Turinys

[Santrumpos 4](#_Toc509153076)

[Įvadas 5](#_Toc509153077)

[Literatūros apžvalga 6](#_Toc509153078)

[1 Įrenginių našūmo kėlimas - organizacinės metodikos 6](#_Toc509153079)

[2 trumpiausio maršruto paieškos problematika 8](#_Toc509153080)

[2.1 Euristiniai metodai 9](#_Toc509153081)

[2.1.1 Artimiausio kaimyno metodas 9](#_Toc509153082)

[2.1.2 2-opt, 3-opt, k-opt metodai 10](#_Toc509153083)

[2.1.3 Lagrandžo skaidymo metodas 10](#_Toc509153084)

[2.2 Meta euristiniai metodai 11](#_Toc509153085)

[2.2.1 Šakų ir rėžių metodas (Branch and Bound) 11](#_Toc509153086)

[2.2.2 Genetiniai algoritmai (Genetic algorithm) 11](#_Toc509153087)

[2.2.3 Practical swarm optimization 12](#_Toc509153088)

[2.2.4 Simuletyvinis atkaitinimas (Simulated annealing) 12](#_Toc509153089)

[2.2.5 Skruzdžių kolonijos algoritmas 12](#_Toc509153090)

[3 Klasterinė analizė ir taikymo metodai 13](#_Toc509153091)

[3.1 K- vidurkio metodas klasterizuojant 14](#_Toc509153092)

[3.2 Afinizmo propogavimas klasterizuojant (Affinity Propagation) 15](#_Toc509153093)

[3.3 Hierarchinis klasterizavimas 16](#_Toc509153094)

[3.4 DBSCAN klasterizavimas 16](#_Toc509153095)

[3.5 HDBSCAN klasterizavimas 17](#_Toc509153096)

[3.6 Birch klasterizavimo metodas 19](#_Toc509153097)

[Literatūros apžvalga (apibendrinimas) 20](#_Toc509153098)

[Medžiagos ir tyrimų metodai 21](#_Toc509153099)

[4 trumpiausio maršruto paieškos problematika 22](#_Toc509153100)

[5 Žvalgomoji duomenų analizė 23](#_Toc509153101)

[6 Duomenų klasterizavimas 23](#_Toc509153102)

[7 Trumpiausio maršruto paieška 23](#_Toc509153103)

[Tyrimo rezultatai ir jų aptarimas 24](#_Toc509153104)

[8 Tyrimo rezultatai 24](#_Toc509153105)

[8.1 Klasterizavimo metodų vertinimas, klasterių interpretavimas 24](#_Toc509153106)

[8.2 Trumpiausio maršruto paieškos metodų vertinimas 24](#_Toc509153107)

[Išvados 25](#_Toc509153108)

[Literatūros sąrašas 26](#_Toc509153109)

[Priedai 30](#_Toc509153110)

****Sąntrauka****

****Summary****

Lentelių Sąršas

Paveikslų sąrašas

Santrumpos

* TSP – Keliaujančio pirklio uždavinys (Travel salesman problem);
* FMS – Lanksčios gamybos sistemos (Flexible Manufacturing Systems);
* CNC – Skaitmeninio valdymo kompiuteriai (Computer Numerical Control);
* SSP – Darbo sekos ir įrankių keitimo problema (Job Sequencing and Tool Switching Problem);
* JSeP – Darbų sekos problema (*Job Sequencing Problem);*
* KTNS *–* Įrankių laikymas kuris reikalingas greičiausiai *(Keep Tool Needed Soonest);*
* NP *–* Klasė nedeterministinio polinominio laiko *(**for nondeterministic polynomial time);*
* ILP – Integruotas tiesinis programavimas (Integer linear programming);
* ILS – Pažingninė vietinė paieška (Iterated Local Search)
* 5S – Nuostolių šalinimo principas organizuojant darbo vietą;
* PDCA – Planuoti, vykdyti, tikrinti, dokumentuoti (Plan, do, check, act);
* SMED – Įrankių keitimas per minutę (Singel Minute Exchange of Die);
* TPM – Pilna gamybos priežiūra (Total productive maintenance);
* MRP – Medžiagų poreikio planavimas (Material Requitement Planning);
* GA – Genetiniai algoritmai *(Genetic algorithm);*
* PSO – Particle swarm optimization (PSO);
* SA – Simuletyvinė analizė (Simulated annealing);
* NNA – Artimiausio kaimyno metodas *(Nearest Neighbor Algorithm)*;
* RNNA – Kartotinas artimiausio kaimyno metodas *(Repeated Nearest Neighbor Algorithm)*;
* AP – Afinizmo propogavimas *(Affinity Propagation);*
* DBSCAN – Tankiu grįstas klasterizavimo metodas su išskirtimis (Density Based Spatial Clustering of Application with Noise);
* HDBSCAN – Hierarchinis tankiu grįstas klasterizavimo metodas su išskirtimis (Herarchical Density Based Spatial Clustering of Application with Noise);
* CFT – Charakteristikų ypatybių medis (Characteristic Feature Tree);
* BIRCH – Balansinis mažinimas arba klasterizavimas naudojant hierarchijas (Balanced iterative reducing and clustering using hierarchies);

Įvadas

Keliama problematika yra siejama su gamybinėmis įmonėmis. Darbui naudojami duomenis „IKEA INDUSTRY Lietuva“ įmonės.

Nuo seno žinoma, kad gamybinis pajėgumas priklauso nuo silpniausios grandies įmonėje. Kiekvienam gamybiniam sraute yra darbo centras / apdirbimo linija ar kt. kas apriboją gamybines apimtis. Vienas esminių klausimų yra kaip sukelti efektyvumą mažiausiai pralaidžioje vietoje. Dažniausi metodai pasitelkiami taikant LEAN modelius modernizuojant darbo centrus taikant 5S, KAIZEN metodologijas ir kt. Darbo tikslas peržvelgti taikomus metodus ir pateikti sprendimus taikant duomenų analitikos priemones, išnaudojant tik esamus duomenis.

Besivystant duomenų analitikos mokslui jis neatsiejamai paliečia ir tokias sferas kaip pramonės sektorius. Platesnis matematikos mokslo panaudojimas siejamas su parametrų derinimų, rizikų vertinimų. Gamybos srautų planavimas iki šiol dažnai atliekamas taikant MPR sistemas kurios tik suderina paralelinius srautus laike ir taikomos apskaitos sekimui.

**Analizuojama veikla ir/ar projekto priežastys: Gamybos sektorius, korpusinių baldų gamyba. Padidinti gamybos pralaidumą ribojančiuose srauto taškuose.**

**Tikslai: Optimizuoti gamybos srautą darbo centruose apribojančiuose gamybos apimtis naudojant duomenų analitikos priemones.**

**Uždaviniai**

* **Išanalizuoti pramonėje taikomas metodikas gamybos efektyvumo didinimui;**
* **Išsiaiškinti optimizavimo priemones taikomas didelės apimties duomenims trumpiausio maršruto uždaviniui spręsti;**
* **Atrinkti technines charakteristikas (požymius) kurių pagrindu galima sudaryti panašumo matricą;**
* **Padalyti gamybos srautą klasterizuojant gaminius remiantis jų techninėmis charakteristikomis;**
* **Parinkti tinkamus metodus trumpiausio maršruto problemai spręsti;**

Literatūros apžvalga

# Įrenginių našūmo kėlimas - organizacinės metodikos

Plačiai gamybos pramonės šakose yra naudojamos tokios organizacinės priemonės kaip LEAN ar AGILE modeliai. Idėja paremta LEAN [14] metodologija „Išnaudojant esamus išteklius ir žinias kurti pridėtinę vertę ir konkurencinį pranašumą“ Esminiai LEAN principai „Nuolatinis esamų procesų efektyvinimas ir nereikalingų veiklų nuostolių šalinimas“ remiantis šiais principais galima bet kūrę veiklą suskaldyti į pagrindines grupes tokias kaip:

* Efektyvus darbo laikas (laikas kuriuo metu vykdomas procesas nešą pridėtinę vertę įmonei) t.y mūsų pasirinktuoju atveju bus kai įrenginys atlieka operacijas: gręžimo ir briaunavimo. Įrenginio efektyvumą galima pagerinti optimizuojant parametrus kiekvienai apdirbamai detalei ir/ar fiziškai peržvelgiant modernizavimą silpniausių gamybinės linijos mazgų, kurie daro lemiama įtaką detalių apdirbimo greičiui;
* Veiksniai nenešantys tiesioginės pridėtinės vertės, bet gamybos procese be jų veikla neatsiejama. Analizuojamoje veikloje galima išskirti tokias grupe:
  + Įrenginio reguliavimo laikas pereinant iš vienos detalės prie kitos;
  + Įrankių (įrangos komponentų) keitimas;
  + Įrangos valymas ir priežiūra;
  + Įrangos testavimo trukmė (atliekama po įrankių keitimo ir/ar jos reguliavimo);
* Pašaliniai veiksniai kurie pasitaiko gamybos procese:
  + Įrenginių gedimai;
  + Kiti veiksniai.

LEAN metodikoje teigiama, kad geriausi sprendimai pagerinti tokius kritinius taškus yra taikyti tokius metodus kaip :

* 5S – Metodo principai šalinti nuostolius organizuojant darbo vietą. Laikomasi penkių pagrindinių principų: rūšiuok, sutvarkyk, išvalyk, standartizuok, laikykis (vertinamas iš japoniškų žodžių);
* Kaizen – Terminas kilęs iš japonų kalbos, termino reikšmė – nuolatinis tobulėjimas. Šis metodas taikomas darbo vietos stebėjimui, ir joje vykstančio proceso ir idėjų siūlymas jos/jo tobulinimui. Pasiūlymai koncentruojasi apie kitų LEAN metodų panaudojimą atitinkamai problemai;
* PDCA - Problemų sprendimo technika arba pastovaus tobulinimo modelis, susidedantis iš logiškos keturių žingsnių sekos: Planuoti (Plan), Įgyvendinti (Do), Patikrinti (Check), Įtvirtinti/Priderinti (Act/Adjust) ir skirtas išmokyti darbuotojus sprendžiant problemas kovoti ne su jų pasekmėmis, o priežastimis.  Taip pat vadinamas Demingo ratu. Metodo taikymo sritis rutininis ciklas darbe;
* Standartizuotas darbas – Geriausias galimas darbo atlikimo būdas, kuris yra nustatomas, kaip sektinas standartas kitiems. Darbo standartizavimas leidžia užtikrinti, kad veikla būtų visada atliekama vienodu būdu, per vienodą laiką ir su tokiu pačiu rezultatu, nepriklausomai nuo to kas ir kada šį darbą atlieka;
* SMED - Antgalio pakeitimas per minutę - Metodai, kaip sureguliuoti stakles per mažiau nei dešimt minučių. Ilgalaikis tokių metodų tikslas - sumažinti nustatymo laiką iki nulio, kai staklės sureguliuojamos akimirksniu, nenutraukiant nuolatinio srauto;
* TPM - gamybinė priežiūra. Jos tikslas - padidinti įrangos efektyvumą ir tarnavimo laiką, daugiau dėmesio skiriant jos priežiūrai ir įtraukiant į tai operatorius. Įvairiais metodais siekiama, kad įranga niekada nesustotų ir gamybos srautas nenutrūktų;
* Kanban - Japoniškas terminas, reiškiantis „signalą“. Kanban naudojama detalių gamybai ir pristatymui valdyti traukos sistemoje;
* ir kiti.

AGILE-gamyba organizacinės metodikos kryptis taikomą gamybos įmonėse. AGILE-gamybos principas, kad gamyba yra terminas, taikomas organizacijai, kuri kuria procesus, įrankius ir mokymus, kad galėtų greitai reaguoti į klientų poreikius ir rinkos pokyčius, tuo pačiu išlaikydama išlaidų ir kokybės kontrolę. Tai siejama tiek su administraciniais tiek su gamybiniais procesais. AGILE-gamyba sekantis žingsnis gamybos procesų efektyvumo kėlimo po LEAN metodologijos taikomų metodų. Tai gamybos metodas, orientuotas į klientų poreikių tenkinimą, tuo pačiu išlaikant aukštus kokybės standartus ir kontroliuojant bendras konkretaus produkto gamybos sąnaudas. Šis požiūris yra skirtas įmonėms, dirbančioms labai konkurencingoje aplinkoje, kai nedideli veiklos rezultatų ir produktų pristatymo skirtumai ilgainiui gali labai pakeisti bendrovės išlikimą ir vartotojų reputaciją.

Ši koncepcija glaudžiai susijusi su LEAN metodologija. Gamyboje bendrovė siekia sumažinti visas išlaidas, kurios nėra tiesiogiai susijusios su produkto gamyba vartotojui. AGILE gamyba gali apimti šią koncepciją, tačiau ji taip pat papildo kitą dimensiją, idėja, kad klientų poreikius reikia greitai ir veiksmingai įgyvendinti. Priešingai nei LEAN metodologijoje žvelgiama ne tik į nuostolių šalinimą kad procesas būtų greitesnis bet ir į patį proceso spartinimą. Tačiau visos organizacines priemonės taikomos AGILE metodologijoje, yra skirtos procesų skatinimui koreguojant procesus tarp skyrių, tiekėjų, pirkėjų t.y. apima tiktai gamybos, tiekimo bendradarbiavimo procesus.

Nagrinėjant organizacines metodologijas įžvelgiama, kad jos neapima procesų pačiuose darbo centruose. Organizacines priemones galima praturtinti papildomai taikant matematinius metodus.

# trumpiausio maršruto paieškos problematika

Lanksčios gamybos sistemos (FMS) kuriose veikia mašinos tipo (CNC) skaitmeninio valdymo kompiuteriai. Mašinos gali apdoroti įvairias darbo vietas, kai mašinų įrankių bloke yra užduočiai atlikti reikalingų apdirbimo įrankių. Konkrečių nagrinėjamų atvejų apdirbimo įrankiai tai skirtingo tipo gręžimo galvos talpinamos į atitinkamus darbo centro įrankių rėmus (blokus). Apdirbimo bloką sudaro C jungčių. Viena jungtis skirta vienai gręžimo galvai.

C taip pat žinomas kaip įrankių rėmo naudingumas (pajėgumas) , atliekamų darbų skaičius negali viršyti C jungčių. Jeigu įrankis (gręžimo galva) nėra apdirbimo bloke tuomet įrankis turi būti pakeistas esamu gręžimo bloke t.y. užimančių vieną ar kelias C jungtį (-is). Vieno įrankio pašalinimas ir kito įtraukimas yra neišvengiamas veiksmas gamybos procese.[20]. Įrankių pakeitimas ir reguliavimas, procesas atimantis daug laiko ir nenešantis tiesioginės pridėtinės vertės, todėl FMS sistemoje svarbus klausimas yra surasti darbo seką, sumažinant bendrą įrankių keitimų skaičių. Ši problema plačiai žinoma kaip darbų sekos ir įrankių keitimo problema (SSP)[16,20] ir yra taikoma įvairiose srityse, tokiose kaip kompiuterio atminties paskirstymo[25], skambučių srautų paskirstymo uždaviniuose. Šis uždavinys plačiai nagrinėjamas įvairiose pramonės šakose tokiose kaip metalo pramone, chemijos pramonė, draudimo, formacijos bendrovės[1,23]. Taip pat viena iš daugiausiai susiduriančių industrijų su SSP yra elektronikos. kai skirtingų tipų spausdintinės plokštės (PCB) turi būti apdorotos komponentų surinkimo mašinoje. Skirtingi PCB gali būti išversti į darbus, o įvairūs elektroniniai komponentai, kuriuos reikia įkelti į mašinos padavimo sistemą, gali būti išversti į įrankius. Tokia formuluotę pateikė Shirazi ir Frizelle [19] kurie ištyrė realaus pasaulio taikomus metodus sprendžiant SSP septyniose skirtingose ​​bendrovėse aviacijos, CNC gamintojų ir įrankių gamybos pramonėje.

Tangas ir Denardoas [20] aptarė, kad SSP gali būti išskaidoma į dvi dedamąsias kurios gali būti sprendžiamos kaip atskiros problemos:

1. Darbo sekos problema (JSeP), kuri yra laiko planavimo problema, kurią galima nustatyti optimalia darbų seka kompiuteryje.
2. Įrankio pakeitimo problema (TRP) reiškia, kad reikia įkelti reikalingus įrankius, kad būtų galima apdoroti numatytą darbų seką (JSeP), siekiant sumažinti bendrą įrankių pakeitimų skaičių.

Y. Crama ir kt.[19] atliekant tyrimus parodė, kad JSeP yra NP-sudėtingumo problema. Keliuose tyrimuose buvo apsvarstyta JSeP ir modeliuojama problema su tiksliu algoritmu. L. Gilbert ir kt.[22] paskelbė optimalų sprendimą JSeP kai kuriems etaloniniams atvejams iki 25 pozicijų. Sprendimus modeliuojant buvo atsižvelgta jų linijinio programavimo (ILP) modelius. Tačiau realiose situacijose sekos taškų skaičius dažnai viršiją šią ribą. Tokiu atveju stengimąsi skaičiavimo sekas apdoroti kitais metodai. Ši problema sprendžiama atliekant skirtingus tyrimus įvairiose publikacijose. Dažnais atvejais JSeP yra prilyginama Keliaujančio pirklio uždaviniui (TSP). Tai grafų teorijos uždavinys kai ieškomas mažiausio svorio Hamiltono kelias pilname svoriniame grafe. Tyrimo metų grafų teorijos terminai siejami taip:

* Taškai – gaminio detalės apdirbamos darbo centre;
* Tiesės – įrenginio reguliavimo laikas pereinant iš vienos detalės prie kitos;
* Ciklas – detalių gamybos eiliškumas darbo centre.

Iš pirmo žvilgsnio paprasčiausias metodas šio uždavinio sprendimui yra visų galimų kombinacijų perrinkimas. Tačiau remiantis gamybos asortimentu išskiriamos ~700 skirtingos detalės. Kurias galima suskirstyti į 4 – 5 savaičių intervalą pagal gaminius. Taip atliekama dėl to kad vienu kartu visas esamas asortimentas nėra užleidžiamas. Toks detalių skaičius apsunkina uždavinio sprendimą.

Remiantis grafų teorija ciklų skaičius yra:

* Esant asimetriniai matricai grafe N taškų sudarys ;
* Esant simetriniai matricai grafe N taškų sudarys [29]

Remiantis nagrinėjamais duomenimis ir ekspertine nuomone nustatoma, kad detalių perstatymas yra simetriškas t.y. ; tokiu atveju taikoma: . Kaip matoma iš pavyzdžio ([2.1.1.1 Lentele. Eksponentinio augumo pavyzdys](#_Lentele._Eksponentinio_augumo)) skirtingų sekų skaičius auga eksponentiškai. Jei per sekundę kompiuteris atliktu vieną milijardą veiksmų, visoms galimoms sekoms ištirti 20-ies taškų sekas prireiktu dvejų metų.[29]. Tokiais atvejais perrinkimo metodas yra netinkamas ir šiam uždaviniui spęsti taikomi, euristiniai, meta-euristiniai metodai.

|  |  |  |
| --- | --- | --- |
| Taškai |  |  |
| 5 | 60 | 120 |
| 6 | 360 | 720 |
| 10 | 1,814,400 | 3,628,800 |
| 11 | 19,958,400 | 39,916,800 |
| 20 | 1,216,451,004,088,320,000 | 2,432,902,008,176,640,000 |
| 21 | 25,545,471,085,854,700,000 | 51,090,942,171,709,400,000 |

#### Lentele. Eksponentinio augumo pavyzdys

## Euristiniai metodai

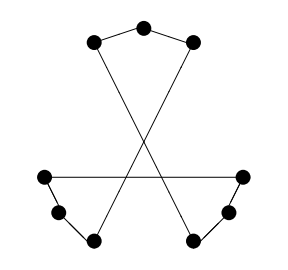
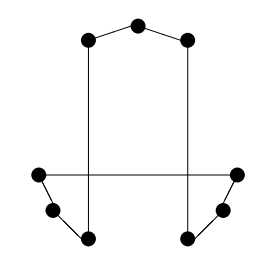
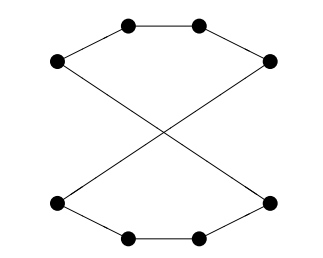
### Artimiausio kaimyno metodas

Tangas ir Denardoas [20] parodė, kad JSeP, gali būti išspręsta taikant godų algoritmą (GR) vadinamu KTNS sprendžiamas daugianario funkcija. Todėl SSP kuris yra kombinuotas gali būti sumažinta iki JSeP ir tai tampa pagrindine problema, kurią reikia spręsti.

G. Gutin ir kt. [18] atliekant kompiuterinius eksperimentus įrodo, kad GR ir artimiausio kaimyno algoritmas (NN), populiarūs euristiniai metodai sudaryti sekai, yra tinkami Euklido TSP lygims. Tačiau rezultatai yra labai prasti tiek simetrinėms, tiek asimetrinėms matricoms sprendžiant TSP (STSP ir ATSP). Įrodoma, kad kiekvienam yra ATSP (STSP) atvejis iš n viršūnių, kai GR nustato blogiausią seką. Tokie pat rezultatai nustatyti ir NN. Šiame tyrime taip pat analizuojamą pasikartojantis NN (RNN) t.y. NN metodas atliekamas pradedant nuo visų galimų viršūnių ir atrenkama geriausia seka. Taip pat jie įrodė, kad "ATSP" taikant RNN visada gaunama geriausia seka, kuri yra ne blogesnė nei mažiausiai kitų sekų, bet tam tikru atveju ji suranda seką, kuri yra ne blogesnė nei ne daugiau kaip kitų sekų , kai . Jie taip pat ištyrė, kad kai kurioms STSP kai atveju RNN duoda gerus rezultatus esant ne daugiau kaip sekų. Šie rezultatai akivaizdžiai skiriasi nuo ankstesnių Tango ir Denardo [20], kurie įrodė, kad visada gaunami geri rezultatai kai sudaroma (n -2)! Sekų.

### 2-opt, 3-opt, k-opt metodai

2-Opt yra turbūt labiausiai paplitęs ir plačiausiai naudojamas vietinės paieškos sistemose, kurios yra integruotos į MRP. Šis euristinis metodas pasiekia gerus rezultatus Euklido atvejų, tiek laiko, tiek artumo atžvilgiu. Šio metodo principinis veikimas: 2-opt algoritmas iš esmės pašalina dvi kraštines iš sekos ir vėl sujungia dvi sukurtas kraštines. Tai dažnai vadinama "2-opt" judesiu. Yra tik vienas būdas iš naujo prijungti dvi kraštines, kad mes vis dar turėtume tinkamą seką (2.1.2.1 Pav. Figūra 1). Tai atliekama tik tuo atveju, jei naujoji kelionė bus trumpesnė. Tas procesas kartojamas tol kol atliekant pakeitimus išimant dvejas kraštines trumpesnio atstumo tarp dviejų taškų nerandama. 3-opt algoritmas veikia panašiai, bet vietoj to, kad pašalinome dvi kraštines, pašaliname tris. Tai reiškia, kad mes turime du būdus, kaip iš naujo sujungti tris viršūnes į naują seką (2.1.2.1 Pav. Figūra 2 ir Figūra 3)



FIGŪRA 1. 2-opt FIGŪRA 2. 3-opt FIGŪRA 3. 3-opt

#### Pav. 2-opt, 3-opt, judėjimo schema

### Lagrandžo skaidymo metodas

D. Catanzaro [5] ištyrė ir pristatė tris ILP formuluotes. Juose pateiktos formuluotės pagerino Laporte Gilbert ir kt.[22] apibrėžtą ribą. Jo atlikti kompiuteriniai eksperimentai rodo, kad ribos, gautos taikant linijinę relaksaciją nagrinėjamų formulių, vidutiniškai pagerėja nuo tų, kur šiuo metu aprašyti literatūroje Laporte Gilbert ir kt.[22], ir tuo pat metu siūlo naujas kryptis tolesnio tikslaus sprendimo būdų kūrimui.

Literatūros apžvelgiančios JSeP sprendimus labai platus spektras, nagrinėjami ir tobulinami esami ir kuriami nauji tiek euristiniai tiek meta-euristiniai metodai. J. Bard’as [2] savo tyrumose problemą formuluoja kaip nelinijinė sveikojo skaičiaus programa ir sprendžiama naudojant dvejopą euristiką. Rezultatai rodo, kad beveik visais atvejais gaunamas optimalus sprendimas. Taip pat ypač sutrumpinamas skaičiavimo laikas. Tai rodo realaus laiko valdymo algoritmo naudojimo galimybes. Tačiau tyrime sekos taškų skaičius imtas iki 25. M. Denizelis [9] JSeP pritaikė Lagrandžo skaidymo metodą ir sekos elementų skaičių padidino iki 30 -ies.

## Meta euristiniai metodai

### Šakų ir rėžių metodas (Branch and Bound)

"Branch & Bound" algoritmai dažnai naudojami rasti kombinatorinius optimizavimo problemų sprendinius. Šis metodas gali būti lengvai taikomas TSP nepriklausomai nuo to, ar tai ATSP ar STSP.

M. Furrer ir T. Mütze [15] pasiūlė sujungtą algoritmą, skirtą įvairiems JSeP tikslams spręsti, susidedanti iš minimizavimo įrankių keitimų ir mašinų sustojimų skaičių per tam tikrą laiką. Šiame darbe pateikiama filtravimo ir susiejimo algoritminė sistema (Branch and Bound) dar kitaip verčiama šakų ir rėžių metodas. Metodas pagrįstas godžiuoju algoritmu (greedy) su grįžimo modifikacija į praeities atliktus žingsnius. Šiuo metu šis metodas pateikiamas kaip pirmaujančių tarp metodų sprendžiančių TSP.

Savo darbe T.Volgenant atliko tyrimus pritaikant 1970 m. Heldo ir Karpo pristatyta Lagrandžo sprendinį STSP. Jie panaudojo šį 1-medžio relaksacijos algoritmą naujame šakų ir rėžių metode. Jis skiriasi nuo kitų algoritmų ne tik schema, bet ir kitimo pobūdžiu, norint apskaičiuoti 1-medžio ribas. Be to, jie nustatome heurinius sprendimus visame skaičiavime, kad pateiktu aukščiausiąsias ribas. Vidutiniškai jų rezultatai nuo Euklido problemų iš literatūros gauna maždaug 60% mažiau šakų, nei geriausiai žinomas algoritmas.

### Genetiniai algoritmai (Genetic algorithm)

Genetiniai algoritmai (GA). GA veikia panašiai kaip gamtos veiksniai. Evoliucinis procesas vyksta formuojant naujus genus. Paprastasis GA prasideda atsitiktinai sugeneruotu sprendimų kandidatų. Kai kurie (arba visi) kanditai sujungiami, kad sudarytu palikuonis, o kai kurie iš jų pereina mutavus procesą. Pasirinkę tinkamiausius kryžminimo ir mutacijos kandidatus, bendras tinkamumas padidėja. GA taikymas TSP apima genų kryžminimą, tinkamumo patikrą, taip pat mutacijas. Kai kurie tyrimai parodė gerus rezultatus. Bet kaip ir Tabu-paieškos, ir SA algoritmo neigiama pusė yra ilga skaičiavimo trukmė.

Peržvelgiant šiuos tįrimų rezultatus galima manyti, kad elementų skaičius yra apribojamas dešimtimis vienetų. Bet A.Konak‘as[21] savo publikacijoje „Minimizing the number of tool switching instants in Flexible Manufacturing Systems“ pristatė tyrimą paremtą E. Denard‘o [26] ir M. Denizel‘io [9] atliktais darbais. Kur sekos elementų sudarė 210. Metodika taikė evoliucinius metodus. Atliekant genetines mutacijas paveldėtojai sugebėjo rasti kokybiškus sprendimus per protingą laiką. Pabrėžtina kokybišką bet nebūtai optimalų.

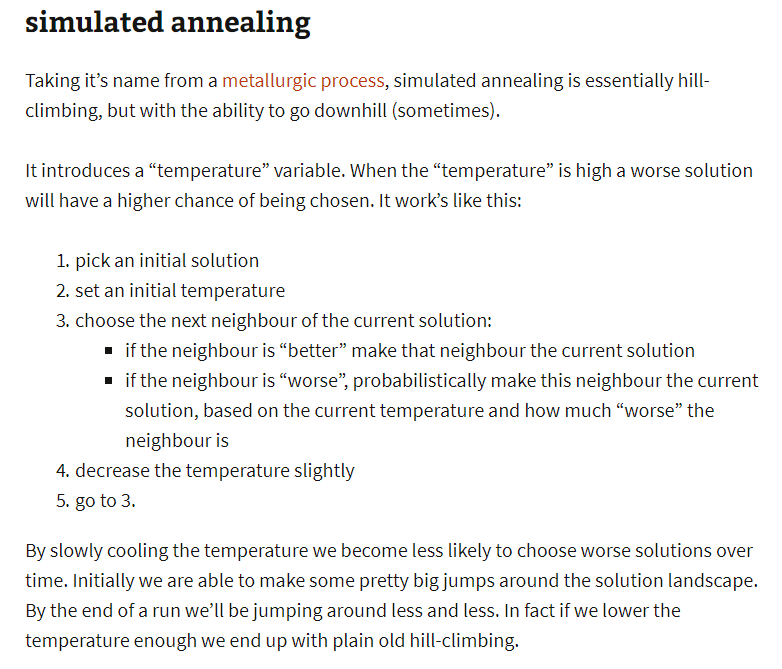
Crama ir kt. [13] pasiūlė keletą euristikos metodų, pagrįstų TSP, ir paskui įvertino jų skaičiavimo rezultatus. Pateiktas kitoks "atstumo" tarp įrankių apibrėžimas. Jie parodė, kad jų euristikos metodai yra pranašesnė. Chaves ir kt. [24], pasiūlė hibridinės euristikos metodą, pagrįstą šališku atsitiktinių genetinių algoritmų generavimu remiantis taškų grupavimo paieška. Jie atliko bandymus, ir kai kuriais atvejais pranešė apie geresnę sprendimo kokybę. Paiva ir kt. [25] siūlomas metaduomenys "Iterated Local Search" (ILS) ir jų rezultatai rodo, kad jų metodas yra pranašesnis už kolegų: Cram‘os ir kt. [13], Yanasse ir kt. [15], Catanzaro ir kt. [16] .

### Practical swarm optimization

Practical swarm optimization (PSO) pasirodė kaip galingas optimizavimas, kurį 1995 m. Sukūrė Kenedis ir Eberhartas[11,13]. PSO, įkvėptas paukščių ir žuvų socialinio elgesio. Jų judėjimas yra sinchronizuotas ir judėjimas be susidūrimo. Paukščiai skrenda ir palaiko tam tikrą atstumą tarp kaimynų. Rezultatai rodo, kad PSO yra galingas algoritmas, kuris sėkmingai sprendžia TSP problemas, naudodamas nedidelį skaičių įvertinimų, kad būtų pasiektas geriausias rezultatas.

### Simuletyvinis atkaitinimas (Simulated annealing)

Simuletyvinis atkaitinimas (SA) buvo sėkmingai pritaikytas pateikti apytikslius TSP sprendimus. SA iš esmės yra atsitiktinių imčių vietinės paieškos algoritmas, leidžiantis judėti neigiama kryptimi t.y palaipsniui mažinant bendrą atstumą visos sekos. Šis metodas pagrįstas tuo, kad renkamas kaimyninis sprendimas atrenkant geriausia sprendinį. Dažnai šie rezultatai sutampa su 2-opt algoritmu. Geresni rezultatai gali būti gaunami didinant SA algoritmo veikimo laiką.



http://www.psychicorigami.com/2007/06/28/tackling-the-travelling-salesman-problem-simmulated-annealing/

### Skruzdžių kolonijos algoritmas

Tyrėjai dažnai bando imituoti gamtą, kai vienas iš tokių pavyzdžių yra labai sėkmingas genetinių algoritmų naudojimas. Dar viena įdomi idėja yra imituoti skruzdžių judesius. M. Darigo [10] buvo pirmasis kuris pritaikė skruzdžių kolonijos algoritmą (ACO) TSP. Šis algoritmas buvo sėkmingas, greitai pateikiant optimalius mažų problemų sprendimus. Skruzdės palieka feromonų taką, kai tyrinėja naujas sritis. Šis takas yra skirtas kitiems skruzdėms nukreipti į galimus maisto šaltinius. Skruzdžių kolonijos stiprumas priklauso nuo jų skaičiaus, o tai pat ir optimizavimas. Paprastai tyrimose pradedama apytiksliai nuo 20. Jie pradeda nuo atsitiktinių viršūnių, o paskui perkeliama į kitas viršūnes. Kartotis tarp viršūnių, negalima. skruzdė, kuri apėjo trumpiausią taką, paliks feromonų taką, atvirkščiai proporcingą kelionių ilgiui. Į šį feromonų taką atsižvelgiama, kai skruzdė pasirenka miestą, į kurį keliauja, todėl jos labiau linkusios vaikščioti kelyje su stipriausiu feromono taku. Šis procesas kartojamas tol, kol randamas trumpiausias maršrutas.

Optimizavimo metodų skirtu TSP spręsti yra įvairių rūšių ir tinkamų metodų parinkimas priklauso labiausiai nuo dalykinės srities ir bendrų keliamų apribojimų kur tie metodai bus taikomi. Remiantis turimais įmonės duomenimis keliamais ribojimais, metodai taikomi ypač didelių duomenų apdirbimui nėra būtini. Taip kaip duomenis reikia klasterizuoti, o optimizavimo procesus atlikti atskirai kiekviename klasteryje. Tai keliamas ribojimas dėl buferinių sandėlių kuriuose gali būti saugoma iki 10 gaminių vienu metu. Atsižvelgiant į tai parenkami metodai kurie išsiskiria tikslumu ir sparta. Taip pat papildomai peržvelgiami naujausi tyrimai susiję su duomenų klasterizavimų.

# Klasterinė analizė ir taikymo metodai

Atsižvelgiant į problematika kuri keliama darbe papildomai peržvelgiame klasterizavimo metodus, kurie skirstomi į:

* Padalijimo metodai;
* Hierarchinei metodai;
* Tankių grysti metodai;
* Tinkleliu grysti metodai;

Klasterizavimas priskiriamas prie „Unsupervised learning“ t.y. mokymasis be mokytojo. Tokiu atveju klasterizavimo metodų vertinimas yra sudėtingas procesas kuris gali remtis į nustatyta vertinimo funkciją ar ekspertinę nuomonę. Plačiausiai šiuo metu taikomi klasterizavimo metodai:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Metodo vardas | Parametrai | Mastelis | Naudojimo atvejis | Metricos |
| K-Means | Klasterių skaičius | Labai didelis n\_samples, vidutinis n\_clusters su MiniBatch kodu | Mažas skaičius klasterių, plokščia geometrija, vienodas klasterių dydis | Atstumas tarp taškų |
| Affinity propogation | Slopinimas, | Nekeičiamo dydžio n\_samples | Daug klasterių, skirtingas dydis klasterių, ne plokščia geometrija | Grafo atstumas (pvz., Artimiausio kaimyno grafas) |
| Mean-shift | Pralaidumas | Nekeičiamo dydžio n\_samples | Daug klasterių, skirtingas dydis klasterių, ne plokščia geometrija | Atstumas tarp taškų |
| Spectral clustering | Klasterių skaičius | Vidutinis n\_samples, mažas n\_clusters | Keli klusteriai, vienodas klasterių dydis, ne plokščia geometrija. | Grafo atstumas (pvz., Artimiausio kaimyno grafas) |
| Ward hierarchical clustering | Klasterių skaičius | Didelis n\_samples, ir n\_clusters | Didelis klasterių skaičius, gali būti jungiamumo apribojimai | Atstumas tarp taškų |
| Agglomerative clustering | Klasterių skaičius, susiejimo tipas, atstumas | Didelis n\_samples, ir n\_clusters | Didelis klasterių skaičius, gali būti jungiamumo apribojimai. Nereikalingi Euklido atstumo matai | Atstumas tarp porų |
| DBSCAN | Kaimynystės dydis | Labai didelis n\_samples, vidutinis n\_clusters | Ne plokščia geometrija, nevienodas klasterių dydis. | Atstumas tarp artimiausių taįkų |
| HDBSCAN |  |  |  |  |
| Gaussian mixtures | Klasterių skaičius | Nekeičiamo dydžio | Plokščia geometrija, geras tankio įvertinimui | Mahalanobis atstumas iki centro |
| Birch | Šakojimosi veiksniai, Slenkstis, pasirenkamas globalus klasteris | Didelis n\_samples, ir n\_clusters | Didelis duomenų rinkinys, pašalinimas, duomenų mažinimas. | Euklido atstumas tarp taįkų |

Neplokščių geometrinių grupių naudojimas yra naudingas, kai klasteriai turi tam tikrą formą, t. y. Ne plokščią atvaizdą, o standartinis Euklido atstumas nėra tinkama metrika.

## K- vidurkio metodas klasterizuojant

K-Vidurkių metodu klasterizuojami duomenys skirstant taškus taip, kad dispersijos būtų lygios. Minimizuojamas inercijos kriterijus arba kvadratinė suma klasteryje. Algoritmas iš anksto nusakomas apsprendžiant klasterių skaičių. P. S. Brandley [3] tyrimose įrodyta, kad K-vidurkių metodas efektyviai dirba su dideliais kiekiais duomenų, dėl šios priežasties K-vidurkių metodas plačiai taikomas įvairiose srityse.

K-vidurkių algoritmas padalina rinkinį iš N elementų į K klasterių, kiekvieną iš jų apibūdina imties vidurkis . Šios priemonės dažniausiai vadinamos klasteriais "centroidais"; K-Vidurkio algoritmas siekia pasirinkti centroidus, kurie mažina inerciją arba suminį kvadratinį vidurkį:

Inercija arba suminis kvadratinis vidurkis pripažintas kaip vidinės darnos klasterių matas. Ji turi įvairių trūkumų:

* Inercija daro prielaidą, kad klasteriai yra išgaubti ir izotropiniai, bet taip būna ne visada. Tai blogai reaguoja į pailgas grupes ar su nelygia forma.
* Inercija tai nenormalizuota metrika: žinoma tik tai, kad mažesnės reikšmės yra geriau ir nuliai yra optimalu. Bet esant daugiadimensiams duomenims, Euklido atstumai tampa išpūti t.y. vadinama dimensijos persislinkimu „curse of dimensionality“. Tokiu atveju reikia taikyti papildomus metodus dimensijos mažinimui tokius kaip PCA. Išsprendžiamas dimensijos persislinkimas ir sukeliamas skaičiavimo greitis.

Algoritmas yra paprastas realizavime jis susideda iš trijų žingsnių.

* Centrinių taškų nustatymas imtyse X kiekvienam klasteriui C.
* Priskiriamas taškas klasteriui kuris yra nepriskirtas ir artimiausias centiniam taškui.
* Sukuriamas naujas centrinis taškas imant visų įeinančių taškų vidurkį. Kartojamas antras žingsnis.

## Afinizmo propogavimas klasterizuojant (Affinity Propagation)

Afinizmą propoguojantis klasterizavimas (AP) - metodas kuria grupes, siunčiant pranešimus tarp taškų porų iki pilno konvergavimo. Duomenų rinkinys apibūdinamas naudojant nedidelį skaičių taškų, kurie yra identifikuojami kaip tie, kurie labiausiai reprezentuoja kitus taškus. Jų atrinkimas vyksta tarp tašku siunčiant pranešimus atsižvelgiant į pranešimų atsakus atnaujinamos grupės t.y. klasteriai. Šis atnaujinimas atitinkamai kartojasi iki pilnos konvergencijos, taigi tuo metu pasirenkami galutiniai pavyzdžiai ir pateikiamas galutinis klasterizavimas.

K.Wang ir kt. [27] savo darbuose paskelbė apie šio metodo limitus: sunku žinoti, kokia parametro "preferencija" vertė gali suteikti optimalų klasterio sprendimą, o svyravimai negali būti automatiškai pašalinami. Šis metodas be papildomų modifikacijų sunkiai pritaikomas automatinėms lanksčioms sistemoms. Skaičiavimo trukmė ir resursų poreikis lyginant su dauguma metodų yra optimalus. Šis metodas gali apdoroti didelius kiekius įrašų taip pat ir požymių (dimensijų). Bet jis nėra integruojamas į dinamines sistemas ir negali būti pritaikytas automatiniams duomenų klasterizavimo procesams

## Hierarchinis klasterizavimas

Hierarchinis klasterizavimas yra bendra klasterizavimo algoritmų grupė, kuri kuria grupes, jas nuosekliai sujungiant arba skaidant. Ši klasterių hierarchija yra pavaizduota kaip medis (arba dendograma).

Vienas iš hierarchinių klasterizavimo būdų Aglomeratyvinis klasterizavimas atliekamas taikant jungimo metodą. Kiekvienas stebėjimas prasideda savo klasteryje, o grupes nuosekliai sujungiamos. Naudojama metriką nusako sujungimo strategiją.

* **Ward** sumažina kvadratų skirtumus visose klasteriuose. Taikant šią metriką suteikiamas reguliarus dydis.
* **Complete linkage** - sumažina maksimalų atstumą tarp stebėjimo porų klasteriuose. Taikoma kai nėra Euklido metrikos.
* **Average linkage** – Sumažina vidutinį atstumą tarp stebėjimo porų klasteriuose. Taikoma kai nėra Euklido metrikos.

I.Davidson [8] nagrinėjant agglomeratyvinio klasterizavimo savybes nustatė jo tinkamumą didelės apimties duomenims kai jis naudojamas kartu su ryšio matrica. Jeigu ji nesudaroma skaičiavimo procesas tiek laiko tiek atminties resursų atžvilgiu smarkiai išauga. Taip pat tyrimai parodė, kad hierarchiniai klasterizavimo metodai yra tinkami kai klasterių dydžiai yra skirtingi (ypač neproporcingi).

## DBSCAN klasterizavimas

DBSCAN Algoritmas peržiūri klasterius kaip didelio tankio sritys, atskirtas mažo tankio sritimis t.y. duomenų tankiu grįstas klasterizavimo metodas. Dėl šios priežasties klasteriai nustatyti gali būti įvairios formos. DBSCAN algoritme klasteris yra pagrindinių pavyzdžių rinkinys, kurių kiekvienas yra arti vienas. Algoritmui yra du parametrai, min\_samples (minimalus skaičius elementų klasteryje) ir eps (spindulys). Didesni min\_samples ar žemesni eps rodo didesnį tankį, būtiną klasteriui sudaryti. Taip pat pagrindinis DBSCAN algoritmo nuo paprastų tankių grįstų metodų, kad jis randa išskirtis t.y. taškus kurių neina priskirti nei vienam klasteriui su nustatytais pradinei parametrais min\_samples ir eps.

S. Chakraborty [6] pateikia DBSCAN pranašumas lyginant su kitais klasterizavimo metodais dinaminėse duomenų bazėse, kas yra aktualu dirbant su kintančiais duomenimis „streaming data“. Jis puikiai tinka labai didelės apimties duomenims t.y. neatima daug atminties resursų ir skaičiavimo trukmę mažai įtakoja duomenų kiekis. Dėl to priežasties jis naudojamas tiesei duomenų kaupyklose „Data Warehouseses“ kur įrašų ir jų požymių kiekiai yra ekstremaliai dideli. Šiam metodui S. Chakraborty [6] išskyrė du esminius trūkumus:

* Jis labai jautrus parametrams. Minimalus parametrų pakeitimas padidina klasterizavimo rezultatų skirtumus.
* Norint nustatyti, ar tai yra objektas yra pagrindinis, reikia aptikti kiekvieną objektą. Tai sudaro daug įvesčių / išvesčių kas didina resursų sąnaudas.

## HDBSCAN klasterizavimas

HDBSCAN - Hierarchinis tankiu grįstas klasterizavimo metodas su išskirtimis, išdirbtas J. Pei ir kt.[24]. Šis algoritmas kombinacija iš hierarchinio klasterizavimo ir DBSCAN. Tokiu būdu apjungtos teigiamos savybės tiek DBSCAN tiek hierarchinio klasterizavimo taip pat kompensuojant vienas kitų trūkumus. Šis algoritmas suderina tvirta bendrąjį ryšį (Robust Single Linkage)[7] su plokščiu grupių ištraukimu (flat cluster extraction)[4]. Šio algoritmo veikimą galima išskirti į pagrindinius 5 žingsnius:

1. Transformuojama erdvė atsižvelgiant į taškų tankį.

Erdvėje atskiriamos dvi grupės: didelio tankio (galimi klasteriai) mažo tankio (galimas triukšmas). Mažo tankio atskyrimas yra atliekamas sudarant naują atstumo matricą literatūroje tai vadinama abipusio pasiekiamumo atstumas (mutual reachability distance):

Čia: yra originali atstumo metrika tarp a ir b taškų.

J. Eldrige ir kt.[12] tarpusavio pasiekiamumo matą išreiškia kaip pagrindinę teoriją, rodančią, kad pasiekiamumo atstumas, kaip transformacija, gerai veikia nustatant lygių rinkinių hierarchiją, nepriklausomai nuo tikrojo tankio pasiskirstymo, iš kurių buvo renkami mūsų taškai.

1. Suformuojamas minimalios apimties medis iš svorinio atstumo grafo;

Čia svorinį grafą sudaro viršūnės iš duomenų taškų o briaunos tai (1.) pateiktas abipusio pasiekiamumo matas. Minimalios apimties medis formuojamas atskiriant silpniausias briaunas iš svorinio grafo ir priskiriant jas prie medžio. Taip sudaroma pilna hierarchija nuo pilnai sujungtų iki pilnai atskirtų elementų. Šis skaičiavimo momentas yra imlus resursams taip kaip kraštinių yra . Todėl naudojamas godusis algoritmas minimalios apimties medžiams formuoti: Prim‘s algoritmas [17]

1. Suformuojama klasterių hierarchija iš sujungtų komponentų t.y. iš minimalios apimties medžių;

Tai lengviau padaryti atvirkštine tvarka: surūšiuojamos medžio kraštinės pagal atstumą (didėjančia tvarka) ir tada pereinama per jį, sukuriant naują klasterį kiekvienai kraštiniai. Vienintelė sunki dalis yra nustatyti dvi grupes, kurios jungsis kartu, tai atliekama surandant duomenų struktūrą. Ir ją apkarpant panaikinat smulkias dendogramos atšakas.

1. Koncentruojama klasterių hierarchija atsižvelgiant į minimalų klasterio dydį;

Pirmasis etapas klasterių išgavime - tai didelės ir sudėtingos klasterių hierarchijos sujungimas į mažesnį medį su šiek tiek daugiau duomenų prie kiekvieno mazgo. Dažniausiai klasterio suskaidymas yra vienas ar du taškai, išsiskiriantys iš klasterio ir tai yra pagrindinis dalykas, o ne tai, kad klasteris skirstomas į du naujus klasterius, kuriuos mes norime laikyti viena nuolatine grupe, kuri yra "prarasti taškai". HDBSCAN algoritme taikoma sąvoka minimalaus klasterio dydžio kuris šiame algoritme išreiškiamas per parametrą minimum\_cluster\_size. Turint minimalaus klasterio dydžio vertę, skaidant hierarchiją t.y apkarpant hierarchijos medį, kiekvienas klasteris t.y jo dydis vertinamas pagal parametrą: minimum\_cluster\_size. Nukirstos šakos sudaro taškus kurie neatitinka parametro ir skaitoma išskirtimis nes nepatenką į sudaromus klasterius. Praėję per visą hierarchiją, gaunamas mažesnis medis su nedideliu skaičiumi mazgų, kurių kiekvienas turi duomenų apie tai, kaip klasterio dydis to mazgo mažėja skirtingu atstumu.

1. Ištraukiami stabilūs klasteriai iš suformuotų medžių.

Intuityviai norime, kad išliktų rinkiniai, kurie išliktų ilgesniam gyvenimui. Trumpalaikiai klasteriai dažniausiai yra tik vieno jungiamojo požymio elementai. Sukuriant plokščią grupę, pridėdamas dar vienas reikalavimas, kad jei pasirinkamas klasteris, negali būti pasirenkamas joks jo palikuonis.

HDBSCAN algoritmas vertina klasterių patvarumą įvedant metriką . Tada mes galime apibrėžti reikšmes  ir . Savo ruožtu, tam tikram grupavimui, kiekvienam taško tame klasteryje apibrėžiama reikšmė . Turint šiuos parametrus galima nustatyti klasterio stabilumą kaip: .

HDBSCAN gali būti taikomas įvairios formos klasteriams, dideliems apimties duomenims, taip pat duomenims kurie apibrėžti dideliu dimensijų skaičiumi.

## Birch klasterizavimo metodas

Birch metodas sukuria medį duotiems duomenims, vadinamą charakteristikos ypatybių medžiu (CFT). CF subklasteriai palaiko reikalingą informaciją klasterizavimui, kuri leidžia nelaikyti visų įvesties duomenų atmintyje. Ši informacija apima:

* Elementų skaičius sub-klasteryje;
* Linijinė suma – n-dimensinis vektorius apimantis suma visų elementų.
* Kvadratinė suma – kvadratinė suma visų elementų L2 regulerizacijos koeeficientų.
* Centroidai –;
* Kvadratiniai normalizuoti centroidai;

Birch algoritmas turi du parametrus, slenkstį ir šakojimą. Šakojimosi veiksnys riboja mazgo pasiskirstymų skaičių, o slenkstis riboja atstumą tarp įvesties pavyzdžio ir esamų poklasių.

Šis algoritmas gali būti laikomas atskiru klasterizavimo atveju arba duomenų mažinimo metodu, nes jis sumažina įvesties duomenis į sub-blokų rinkinį, kuris gaunamas tiesiai iš CFT. Šiuos sumažėjusius duomenis galima toliau apdoroti. T. Zhang ir kt. [28] pristatė šį algoritmą kai klasterizavimo sprendimą esant dideliam požymių (dimensijų) skaičiui, dideliems duomenų apimtims. Metodas pasižymėjo savo dinamiškumu sprendžiant klasterizavimo problemas išnaudojant turimus laiko ir atminties resursus. BIRCH metodas dažniausiai aptinka geriausius klasterizavimo sprendinius atliekant viena duomenų skanavimą t.y. pa žingsninį įrašų perrinkimą. Atliekant papildomus įrašų perrinkimus klasterizavimo rezultatai tik patikslinami. Taip pat šio metodo dėka galima atskirti išskirtis iš bendros duomenų imties.

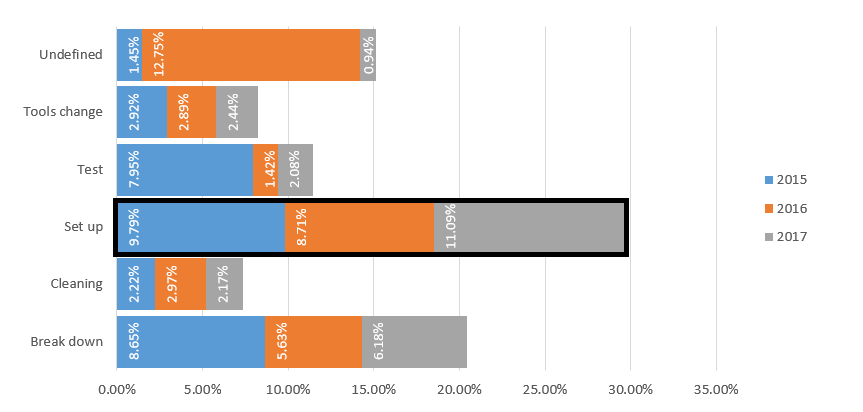
Literatūros apžvalga (apibendrinimas)

Darbo tema yra gamybinio srauto optimizavimas remiantis turima gamybine / technologine informaciją taikant tik matematinius sprendimus. Literatūros apžvalgos tikslas patikrinti organizacinius metodus taikomus gamybiniam sektoriuje. Jų esmę ir taikymo sritį. Įsitikinta, kad organizaciniai metodai tokie kaip LEAN ir AGILE neapima pilno gamybos optimizavimo proceso. Išvada, kad optimalu taikyti organizacinius metodus papildant matematiniais optimizavimo sprendimais. Taip kaip jų veikimas ir taikymo sritys nepersidengia o tik papildo viena kitas. Nagrinėjama problema optimizavime išskiriama kaip atskira šaka JSeP kuri sprendžiama ją interpretuojant kaip TSP. Tai NP sudėtingumo problema kuri jautri įrašų skaičiui. Peržvelgiant tiek euristinius / meta-euristinsius metodus nustatyta kad ir geriausiu atveju maksimalus sekos taškų galimas 210. Mūsų duomenyse sekos taškų sudaro ~700. Šiuo atveju problemą reikia nagrinėti kombinuojant skirtingus metodus. Atsižvelgiant į gamybinius srautus ir buferinių sandėlių apribojimus optimalus sprendimas gamybą skaldyti porcijomis. O tam tikrose porcijose atlikti TSP sprendimą taikant optimizavimo algoritmus tokius kaip: GA, RNN, k-opt, Simulated annealing. Bendras gamybos skaldymas turėtų būti atliekamas taikant klasterizavimo metodus o jų tikslo funkcija TSP sprendinio vertė artėjanti prie 0. Peržvelgiant literatūra matoma, kad klasterizavimo algoritmai yra jautrūs parametrams: įrašų skaičiui, požymių skaičiui, klasterių formai / dydžiui. Nagrinėjami duomenys išsiskiria: žemas įrašų skaičius (~700), ekstremaliai didelis požymių skaičius (~5000), neapibrėžtas klasterių dydis ir forma. Remiantis šiais duomenimis iš populiarių klasterizavimo metodų pritaikomų konkrečiai problemai lieka: HDBSCAN, DBSCAN, BIRCH. Kitais klasterizavimo atvejais dėl didelio požymių (dimensijų) skaičiaus reikia taikyti dimensijų mažinimo priemones tokias kaip PCA ar kt.

Medžiagos ir tyrimų metodai

# Pradiniai duomenys ir jų vertinimas

Keliama problema pagrysta baldų pramonės įmonės duomenimis. Remiantis ekspertine nuomonę išskiriamas gamybos srautą ribojantis darbo centras – HOMAG4 (Gręžimo ir briaunavimo apdirbimo linija). Tarp gaminio efektyvumą apsprendžiančių veiksnių labiausiai išsiskiriantis yra įrenginio reguliavimo laikas keičiant gamybos srautą iš vienos detalės į kitą ([3.6.1.1 Pav. Įrenginio prastovų pasiskirstymas](#_Pav._Įrenginio_prastovų)). Šis segmentas yra potenciali vieta gerinant įrenginio efektyvumo rezultatus.



#### Pav. Įrenginio prastovų pasiskirstymas

Detalių perstatymo trukmė priklauso nuo veiksmų kiekio kuriuos reikia atlikti reguliuojant įrenginius iš vienos detalės į kitą. Veiksmų kiekis priklauso nuo detalių panašumo iš kurios į kūrę pereinama. Detalių panašumas apsprendžiamas įvairiais technologiniais parametrais konkrečiu atveju galima išskirti:

* Gręžimo įranga. Pagrindiniai parametrai (Tiesiogiai nuskaitomi iš gręžimo įrangos duomenų bazės. Duomenų formatas .b2; Priedas Nr.1). Parametrai: gręžimo galvų modeliai ir gręžimo galvos surinkimas (t.y. gręžimo galvą sudaro nustatytas skaičius špindelių nustatytų žingsniu. Kiekvienai detaliai individualiai priskiriama kurie špindeliai montuojami ir kokio skersmens/ilgio grąžtais);
* Briaunavimo įranga. Pagrindiniai parametrai (Tiesiogiai nuskaitomi iš technologinės gaminio informacijos duomenų bazės. Duomenų formatas .accdb): Briaunų profilio tipai, Briaunų pločiai
* Pagalbinė įranga. Pagrindiniai parametrai (Tiesiogiai nuskaitomi iš technologinės gaminio informacijos duomenų bazės. Duomenų formatas .accdb):
  + Pakėlimo liftas (įėjimo/išėjimo greičiai m/min);
  + Detalių dalinimas (įėjimo/išėjimo greičiai m/min).

Šiame darbe JSeP problema formuluojama kaip TSP o atstumas tarp taškų (detalių) formuluojamas kaip detalės panašumo matas išreikštas atsižvelgiant į detalės charakteristikas darbo centre (gręžimo galvas, nustatymų koordinatės, briaunavimo profiliai, pastūmos greičiai ir kt.) . Darbo metu tikrinami panašumo matai Euklido ir Mahalanobis. Problemos sprendimas išskaidomas į dvi dedamąsias: Gaminių klasterizavimas, TSP sprendimas klasteryje.

## Duomenų apdorojimas / transformacija

Kaip buvo minėta duomenys suskirstyti pagal gavimo vietą į dvi grupes:

* Gręžimo / Briaunavimo įrenginių duomenų bazė – duomenų failai .b2 formatu.

REIKIA PARAŠYTI SITA

* Gaminio technologinės informacijos duomenų bazė .accdb formatu.

Duomenims išgauti naudojama Python programavimo kalba, paketas pyodbc. Pastarasis yra skirtas apdoroti prisijungimo duomenis (Connection string) ir suformuoti sąsaja su SQL tipo duomenų bazėmis. Duomenų nuskaitymui taikomas paketas pandas. Paketas skirtas duomenų nuskaitymu ir transformavimui į atitinkamą formatą tolimesniems veiksmams. Programinis kodas pateikiamas priede Nr. 2. (Programinis kodas technologinių duomenų nuskaitymui). Išgauti duomenys pateikti priede Nr. 3 (Technologinių duomenų aprašymai).

Duomenims papildomai atliktas normalizavimas ir duomenų transformavimas t.y. kategoriniai kintamieji paversti į binarinius. Pašalinami neaktualūs kintamieji tokie kaip detalių pavadinimai, jų darbo centras kuriame jos apdirbamos (nagrinėjamos detalės apdirbamos viename darbo centre) ir agreguotos detales kodas (darbui naudojamas tik išplėstinis). Taip pat patikrinami įrašai ar nėra trūkstamų reikšmių, nustatyta, kad tokių reikšmių nėra. Taikomas programinis kodas pateikiamas priede Nr.4. Paruošimo veiksmai atliekami taikant python programavimo kalbą naudojami paketai: numpy , pandas, scipy.

## Žvalgomoji duomenų analizė

Žvalgomoji analizė koncentruojama į technologinius duomenis aprašančius gaminio detales. Remiamasi tuo, kad nevisi parametrai gali būti reikšmingi esamoje problemoje. Jie netiesiogiai susiję su įrenginių perreguliavimu tuo tarpu kai parametrai iš gręžimo įrenginių yra tiesioginiai duomenys įrenginio perreguliavimui. Duomenų analizei taikomas python paketas scipy o vizualizavimui seaborn ir matplotlib.

Žvalgomosios analizės

# Tyrimo metodai

## Duomenų klasterizavimas

HDBSCAN, DBSCAN, BIRCH.

DBSCAN klasterizavimo metodas realizuotas taikant scikit-learn – python programinį paketą. Parinkti nustatymai:

* eps – maksimali distancija tarp vektorių įeinančiu į bendrą klasterį. Tyrimui parinkta reikšmių gardelė [];
* min\_samples – minimalus taškų skaičius sudarantis klasterį. Tyrimui parinkta reikšmių gardelė [];
* metric – tyrimų gardelę sudaryta iš dviejų metrikų: Euklido ir Mahalanobis.
* algorithm - modulyje parinktas algoritmas "The Nearest Neighbors" t.y artimiausio kaimyno metodas, skirtas apskaičiuoti momentinius atstumus ir rasti artimiausius kaimynus.
* n\_jobs – paralelinių veiksmų skaičius, parenkamas parametras -1 t.y išskaidomi veiksmai atsižvelgiant į procesoriaus branduolių skaičių.

BIRCH klasterizavimo metodas realizuotas taikant scikit-learn – python programinį paketą. Parinkti nustatymai:

http://scikit-learn.org/stable/modules/generated/sklearn.cluster.Birch.html

* threshold
* branching\_factor
* n\_clusters
* compute\_labels

HDBSCAN

## Trumpiausio maršruto paieška

<https://ericphanson.com/posts/2016/the-traveling-salesman-and-10-lines-of-python/>

<https://gist.github.com/turbofart/3428880>

https://www.localsolver.com/documentation/index.html

Tyrimo rezultatai ir jų aptarimas

# Tyrimo rezultatai

## Klasterizavimo metodų vertinimas, klasterių interpretavimas

## Trumpiausio maršruto paieškos metodų vertinimas

Išvados

Literatūros sąrašas

1. AMAYA, J.E. et al. A memetic algorithm for the tool switching problem. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* [interaktyvus]. [s.l.]: Springer, Berlin, Heidelberg, 2008. p. 190–202. [žiūrėta 2018-03-04]. Prieiga per internetą: <http://link.springer.com/10.1007/978-3-540-88439-2\_14>.

2. BARD, J.F. A heuristic for minimizing the number of tool switches on a flexible machine. In *IIE Transactions* [interaktyvus]. 1988. Vol. 20, no. 4, p. 382–391. [žiūrėta 2018-03-05]. . Prieiga per internetą: <http://www.tandfonline.com/doi/abs/10.1080/07408178808966195>.

3. BRADLEY, P.S. et al. Constrained k-means clustering. In *Microsoft Research* [interaktyvus]. 2000. p. 1–8. [žiūrėta 2018-03-14]. . Prieiga per internetą: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.33.3257>.

4. CAMPELLO, R.J.G.B. et al. A framework for semi-supervised and unsupervised optimal extraction of clusters from hierarchies. In *Data Mining and Knowledge Discovery* [interaktyvus]. [s.l.]: Springer US, 2013. p. 344–371. [žiūrėta 2018-03-16]. Prieiga per internetą: <http://link.springer.com/10.1007/s10618-013-0311-4>.

5. CATANZARO, D. et al. Improved integer linear programming formulations for the job Sequencing and tool Switching Problem. In *European Journal of Operational Research* [interaktyvus]. 2015. Vol. 244, no. 3, p. 766–777. [žiūrėta 2018-03-05]. . Prieiga per internetą: <https://www.sciencedirect.com/science/article/pii/S0377221715001198>.

6. CHAKRABORTY, S. - NAGWANI, N.K. Analysis and Study of Incremental DBSCAN Clustering Algorithm. In *International Journal of Enterprise Computing and Business Systems* [interaktyvus]. 2011. Vol. 1, no. 2. [žiūrėta 2018-03-15]. . Prieiga per internetą: <http://arxiv.org/abs/1406.4754>.

7. CHAUDHURI, K. - DASGUPTA, S. Rates of convergence for the cluster tree. In *Advances in Neural Information Processing Systems 23* [interaktyvus]. 2010. p. 343–351. [žiūrėta 2018-03-16]. ISBN 9781617823800Prieiga per internetą: <http://cseweb.ucsd.edu/~dasgupta/papers/tree.pdf>.

8. DAVIDSON, I. - RAVI, S.S. Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* [interaktyvus]. [s.l.]: Springer, Berlin, Heidelberg, 2005. p. 59–70. [žiūrėta 2018-03-15]. Prieiga per internetą: <http://link.springer.com/10.1007/11564126\_11>.

9. DENIZEL, M. Minimization of the Number of Tool Magazine Setups on Automated Machines: A Lagrangean Decomposition Approach. In *Operations Research* [interaktyvus]. 2003. Vol. 51, no. 2, p. 309–320. [žiūrėta 2018-03-05]. . Prieiga per internetą: <https://pubsonline.informs.org/doi/abs/10.1287/opre.51.2.309.12784>.

10. DORIGO, M. - CARO, G. DI Ant Colony Optimization: A new meta-heuristic. In *Congress on Evolutionary Computation* [interaktyvus]. 1999. p. 1470–1477. [žiūrėta 2018-03-09]. Prieiga per internetą: <http://ieeexplore.ieee.org/abstract/document/782657/>.

11. EBERHART, R. - KENNEDY, J. A new optimizer using particle swarm theory. In *MHS’95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science* [interaktyvus]. p. 39–43. [žiūrėta 2018-03-09]. Prieiga per internetą: <http://ieeexplore.ieee.org/abstract/document/494215/>.

12. ELDRIDGE, J. et al. Beyond Hartigan Consistency: Merge Distortion Metric for Hierarchical Clustering. In [interaktyvus]. 2015. [žiūrėta 2018-03-17]. . Prieiga per internetą: <https://arxiv.org/pdf/1506.06422v2.pdf>.

13. FAN, H. Discrete Particle Swarm Optimization for TSP based on Neighborhood. In *Journal Of Computational Information Systems* [interaktyvus]. 2010. Vol. 10, p. 3407–3414. [žiūrėta 2018-03-09]. . Prieiga per internetą: <https://pdfs.semanticscholar.org/63a4/1c7e9db00baacb249668ba112b7cf1830e87.pdf>.

14. FLEMING, D. *Lean Logic: A Dictionary for the Future and How to Survive It* [interaktyvus]. . 2016. 658 p. ISBN 978-1-60358-648-1.

15. FURRER, M. - MÜTZE, T. An algorithmic framework for tool switching problems with multiple objectives. In *European Journal of Operational Research* [interaktyvus]. 2017. Vol. 259, no. 3, p. 1003–1016. [žiūrėta 2018-03-05]. . Prieiga per internetą: <https://www.sciencedirect.com/science/article/pii/S0377221716309596>.

16. GHIANI, G. et al. An exact solution to the TLP problem in an NC machine. In *Robotics and Computer-Integrated Manufacturing* [interaktyvus]. 2007. Vol. 23, no. 6, p. 645–649. [žiūrėta 2018-03-04]. . Prieiga per internetą: <https://www.sciencedirect.com/science/article/pii/S0736584507000282>.

17. GREENBERG, H.J. Greedy Algorithms for Minimum Spanning Tree Proof of Optimality. In *Proceedings of the American Mathematical Society* [interaktyvus]. 1998. p. 2–4. [žiūrėta 2018-03-17]. . Prieiga per internetą: <http:::www.cudenver.eduu>.

18. GUTIN, G. et al. Traveling salesman should not be greedy: Domination analysis of greedy-type heuristics for the TSP. In *Discrete Applied Mathematics* [interaktyvus]. 2002. Vol. 117, no. 1–3, p. 81–86. [žiūrėta 2018-03-06]. . Prieiga per internetą: <https://www.sciencedirect.com/science/article/pii/S0166218X01001950>.

19. YVES CRAMA, ANTOON W. J. KOLEN, ALWIN G. OERLEMANS, F.C.R.S. Minimizing the Number of Tool Switches on a Flexible Machine. In [interaktyvus]. 1994. Vol. 6, p. 1987–1988. [žiūrėta 2018-03-04]. . Prieiga per internetą: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.867.7911&rep=rep1&type=pdf>.

20. KARAKAYALI, I. - AZIZOǦLU, M. Minimizing total flow time on a single flexible machine. In *International Journal of Flexible Manufacturing Systems* [interaktyvus]. 2006. Vol. 18, no. 1, p. 55–73. [žiūrėta 2018-03-04]. . Prieiga per internetą: <http://link.springer.com/10.1007/s10696-006-9000-6>.

21. KONAK, A. et al. Minimizing the number of tool switching instants in Flexible Manufacturing Systems. In *International Journal of Production Economics* [interaktyvus]. 2008. Vol. 116, no. 2, p. 298–307. [žiūrėta 2018-03-05]. . Prieiga per internetą: <https://www.sciencedirect.com/science/article/pii/S0925527308003034>.

22. LAPORTE, G. et al. Exact algorithms for the job sequencing and tool switching problem. In *IIE Transactions (Institute of Industrial Engineers)* [interaktyvus]. 2004. Vol. 36, no. 1, p. 37–45. [žiūrėta 2018-03-05]. . Prieiga per internetą: <http://www.tandfonline.com/doi/abs/10.1080/07408170490257871>.

23. MÜTZE, T. Scheduling with few changes. In *European Journal of Operational Research* [interaktyvus]. 2014. Vol. 236, no. 1, p. 37–50. [žiūrėta 2018-03-04]. . Prieiga per internetą: <http://page.math.tu-berlin.de/~muetze/papers/scheduling.pdf>.

24. PEI, J. et al. *Advances in Knowledge Discovery and Data Mining* [interaktyvus]. . 2014. ISBN 978-3-319-06607-3.

25. PRIVAULT, C. - FINKE, G. K-Server problems with bulk requests: An application to tool switching in manufacturing. In *Annals of Operations Research* [interaktyvus]. 2000. Vol. 96, no. 1/4, p. 255–269. [žiūrėta 2018-03-04]. . Prieiga per internetą: <http://link.springer.com/10.1023/A:1018939132489>.

26. TANG, C. - DENARDO, E. Models Arising fro a Flexible Manufacturing Machine, Part I: Minimization of the Number of Tool Switches. In *Operations Research* [interaktyvus]. 1988. Vol. 35, no. 5, p. 767–777. [žiūrėta 2018-03-05]. . Prieiga per internetą: <https://pubsonline.informs.org/doi/abs/10.1287/opre.36.5.778>.

27. WANG, K. et al. Adaptive Affinity Propagation Clustering. In *Automatica* [interaktyvus]. 2007. Vol. 33, no. 12, p. 1242–1246. [žiūrėta 2018-03-15]. . Prieiga per internetą: <http://arxiv.org/abs/0805.1096>.

28. ZHANG, T. et al. BIRCH: an efficient data clustering method for very large databases. In *Proc. ACM-SIGMOD Int. Conf. Management of Data* [interaktyvus]. 1996. p. 103–114. [žiūrėta 2018-03-16]. Prieiga per internetą: <http://delivery.acm.org/10.1145/240000/233324/p103-zhang.pdf?ip=193.219.170.65&id=233324&acc=ACTIVE SERVICE&key=1FA3353941FE8055.4A4E353D8E57EBBB.4D4702B0C3E38B35.4D4702B0C3E38B35&\_\_acm\_\_=1521191017\_aff6d364840f053eb21a719d05bdb293>.

29. *Graph Theory and Network Flows*. . 49-78 p. ISBN 2850476641.

Priedai

1. Pavyzdys CNC duomenų failo aprašo detalei priskirta gręžimo galvą (darbo įrankio charakteristika):

\_B\_O\_H\_R\_K\_O\_P\_F\_D\_A\_T\_E\_N\_ \_F\_U\_E\_R\_ \_V\_A\_R\_I\_A\_N\_T\_E\_N\_P\_R\_O\_G\_R\_A\_M\_M\_.\_

\_INHALTSVERZEICHNIS : >< KUNDE : >XXXXXXXXXXXXXXXXXX<

\_MASCHINENNUMMER : >0-251-11-0950/51< SACHBEARB.: >BSTWIN32<

\_PRODUKTSCHLUESSEL : >BST 500< DATUM : >03-02-2017<

---------------------------------------------------------------------------

\*BOHRKOPF !SPINDEL!WK!DELTA-X !DELTA-Y !DURCHMESSER! KOMMENTAR

----------+----------+----------+----------+-----------+-------------------

[V11]

A11 ! 2!62! 0.000! -128.000! 5.000! 70.0

A11 ! -11!62! 0.000! 160.000! 5.000! 70.0

[V11]

A12 ! -5!62! 0.000! -32.000! 15.000! 70.0

[V11]

A22 ! -1!62! 0.000! -160.000! 15.000! 70.0

[V11]

A13 ! 2!62! 0.000! -128.000! 5.000! 70.0

A13 ! -3!62! 0.000! -96.000! 8.000! 70.0

[V11]

A14 ! 2!62! 0.000! -128.000! 5.000! 70.0

A14 ! -11!62! 0.000! 160.000! 5.000! 70.0

[V11]

A15 ! 2!62! 0.000! -128.000! 5.000! 70.0

A15 ! -11!62! 0.000! 160.000! 5.000! 70.0

---------------------------------------------------------------------------

\*BOHRKOPF !SPINDEL!WK!DELTA-X !DELTA-Y !DURCHMESSER! KOMMENTAR

----------+----------+----------+----------+-----------+-------------------

[NoValidForUse]

O11 ! 1!62! 0.000! 0.000! 0.000!

[NoValidForUse]

O21 ! 1!62! 0.000! 0.000! 0.000!

[NoValidForUse]

O12 ! 1!62! 0.000! 0.000! 0.000!

[NoValidForUse]

O22 ! 1!62! 0.000! 0.000! 0.000!

[NoValidForUse]

O13 ! 1!62! 0.000! 0.000! 0.000!

[NoValidForUse]

O23 ! 1!62! 0.000! 0.000! 0.000!

---------------------------------------------------------------------------

\*BOHRKOPF !SPINDEL!WK!DELTA-X !DELTA-Y !DURCHMESSER! KOMMENTAR

----------+----------+----------+----------+-----------+-------------------

[h11]

H11 ! 6!62! 0.000! 160.000! 8.000! 70.0

H11 ! 8!62! 0.000! 224.000! 8.000! 60.0

[h11]

H21 ! -1!62! 0.000! 0.000! 8.000! 60.0

H21 ! -3!62! 0.000! 64.000! 8.000! 70.0

1. Programinis kodas technologinių duomenų nuskaitymui.

sql = """

SELECT

ITEMCHAR.WC as WorckCenter,

ITEMCHAR.ITEM\_ID as ProductAGR,

ITEMCHAR.ITEM\_NAME as ProductName,

ITEMCHAR.NPC\_MAX as NPC,

ITEMCHAR.QTY\_BLANK\_1 as Parts,

ITEMCHAR.MARSRUTAS as Path,

ITEMCHAR.WIDTH\_1 as Width,

ITEMCHAR.LENGHT\_1 as Lenght,

ITEMCHAR.Liftas\_Past as LiftPushSpeed,

ITEMCHAR.Liftas\_nuos\_veikimas as LiftRegularSpeed,

ITEMCHAR.pas\_stal\_iejimas as RT1EntranceSpeed,

ITEMCHAR.pas\_stal\_isejimas as RT1ExitSpeed,

ITEMCHAR.[2\_pas\_stal\_iejimas] as RT2EntranceSpeed,

ITEMCHAR.[2\_pas\_stal\_isejimas] as RT2ExitSpeed,

ITEMCHAR.[1\_krast\_stak\_pastuma] as EM1PushSpeed,

ITEMCHAR.[1\_krast\_stak\_Ciklas] as EM1PushCycle,

ITEMCHAR.[2\_krast\_stak\_pastuma] as EM2PushSpeed,

ITEMCHAR.[2\_krast\_stak\_Ciklas] as EM2PushCycle,

ITEMCHAR.[1\_krast\_stak\_Kumstelio\_sk] as Position,

LEFT(ITEM\_ID, inStr(ITEM\_ID, '-')+2) +

COLLORS.COLINX +

RIGHT(ITEM\_ID, LEN(ITEM\_ID)-(inStr(ITEM\_ID, '-')+3)) as Product,

LEFT(ITEM\_ID, inStr(ITEM\_ID, '-')-1) as BR

FROM ITEMCHAR, COLLORS

WHERE

ITEMCHAR.COLOR\_INDEX = COLLORS.UNKNOWN AND

WC in (12010) AND

ITEMCHAR.archivuotas = False

"""

data = pd.read\_sql(sql,cnxn);

1. Technologinių duomenų aprašymai

|  |  |
| --- | --- |
| WorckCenter | Darbo centro numeris kuriuo įrenginys identifikuojamas; |
| ProductAGR | Produkto/Detalės ID su eliminuota apdailos spalva 'X' *pavyzdys: BR3-ITX104 spalvos W (balta), B (juoda), O (ąžuolo) ar kt.* |
| Product | išplėstinis produkto/detalės ID atsižvelgiant į spalvas |
| BR | bazinis detalės lygis |
| ProductName | produkto pavadinimas |
| NPC | detalių skaičius per minutę |
| Parts | detalių skaičius gaunamas iš ruošinio |
| Width | detalės plotis (mm.) |
| Lenght | detalės ilgis (mm.) |
| LiftPushSpeed | Pakrovimo lifto pastūmos greitis (m/min.) |
| LiftRegularSpeed | Pakrovimo lifto regulerus veikimas (m/min.) |
| RT1EntranceSpeed | pirmojo pasukimo stalo pastūmos greitis įėjimo pusėje (m/min.) |
| RT1ExitSpeed | pirmojo pasukimo stalo pastūmos greitis išėjimo pusėje (m/min.) |
| RT2EntranceSpeed | antrojo pasukimo stalo pastūmos greitis įėjimo pusėje (m/min.) |
| RT2ExitSpeed | antrojo pasukimo stalo pastūmos greitis išėjimo pusėje (m/min.) |
| EM1PushSpeed | pirmų briaunavimo staklių pastūmos greitis (m/min.) |
| EM1PushCycle | pirmų briaunavimo staklių ciklas (vnt/min.) |
| EM2PushSpeed | antrų briaunavimo staklių pastūmos greitis (m/min.) |
| EM2PushCycle | antrų briaunavimo staklių ciklas (vnt/min.) |
| Position | atraminiai kumšteliai |

1. Programinis kodas technologinių duomenų paruošimui.

# Check data sets for missing values

print('Bad rows (with missing values) caunt: ', data.shape[0]-data.dropna().shape[0],

'\nRaw data shape: ', data.shape,

'\nClean data shape: ', data.dropna().shape,);

>>> Bad rows (with missing values) caunt: 0

>>> Raw data shape: (795, 21)

>>> Clean data shape: (795, 21)

# Seperate data to differents categorys by type

continuous = ['NPC', 'Width', 'Lenght', 'LiftPushSpeed', 'LiftRegularSpeed',

'RT1EntranceSpeed', 'RT1ExitSpeed', 'RT2EntranceSpeed', 'RT2ExitSpeed',

'EM1PushSpeed', 'EM1PushCycle', 'EM2PushSpeed', 'EM2PushCycle', 'Parts']

id = ['Product']

categorical = ['Position', 'BR', 'Color']

binary = ['Path', 'Chipboard']

drop = ['ProductAGR', 'ProductName', 'WorckCenter']

# functions to menage original data

def set\_chipboard(row):

if row['Product'][-1] == 'A': return 1

else: return 0

def set\_color(row):

if row['Product'][6] == 'W': return 'white'

elif row['Product'][6] == 'B': return 'black'

elif row['Product'][6] == 'O': return 'oack'

elif row['Product'][6] == 'C': return 'chary'

elif row['Product'][6] == 'X': return 'none'

else: return ''

def set\_path(row):

if row['Path'] == 1: return 1

else: return 0

def set\_dummies(data, dummies):

df = data

for dummy in dummies:

df\_dummies = pd.get\_dummies(data[dummy], prefix=dummy)

df = pd.concat([df, df\_dummies], axis=1)

return df

# Data normalization

df[continuous] = df[continuous].apply(lambda x: (x - np.mean(x)) / (np.max(x) - np.min(x)))

1. Programinis kodas duomenų vizualizavimo – QQ Plot.

def QQplot(attribute, options):

probscale.probplot(

df[attribute].tolist(),

plottype='qq',

bestfit=True,

estimate\_ci=True,

problabel='Standard Normal Quantiles',

datalabel=attribute,

\*\*options

)

ax.legend(loc='lower right')

ax.set\_ylim(bottom=-1, top=1)

sns.despine(fig)

scatter\_options=dict(

marker ='o',

markerfacecolor='none',

markeredgecolor='firebrick',

markeredgewidth=1.25,

linestyle='none',

alpha=0.35,

zorder=5,

label='Observations'

)

line\_options=dict(

label = 'BF Line',

color = 'black',

linestyle=':',

linewidth = 1.5

)

options = dict(

scatter\_kws = scatter\_options,

line\_kws = line\_options,

ax = None

)

fig, axes = plt.subplots(5,3, figsize=(30,45), sharex=True)

axes = axes.flatten();

for ax, attribute in zip(axes, continuous):

options['ax'] = ax

QQplot(attribute, options)