Liet. matem. rink, **46***, spec. nr.*, 2006, 366–370

Stočių išdėstymo algoritmo analizė

Saulius LAZARAVIČIUS, Narimantas LISTOPADSKIS (KTU)
el. paštas: saulaz@gmail.com, narlis@ktu.lt

Problemos aprašymas

„Set Covering“ tipo užduotys yra žinomos ir sprendžiamos jau kuris laikas. Šiame straipsnyje išnagrinėsime algoritmo, skirto „Set Covering“ problemai spręsti, taikymą naujame ir aktualiame nūdienai uždavinyje – optimaliame stočių (siųstuvų) išdėstyme. Dėl spartaus technologijų vystymosi ši problema pramonėje yra labai aktuali, todėl būtina sukurti automatizuotas skaičiavimo procedūras šiai problemai spręsti.

Problemos esmė ta, kad turime teritoriją, kurioje norime teikti mobiliojo ryšio paslaugą. Tam tikslui, šioje teritorijoje turime išdėstyti siųstuvus. Iškart iškyla klausimas kiek tokių siųstuvų reikės ir kaip juos išdėstyti teritorijoje? Visų pirma, tai priklauso nuo eilės siųstuvo parametrų. Bene pagrindinis iš jų, tai teritorijos, kurią siųstuvas gali aptarnauti stovėdamas tam tikroje vietoje, ploto. Kitas, nemažiau svarbus siųstuvo parametras, tai siųstuvo pastatymo kaina tam tikroje teritorijoje.

Akivaizdu, kad geriausia būtų, jog suminė pastatytų siųstuvų kaina būtų kiek galima mažesnė.

Užduoties aprašymas

Duota:

- Teritorija, kurioje reikia teikti mobiliojo ryšio paslaugą. Ši teritorija yra suskirstyta pagal pasirinktą tinklelį. Pvz., kvadratais, apkritimais, aštuoniakampiais.
- Potencialios siųstuvų buvimo vietos (tinklelio elementai, kuriuose galima statyti siųstuvus).
- Siųstuvo pastatymo kaina konkrečiame teritorijos elemente.
- Teritorija, kurią siųstuvas gali aptarnauti, būdamas konkrečiame teritorijos elemente.

Rasti:

Teritorijos elementus, kuriuose pastatę siųstuvus padengsime norimą procentą teritorijos ir tai padarysime, su kiek galima mažesniu kiekiu pinigų.

„Skruzdžių kolonijos“ algoritmas

Matematinė uždavinio formuluotė:

Duota: $A = (a_{ij})$, čia A – $m \times n$ matrica, kur

$$a_{ij} = \begin{cases} 1, & \text{jei } j\text{-is stulpelis padengia } i\text{-ją eilutę,} \\ 0, & \text{kitais atvejais;} \end{cases}$$

b_j – kiekvienam stulpeliui priskirti stoties pastatymo kaštai, $\forall j \;\; b_j \geqslant 0$.

Rasti:

$$f(y) = \sum_{j=1}^{n} b_j \cdot y_j \to \min,$$

kai galioja apribojimai:

$$\sum_{j=1}^{n} a_{ij} \cdot y_j \geqslant 1, \quad i = \overline{1, m},$$

$$y_j = \begin{cases} 1, & \text{jei } j\text{-asis stulpelis priklauso sprendiniui,} \\ 0, & \text{kitais atvejais;} \end{cases}$$

$$y_j \in \{0, 1\}, \quad j = \overline{1, n}$$

Čia matrica A – „teritorijos žemėlapis“, kur eilutės atitinka teritorijos tinklelio elementus, o stulpeliai atitinka galimas siųstuvų dislokacijos vietas. a_{ij} parodo, ar i-tąjį teritorijos tinklelio elementą gali aptarnauti j-sis siųstuvas. Taigi, matrica A vienareikšmiškai nusako teritorijos žemėlapį ir siųstuvų aptarnaujamas teritorijas.

Apribojimų sistema garantuoja, kad absoliučiai visa teritorija yra padengiama. Jei turime teritoriją, kurioje ryšio poreikis yra netolygus, tai keisdami apribojimų sistemą nesunkiai algoritmą modifikuosime.

Pavyzdys. Tarkim k-tojoj teritorijos dalyje vartotojų koncentracija labai didelė ir vienas siųstuvas negali aptarnauti.

Tuomet įtrauksime sąlygą: $\sum_{j=1}^{n} a_{kj} \cdot y_j \geqslant 2$.

Jei k-tojoj teritorijos dalyje yra negyvenamos teritorijos, tai įtrauksime sąlygą: $\sum_{j=1}^{n} a_{kj} \cdot y_j \geqslant 0$.

Taigi matome, kad tokia uždavinio matematinė formuluotė yra ekvivalenti problemos formuluotei.

Bendras „Skruzdžių kolonijos“ algoritmo pseudo-kodas

```
„Skruzdžių kolonijos“ algoritmas {
    Nustatyti pradinius parametrus
    Kol neįvykdytas numatytas iteracijų skaičius Kartoti {
        Nuo k=1 Iki „skruzdėlių“ skaičius Kartoti {
            Kol sprendinys nesukonstruotas Kartoti {
                Įtraukti stulpelį į sprendinį
```

Įtraukti stulpelį į „tabu“ sąrašą
}
Taikyti Lokalios paieškos procedūrą
}
Atnaujinti geriausią sprendinį
Atnaujinti „feromonus“ kiekvienam stulpeliui
}
}

čia:

- Pradinių parametrų parinkimas apima „skruzdėlių“ skaičiaus nustatymą, bei svarbiausia, taisyklės, pagal kurią sprendžiama kurį stulpelį traukti į sprendinį, o kurio ne.
- Algoritmo stabdymo strategijų gali būti įvairių: fiksuojamas iteracijų skaičius, kartojama tol kol geriausias sprendinys nebegerėja ir pan.
- Ar stulpelis bus traukiamas į sprendinį yra nustatoma atsitiktinių skaičių generatoriaus pagalba. Tikimybė stulpeliui patekti į sprendinį pagrinde priklauso nuo dviejų veiksnių:

$$p = p(\text{euristinė informacija, „feromonų“ informacija}).$$

- „Feromonų“ atnaujinimas, tai procesas kurio metu stulpeliam priskiriama informacija apie jų „populiarumą“, renkant juos į sprendinius.

Tikimybės patekti į sprendinį sudarymo pavyzdys

$$p_i^k(t) = \frac{[\tau_i(t)] \cdot [\eta_i]^\beta}{\sum_{l \in N^k} [\tau_l(t)] \cdot [\eta_l]^\beta}, \quad \text{jei } i \in N^k,$$

čia

- k – „skruzdėlės“ numeris,
- i – nagrinėjamo stulpelio numeris,
- t – iteracijos numeris,
- τ_j – „feromonų“ informacija j-tąjam stulpeliui,
- η_j – euristinė informacija j-tąjam stulpeliui, pvz., $\eta_j = \frac{1}{b_j}$,
- N_k – leistinų k-tąjai „skruzdėlei“ stulpelių aibė, kurie padengia bent po vieną dar nepadengtą matricos A eilutę.

***j*-tojo stulpelio feromonų atnaujinimo taisyklės pavyzdys**

$$\tau_j = (1 - \rho) \cdot \tau_j + \rho \cdot \Delta\tau_j,$$

čia:

- ρ – „išgaravimo“ koeficientas (nurodomas kaip algoritmo parametras),
- $\Delta\tau$ – j-tojo stulpelio dažnis „skruzdėlių“ suformuotose sprendiniuose.

Duomenų ir rezultatų pavyzdžiai

Nagrinėkime teritoriją, nusakomą tokia matricos A išraiška:

$$A = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Siųstuvų kainų vektorius b turi išraišką:

$$b = (100 \;\; 300 \;\; 300 \;\; 1000 \;\; 100 \;\; 150 \;\; 300 \;\; 1000 \;\; 300 \;\; 300 \;\; 100 \;\; 300),$$

čia b_j – j-tojo siųstuvo kaštai 100 tūkst. Lt.

Algoritmo parametrų reikšmės:

- iteracijų skaičius = 20,
- „skruzdėlių“ skaičius = 12,
- $\beta = 0,5$,
- $\rho = 0,4$.

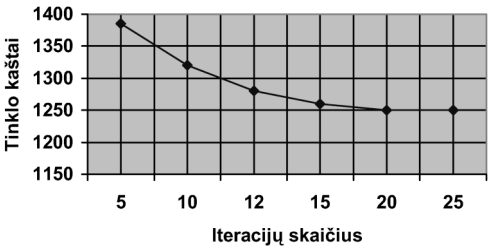
Esant šioms algoritmo parametrų reikšmėms gauname šiuos rezultatus:

Siųstuvus reikia statyti: **1**-ame, **5**-tame, **6**-tame, **7**-tame, **9**-tame, bei **10**-tame teritorijos tinklelio elementuose. Šitaip įrengto tinklo kaina bus 1250000Lt.

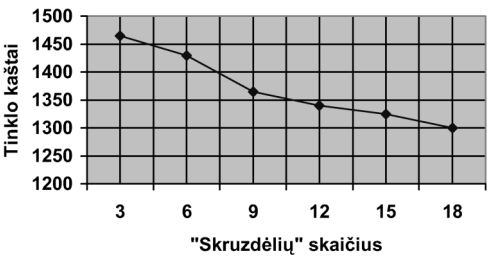
Iš duotos matricos A pašalinę visus likusius stulpelius gauname sprendinį reprezentuojančią matricą A^*:

$$A^* = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Kaip matome sprendinys tenkina ir apribojimų sistemą, t.y. kiekvienoje eilutėje yra bent vienas vienetas. Tai reiškia, kad bet kurioje duotos teritorijos vietoje įmanomas ryšys bent su vienu siųstuvu.



1 pav. Tinklo diegimo kaštų priklausomybės nuo algoritmo iteracijų skaičiaus grafikas.



2 pav. Tinklo diegimo kaštų priklausomybės nuo naudojamų „skruzdėlių“ skaičiaus grafikas.

Pateikiame vidutinės kaštų reikšmės priklausomybės nuo iteracijų skaičiaus grafiką (1 pav.). Kiekvienai iteracijos reikšmei kaštų reikšmė buvo apskaičiuota po 10 kartų, kai “skruzdėlių“ skaičius = 12, $\beta = 0,5$ ir $\rho = 0,4$.

Pateikiame vidutinės kaštų reikšmės priklausomybės nuo „skruzdėlių“ skaičiaus grafiką (2 pav.). Kiekvienai „skruzdėlių“ skaičiaus reikšmei kaštų reikšmė buvo apskaičiuota po 10 kartų, kai iteracijų skaičius = 10, $\beta = 0,5$ ir $\rho = 0,4$.

Iš grafikų matyti, kad algoritmas yra jautrus pradiniam parametrų nustatymui. Taigi labai svarbu tinkamai juos parinkti, atsižvelgiant į teritorijos žemėlapio dydį ir sudėtingumą.

Literatūra

1. M. Dorigo, T. Stutzle, *The Ant Colony Optimization Metaheuristic*: *Algorithms, Applications, and Advances*.
2. L. Lessing, I. Dumitrescu, T. Stutzle, *A Comparison Between ACO Algorithms for the Set Covering Problem*.
3. B. Crawford, C. Castro, *ACO with Lookahead Procedures for Solving Set Partitioning and Covering Problems*.

SUMMARY

S. Lazaravičius, N. Listopadskis. The analysis of algorithm for transmitters locating problem

The analysis of ACO algorithm for transmitters locating problem is presented in this paper.

Keywords: combinatorial optimization, metaheuristic, ant colony optimization, ACO, transmitterslocating problem, set covering problem.