Мс Марија ЈОРДАНОВИЋ
Учитељски факултет у Врању
Универзитет у Нишу

Мс Никола МИЛЕНКОВИЋ
„CubeSoft“ у Београду

УДК 005.31
-стручни рад-

РЕШАВАЊЕ ПРОБЛЕМА ТРГОВАЧКОГ ПУТНИКА МЕТОДОМ СИМУЛИРАНОГ КАЉЕЊА

Сажетак: *Математичка оптимизација или математичко програмирање је избор најбољег елемента из неког скупа алтернатива, у складу са одређеним критеријумима. Математичком оптимизацијом се могу решити изузетно комплексни проблеми, који су постали веома популарни са развојем рачунарства. Уколико није могуће одредити тачно решење проблема помоћу класичних метода, приступа се хеуристичком решавању проблема, односно одређивању приближног решења неком од метода хеуристике. Једну врсту хеуристика чине метахеуристике. Метахеуристике представљају веома користан приступ оптимизационим проблемима јер, проналажењем великог скупа могућих решења, могу довести до доброг решења са далеко мањим рачунарским „напором“ него алгоритми, итеративне методе и обичне хеуристике. Симулирано каљење (Simulated annealing) је врста метахеуристике,инспирисана процесом каљења метала, који подразумева грејање и контролисано хлађење метала ради измене физичких особина услед промена које настају у унутрашњој структури материјала. Алгоритам симулираног каљења погодан је за решавање НП-тешких проблема, као што је проблем трговачког путника (Travelling Salesman Problem – TSP). У овом раду ће бити приказан псеудокод за решавање проблема трговачког путника помоћу алгоритма симулираног каљења.*

Кључне речи: *математичка оптимизација, метахеуристике, симулирано каљење, проблем трговачког путника.*

Оптимизација

Математичка оптимизација или математичко програмирање је избор најбољег елемента из неког скупа алтернатива, у складу са одређеним критеријумима. Методе оптимизације омогућују налажење најбољих, односно оптималних решења различитих врста проблема. Решења проблема морају задовољити одговарајуће критеријуме којима се одређује њихов квалитет, како бисмо били у могућности да, из скупа свих решења проблема, издвојимо оптимално решење. Из тог разлога, у математичком моделу постоји функција која сваком решењу придружује његову меру квалитета.

Поменута функција се назива функција циља или критеријум. У том случају, проблем оптимизације се своди на проналажење решења које задовољава одговарајућа ограничења и у коме функција циља достиже екстремну вредност. Стога, у оквиру оптимизације разликујемо максимизацију – тражење највеће вредности функције циља и минимизацију – тражење најмање вредности функције циља.

Да бисмо задали проблем математичког програмирања, неопходно је:

1) одабрати једну или више оптимизационих променљивих;
2) одабрати функцију циља;
3) формирати скуп ограничења.

Математичком оптимизацијом се могу решити изузетно комплексни проблеми, који су постали веома популарни са развојем рачунарства. Неки проблеми се решавају претраживањем целокупног простора решења (енгл. *brute-force*), што је омогућено повећањем брзине процесора, односно развојем рачунара. Један од таквих проблема је распоређивање нераспоређених студената у групе за предавања. Овај проблем се јавља након накнадног уписа студената на предмете. Нераспоређене студенте је потребно распоредити у групе тако да број чланова групе не буде већи од максималног броја студената, одређеног просторијом, као и да се обезбеди да студент нема преклапања са другим додељеним обавезама. Овакви проблеми се решавају тражењем оптималног решења, тј. помоћу метахеуристика.

Хеуристички приступ решавању проблема

Дефиниција. Алгоритме који проналазе задовољавајуће добра решења, али не нуде никакве гаранције да ће пронаћи оптимално решење и, притом, имају релативно ниску рачунску сложеност (полиномијалну сложеност) називамо хеуристичким методама или хеуристикама. [2]

Дакле, хеуристика је техника налажења приближног решења датог проблема у случају када је немогуће одредити тачно решење помоћу класичних метода. Циљ хеуристике је проналажење довољно доброг решења које не мора бити најбоље, већ може бити само апроксимација тачног решења. То не умањује значајност хеуристичких метода јер су такве методе најпогодније за решавање НП-тешких проблема. Формализацијом неких „здраворазумских“ правила, као и претходних знања и искустава експерата у решавању проблема, може се очекивати да добијена решења буду блиска оптималном.

Развој комбинаторне оптимизације, теорије алгоритама, као и рачунара, утиче на све већу примену хеуристичких метода за решавање комплексних проблема са којима се срећемо у свакодневном животу. Хеуристичке методе се примењују у следећим случајевима:

1) за решавање проблема који се не могу решити помоћу егзактних алгоритама. У питању су тзв. слабо структуирани проблеми код којих се не може извршити потпуна формализација и формирати прецизни математички модел. Такође, присутни су и елементи неизвесности, неодређености и субјективне процене;
2) за решавање проблема за које постоје егзактни алгоритми, али нису ефикасни за велике димензије проблема;
3) као допуна егзактним алгоритмима, нпр. за налажење почетног решења за проблем линеарног програмирања;
4) за тражење задовољавајућег решења проблема у случају ограниченог времена, простора или финансијских средстава.

Хеуристике се деле на:

1) конструктивне алгоритме (генеришу само једно допустиво решење);
2) методе локалног претраживања (генеришу низ допустивих решења, где је свако следеће решење добијено побољшањем претходног по неком локалном критеријуму);
3) методе декомпозиције (разбијају проблем на више проблема мањих димензија);
4) еволутивне методе (генеришу популацију допустивих решења у свакој итерацији где је свака следећа популација боља од претходне);
5) метахеуристике (опште методе новијег датума).

Метахеуристике

Дефиниција. Метахеуристика је алгоритамски оквир, независан од проблема који обезбеђује скуп смерница или стратегија за развој хеуристичких оптимизационих алгоритама. [3] То је врста хеуристике, намењена за проналажење, генерисање или бирање алгоритма делимичне претраге, који може обезбедити довољно добро решење за оптимизацију проблема, посебно оних са непотпуним информацијама или ограниченим капацитетом рачунара.

Ипак, може се рећи да метахеуристика генерално није алгоритам, тј. не чини је низ акција које је потребно следити једну за другом; пре би могла бити конзистентан скуп идеја, концепата и операција које се могу употребити за креирање хеуристичких оптимизационих алгоритама.

Метахеуристике представљају веома користан приступ оптимизационим проблемима јер проналажењем великог скупа могућих решења могу довести до доброг решења са далеко мањим „рачунарским напором“ него алгоритми, итеративне методе и обичне хеуристике. Ове методе не гарантују проналажење глобалног оптималног решења у некој класи проблема, али су веома погодне за различите проблеме јер могу да обезбеде неколико претпоставки оптимизације решења проблема.

Особине које карактеришу већину метахеуристика су:

– метахеуристике представљају стратегије које спроводе процес претраживања;
– њихов циљ је ефикасно истраживање простора за претрагу ради проналажења решења која су близу оптималног;
– технике које чине метахеуристичке алгоритме могу бити једноставне процедуре локалне претраге, али и сложени процеси учења;
– алгоритми метахеуристике су апроксимативни и обично су недетерминистички.

Постоји широк спектар различитих метахеуристика. Оне се могу класификовати на основу различитих критеријума. Једна од класификација извршена је на основу врсте стратегија за претрагу. Стога, једну врсту стратегија чине побољшани једноставни алгоритми локалне претраге. Ту спадају:

– симулирано каљење (Simulated annealing);
– табу претраживање (Tabu search);
– итеративна метода локалне претраге (Iterated local search);
– метода променљивих околина (Variable neighborhood search);
– процедура случајне, „похлепне“, адаптивне претраге (Greedy randomized adaptive search procedure).

Стратегије друге врсте садрже компоненту учења. То су:

– мрављи алгоритми (Ant colony optimization);
– алгоритми еволуцијског рачунања (Evolutionary computation);
– генетски алгоритми (Genetic algorithms).

Метахеуристике се, такође, могу поделити на алгоритме који раде над једним решењем (симулирано каљење, табу претраживање) и на алгоритме који раде са скуповима решења (алгоритми еволуцијског рачунања). Трећа подела метахеуристика је на хибридне и паралелне.

Симулирано каљење

Идеја за креирање алгоритма симулираног каљења потиче из рада који је објавио Н. Метрополис са сарадницима 1953. године. Алгоритам симулираног каљења првобитно је био инспирисан процесом каљења метала који подразумева грејање и контролисано хлађење метала ради измене физичких особина, услед промена које настају у унутрашњој структури материјала. Топлота утиче на атоме тако што их ослобађа од њиховог почетног положаја и допушта им да се слободно крећу ка вишим енергетским нивоима. Постепено хлађење омогућава повећање вероватноће да атоми пронађу распоред у којем ће имати мању унутрашњу енергију него пре почетка каљења. Мања унутрашња енергија атома уједно значи и већа стабилност и боље карактеристике састава. Са хлађењем, структура метала постаје фиксна, што проузрокује задржавање новодобијених својстава метала. Код каљења метала температура се мења како би се стимулисао процес загревања. У почетном стадијуму, температура је висока, а затим се постепено хлади.

Аналогно ради и алгоритам симулираног каљења. У току извршавања алгоритма, тренутно решење замењује се неким „блиским“, случајно одабраним решењем. Вероватноћа избора алтернативног решења зависи од тога колика је разлика у енергетским нивоима (односно у адекватности и квалитету) између тренутног и алтернативног решења. Та вероватноћа зависи и од глобалног параметра Т који означава температуру и постепено се смањује током рада алгоритма. Тренутно решење се замењује новим када параметар Т има велику вредност, независно од тога да ли је новоизабрано решење бољег или лошијег квалитета од тренутног. Када Т има нижу вредност, инсистира се на томе да ново решење буде увек боље од тренутног. Док је температура висока, може се чешће десити да алгоритам прихвати решења која су лошија од тренутног. Ово својство омогућава наставак претраге ако у раном извршавању пронађе локални оптимум. Како се температура смањује, смањује се и могућност прихватања лошијих решења.

Претрага се фокусира на подручје у коме постоји могућност за проналажење оптималног решења. Управо овај процес постепеног хлађења чини алгоритам симулираног каљења посебно ефикасним за проналажење локалног оптимума, када су у питању велики проблеми који садрже већи број локалних оптимума.

Дакле, алгоритам најпре врши проверу да ли је суседно решење (сусед) боље од тренутног. Ако јесте, алгоритам безусловно прихвата ново решење. У супротном, разматра се неколико фактора, а пре свега, да ли је разлика у квалитету између тренутног и новог решења велика и колико је висока тренутна температура нашег система. На високим температурама, систем прихвата лошија решења. Математички запис за претходну констатацију је:

$$\exp\left(\frac{\text{енергијаРешења} - \text{енергијаСуседа}}{\text{температура}}\right)$$

У суштини, што је мања промена у енергији (квалитету решења) и виша температура, већа је вероватноћа да ће алгоритам прихватити решење. Алгоритам симулираног каљења у својој основној имплементацији изгледа једноставно:

- најпре је потребно изабрати почетну температуру и случајно изабрати почетно решење;
- затим, алгоритам почиње да се одвија док не наиђе на услов за заустављање, обично док се систем прилично не охлади или док се не пронађе довољно добро решење;
- од тог тренутка, бирамо суседа мењајући наше тренутно решење;
- затим, одлучујемо да ли да се померамо ка суседном решењу;
- на крају, смањујемо температуру и настављамо петљу.

Псеудокод алгоритма симулираног каљења приказан је у наставку рада.

```
процедура Симулирано каљење(i_0, c_0)
i = i_0; // почетно решење
c = c_0; // почетна температура
Ci = C(i); // C – оптимизацијска функција
    понављај
        понављај
        j = суседно решење(i);
        Cj = C(j);
        ΔC = C(j) − C(i);
        прихвати = FALSE;
            ако је ΔC < 0
                онда прихвати = TRUE;
            иначе
                ако је exp(−ΔC/c) > рандом [0,1]
                    онда прихвати = TRUE;
            ако је прихвати = TRUE
                онда i = j;
                     Ci = Cj;
        до температурне равнотеже
            c = c · α;  // смањи температуру
    до замрзавања
крај
```

Псеудокод 1. Алгоритам симулираног каљења

Поступно каљење симулирамо функцијом *C*. Упоређивање се врши са неком случајном вредношћу, што уводи меру случајности у цео поступак и тако конкретне изборе препушта вероватноћи.

Проблем трговачког путника представља одличан пример за примену описаног алгоритма.

Проблем трговачког путника

Метахеуристике се користе за комбинаторну оптимизацију, где се оптимално решење тражи у дискретном простору за претрагу. Проблем трговачког путника (*Travelling Salesman Problem – TSP*) један је од најпознатијих проблема који се решавају методама метахеуристика. Овај проблем припада класи НП-тешких проблема и може се формулисати на следећи начин:

– Трговачки путник креће из једног града, обилази $n-1$ градова и на крају се враћа у почетни град.
– Путник мора обићи свих n градова, при чему у сваки град може доћи само једном, а дужина његове кружне путање (или трошкови његовог путовања) мора бити минимална.

Дакле, потребно је минимизирати укупну удаљеност коју трговачки путник треба да пређе како би посетио сваки од n задатих градова тачно једном и вратио се у полазни град. То је тзв. симетрични проблем трговачког путника. Из ове интерпретације потиче и сам назив проблема. Најранији резултат у вези са проблемом трговачког путника сеже у 1759. годину, када је Ојлер објавио решење за проблем кретања скакача по шаховској табли (*Knight's Tour Problem*). Проблем је био формулисан на следећи начин: „Да ли постоји пут којим скакач може посетити свако поље шаховске табле тачно једанпут, полазећи с произвољног поља и крећући се према шаховским правилима.“

Решавање проблема трговачког путника је велики изазов за математичаре и програмере, делом због његове тежине, а делом због повезаности са занимљивим практичним и теоријским питањима. Данас је то референтни проблем за оцену квалитета разних алгоритама и оптимизацијских метода.

Овако постављен проблем може имати много решења, а с повећањем димензије проблема број могућих кружних путања расте брзином већом од експоненцијалне. Таквих путања има $(n-1)!$ што представља препреку егзактном тражењу минималног пута. Проблем трговачког путника може се интерпретирати на потпуном тежинском графу G са чворовима $1, 2, 3, \dots, n$ и са матрицом тежина грана $D = [d_{ij}]_{m \times n}$.

Дефиниција. Хамилтонов пут (Хамилтонова контура) јесте пут који пролази кроз све чворове графа тачно једанпут. Пут који се завршава у почетном чвору, а кроз све остале чворове пролази само једанпут, назива се затворен Хамилтонов пут. Граф који има Хамилтонову контуру назива се Хамилтоновим графом. [1]

С обзиром на то да је Хамилтонова путања пут који кроз сваки чвор графа пролази тачно једном, пут трговачког путника можемо посматрати као Хамилтонов пут, па се проблем трговачког путника своди на следећи проблем:

Проблем 1. Одредити најкраћи Хамилтонов пут у потпуном тежинском графу G.

Постоје различите подврсте проблема трговачког путника. Поред наведеног проблема, постоји и тзв. асиметрични проблем трговачког путника. У том случају, дужина бар једног пута има различите вредности, у зависности од смера кретања путника, а као модел се користи усмерени граф. Проблем трговачког путника се може и генерализовати: m трговачких путника креће из истог почетног града, обилази задати скуп градова и враћа се у почетни град; потребно је одредити обиласке за све трговачке путнике, тако да сваки град буде посећен само једном, а дужина пређених путева трговачких путника буде минимална.

Постоје и неке још компликованије интерпретације претходно поменутог проблема. Постоји неколико полазних градова и одређени број трговачких путника креће из сваког од њих. Након завршетка обиласка, трговачки путници се враћају или сваки у свој полазни град, или у било који од полазних градова, уз услов да на крају у сваком полазном граду буде исти број путника као и на почетку. Могу се увести и друге рестрикције, као што су ограничен број градова које трговачки путник може обићи, минимална или максимална удаљеност коју неки од њих морају прећи итд.

У наставку рада биће приказано решење симетричног проблема трговачког путника методом симулираног каљења. Параметри који се користе у програму су $i = i_0$ као почетно решење, $c = c_0$ као почетна температура и C као функција чији минимум тражимо. Почетно решење приказујемо као низ бројева $i_0 = (1, 2, 3, \dots, N)$ који представља путовање градовима с лева на десно. Суседно решење се формира бирањем два случајна броја из интервала $[1, N]$ и заменом редоследа чланова подниза чији су гранични елементи изабрани бројеви. На пример, ако је почетно решење $i_0 = (7, 1, 3, 2, 12, 6, 4, 8, 9, \dots, N)$, а изабрани бројеви су 4 и 8, суседно решење може бити $i_0 = (7, 1, 3, 8, 4, 6, 12, 2, 9, \dots, N)$.

Унутрашња и спољашња петља су задате параметрима KTL (коефицијент температурне равнотеже) који може имати вредности од 0,1 до 0,5 и S (број понављања спољашње петље) који обично износи 100. Вероватноћа прихватања лошијег решења означена је са $p_0(0{,}7 - 0{,}8)$, а коефицијент смањивања температуре са $\alpha(0{,}5 - 0{,}99)$. Потребно је израчунати матрицу $D[N, N]$ која представља међусобне удаљености свих градова. У наставку рада дат је псеудокод решења проблема трговачког путника за имплементацију у било ком програмском језику.

```
процедура TSP(N, S, p0, α, KTL)
    // задате су координате N градова
    израчунај_матрицу D[N, N];
    пут = почетно_решење; // (1,2,3 …, N)
    дпут = дужина (пут);
    // одређивање просечне температуре
    бројач = 0; просек = 0;
        за i = 1 до 100 ради {
            пут2 = случајно_суседно_решење(пут);
            дпут2 = дужина(пут2);
                ако дпут2 > дпут
                онда просек += (дпут2 - дпут);
                    бројач ++;
        }
    c = (просек/бројач) / (ln(1/p0));
        за i = 1 до S ради {
            за j = 1 до (KTL * N * N) ради {
                пут2 = случајно_суседно_решење(пут);
                дпут2 = дужина(пут2);
                промена = дпут2 - дпут;
                прихвати = FALSE;
                ако промена < 0
                    онда прихвати = TRUE;
                иначе
                ако exp(−промена/c) > рандом[0, 1]
                    онда прихвати = TRUE;
                ако прихвати = TRUE
                    онда пут = пут2;
                        дпут = дпут2;
            } // унутрашња
            c = c · α; // смањујемо температуру
        }
    крај
```

Псеудокод 2. Решење проблема трговачког путника методом симулираног каљења

Литература

1. Glover, F., Kochenberger, G. A. (2003). *Handbook of Metaheuristics.* Kluwer Academic Press
2. Reinelt, G. (1994). *The Traveling Salesman: Computational Solutions for TSP Applications (Lecture Notes in Computer Science)*. New York: Springer
3. Sorensen, K. (2012). Metaheuristics - the Metaphor Exposed, *International Transactions in Operational Research* doi: 10.1111/ itor.12001
4. Čupić, M. (2012). *Prirodom inspirirani optimizacijski algoritmi. Metaheuristike*, преузето 06. 10. 2013. са WWW: http://java.zemris.fer.hr/nastava/pioa/knjiga-0.1.2012-08-28.pdf
5. Станимировић, З. *Математичко програмирање и оптимизација*, преузето 29. 09. 2013. са WWW: http://www.math.rs/p/files/16-05V__MPIO_ (problem_trgovackog_putnika,_ k_trgovackih_putnika _i_problem_ranca).pdf

Ms Marija Jordanovic
Ms Nikola Milenkovic

SOLVING THE TRAVELING SALESMAN PROBLEM USING SIMULATED ANNEALING

Summary: *Mathematical optimization or mathematical programming is the choice of the best elements from a set of alternatives, according to certain criteria. Extremely complex problems that have become very popular with the development of computer science can be solved by using Mathematical optimization.If it is not possible to determine the exact solution of the problem using traditional methods, we can approach to problem solving by using heuristic methods. Thereby we can determine an approximate solution. One type of the heuristics are metaheuristics. The metaheuristics are very useful approach to the optimization problems, because finding a large set of possible solutions can lead to a good solution with less computing „effort“ than algorithms, iterative methods and simple heuristics. Simulated Annealing is a type of metaheuristics inspired by the process of the metals annealing that involves heating and controlled cooling of metals to modify the physical properties due to changes occurring in the internal structure of materials.Simulated annealing is a suitable algorithm for solving NP - hard problems such as the Traveling Salesman Problem. This paper presents the pseudocode for solving the Traveling Salesman Problem using the simulated annealing algorithm.*

Key words: *Mathematical Optimization, Metaheuristics, Simulated Annealing, the Traveling Salesman Problem.*

Примљено: 29. 10. 2013. године.
Одобрено за штампу: 28. 11. 2013. године.