվեն հումքի վրա:

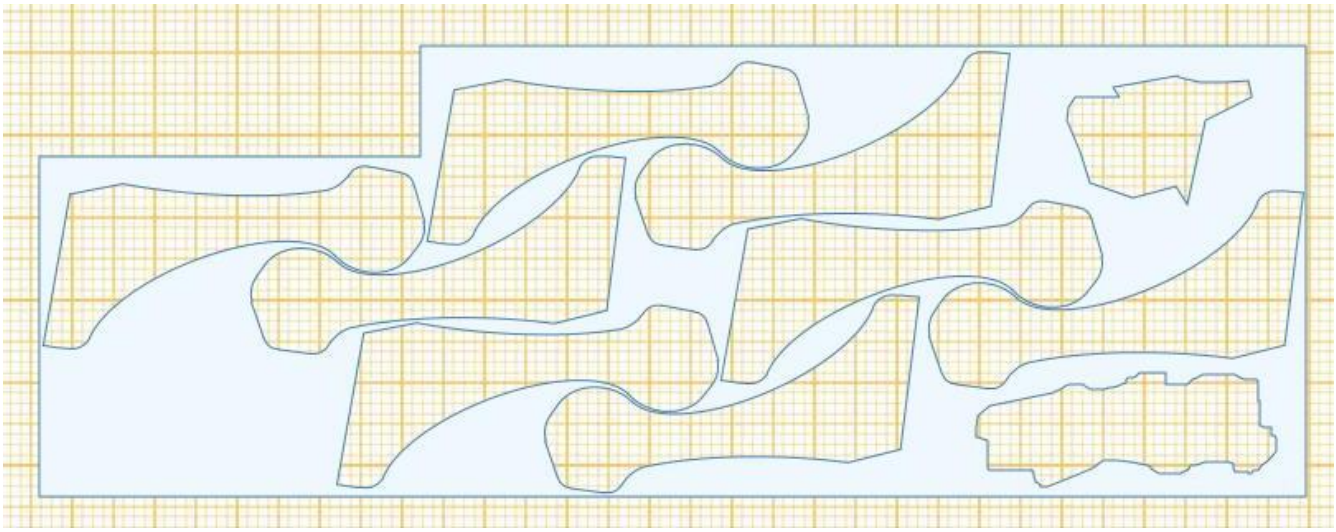


Նկ. 4.3. Դետալների բաշխումը

Նկար 4.3 –ում բերված է նկար 4.1-ում դիտարկված հումքի և դետալների բաշխման արդյունքը:

Ալգորիթմը իրականացված է ծրագրային համակարգում, որի նկարագրությունը բերված է հաջորդ բաժիններում:

Ալգորիթմի ավարտից հետո դետալները վերականգնվում են ըստ իրենց համարների: Ձևման վերջնական տեսքի օրինակ պատկերված է նկար 4.4-ում:



Նկ. 4.4. Ձևման վերջնական արդյունքը

```
public class MartixHelper
{
    public double GetFit(int[,] left, int[,] right)
    {
```

```
    var edgeLenght = GetCommonEdgeLenght(left, right);

    var leftCellCount = GetFullCellCountOfEdge(left, edgeLenght, left.GetLength(1) - 1);
    var rightCellCount = GetFullCellCountOfEdge(right, edgeLenght, 0);

    return ((leftCellCount + rightCellCount) / (double)(2 * edgeLenght)) * 100;
}

private int GetCommonEdgeLenght(int[,] left, int[,] right)
{
    var leftLenght = left.GetLength(0);
    var rightLenght = left.GetLength(0);

    return Math.Min(leftLenght, rightLenght);
}
private int GetFullCellCountOfEdge(int[,] matrix, int rowCount, int edgeColumn)
{
    int fullCellCount = 0;

    for (int i = 0; i < 5; i++)
    {
        if (matrix[i, edgeColumn] == 1)
            ++fullCellCount;
    }
    return fullCellCount;
}
```

}

4.2. Օգտագործողի ինտերֆեյսի նկարագրությունը

Մշակվել է պարզ և մատչելի օգտվողի ինտերֆեյս, որը բաղկացած է հետևյալ գործիքամիջոցներից.

1. **Էլեմենտներ ստեղծող**: Սրանց մասին նկարագրվել է առաջին գլխում, օգտագործվում են էլեմենտների ստեղծելու համար:
2. **Մոնիպուլացոն**: Օգտագործվում են ձևափոխություններ կատարելու համար:
3. **Մուտքի – ելքի**: Կարելի է հիշել պատրաստի դետալը և այնուհետև նրան օգտագործել բազմակի անգամ: «Details» կոճակի վրա սեղմելուց հետո կբացվի դիալոգային պատուհանը, որի վրայից կարելի է ընտրել այն բոլոր դետալները, որոնք մասնակցելու են բաշխմանը:
4. **Խմբավորման**: Հնարավորություն է տալիս խմբավորել դետալները: (Նկարագրված 1 առաջին գլխում):
5. **Տեղաշարժման և խոշորացման**: Սրանք արված են հարմարավետության համար՝ ապահովում են ավելի մեծ տարածք աշխատանքի համար: Հնարավորություն է ստեղծված խոշորացնել ամբողջ տարածքը՝ առավել մանրակրկիտ աշխատանքի համար:
6. **Ինֆորմացիոն մուտք – ելք**: Օգտագործողին է ինֆորմացիա է տալիս ծրագրի կարգավիճակի կամ առաջընթացի մասին, ինչպես նաև օգտագործողը ներմուծում է տվյալներ:
7. **Ծրագրի կառավարման**: Սրանք ստանդարտ են համարվում: Բաղկացած է երեք կոճակից՝ մինիմացնել ծրագիրը, վերաբեռնավորել այն և անջատել:

4.3. Ծրագրային ապահովման տեխնիկական նկարագիրը

MIC Host ծրագրային համակարգը մշակվել է Windows օպերացիոն համակարգի համար: Համակարգի նախագծման համար օգտագործվել են .NET պլատֆորմի C#, բազաների հետ աշխատելու համար SQL և ինտերֆեյսի նկարագրման համար HTML լեզուները և Visual Studio ծրագրային միջավայրերը:

Ներկայացված ալգորիթմը իրականացնելու համար ընտրվել է .NET framework-ի C# լեզուն, քանի որ այն խիստ տիպիզացված և օբյեկտ կողմնորոշված լեզու է: Օբյեկտ կողմորոշված են համարվում այն ծրագրավորման լեզուները, որոնց հիմքում ընկած են կլասները: Դրանք իրենցից ներկայացնում են տիպեր, այսինքն պարունակում են կոնկրետ հասկացություն նկարարագրող բոլոր հատկությունները և այդ հասկացության հետ գործողություններ կատարելու համար ֆունկցիաներ կամ մեթոդներ: Այդ տիպին պատկանող ցանկացած օբյեկտ տվյալ տիպի մնացած օբյեկտներից անկախ է: Միաժամանակ կարող է գոյություն ունենալ մի տիպի պատկանող մեկից ավելի օբյեկտներ, որոնք բոլորը ունեն նախապես կլասում տրված նկարագիրը և ֆունկցիաները:

Որպեսզի ծրագրավորմանն լեզուն կոչվի օբյեկտ կողմորոշված, այն պետք է ունենա հետևյալ հատկությունները:

Աբստրակցիա : Սա օբյեկտի կարևոր հատկությունների առանձնացումն է, հաշվի չառնելով երկրորդականները:

Ինկապսուլյացիա: Սա համակարգի հատկություն է, որը թույլ է տալիս միավորել դասի տվյալները և դրանց հետ աշխատող մեթոդները՝ միաժամանակ անտեսանելի պահելով իրականացման առանձնահատկությունները:

Ժառանգում : Սա համակարգի հատկություն է, որը թույլ է տալիս նկարագրել նոր դաս՝ մեկ այլ դասի հիմքի վրա, ֆունկցիոնալության մասնակի կամ լրիվ փոխառնմամբ: Այն դասը, որից կատարվում է ժառանգում կոչվում է բազային կամ ծնող դաս: Իսկ նոր դասը կոչվում է ածանցյալ կամ ժառանգ դաս:

Պոլիմորֆիզմ (Բազմաձևություն): Սա համակարգի հատկություն է, որը թույլ է տալիս միանման ինտերֆեյսով օբյեկտներ ՝ առանց ներքին կառուցվացքի ու տիպի մասին պատկերացում ունենալու:

Դաս : Դասը հանդիսանում է ելակետային կոդի տերմինաբանությամբ նկարագրված, դեռ գոյություն չունեցող օբյեկտի մոդել: Փաստացի այն նկարագրում է օբյեկտի կառուցվածքը, հանդիսանալով օբյեկտի "գծագիր":

Օբյեկտ : Հաշվողական համակարգի հասցեային տիրույթի իմաստային հատված է, որը առաջանում է դասի օրինակ (օբյեկտ) ստեղծելիս:

Նախատիպ : Նախատիպը օբյեկտ-օրինակ է, որի օրինակով ու նմանությամբ ստեղծվում են ուրիշ օբյեկտներ:

Աբստրակտ: Աբստրակտ կլասը իրենից ներկայացնում է նախնական հատկությունների և ֆունկցիաների կրող, սակայն նա չի կարող ունենալ օբյեկներ, այն նախնական հասկացությունների կրող է, և կարող է օգտագործվել միայն ժառանգության համար, որպեսզի իր հատկությունները փոխանցի իրենից ժառանգվող կլասսներին:

```
abstract class ShapesClass
{
    abstract public int Area();
}
class Square : ShapesClass
{
```

```
int side = 0;

public Square(int n)
{
    side = n;
}
// Area method is required to avoid
// a compile-time error.
public override int Area()
{
    return side * side;
}

static void Main()
{
    Square sq = new Square(12);
    Console.WriteLine("Area of the square = {0}", sq.Area());
}

interface I
{
    void M();
}
abstract class C : I
{
    public abstract void M();
```

```
    }
}
```

Ստատիկ: Սրանք կլասսների տեսակներ են, որոնք ունեն ստատիկ հատկություններ և փոփոխականներ, ի տարբերություն հասարակ ֆուկցիաների, որոնց օգտագործման համար անհրաժեշտ է ստեղծել օբյեկտ, որը պատկանում է այդ կլասսին: Սրանք կարելի է օգտագրոծել առանց օբյեկտ ստեղծելու: Այս տիպի կլասսները հիմանականում լինում են օգնական ֆունկցիաների գրադարաններ: Իհարկե կարելի է ստեղծել նաև օբյեկտներ, սակայն այս դեպքում այդ օբյեկներից ցանկացած մեկում փոփոխություն անելու դեպքում կփոխվի նաև մնացած բոլոր օբյեկտների հատկությունները, քանի որ նրանք ընդհանուր են:

```
public class Automobile
{
    public static int NumberOfWheels = 4;
    public static int SizeOfGasTank
    {
        get
        {
            return 15;
        }
    }
    public static void Drive() { }
    public static event EventType RunOutOfGas;
}
```

Հասարակ: Ունի վերը նշված կլասսների հատկությունները, բայց կարող է օգտագործվել որպես անկախ տիպ:

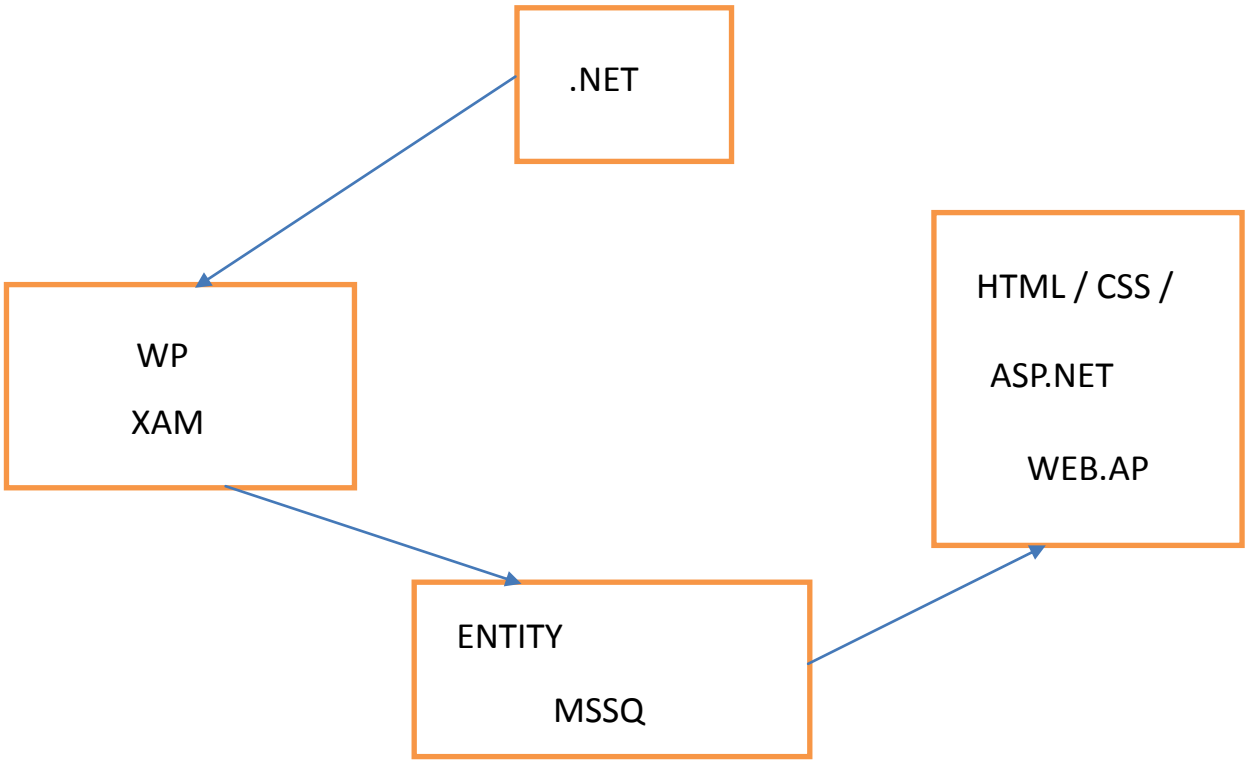
```
public class Customer
{
    //Fields, properties, methods and events go here...
}
```

Տվյալ խնդրի շրջանակներում հայտարարված է Martix կլասը, որը ունի երկարություն և լայնություն հատկություններ, և անհրաժեշտ բոլոր ֆունկցիաները՝ նրանց հետ աշխատելու համար: Դրանցից են՝ պպտել մատրիցը, վերցնել եզրայնի սյունակը և այլն:

Քանի որ C# լեզվում չկա անհրաժեշտ գրադարան, որը կարող է մատրիցի հետ կատարել բոլոր անհրաժեշտ գործողությունները, որոշվեց ստեղծել գրադարան MartixHelper անունով: Այն ևս իրենից ներկայացնում է կլաս, և իր մեջ ներառում է մի քանի մատրիցների հետ աշխատելու համար անհրաժեշտ ֆունկցիաները, որոնցից են՝

- գտնել երկու հարևան մատրիցների ընդհանուր կողերի 1-երի քանակը;
- հաշվել երկու հարևան մատրիցների ֆիթը:

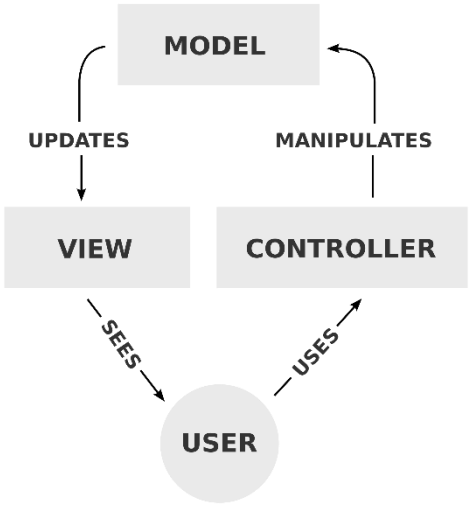
Ծրագրային հատվածը կարելի է ներկայացնել հետևյալ սխեմայի միջոցով:



Նկ. 4.5. Տեխնոլոգիական լուծումները

Գրաֆիկական մասի համար պատասխանատու է WPF (windows presentation foundation), սա պլատֆորմ է արտացանցային ծրագրեր գրելու համար, սրա միջոցով կառուցված էլեմենտները XAML ֆորմատով պահվում են տվյալների բազայում ENTITY FRAMEWORK-ի միջոցով, որը .NET-ի նորագույն ORM (Object relational mapping)-ն է,

որը կլասները կապում է տվյալների հենքերի օբյեկտների հետ: Մատրիցային բաշխման ինտերֆեյսի համար օգտագործվել է MVC (Model view controller) նմուշը, այն համացանցային ծրագրերի համար, նա ունի հետևյալ կառուցվածքը:



4.4. Եզրակացություն

Այսպիսով, ներկայացվեց նոր էվրիստիկ ալգորիթմը և մշակված MIC Host ծրագրային համակարգը [67]: Ալգորիթմի իրականացման համար մշակվել է matrix.dll մատրիցների հետ աշխատելու ֆունկցիաների գրադարանը, որը ներդրվել է github ազատ օգտագործման գրադարանների ռեսուրսում: Ծրագրային համակարգի աշխատանքի արդյունավետությունը ստուգելու համար այն փորձարկվել է և ներդրվել երկու փայտամշակման մասնավոր արդյունաբերության արտադրամասերում: Համակարգի կիրառման շնորհիվ բարձրացել է արտադրության արդյունավետությունը, կրճատվել են ծախսերը ավելի քան 20 տոկոսով:

ԵԶՐԱԿԱՑՈՒԹՅՈՒՆ

Այսպիսով, աշխատանքում ստացվել են հետևյալ հիմնական արդյունքները.

- Մշակվել է եղանակ և ծրագրային միջավայր, որը մոդելավորում է կամայական ուրվագծով երկչափ օբյեկտները: Այն ներառում է գործիքներ պարզ երկրաչափական օբյեկտների ստեղծման համար, դրանց հետ փոփոխություններ կատարելու, ինչպես նաև բարդ ուրվագծեր ստեղծելու համար: [65]
- Առաջարկվել է եղանակ երկրաչափական մոդելների դիսկրետացման համար: [66]
- Առաջարկվել է էվրիստիկ արտացանց ալգորիթմ կամայական ուրվագծով երկչափ մարմինների ձևման խնդրի արդյունավետ լուծման համար:[67]
- Մշակվել է matrix.dll մատրիցների հետ աշխատելու ֆունկցիաների գրադարան:[67]
- Մշակվել է ծրագիր, որը իրականացնում է կամայական ուրվագծով երկչափ մարմինների ձևումը տված հումքի պայմաններում:[67]

ՕԳՏԱԳՈՐԾՎԱԾ ԳՐԱԿԱՆՈՒԹՅԱՆ ՑԱՆԿԸ

1. Դերձյան Հ. Ս. Ռեսուրսների՝ ըստ ժամանակի հավասարաչափ բաշխման խնդրի մաթեմատիկական մոդելի և համապատասխան ալգորիթմի ու ծրագրային փաթեթի մշակումը, թեկնածուական ատենախոսություն, Երևան, 2015.
2. Էքսուզյան Ս. Հ. Ձևման խնդիրների մոդելավորումը և դրանց կիրառումը տեխնոլոգիական գործընթացում, թեկնածուական ատենախոսություն, Երևան, 2011.
3. Бабаев Ф.В. Оптимальный раскрой материалов с помощью ЭВМ/ Ф.В. Бабаев. М: Машиностроение, 168 с., 1982.
4. Гимади Э.Х.. О некоторых математических моделях и методах планирования крупномасштабных проектов. Модели и методы оптимизации. Труды Института математики. Новосибирск. Наука. Сиб. Отд–ние. с. 89–115. 1988.
5. Канторович Л. В., Залгаллер В. А. Рациональный раскрой промышленных материалов. Новосибирск: Наука, 1971.
6. Канторович Л. В, Математические методы организации и планирования производства, Ленинградский Гос. Унив., 1939.
7. Скатерной В.А., Оптимизация раскроя материалов в легкой промышленности, Изд. Легпромбытиздат, 143с., 1989.
8. Смирнов А. В.. О задаче упаковки в контейнеры. УМН, том 46, выпуск 4(280), с. 173–174, 1991.
9. Allen S.D., Burke E.K., Mareček J. A space-indexed formulation of packing boxes into a larger box. Operations Research Letters. Vol. 40. Is. 1. P. 20–24., 2012. Инженерный вестник, №06, 2015 579
10. Alvarez-Valdes R., Parreno F., Tamarit J.M. A GRASP algorithm for constrained two-dimensional non-guillotine cutting problems. Journal of Operational Research Society, No. 56(4), 414–425, 2005.

11. Alvarez-Valdes R., Parreno F., Tamarit J.M. A tabu search algorithm for two -dimensional non-guillotine cutting problems. European Journal of Operational Research, No. 183(3), 1167–1182, 2007.
12. Araya I., Riff M.C. A beam search approach to the container loading problem. Computers & Operations Research. Vol. 43. P. 100–107. 2014.
13. Bansal N., Caprara A. and Sviridenko M., A new approximation method for set covering problems with applications to multidimensional bin packing. SIAM J. Comput. 39, 4, pp. 1256–1278, 2009.
14. Belov G., Problems, models and algorithms in one- and two-dimensional cutting, Dresden, 118p, 2004.
15. Bengtsson B. E., Packing rectangular pieces. A heuristic approach, The Computer Journal, vol. 25, No. 3, pp. 353-357, 1982.
16. Bin packing problem, https://en.wikipedia.org/wiki/Bin_packing_problem
17. Bischoff E.E. Three-dimensional packing of items with limited load bearing strength. European Journal of Operational Research. Vol. 168. Is. 3. P. 952– 966. 2006.
18. Bischoff E.E., Marriott M.D. A comparative evaluation of heuristics for container loading. European Journal of Operational Research. Vol. 44. Is. 2. P. 267–276, 1990.
19. Bischoff E.E., Ratcliff M.S.W. Issues in the development of approaches to container loading. Omega-international Journal of Management Science. Vol. 23. Is. 4. P. 377–390. 1995.
20. Bortfeldt A. A heuristic for multiple container loading problems. OR Spektrum. Vol. 22. Is. 2. P. 239–261. 2000.
21. Bortfeldt A., Gehring H. A hybrid genetic algorithm for the container loading problem. European Journal of Operational Research. Vol. 131. Is. 1. P. 143– 161. 2001.
22. Bortfeldt A., Gehring H. Applying tabu search to container loading problems // (Symposium on Operations Research (SOR'97) Jena, September 3–5, 1997). Operations Research Proceedings 1997. Springer Berlin Heidelberg. P. 533–538. 1998.

23. Bortfeldt A., Gehring H., Mack D. A parallel tabu search algorithm for solving the container loading problem. Parallel Computing. Vol. 29. Is. 5. P. 641–662. 2003.
24. Bortfeldt A., Wascher G. Constraints in container loading: a state-of-the-art review, European Journal of Operational Research, Vol. 229, Is.1, pp. 1–20, 2013.
25. Caprara A., Packing 2-dimensional bins in harmony, Proceedings of the Symposium on Foundations of Computer Science (FOCS), pp. 490–499, 2002.
26. Che C. H., Huang W., Lim A., Zhu W. The multiple container loading cost minimization problem. European Journal of Operational Research. Vol. 214. Is. 3. P. 501–511. 2011.
27. Chen C.S., Lee S.M., Shen Q.S. An analytical model for the container loading problem. European Journal of Operational Research. Vol. 80. Is. 1. P. 68–76. 1995.
28. Chien C.F., Deng, J.F. A container packing support system for determining and visualizing container packing patterns, Decision Support Systems, vol 37, No. 1, pp. 23–34, 2004.
29. Chien C.F., Lee C.Y., Huang Y.C., Wu W.T. An efficient computational procedure for determining the container loading pattern. Computers & Industrial Engineering. Vol. 56. Is.3. P. 965–978. 2009.
30. Chien C.F., Wu W.T. A recursive computational procedure for container loading. Computers & Industrial Engineering. Vol. 35. Is. 1-2. P. 319–322. 1998.
31. Chua C.K., Narayanan V., Loh J. Constraint-based spatial representation technique for the container packing problem. Integrated Manufacturing Systems. Vol. 9. Is. 1. P. 23–33. 1998.
32. Chung F., Garey M. and Johnson D., On packing two-dimensional bins, SIAM J. Alg. Disc. Meth. 3, 1, pp. 66–76, 1982.
33. Chwatal A. M., Pirkwieser S., Solving the Two-Dimensional Bin-Packing Problem with Variable Bin Sizes by Greedy Randomized Adaptive Search Procedures and Variable Neighborhood Search. EUROCAST (1) 2011, pp. 456-463, 2011.

34. Crainic T.G., Perboli G., Tadei R. Extreme point-based heuristics for three-dimensional bin packing . INFORMS Journal on Computing. Vol. 20. Is. 3. P. 368–384. 2008.
35. Crainic T.G., Perboli G., Tadei R. TS2PACK: a two-level tabu search for the three-dimensional bin packing problem. European Journal of Operational Research. Vol. 195. Is. 3. P. 744–760. 2009.
36. Cutting stock problem. https://en.wikipedia.org/wiki/Cutting_stock_problem
37. Davies A.P., Bischoff E.E. Weight distribution considerations in container loading. European Journal of Operational Research. Vol. 114. Is. 3. P. 509– 527., 1999. Инженерный вестник, №06, 2015.
38. Dereli T., Das G.S. A hybrid 'bee(s) algorithm' for solving container loading problems. Applied Soft Computing. Vol. 11. Is. 2. P. 2854–2862. 2011.
39. Dyckhoff H., A typology of cutting and packing problems, European Journal of Operational Research, vol. 44, pp. 145-159, 1990.
40. Egeblad J., Pisinger D. Heuristic approaches for the two- and three-dimensional knapsack packing problem, Computers & Operations Research, vol. 36, Is. 4, pp. 1026–1049, 2009.
41. Eisenbrand F., Palvoelgyi D. and T. Rothvoss, Bin Packing via Discrepancy of Permutations, Symposium on Discrete Algorithms (SODA 2011), San Francisco, USA, pp. 22-25, 2011.
42. Eley M. A bottleneck assignment approach to the multiple container loading problem. OR Spektrum. Vol. 25. Is. 1. P. 45–60. 2003.
43. Eley M. Solving container loading problems by block arrangement. European Journal of Operational Research. Vol. 141. Is. 2. P. 393–409. 2002.
44. Faina L. A global optimization algorithm for the three-dimensional packing problem. European Journal of Operational Research. Vol. 126. Is. 2. P. 340– 354. 2000.
45. Farøe O., Pisinger D., Zachariasen M. Guided local search for the three-dimensional bin-packing problem. INFORMS Journal on Computing. Vol. 15. Is. 3. P. 267–283. 2003.

46. Fekete S.P., Schepers J. A combinatorial characterization of higher-dimensional orthogonal packing. Mathematics of Operations Research. Vol. 29. Is. 2. P. 353–368. 2004.
47. Fekete S.P., Schepers J. A new exact algorithm for general orthogonal d-dimensional knapsack problems. Conference: European Symposium on Algorithms. (5th Annual European Symposium Graz, Austria, September 15–17, 1997) Proceedings. Berlin: Springer - Heidelberg. P. 144 – 156. 1997.
48. Gehring H., Bortfeldt A. A parallel genetic algorithm for solving the container loading problem. International Transactions in Operational Research. Vol. 9. Is. 4. P. 497–511. 2002.
49. Gehring H., Bortfeldt A., "A genetic algorithm for solving the container loading problem", International Transactions in Operational Research, vol 4., Is.5-6, pp. 401–418, 1997.
50. Gehring H., Menschner K., Meyer M. A computer-based heuristic for packing pooled shipment containers. European Journal of Operational Research. Vol. 44. Is. 2. P. 277–288. 1990.
51. George J.A. A method for solving container packing for a single size of box. Journal of the Operational Research Society. Vol. 43. Is. 4. P. 307–312. 1992.
52. George J.A., Robinson D.F. A heuristic for packing boxes into a container. Computers & Operations Research. Vol. 7. Is. 3. P. 147–156. 1980.
53. Gonçalves J.F., Resende M.G. A biased random key genetic algorithm for 2D and 3D bin packing problems. International Journal of Production Economics. Vol. 145. Is. 2. P. 500–510. 2013.
54. Gonçalves J.F., Resende M.G.C. A parallel multi-population biased random-key genetic algorithm for a container loading problem. Computers & Operations Research. Vol. 39. Is. 2. P. 179–190. 2012.
55. Haessler R.W., Talbot F.B. Load planning for shipments of low density products, European Journal of Operational Research, vol 44, Is. 2, pp. 289– 299, 1990.

56. Han C.P., Knott K., Egbelu P.J. A heuristic approach to the three dimensional cargo loading problem. Computers & Industrial Engineering. Vol. 11. Is. 1 - 4. P. 109–113. 1986.
57. He K., Huang W. A caving degree based flake arrangement approach for the container loading problem. Computers & Industrial Engineering. Vol. 59. Is. 2. P. 344–351. 2010.
58. He K., Huang W. An efficient placement heuristic for three-dimensional rectangular packing. Computers & Operations Research. Vol. 38. Is.1. P. 227– 233. 2011. Инженерный вестник, №06, 2015.
59. Hemminki J. Container Loading with Variable Strategies in Each Layer. Institute for Applied Mathematics, University of Turku, Turku, Finland. 1994.
60. Hopper E., Turton B.C.H. An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. European Journal of Operational Research, No. 128(1), 34–57, 2001.
61. Huang W., He K. A caving degree approach for the single container loading problem. European Journal of Operational Research. Vol. 196. Is. 1. P. 93– 101. 2009.
62. Huang W., He K. A new heuristic algorithm for cuboids packing with no orientation constraints. Computers & Operations Research. Vol. 36. Is. 2. P. 425–432. 2009.
63. Ivancic N.J., Mathur K., Mohanty B.B. An integer programming based heuristic approach to the three-dimensional packing problem. Journal of Manufacturing and Operations Management. Vol. 2. Is. 4. P. 268–298. 1989.
64. Jin Z., Ito T., Ohno K. The three-dimensional bin packing problem and its practical algorithm. JSME International Journal Series C - Mechanical Systems, Machine Elements and Manufacturing. Vol. 46. Is. 1. P. 60 – 66. 2003.
65. Keyan A. Modeling of Irregularly Shaped Two – Dimensional Objects, Transactions of IIAP NAS RA, Mathematical Problems of Computer Science, vol. 44, pp. 154-160, 2015.
66. Keyan A. Discretization of geometric models, Transactions of IIAP NAS RA, Mathematical Problems of Computer Science, vol. 45, pp. 99-105, 2016.

67. Keyan A. Single large bin packing problem of irregularly shaped two – dimensional objects, Материалы Второго Международного студенческого симпозиума “Математика и информационные технологии в приложениях”, стр. 74-76, Сочи, 2016.
68. Khan A. Approximation algorithms for multidimentional bin packing, PhD thesis, 2015.
69. Kotov V. M., Dayong Cao. A heuristic algorithm for the two-dimensional single large bin packing problem. Buletinul Academiei de Stinte a Republicii Moldova. Matematica, No. 3(64), pp. 23-28, 2010.
70. Lai K.K., Chan J.W.M. Developing a simulated annealing algorithm for the cutting stock problem. Computers & Industrial Engineering. Vol. 32. Is. 1. P. 115–127. 1997.
71. Lai K.K., Xue J., Xu B. Container packing in a multi-customer delivering operation. Computers & Industrial Engineering. Vol. 35. In. 1-2. P. 323–326. 1998.
72. Layeb A., Chenche S., A Novel GRASP Algorithm for Solving the Bin Packing Problem, International Journal of Information Engineering and Electronic Business (IJIEEB), Vol. 4, No. 2, pp. 8-14, 2012.
73. Lim A., Rodrigues B., Wang Y. A multi-faced buildup algorithm for three-dimensional packing problems. Omega. Vol. 31. Is. 6. P. 471–481. 2003.
74. Lim L.C.A., Ma H., Xu J., Zhang X. An iterated construction approach with dynamic prioritization for solving the container loading problems. Expert Systems with Applications. Vol. 39. Is. 4. P. 4292–4305. 2012.
75. Lins L., Lins S., Morabito R. An n-tet graph approach for non-guillotine packings of n-dimensional boxes into an n-container. European Journal of Operational Research. Vol. 141. Is. 2. P. 421–439. 2002.
76. Liu J., Yue Y., Dong Z., Maple C., Keech M. A novel hybrid tabu search approach to container loading. Computers & Operations Research. Vol. 38. Is. 4. P.797–807. 2011.
77. Lodi A., Martello S. and D. Vigo, Approximation algorithms for the oriented two-dimensional bin packing problem, European Journal of Operational Research, Vol. 112, pp. 158-166. 1999.

78. Lodi A., Martello S., Monaci M., Two-dimensional packing problems: A survey. European Journal of Operational Research (Elsevier), vol. 141, pp. 241–252, 2002.
79. Lodi A., Martello S., Vigo D. Heuristic algorithms for the three-dimensional bin packing problem. European Journal of Operational Research. Vol. 141. Is. 2. P. 410–420. 2002.
80. Lodi A., Martello S., Vigo D. , Recent advances on two-dimensional bin packing problems, Discrete Applied Mathematics, Vol. 123, Issues 1–3, 15 November, P. 379–396, 2002.
81. Loh T.H., Nee A.Y.C. A packing algorithm for hexahedral boxes. Proceedings of the Conference of Industrial Automation. Singapore. P. 115–126. 1992.
82. Mack D., Bortfeldt A., Gehring H. A parallel hybrid local search algorithm for the container loading problem. International Transactions in Operational Research. Vol. 11. Is. 5. P. 511–533. 2004.
83. Martello S., Pisinger D., Vigo D. The three-dimensional bin packing problem. Operations Research. Vol. 48. Is. 2. P. 256–267. 2000.
84. Mohanty B.B., Mathur K., Ivancic N.J. Value considerations in three-dimensional packing: a heuristic procedure using the fractional knapsack problem. European Journal of Operational Research. Vol. 74. Is. 1. P. 143 – 151. 1994.
85. Morabito R., Arenales M. An AND/OR-graph approach to the container loading problem, International Transactions in Operational Research, Vol. 1, Is. 1, pp. 59–73, 1994.
86. Moura A., Oliveira J.F. A GRASP approach to the container-loading problem. IEEE Intelligent Systems. Vol. 20. P. 50–57. 2005.
87. Ngoi B.K.A., Tay M.L., Chua E.S. Applying spatial representation techniques to the container packing problem. International Journal of Production Research. Vol. 32. Is. 1. P. 111–123. 1994.
88. Padberg M. Packing small boxes into a big box. Mathematical Methods of Operations Research. Vol. 52. Is. 1. P. 1–21. 2000.

89. Parreño F., Alvarez-Valdes R., Oliveira J., Tamarit J. Neighborhood structures for the container loading problem: a VNS implementation. Journal of Heuristics. Vol. 16. Is. 1. P. 1–22. 2010.
90. Parreño F., Alvarez-Valdes R., Tamarit J.M., Oliveira J.F. A maximal-space algorithm for the container loading problem. INFORMS Journal on Computing. Vol. 20. Is. 3. P. 412–422. 2008.
91. Pisinger D. Heuristics for the container loading problem. European Journal of Operational Research. Vol. 141. Is. 2. P. 382–392. 2002.
92. Ratcliff M.S.W, Bischoff E.E. Allowing for weight considerations in container loading. OR Spektrum. Vol. 20. Is. 1. P. 65–71. 1998.
93. Ren J., Tian Y., Sawaragi T. A tree search method for the container loading problem with shipment priority. European Journal of Operational Research. Vol. 214. Is. 3. P. 526–535. 2011.
94. Scheithauer G. Algorithms for the container loading problem, Operations Research Proceedings, pp. 445–452, 1991.
95. Takahara S. A multi-start local search approach to the multiple container loading problem. Greedy Algorithms / Witold Bednorz. Vienna: IN-TECH. 586 p. Ch. 4. P. 55–68, 2008.
96. Takahara S. A simple meta-heuristic approach for the multiple container loading problem. Browse Conference Publications. IEEE International Conference on Systems, Man and Cybernetics. 8-11 Oct. 2006. Taipei. Vol. 3. P. 2328–2333. 2006.
97. Wang P., Valenzeva L., Data set generation for rectangular placement problems, European Journal of Operational Research, vol. 134, no. 2, pp.378-391, 2001.
98. Wang Z., Li K.W., Levy J.K. A heuristic for the container loading problem: a tertiary-tree-based dynamic space decomposition approach, European Journal of Operational Research, vol 191, Is.1, pp. 86–99, 2008.

99. Wang P., Wascher G., Special issue: Cutting Packing Problems, Europen Journal of Operational research. 141p. 2002.

100. Wascher G., Haussner H., Schumann H., “An improved typology of cutting and packing problems”. European Journal of Operational Research, No. 183(3), pp. 1109–1130, 2007.

101. Wu Y., Li W., Goh M., De Souza R. Three-dimensional bin packing problem with variable bin height. European Journal of Operational Research. Vol. 202. Is. 2. P. 347–355. 2010.

102. Xue J., Lai K.K. Effective methods for a container packing operation. Mathematical and Computer Modelling. Vol. 25. Is. 2. P. 75–84. 1997.

103. Zhang D., Peng Y., Leung S.C.H. A heuristic block-loading algorithm based on multi-layer search for the container loading problem. Computers & Operations Research. Vol. 39. Is. 10. P. 2267–2276. 2012.

104. Zhu W., Huang W., Lim A. A prototype column generation strategy for the multiple con-tainer loading problem. European Journal of Operational Research. Vol. 223. Is. 1. P. 27–39. 2012.

105. Zhu W., Lim A. A new iterative-doubling Greedy-Lookahead algorithm for the single con-tainer loading problem. European Journal of Operational Research. Vol. 222. Is. 3. P. 408–417. 2012.

106. Zhu W., Oon W.C., Lim A., Weng Y. The six elements to block-building approaches for the single container loading problem. Applied Intelligence. Vol. 37. Is. 3. P. 431–445. 2012.