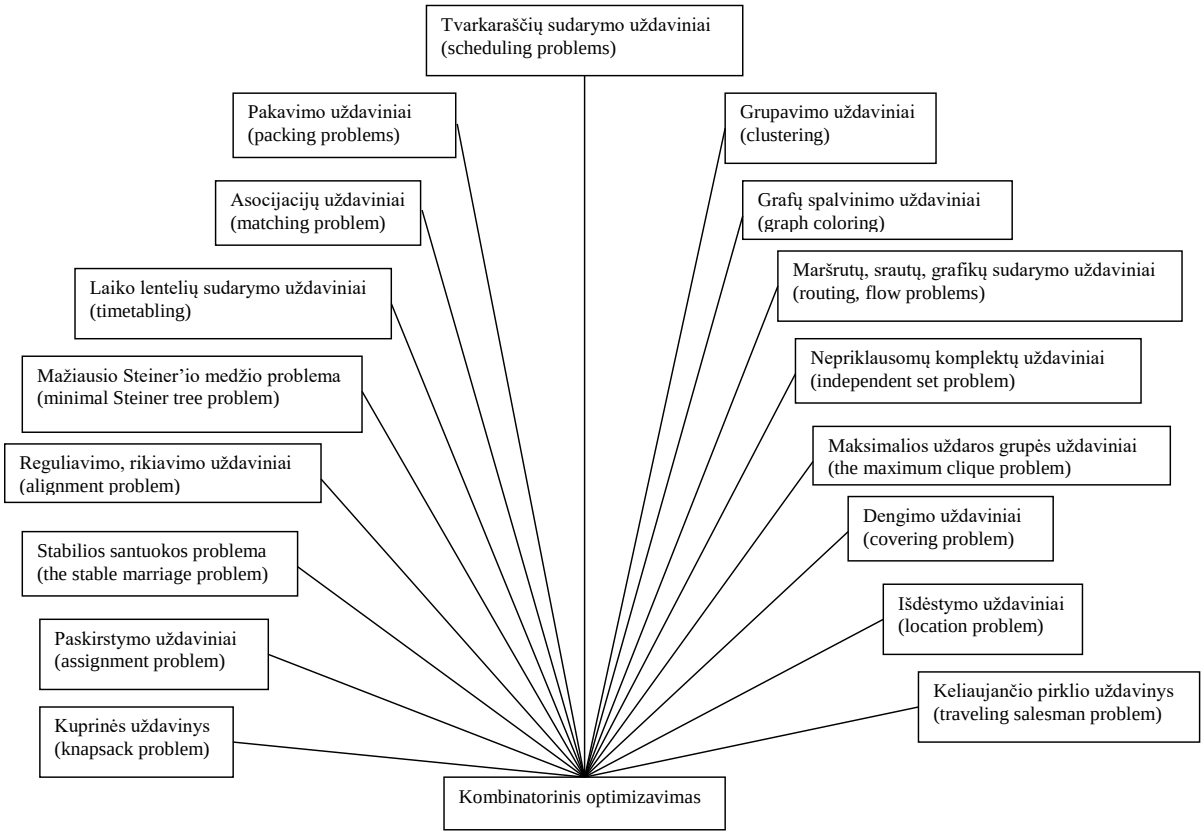
Optimizavimu grįsti uždaviniai

Formaliai optimizavimo uždavinį galima aprašyti pora (S,f), kur S yra leistinų sprendinių aibė, f yra funkcija, kurios apibrėžimo sritis yra aibė S, o šios funkcijos reikšmių aibė yra realieji skaičiai. Ši funkcija yra vadinama tikslo funkcija. Sprendinių aibė priklauso nuo sprendžiamo uždavinio, pavyzdžiui, tai gali būti leistini laikai, kažkokia parametrų erdvė ar norimi prioritetiniai pasirinkimai. Tikslo funkcija yra taip pat priklauso nuo uždavinio ir jeigu sprendžiamas uždavinys, kuriame reikia maksimizuoti pelną, tokiu atveju ieškoma tos funkcijos maksimumo. Jeigu norima sumažinti išlaidas arba rasti geriausią sprendinį, kur sąnaudos būtų minimalios, tada sprendžiamas minimizavimo uždavinys.

Apibendrinant, bet kokios optimizavimo problemą aprašo uždavinį, kuriame siekiama maksimizuoti arba minimizuoti tikslo funkciją leistinų sprendinių aibėje.

Taigi, kokios problemos yra optimizavimo uždaviniai? Pavyzdžiui, kombinatorinio optimizavimo uždavinius galima atvaizduoti tokia diagrama.



Optimizavimo uždaviniai gali būti skirstomi pagal sudėtingumą, o sudėtingumas apibrėžiamas kaip sprendinio radimui sunaudoto laiko arba atminties kiekis. Priklausomai nuo uždavinių, tai gali būti daugianarinis, faktorialinis arba eksponentinis sudėtingumas. Realiems uždaviniams yra sukurti tik eksponentinio arba faktorialinio sudėtingumo algoritmai. Kadangi jie

reikalauja labai daug sąnaudų, todėl mes juos sprendžiam apytiksliai: perdarome į uždavinius, kurie gali būti polinominio sudėtingumo.

Tiriamas sprendimo laiko ar sąnaudų padidėjimas, augant duomenų skaičiui, pavyzdžiui, turint daugianarinį sudėtingumą n^2 ar n^3 sprendimo laikas dar yra priimtinas, bet taikomuosiuose uždaviniuose dažniausiai sudėtingumas būna eksponentinis, pavyzdžiui 2^n arba faktorialinis. Pasižiūrėjus į lentelę, matoma, kaip išauga sprendimo laikas, turint ne 10 o 100 pradinių duomenų, o jeigu nagrinėjama dar daugiau duomenų tai sprendimo laikas gali viršyti netgi visatos amžių.

Vienas iš lengviausiai suprantamų ir įsivaizduojmų yra kuprinės uždavinys arba kitaip vadinamas – vagies uždavinys. Uždavinio formulavimas yra toks: turim kelis daiktus su dydžiais ir vertėm, o tikslas – į kuprinę ar tam tikrą talpą sudėti daiktus, kurių vertė būtų didžiausia, bet neviršijant kuprinės talpos. Šis uždavinys vadinamas vagies uždaviniu todėl, kad vagiui atėjus į parduotuvę, jis turi pasirinkti, kokius daiktus jam reikės neštis. Pavyzdžiui, atėjus į buitinės technikos parduotuvę, tikėtina, kad jis į savo kuprinę susidėtų mobilius telefonus, nes jie užima mažai vietos ir turi didelę vertę, bet neimtų plazminio televizoriaus, kuris nors ir turi didelę vertę, bet yra per didelis, kad tilptų į kuprinę.

Nepaisant to, kad kuprinės uždavinys yra formalizuotas būtent taip, bet jis gali būti taikomas ir kitose srityse, pavyzdžiui konteinerio užpildymui, investicijų portfelio sudaryme, kur daiktus attinka skirtingų tipų investicijos, personalo samdymo problemose.

Kitas populiarus problemų tipas yra tvarkaraščių sudarymo uždaviniai. Su tvarkaraščių sudarymo uždaviniais jūs esate tiesiogiai susidūrę mokykliniuose ar universitetiniuose tvarkaraščiuose, taip pat viešbučiai, sanatorijos, aptarnavimo punktai irgi sudaro atvykimo ir veiklų pasiskirstymo tvarkaraščius, pavyzdžiui suskirsto procedūras pagal laiką ir galimybes, kad būtų kuo daugiau personalo būtų užimta, nebūtų prastovų ir kad klientų poreikiai būtų patenkinti. Mokyklinio tvarkaraščio atveju tikslas yra kad mokiniai neturėtų laisvų pamokų, pasirūpinti, kad mokytojui nebūtų didelio tarpo tarp pamokų, pavyzdžiui, kad užimta tik pirma ir penkta pamoka, ir taip pat svarbu atkreipti dėmesį, kad mokiniams tą pačią dieną nebūtų kelios iš eilės tos pačios rūšies pamokos.

Pakavimo uždaviniai siejami su kuprinės uždaviniu, tačiau yra tam tikrų skirtumų, čia problema formuluojama, pavyzdžiui, kaip sudėlioti prekes, kad dėžėje nebūtų tuščių vietų ir kad mūsų išlaidos pakavimui būtų minimizuojamos, įvertinant sandėliavimo kaštus.

Kitas optimizavimo problemų tipas yra grupavimo uždaviniai, kai norimaa suskirstyti objektus į tam tikrus klasterius. Tokie uždaviniai yra populiarūs vaizdų apdorojime, signalų apdorojime, kai norima nuspręsti, į kurią grupę paklius gautas naujas objektas.

Kita optimizavino problema yra vadinama keliaujančio pirklio uždaviniu. Jo tikslas yra kuo mažiausiu atstumu apkeliauti visus objektus ir grįžti į pradinį tašką , tai galima sieti pavyzdžiui su minimizuojamomis degalų sąnaudomis skirstant prekes po miestus. Šitas uždavinys yra aktualus logistikai, kai įmonė nori geriausiu maršrutu išvežioti prekes, bet taikomas ir mikroshemų gamyboje, optimaliam gręžiamų skylių išdėstymui, kristalografijoje ir kitur.

Kitas optimizavmo problema vadinama pjaustymo uždaviniu. Dažniausiai jis taikomas gamyboje, kai tarkim turima tam tikra plokštė, ir iš jos norima išpjauti atitinkamą skaičių detalių.

Pagrindinis uždavinio tikslas yra minimizuoti laisvą plotą. Vienas iš pavyzdžių, kurį galima būtų įsivaizduoti, tai yra siuvėjas, kuris turi ruloną medžiagos ir nori padaryti iškarpas, todėl jam reikia rasti iškarpų išdėliojimo sprendinį, kad medžiagos atliktų mažiausiai.

Dengimo uždaviniai dažniausiai taikomi mobilių operatorių, turbūt visiems teko susidurti su situacija, kad kai kuriose Lietuvos vietose nėra mobiliojo ryšio. Todėl mobilių operatorių tikslas yra sudėti bokštus taip, kad neliktų nepadengtų vietų.

Vienas iš dažniausiai taikomų yra gamybos planavimo uždavinys, kai mes turim skirtingus daiktus, kurie gaminami skirtingose mašinose. Šio uždavinio pagrindinis tikslas yra minimizuoti gamybos laiką ir atitinkamai maksimizuoti visų mašinų užimtumą, kad nebūtų prastovų, ir užtektų turimų laiko ir mašinų resursų: ir žinoma, kad produkcijos kiekis pagal užsakymus būtų įgyvendintas.

Dažniausiai gamybos planavimo uždaviniuose sprendinys yra vaizduojamas *Gant* diagramomis, kur matosi mašinų užimtumai skirtingais produktais. Grafinis vaizdas leidžia pamatyti sprendinio kokybę, kuri nusakoma reikalavimu, kad diagramoje baltų tarpų būtų kuo mažiau.

Optimizavimo metodai

Skiriamos trys optimizavimo uždavinių sprendimo klasės: tikslieji algoritmai, euristiniai algoritmai ir meta-euristiniai algoritmai.

Vienas iš paprasčiausiai suprantamų tiksliųjų algoritmų yra: pilnojo perrinkimo – kai išbandomi visi variantai ir iš jų išsirenkamas geriausias. Kiti yra susiję su tiesinio programavimo uždaviniu – kai turima tam tikrą sritis erdvėje, turimi tiesiniai apribojimai ir bandoma surasti geriausią sprendinį. Pilnojo perrinkimo algoritmas suveikia visada, tik yra viena problema: jeigu turim labai daug duomenų, sprendinio radimo laikas bus neprrimtinas.

Dėl didelio duomenų kiekio realiuose uždaviniuose vietoj tiksliųjų algoritmų yra taikomi euristiniai arba metaeuristiniai metodai. Jų tikslas sumažinti sprendimo algoritmo sudėtingumo klasę nuo eksponentinio arba faktorialinio iki polinominio. Pagrindinis euristinių metodų tikslas: greitai gauti gerą sprendinį, vadinasi, taupyti laiko ir resursų sąnaudas, o gautas sprendinys nebūtinai turi būti geriausias, bet pakankamai geras pagal užduotus kriterijus.

Pagrindiniai euristiniai metodai yra atsitikinių paieškų algoritmai ir jų modifikacijos: pavyzdžiui lokalios paieškos algoritmas, prisitaikančios atminties ir kiti. Gali būti, kad lokaliojoje paieškoje, kaip ir kituose metoduose galima kartai nuklysti į kitą pusę, tai yra užstrigti lokaliame minumume, bet tai yra visų metodų bėda, kuri sprendžiama pavyzdžiui paleidžiant algoritmo veikimą iš naujo. Dažniausiai realiuose uždaviniuose taikoma meta-euristiniai algoritmai, kurie sujungia kelis paprastesnius algoritmus ir dažnai šie algortimai remiasi gamtos procesais, kadangi gamta yra įpratusi sutvarkyti viską taip, kaip jai geriausia.

Keli meta-euristiniai algoritmai: modeliuojamas atkaitinimas, kuris remiasi fizikiniu procesu, kai mes po truputėlį šildom ir šaldom medžiagą, ji susidėlioja pagal geriausią struktūrą; genetiniai algoritmai, pagrįsti Darvino evoliucijos principais; paieška su apribojimais, Tabu paieška, kurios panašios į lokalią paiešką, bet negalima grįžti atgal.

Ne tik genetinių algoritmų idėja kilusi iš natūralios atrankos gamoje, bet iš gamtos procesų gauti ir daugiau meta-euristinių algoritmų: pavyzdžiui skruzdžių kolonijos optimizavimo metodas, dalelių spiečiaus algoritmas, kurį galima iliustruoti žuvų būrelio elgesiu įmetus maisto į vandenį; kintamų aplinkų algoritmaskuris atitinka gamtos objektų prisitaikymą prie pasikeitusių aplinkos ir gyvenimo sąlygų, Tai pat galima naudti ir ne su gamta susijusius algoritmus: dirbtinius neuroninius tinklus bei įvairius algoritmų hibridus.

Genetiniai algoritmai atsirado pakankamai seniai ir tai yra atskira klasė evoliucinių algoritmų, kurie yra pagrįsti biologija. Į genetinį algoritmą įeina tokie elementai, kaip paveldėjimas, mutacija, atranka ir kryžminimas. Šio algoritmo esmė: sugeneruojama kelių pradinių sprendinių populiacija, iš jų atrenkami geriausieji, tada atliekamos kryžminimo ir mutacijos procedūros, gaunama nauja sprendinių aibė, iš jos vėl išrenkami geriausi, ir po kelių epochų gaunamas geriausias sprendinys. Čia ir yra algoritmo panašumas į gamtą, kur per paveldėjimą išlieka tik stipriausi, turintys, tarkim, šnekant apie gazeles, tik gražiausius ragus ir didžiausią greitį, kuriuo naudojasi bėgdami nuo liūtų.

Skruzdžių kolonijos algoritmas yra pagrįstas skruzdžių maisto ieškojimo procesu. Ko gero, yra tekę stebėti skruzdžių takus, kai iš pradžių skruzdės išeina ieškoti maisto atsitiktinai, o jeigu randa maisto šaltinį, grįžta, praneša kitoms naudodamos feromonus ir po truputėlį visos pradeda eiti tuo pačiu keliu iki rasto maisto šaltinio, kol jis neišsenka. Optimizavimo metode viskas vyksta panašiai, tai yra jeigu rastas vieną iš geresnių sprendinių, kuris atitinka maisto šaltinį, tai aplink jį ir vyksta tolimesnė paieška.

Kitas gamtos procesais pagrįstas algoritmas yra dirbtinio bičių spiečiaus algoritmas, kuris savo veikimu panašus į bičių spiečiaus elgesį. Įsivaizduokite avilio darbą, kur vienos bitės yra darbininkės, kitos – žvalgai, ir yra motinėlė, kuri visus prižiūri. Geriausi sprendinio ieškojimas pagrįstas tuo, kad bitės darbininkės, prieš tai informuotos bičių žvalgių suneša nektarą iš geriausių vietų, o nektaro kiekis atitinka sprendinio gerumą optimizavimo uždavinyje.

Dalelių spiečiaus algoritmas savo esme panašus į atsitiktinę paiešką, kur judama link geriausio sprendinio. Šiu atveju į vietą, kur surastas geras sprendinys, atitinkantis maisto šaltinį, sumigruoja daugiau dalelių, pavyzdžiui kaip žuvų susibūrime maisto vietoj arba vieningame paukščių skridime į Pietus.

Vienas iš įdomesnių algoritmų yra liūtų algoritmas, kuris yra pagrįstas dominavimu, patelių paieška ir teritorijos žymėjimu. Kaip ir gamtoj išlieka geriausi liūtų patinai, tai ir čia išlieka tik geriausi sprendiniai, kurie yra dominuojantys, ir iš kurių gaunama nauja sprendinių karta.

Apibendrinant, kiekvienam konkrečiam uždaviniui galima atrasti metodą, kuris jam tinka geriausiai. Dėl to dažniausiai optimizavimo uždaviniuose yra lyginami keli metodai ir nusprendžiama, kuris iš jų bei su kokiais parametrais, yra labiausiai tinkamas. Parametrų pavyzdžiai gali būti pradinės populiacijos skaičius, įvedamos atsitktinio judėjimo, mutacijos, kryžminimo tikimybės, metodo sustojimo sąlyga. Iš praktinių uždavinių sprendimo yra rastas tam tikras paradoksas, kuris vadinamas *no-freelanch*, kurį trumpai galima apibūdinti: jeigu surastas vienas metodas, kuris konkrečiam uždaviniui veikia labai gerai, tai nereiškia, kad jis gerai veiks ir kitiems uždaviniams.