Rīgas Valsts 3. ģimnāzija

Zinātniskās pētniecības darbs

Matemātika
Programmēšana

Minimālā Eiklīda aptverošā koka garuma atkarība no papildus punktu skaita

Darba autors: Ēriks Cepurīts
11.A klases skolnieks

Darba vadītāja: Magone Una Rauba
programmēšanas skolotāja

Konsultants: Sergejs Meļņiks
matemātikas maģistrs

Rīga
2020. gads

Anotācija

Darba autors ir Rīgas Valsts 3. ģimnāzijas 11.A klases skolnieks Ēriks Cepurīts. Darbs ir veikts ar darba vadītājas Magones Unas Raubas un konsultanta Sergeja Meļņika palīdzību par tēmu "Minimālā Eiklīda aptverošā koka garuma atkarība no papildus punktu skaita".

Darba mērķis bija izpētīt, cik liela nozīme ir kustīgajām virsotnēm un to skaitam noteiktajā grafā, veidojot minimālo Eiklīda aptverošo koku un nosakot to garumu. Tika secināts, ka konfigurācijās ar 10, 15, 20 un 30 fiksētām virsotnēm, visoptimālākais papildus virsotņu skaits ir 50%-70% no fiksēto punktu skaita, 40 fiksētām virsotnēm – 38%, 50 un 60 fiksētām virsotnēm – 10%-20%.

Galvenie atslēgvārdi: minimālais Eiklīda aptverošais koks, Neldera-Mida algoritms.

Annotation

The following research was made by Eric Tsepurit student of Riga State Gymnasium No.3 from form 11A with help of Sergey Melnik as a consultant and Magone Una Rauba as a project leader. The research topic is "Minimal Euclid spanning tree's length depending on the number of additional apexes".

The aim was to evaluate the importance of additional apexes and their count creating minimal Euclid spanning tree and calculating it's length. According to the results, the most optimal additional apex number for configurations with 10, 15, 20 and 30 fixed apexes is 50%-70% from fixed point number, for configuration with 40 fixed apexes – 38%, for 50 and 60 – 10%-20%.

Key words: minimal Euclid spanning tree, Nelder-Mead algorithm.

Satura rādītājs

Ievads

Tēma: Minimālā Eiklīda aptverošā koka garums atkarībā no papildus punktu skaita.

Pētījuma jautājums: Kā papildus punktu skaits ietekmē minimālā Eiklīda aptverošā koka garumu?

Hipotēze: Ja papildus punktu skaits sastāda 30%-40% no fiksēto punktu skaita, tad minimālā Eiklīda aptverošā koka garums ir vismazākais.

Optimizācija un lietderīga resursu izmantošana ir ļoti svarīgas lietas mūsdienās. Īsākā savienojuma jeb minimālā Eiklīda aptverošā koka atrašana *(Stainer tree problem)* ir sens uzdevums, kas vēl nav pilnīgi atrisināts. Ir vairāki punkti uz plaknes, kurus ir jāsavieno tā, lai varētu no jebkura nokļūt uz jebkuru un ceļu garums, kas šos punktus savieno būtu minimāls (var pievienot papildus punktus). Šāda veida algoritma nepieciešamība ir vairākās nozarēs: gan būvniecībā (ceļi, elektrolīnijas, caurules, kabeļi), gan inženierzinātnē (neironu tīkli, mazu mikroshēmu izstrāde) utt.

Darba uzdevumi:

1. Sameklēt informāciju par īsākā savienojuma uzdevumu.
    a. Sameklēt informāciju par Prima algoritmu.
    b. Sameklēt informāciju par Neldera-Mida algoritmu.
2. Apkopot savākto informāciju.
3. Izveidot programmu, kas veido minimālo Eiklīda aptverošo koku, izmantojot Prima algoritmu *(Prim's algorithm)*. Tiks izmantota Java.
4. Izvedot programmu, kas izskaitļo koka garumu. Tiks izmantota Java.
5. Izveidot programmu, kas atrod optimizētāku vietu papildus punktiem, izmantojot Neldera-Mida algoritmu *(Nelder–Mead method)*. Tiks izmantota Java.
6. Izveidot programmu, kas vizualizē visu procesu.
7. Veikt datu ieguvi attiecīgi pētījuma gaitas plānam.

1. Grafu teorija

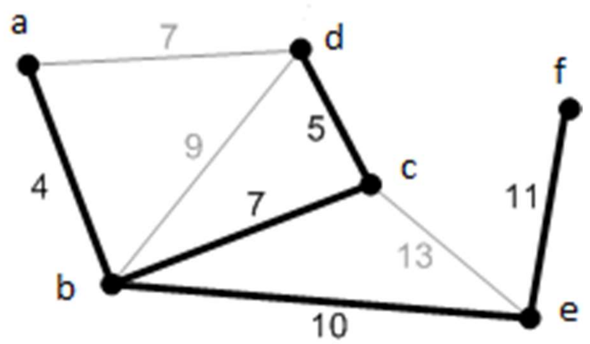
1.1. Grafu teorijas elementi

Grafu teorija – diskrētās matemātikas nozare, kas pēta grafu kombinatoriskās un topoloģiskās īpašības.
Grafs – punktu (virsotņu) kopa kopā ar šķautnēm, kas tos savieno.
Sakarīgs grafs – grafs, starp kura jebkurām divām virsotnēm ir kaut viens savienojums.
Neorientēts grafs – grafs, kura šķautnēm nav virziena.
Koks – grafs, starp kura jebkurām divām virsotnēm ir tikai viens savienojums.
Minimālais Eiklīda aptverošais koks – minimāla garuma koks Eiklīda metrikā.

1.2. Īsākā savienojuma atrašana *(Stainer Tree Problem)*

Uzdevums ir nosaukts par godu Šveices zinātniekam un matemātiķim Jakobam Šteineram *(Jakob Steiner).*

Ideja: Ir vairāki punkti uz plaknes, kurus ir jāsavieno tā, lai varētu no jebkura nokļūt uz jebkuru un ceļu garums, kas šos punktus savieno būtu minimāls (var pievienot papildus punktus). Citiem vārdiem, dotas grafa virsotnes un ir jāizveido minimālais Eiklīda aptverošais koks (skat. attēlu Nr. 1).



Attēls Nr. 1

2. Algoritmi

Šobrīd jau eksistē dažādi algoritmi un risināšanas veidi, kas ļauj iegūt minimālo Eiklīda aptverošo koku. Konkrēti būs izmantoti Prima algoritms minimālā koka atrašanai un Neldera-Mida algoritms papildus punktu izvietošanai.

2.1. Prima Algoritms

Algoritms ir nosaukts par godu amerikāņu matemātiķim Robertam Primam *(Robert Prim)*, kurš atklāja šo algoritmu 1957. gadā. Interesanti, ka šo algoritmu atklāja Čehijas matemātiķis Vojteks Jarņiks *(Vojtěch Jarník)* jau 1930 gadā. Nedrīkst nepieminēt Edgaru Dejkstru *(Edsger Dijkstra)*, kas 1959. gadā izveidoja tādu pašu algoritmu.

Dotie lielumi:

Dots sakarīgs neorientēts grafs *G* ar *n* virsotnēm un *m* šķautnēm (katrai šķautnei dots garums). Ir jāatrod minimāls Eiklīda aptverošais koks, kas saista visas grafa virsotnes un ir minimāli garš (šķautņu garumu summa ir minimāla).

Algoritms

1. Izvēlas jebkuru grafa *G* virsotni.
2. Atrod grafa virsotni, kas ir vistuvāk sākotnēji izvēlētai virsotnei.
3. Savieno atrasto grafa virsotni un sākotnēji izvēlēto virsotni. Izveidojas koks.
4. Atrod grafa virsotni, kas ir vistuvāk kādai no koka virsotnēm.
5. Savieno atrasto grafa virsotni ar vistuvāko koka virsotni.
6. Atkārto 4. un 5. soli, kamēr visas virsotnēs nebūs iesaistītas šajā kokā.
7. Iegūtais koks ir minimālais aptverošais koks dotajam grafam.[1]

2.2. Neldera-Mida algoritms

Optimizācijas metode, kas tiek izmantota funkcijām ar vairākiem argumentiem ($f(x_1, x_2, \ldots, x_n)$). Algoritma ideja ir simpleksa *(skat. 2.4. Terminu skaidrojums [1])* pārvietošana un deformēšana ekstrēmā punkta *(skat. 2.4. Terminu skaidrojums [2])* tuvumā. Vizuāls algoritma attēlojums ir parādīts blok-shēmā tālāk (skat. attēlu Nr. 6).

Dotie lielumi:

Dota funkcija $f(x_1, x_2, \ldots, x_n)$ ar n argumentiem. Ir jāatrod funkcijas minimums.

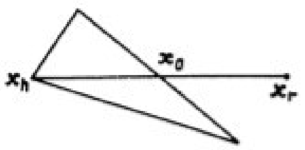
Atspoguļošanas koeficients $\alpha > 0$, parasti $\alpha = 1$.
Saspiešanas koeficients $1 > \beta > 0$, parasti $\beta = 0{,}5$.
Izstiepšanas koeficients $\gamma > 0$, parasti $\gamma = 2$.

[1] Minimālais aptverošais koks. Prima algoritms. [tiešsaite]

Algoritms

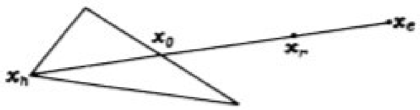
1. Izvēlas jebkuras n vērtības x.
2. Pieņem sākuma punktu ar koordinātām $p0\ (x_1;\ x_2;\ \dots;\ x_n)$
3. Atrod $f(p0)$.
4. Izveido vēl n+1 tādus p punktus, pieskaitot katrai nākamajai koordinātai +1.
   a. $p_1\ (x_1 + 1;\ x_2;\ \dots;\ x_n)$
   b. $p_2\ (x_1;\ x_2 + 1;\ \dots;\ x_n)$
   c. $p_n\ (x_1; x_2; \dots; x_n + 1)$

   $p_0, p_1, p_2, \dots, p_n$ – n-dimensionālā deformējamā simpleksa virsotņu koordinātas.
5. Izvēlas lielāko, otro pēc lieluma un vismazāko funkcijas vērtības no $f(p_1)$, $f(p_2)$, …, $f(p_n)$ un tām atbilstošos punktus p.
6. Apzīmē:
   a. f_h – vislielākā funkcijas vērtība
   b. x_h – p virsotne, ar kuru $f(p) = f_h$
   c. f_g – otrā lielāka funkcijas vērtība
   d. x_g – p virsotne, ar kuru $f(p) = f_g$
   e. f_l – vismazākā funkcijas vērtība
   f. x_l – p virsotne, ar kuru $f(p) = f_l$
7. Atrod smaguma centru visām x virsotnēm, izņemot x_h.
   $$x_0 = \frac{1}{n}\sum_{i \neq h} x_i$$
8. Atrod $f(x_0)$.
9. Atspoguļo virsotni x_h pret punktu x_0 un iegūst x_r.
   $x_r = (1+\alpha)x_0 - \alpha x_h$ (skat. attēlu Nr. 2)



Attēls Nr. 2

10. Atrod $f(x_r)$.
11. Salīdzina f_r un f_l:
    a. Ja $f_r < f_l$, tad ir iegūts jauns punkts, kur funkcija ir vismazākā. Tātad šis pārvietošanās virziens ir labvēlīgs. Atrod x_e.
    $x_e = \gamma x_r + (1-\gamma)x_0$. Atrod $f(x_e)$. (skat. attēlu Nr. 3)

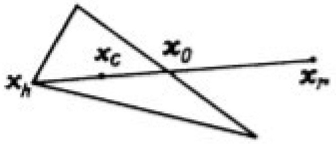


Attēls Nr. 3

       i. Ja $f_e < f_l$, tad maina virsotni x_h uz punktu x_e. Ja rezultāts nav pietiekami precīzs, atgriežas uz soli 5.
       ii. Ja $f_e \geq f_l$, tad atmetam punktu x_e, jo tas ir par tālu. Tātad maina virsotni x_h uz x_r. Ja rezultāts nav nepietiekami precīzs, atgriežas uz soli 7.
    b. Ja $f_r > f_l$, bet $f_r \leq f_g$, tad nomaina x_h uz x_r. Ja rezultāts nav nepietiekami precīzs, atgriežas uz soli 5.
    c. Ja $f_r > f_l$ un $f_r > f_g$, tad iet tālāk uz soli 12.
12. Salīdzina f_r un f_h:

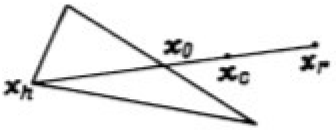
a. Ja $f_r > f_h$, tad iet tālāk uz soli 13.
b. Ja $f_r < f_h$, tad nomaina x_h uz x_r un f_h vērtību uz f_r.

13. Šajā brīdī $f_r > f_h$, tātād pārvietojums ir pārāk liels. Lai kompensētu šo faktu, izmanto saspiešanu:
    a. Atrod x_c.
        i. Ja $f_r > f_h$, tad
        $x_c = \beta x_h + (1 - \beta)x_0$ (skat. attēlu Nr. 4)



Attēls Nr. 4

        ii. Ja $f_r < f_h$, tad nomaina punktu x_h uz x_r un
        $x_c = \beta x_r + (1 - \beta)x_0$ (skat. attēlu Nr. 5)



Attēls Nr. 5

    b. Atrod $f(x_c)$.

14. Salīdzina f_c un f_h:
    a. Ja $f_c < f_h$, tad nomaina punktu x_h uz x_c. Ja rezultāts nav pietiekami precīzs, atgriežas un soli 5.
    b. Ja $f_c > f_h$, tad atkal nav atrasta mazāka vērtība. Pāriet uz soli 15.

15. Ja neviens paņēmiens nevar dot nepieciešamos rezultātus, ir jāsamazina deformējamais simplekss par divām reizēm ar atskaites punktu x_l.

    $Katra\ virsotne\ x_i, izņemot\ x_l, = \frac{x_i + x_l}{2}, kur\ i = 1, 2, \dots, n + 1$

    $Katra\ funkcijas\ vērtība\ f_i = f(x_i), kur\ i = 1, 2, \dots, n + 1$

    Ja rezultāts nav pietiekami precīzs, tad atgriežas uz soli 7. Pretējā gadījumā virsotnes atrodas ļoti tuvu viena otrai. Var secināt, ka funkcijas minimums ir ļoti tuvu. Tātad x_l ir funkcijas minimums ar tuvinājumu.[2]

2.3. Precizitātes pārbaude

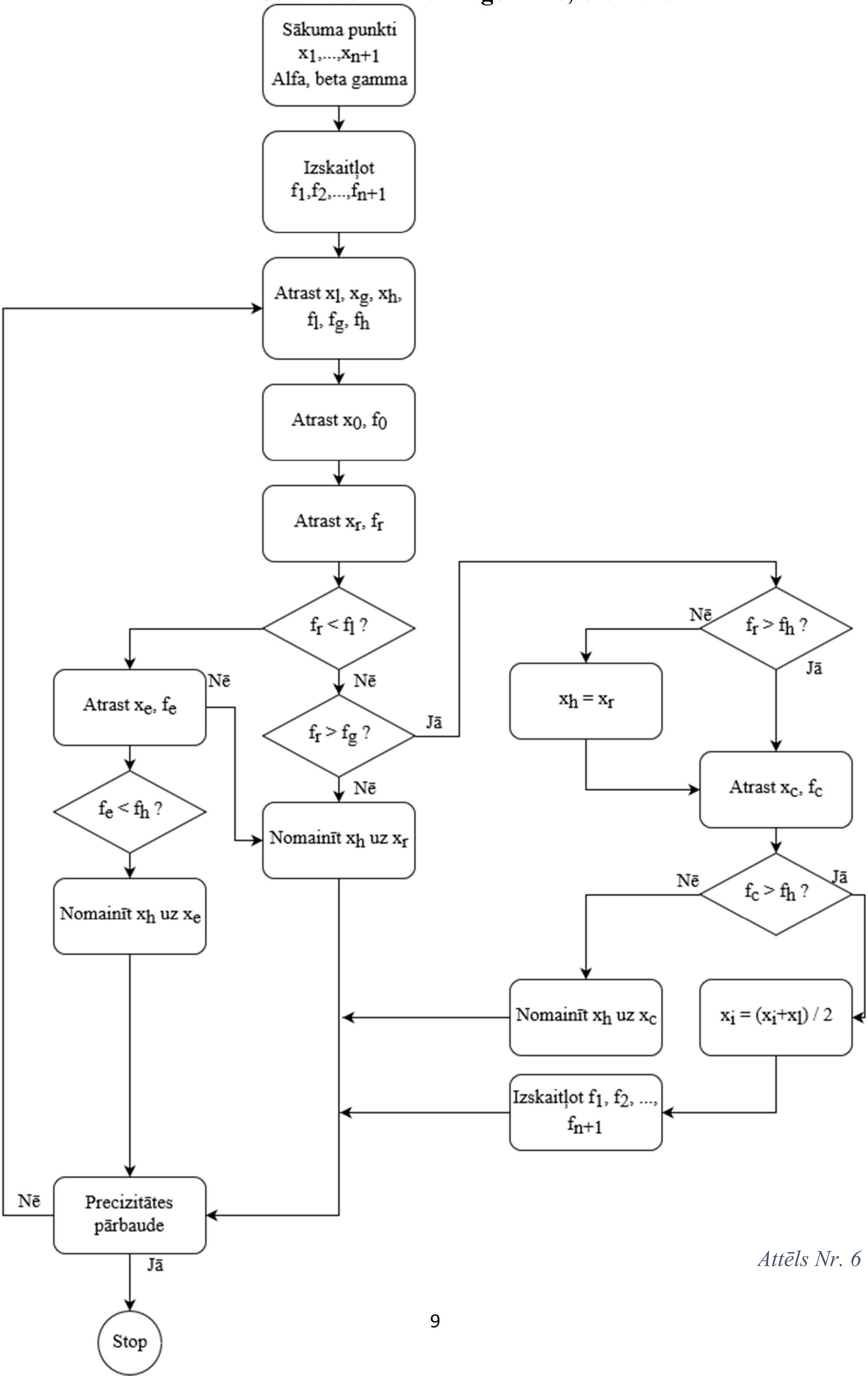
Šis algoritms minimizē funkciju ar tuvinājumu un nedod precīzu atrisinājumu, toties šo tuvinājumu var definēt pirms algoritma sākšanas. Dotajā pētījumā algoritms apstāsies pie nosacījuma, kad simpleksa šķautņu garumu summa būs mazāka par noteiktu vērtību *(epsilonu)*.

2.4. Terminu skaidrojums

1. Simplekss – ģeometriska figūra, trijstūris n-dimensiju telpā.
2. Ekstrēmais punkts – punkts, kurā funkcija maina savu dinamiku (beidzas augšana un sākas dilšana vai otrādi).

[2] Basic Optimisation Methods. London: Edward Arnold (Publishers) Limited, 1984

2.5. Neldera-Mida algoritms, blok-shēma



Attēls Nr. 6

3. Pētījuma gaita

Šajā darbā tiks pētīts, kā mainās minimālā aptverošā koka garums atkarībā no papildus punktu skaita.

1. Izvieto fiksētos punktus uz plaknes.
2. Izmantojot Prima algoritmu, iegūst minimālo Eiklīda aptverošo koku.
3. Nosaka to garumu, saskaitot visu šķautņu garumu summu.
4. Tad izvēlas papildus punktu skaitu.
5. Ar Neldera-Mida algoritma palīdzību atrod labākās vietas visiem papildus punktiem.
6. Izmantojot Prima algoritmu, iegūst jaunu minimālo Eiklīda aptverošo koku.
7. Nosaka to garumu, saskaitot visu šķautņu garumu summu.
8. Salīdzina iegūto koku garumus.
9. Atkārto, izvēloties dažādu papildus punktu skaitu atšķirīgās fiksēto punktu konfigurācijās.
10. Izpēta, kā papildus punktu skaits ietekmē minimālā Eiklīda aptverošā koka garuma maiņu.
11. Pārbauda savu programmu ar konfigurācijām, kurām jau ir atrasts atrisinājums.
12. Mēģina uzlabot algoritmu efektivitāti.

Pieņēmums: Es domāju, ka hipotēze apstiprināsies, jo jau gatavās konfigurācijās punktu skaits ir 30%-40% robežās.

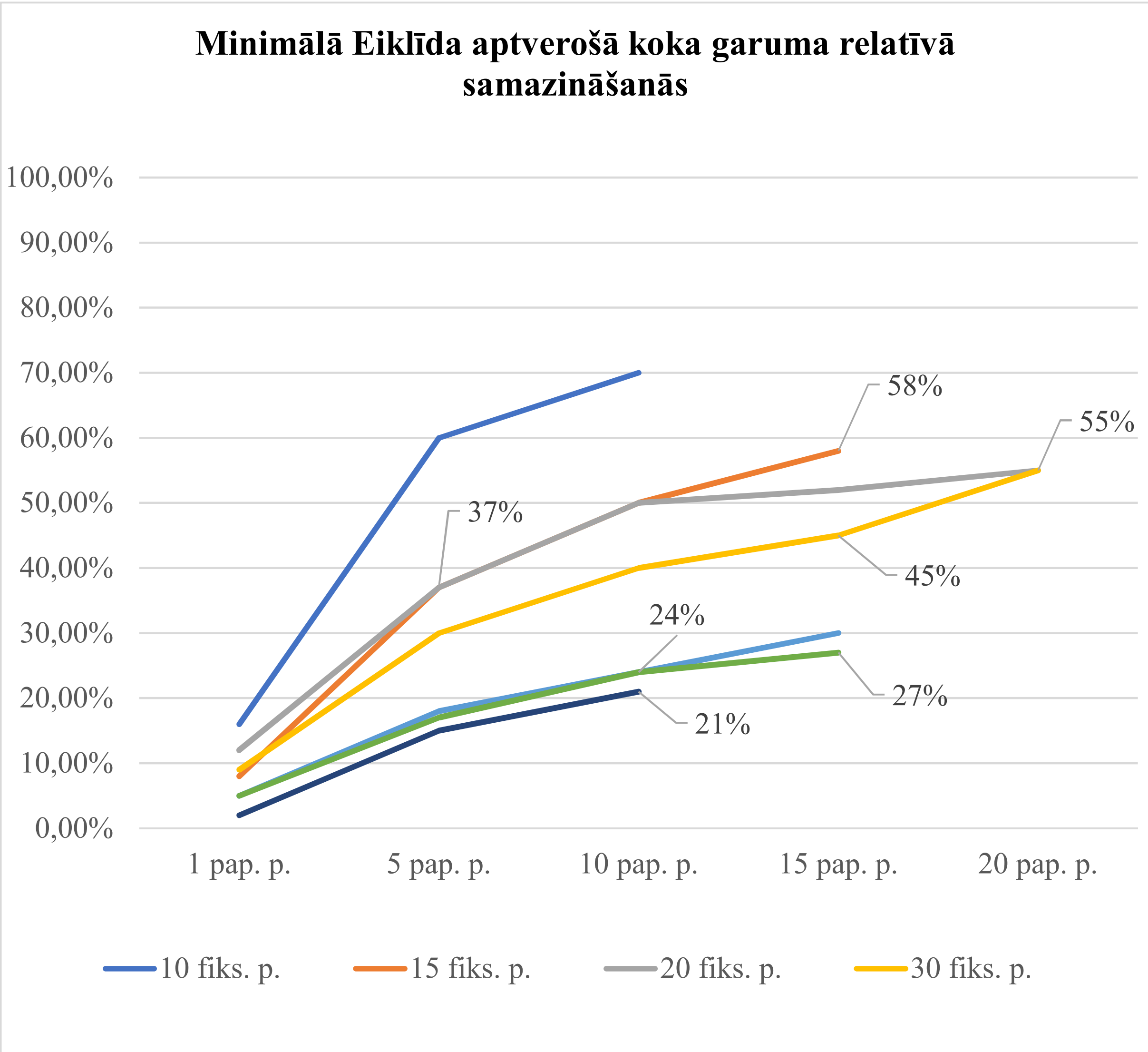
4. Datu iegūšana un apstrāde

Iegūto datu apjoms bija ļoti liels, tāpēc zemāk tiks aplūkota tabula ar jau apkopotiem datiem.

Minimālā Eiklīda aptverošā koka garuma relatīvā samazināšanās

Fiksētu punktu skaits	Ar 1-5 papildus punktiem	Ar 6-10 papildus punktiem	Ar 11-15 papildus punktiem	Ar 16-20 papildus punktiem
10	16% – 60%	61% – 70%	-	-
15	8% – 37%	38% – 50%	51% – 58%	-
20	12% – 37%	38% – 50%	51% – 52%	53% – 55%
30	9% – 30%	31% – 40%	41% – 45%	46% – 55%
40	5% – 18%	19% – 24%	25% – 30%	-
50	5% – 17%	18% – 24%	25% – 27%	-
60	2% – 15%	16% – 21%	-	-

Attēls Nr. 7

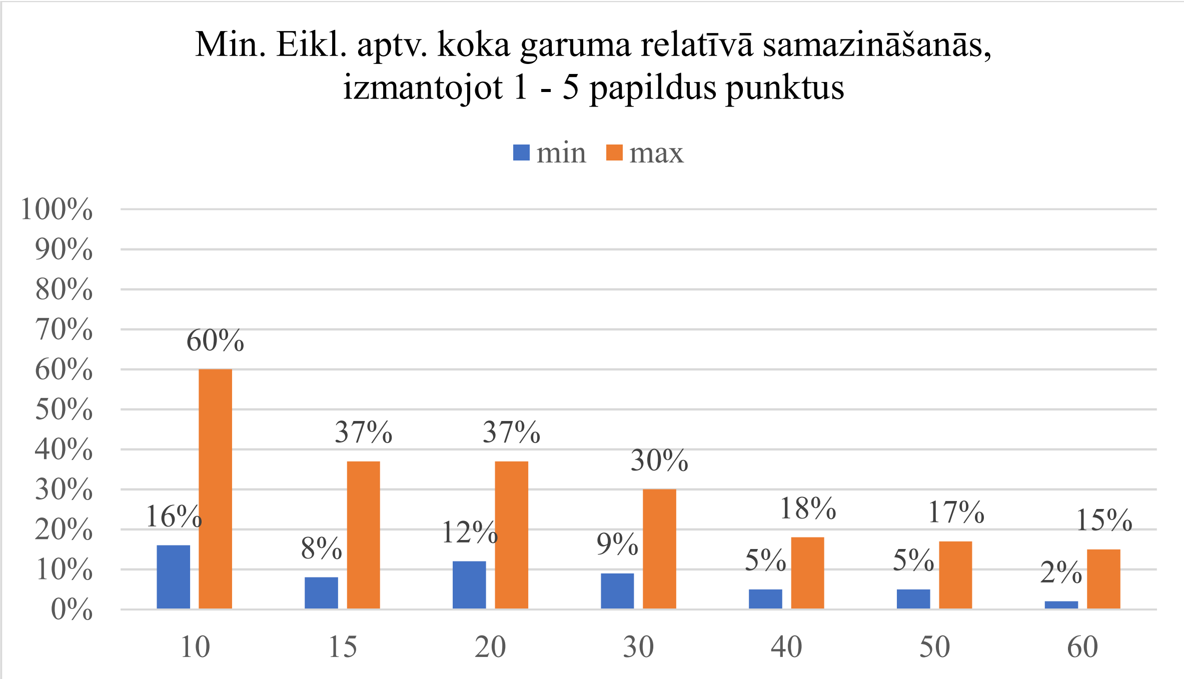


Attēls Nr. 8

5. Rezultātu analīze

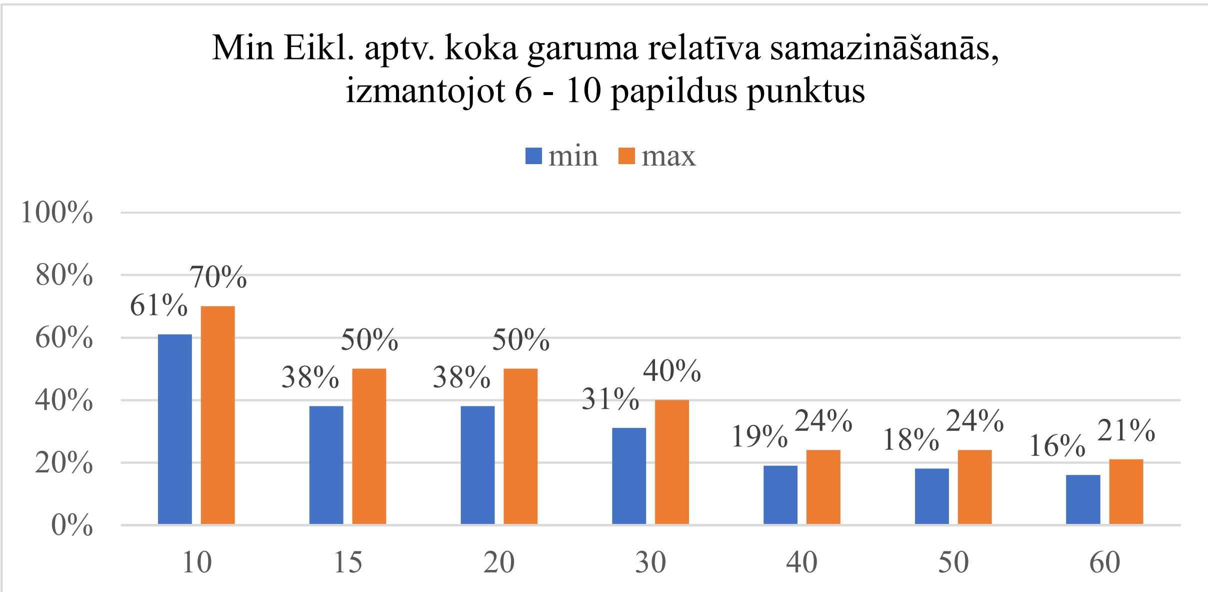
Šajā darbā tiek pētītas konfigurācijas ar 10, 15, 20, 30, 40, 50 un 60 fiksētiem punktiem. Visas zemāk izmantotās vērtības atrodas tabulā "Minimālā Eiklīda aptverošā koka garuma relatīvā samazināšanās" (skat. attēlu Nr. 7).

1. No tabulas (skat. attēlu Nr. 7) var redzēt, ka visefektīvāk Neldera-Mida algoritms strādā ar 10 fiksētiem punktiem un minimālā Eiklīda aptverošā koka garums samazinās par līdz pat 70%, izmantojot 10 papildus punktus. Savukārt starpība starp 5 un 10 papildus punktiem ir 10%. Šajā gadījumā iegūtie 10% nav noteicoši, tāpēc var uzskatīt, ka visoptimālāk ir izmantot 5 papildus punktus ar 60% samazināšanos. Papildus punktu, skaits lielāks par 10, koka garuma samazināšanos neveido.
2. Konfigurācijā ar 15 fiksētiem punktiem min. Eikl. aptv. koka garums samazinās par līdz 58%, izmantojot 15 papildus punktus. Starpība starp 10 un 15 papildus punktiem ir tikai 8%, bet starp 5 un 10 papildus punktiem jau 13%, tātad var uzskatīt, ka optimālākais risinājums konfigurācijā ar 15 fiksētiem punktiem ir 10 papildus punkti. Papildus punktu, skaits lielāks par 15, koka garuma samazināšanos neveido.
3. Konfigurācijā ar 20 fiksētiem punktiem lielākais iegūtais min. Eikl. aptv. koka samazinājums ir 55%, izmantojot 20 papildus punktus. Starpība starp 10 un 15 papildus punktiem ir tikai 2%, bet starp 15 un 20 papildus punktiem – 3%. 5% samazinājums ir pārāk mazs, lai izmantotu vēl papildus 10 punktus, tātad var uzskatīt, ka optimālākais risinājums konfigurācijā ar 20 fiksētiem punktiem ir 10 papildus punkti. Papildus punktu, skaits lielāks par 20, koka garuma samazināšanos neveido.
4. Konfigurācijā ar 30 fiksētiem punktiem samazinājums ir vienmērīgāks nekā konfigurācijā ar 20 punktiem, bet lielākais iegūtais samazinājums ir arī 55%. Tomēr šajā gadījumā starpība starp 15 un 20 papildus punktiem ir 10%. Ņemot vērā vienmērīgu samazināšanos šie 10% stipri ietekmē kopējo minimālā aptverošā koka garumu, tāpēc visoptimālākais risinājums dotajā konfigurācijā ir 20 papildus punkti. Papildus punktu, skaits lielāks par 20, koka garuma samazināšanos neveido.
5. Konfigurācijās ar 40 un 50 fiksētiem punktiem iegūtie dati ir ļoti līdzīgi, tāpēc šie divi gadījumi tiks aplūkoti kopā. Vislielākais min. Eikl. aptv. koka samazinājums ir 30% un 27% attiecīgi. Samazinājums ir vienmērīgs, bet diezgan mazs, ap 5%. Var secināt, ka ar šādu punktu skaitu Neldera-Mida algoritms strādā diezgan neefektīvi. No iegūtiem datiem var secināt, ka 1. gadījumā (40 fiksēti punkti) optimālākais variants ir 15 papildus punkti, bet 2. gadījumā (50 fiksēti punkti) – 10 papildus punkti. Papildus punktu, skaits lielāks par 15, abos gadījumos koka garuma samazināšanos neveido.
6. Konfigurācijā ar 60 fiksētiem punktiem vislielākais samazinājums ir 21%, izmantojot 10 papildus punktus. Šis ir jau pavisam mazs samazinājums ar tādu papildus punktu skaitu, salīdzinot ar pārējām konfigurācijām, tāpēc ar vairāku fiksētu punktu skaitu netiek pētīts tālāk. Šajā konfigurācijā visoptimālākais risinājums ir 10 papildus punkti ar 21% samazinājumu. Papildus punktu, skaits lielāks par 10, koka garuma samazināšanos neveido.



Attēls Nr. 9

Šis grafiks (skat. attēlu Nr. 9) norāda uz to, ka vislielākā ietekme uz min. Eikl. aptv. koka garumu ir pirmiem 5 papildus punktiem. Vidējais samazinājums ir 22%, bet, ja ņemt vērā konfigurācijas tikai ar 10, 15, 20 un 30 fiksētiem punktiem, tad vidējais samazinājums jau sastāda 30%. Ir redzama Neldera Mida algoritma efektivitātes krišana, palielinoties fiksēto punktu skaitam. Šis grafiks pastiprina secinājumus, par to, ka 40 fiksētie punkti ir algortima efektivitātes saprātīga robeža, jo samazīnājums sastādā mazāk par 13%.



Attēls Nr. 10

Šis grafiks (skat. attēlu Nr. 10) uzskatāmi attēlo efektivitātes kritumu, bet konfigurācijās ar 15 un 20 fiksētiem punktiem tas nav tik liels, kā pārējos gadījumos. 6-10 papildus punkti dod 12% min. Eikl. aptv. Koka garuma samazināšanos.

6. Secinājumi

Hipotēze ir daļēji apstiprinājusies, jo, balstoties uz grafikā apkopotiem datiem (skat. attēlu Nr 8), tikai konfigurācijā ar 40 fiksētiem punktiem, optimālākais papildus punktu skaits atrodas intervālā 30%-40%:

- 10 fiksētiem punktiem – 5 papildus punkti / 50%
- 15 fiksētiem punktiem – 10 pappildus punkti / 67%
- 20 fiksētiem punktiem - 10 papildus punkti / 50%
- 30 fiksētiem punktiem - 20 papildus punkti / 67%
- 40 fiksētiem punktiem - 15 papildus punkti / 38%
- 50 fiksētiem punktiem - 10 papildus punkti / 20%
- 60 fiksētiem punktiem - 10 papildus punkti / 17%

Ja skatīt konfigurācijas tikai ar 10, 15, 20 un 30 fiksētiem punktiem, tad var secināt, ka, ja papildus punktu skaits sastāda 50%-70% no fiksēto punktu skaita, tad minimālā Eiklīda aptverošā koka garums ir vismazākais.

Savukārt ja skatīt konfigurācijas ar 50 un 60 fiksētiem punktiem, tad var secināt, ka , ja papildus punktu skaits sastāda 10%-20% no fiksēto punktu skaita, tad minimālā Eiklīda aptverošā koka garums ir vismazākais.

Var pieņemt, ka, ja ir daudz fiksētu punktu, tad katram papildus punktam ir ļoti mazs īpatsvars kopējā koka garumā, tāpēc arī nav tik lielas atšķirības ar dažādu papildus punktu skaitu.

Šis darbs bija vērsts uz kopējās situācijas analīzi (kvantitatīvais pētījums). Tika apskatīts pēc iespējas vairāk dažādu konfigurāciju. Bija padarīts ļoti liels darbs, ievācot milzīgus datu masīvus. Bet tādā arī bija pētījuma ideja, tāpēc iegūtie rezultāti atbilst darba uzdevumam.

Ir vērts sīkāk izpētīt likumsakarības pie 10 fiksētiem punktiem un pie 50 un 60. Pamēģināt izmantot citus optimizācijas algoritmus, kā arī minimizēt nevis funkciju, kas skaitļo koka garumu, bet kādu praktiskāku dzīves sakarību, aprakstītu ar matemātisku funkciju.

Avoti

1. Minimālais aptverošais koks. Prima algoritms. [tiešsaite]. [Skatīts 08.05.2019]. Pieejams: https://e-maxx.ru/algo/mst_prim
2. Brian D. Bunday (1984). Basic Optimisation Methods. Edward Arnold (Publishers) Limited, 41 Bedford Square, London WC1B 3DQ.
3. Tmeladze Z. Nelineārā programmēšana. Kvants, 1976, 29.-34. lpp.
4. Neldera-Mida optimizācijas piemērs. [tiešsaite]. [Skatīts 08.05.2019]. Pieejams: https://habr.com/ru/post/332092/