KORIŠĆENJE GENETSKOG ALGORITMA ZA REŠAVANJE MINIMUM VERTEX COVER PROBLEMA

Cerović Mina 9/2019

MATEMATIČKI FAKULTET 2023.

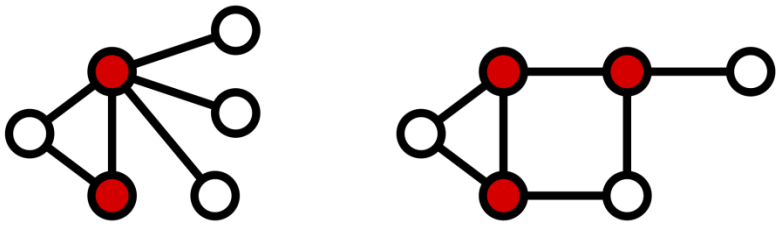
MINIMUM VERTEX COVER PROBLEM

Minimum vertex cover problem je dobro poznat kombinatorni optimizacioni problem, sa primenama u umrežavanju i pravljenju rasporeda. On je NP težak, pa se ne može rešiti polinomijalnim algoritmom, osim ako je P=NP. Štaviše, teško ga je aproksimirati – ne može se aproksimirati sa faktorom manjim od 2.

Formulisan kao problem odlučivanja, ovaj problem je NP kompletan. 1972. godine Karp je predstavio listu 21 NP kompletnog problema [1], a odlučivanje da li graf G poseduje vertex cover veličine najviše k bio je jedan od njih.

Kao optimizacioni problem, minimum vertex cover je NP težak i formulisan je na sledeći način:

Dat je neusmereni graf G(V, E), gde je V skup čvorova, a E skup grana. Naći najmanji podskup V′⊆V takav da je svaka grana iz E incidentna sa najmanje jednim čvorom iz V′.



U nastavku je navedeno nekoliko načina za rešavanje minimum vertex cover problema.

Brute force

Ako bismo želeli da proverimo za svaki mogući podskup čvorova u grafu da li je minimum vertex cover, morali bismo da ispitamo 2^n skupova. Kako se broj čvorova (n) povećava, postaje nemoguće brzo naći min. vertex cover brute force tehnikom.

2-Approximation Algoritam

Postoji vise algoritama koji aproksimiraju rešenje minimum vertex cover problema, a jedan od njih je sledeći:

Za graf G=(V, E) algoritam dodaje čvorove u inicijalno prazan skup S, što će biti vertex cover kada algoritam završi sa radom.

Biramo granu e = {v_i, v_j}. Dodajemo v_i i v_j u S. Pošto je svaka grana incidentna sa v_i i v_j sada pokrivena sa S, uklanjamo te grane kao i dva čvora iz G. Ponavljamo proces dok E nije prazno.

Ovaj algoritam predstavljen je u knjizi Introduction to Algorithms [2], gde je dat i dokaz da je vertex cover koji on proizvodi najviše dva puta veći od minimalnog.

Greedy Algoritam

Ovaj algoritam u svakom koraku bira čvor sa najvećim stepenom, tj. onaj koji pokriva najviše preostalih grana, i dodaje ga u vertex cover.

*Papadimitriou i Steiglitz [3] pokazali su da ovo nije dobra strategija. Oni su razmatrali graf koji se sastoji od 3 nivoa, od kojih prva dva imaju isti broj čvorova, dok treći ima 2 manje. Minimum vertex cover se sastoji od k+2 čvorova sa drugog nivoa, a greedy algoritam će izabrati 2*k+2 čvorova sa drugog i trećeg nivoa.*

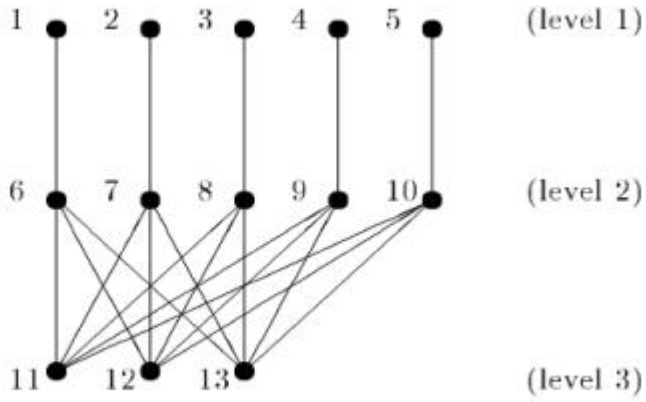


Figure 1: Construction of the regular graph after Papadimitriou and Steiglitz.

REŠAVANJE PROBLEMA KORIŠĆENJEM GENETSKOG ALGORITMA

Opšti rad algoritma

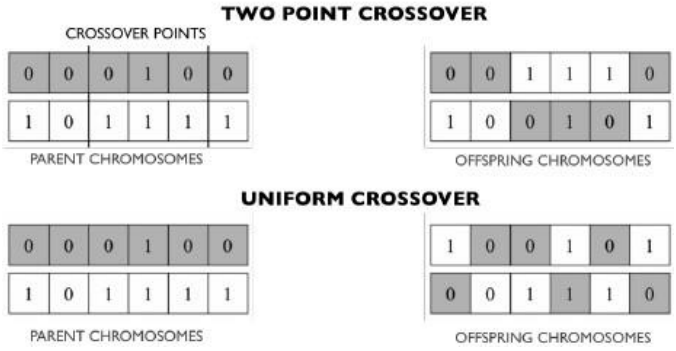
Genetski algoritam razvijen je u Americi 1970-ih godina. Njegovi ključni autori bili su J. Holland, K. DeJong, D. Goldberg. On imitira proces evolucije tako što, umesto organizama, jedinke u populaciji kodiraju rešenja nekog problema. Nakon nekog vremena, rešenja evoluiraju u smeru poboljšanja.

Genetski algoritam sastoji se od 3 glavna dela: selekcije, ukrštanja I mutacije.

***Selekcija** je izbor jedinki iz populacije koje će učestvovati u ukrštanju, odnosno ostaviti potomstvo. Veća šansa se daje boljim jedinkama, kako bi se dobri geni preneli u narednu generaciju.*

U implementiranom algoritmu korišćena je turnirska selekcija – od n nasumično izabranih jedinki iz populacije bira se najbolja (ona sa najboljim fitnessom).

***Ukrštanje** dve jedinke proizvodi jednu ili više jedinki koje se dodaju u populaciju. Za rešavanje ovog problema korišćeno je ravnomerno i dvopoziciono ukrštanje.*



***Mutacija** je mala izmena gena. Ona je eksplorativna, tj. proširuje prostor pretrage. Implementirana je tako da se svaki bit u rešenju invertuje sa nekom verovatnoćom.*

*Još jedna važna stavka kod genetskg algoritma je **selekcija preživelih**. Ona govori na koji način se smenjuju generacije. U implementiranom algoritmu korišćen je tzv. generacijksi model, zajedno sa elitizmom. Dakle, pre generisanja nove generacije kopira se 5 najboljih jedinki, a zatim se ostatak populacije popunjava novim jedinkama.*

Broj generacija koje će algoritam izgenerisati, kao i broj jedinki (n) u populaciji je zadat unapred. Na samom početku se inicijalizuje početna populacija, koja sadrži n-1 nasumičnu jedinku i jednu koja predstavlja rešenje dobijeno greedy algoritmom. Na taj način je obezbedjeno da konačno rešenje genetskog algoritma bude dopustivo i da ne bude gore od rešenja koje daje greedy algoritam.

Način kodiranja rešenja

Za dati graf G=(V, E) možemo urediti čvorove označavajući ih sa 0,1,…,n-1 gde je n=|V|, a zatim ćemo koristiti binarno kodiranje.

Rešenje ćemo predstaviti kao niz True i False vrednosti dužine n, gde i-ta pozicija odgovara i-tom čvoru u grafu. Ako je na i-tom mestu u nizu True, onda je i-ti čvor uključen u rešenje.

Fitness funkcija (funkcija cilja)

Želimo da definišemo fitness funkciju tako da postoji kazna ako neka grana nije incidentna ni sa jednim čvorom iz rešenja. Ona takodje mora biti povezana sa brojem čvorova u rešenju.

$$f(\vec{x}) = \sum_{i=1}^{n} \left(x_i + n \cdot (1 - x_i) \cdot \sum_{j=1}^{n} (1 - x_j) \cdot e_{ij} \right)$$

Vidimo da je prvi deo funkcije $\sum x_i$, sto je broj čvorova uključenih u rešenje.

Drugi deo možemo napisati kao $n \cdot \sum (1 - xi) \cdot \sum (1 - xj) \cdot eij$.

Vidimo da ako e={i, j} nije grana u G onda je $e_{ij} = 0$ i ovaj proizvod je jednak 0. (ne želimo da dodeljujemo kaznu za grane koje ne postoje)

Ako e={i, j} jeste grana u G, onda je $e_{ij}=1$. Ako rešenje uključuje jedan od čvorova i i j, x_i ili x_j će biti 1, pa će drugi deo funkcije biti 0.

Ako nijedna krajnja tačka neke grane nije uključena u rešenje, onda je $x_i = x_j = 0$. Tada kaznu množimo sa n jer je to dužina najvećeg mogućeg vertex cover-a u grafu.

Funkciju cilja definisaćemo kao –f(x), a zatim ćemo tražiti njen maksimum.

REZULTATI

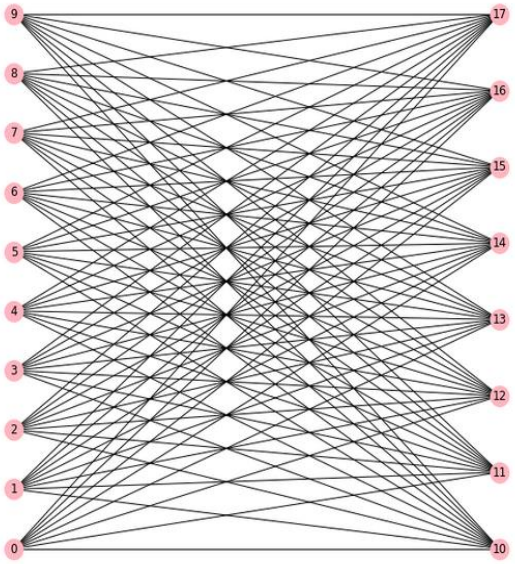
MALI GRAFOVI

Rad algoritma testiran je na grafovima sa do 7 čvorova, dobijenim pomoću biblioteke networkx. Za 500 takvih grafova pokrenut je brute-force algoritam, kako bi se uporedilo dobijeno rešenje. Genetski algoritam je imao tačnost 100% na ovom skupu, dok je greedy heuristika imala 96.2%.

KOMPLETAN BIPARTITAN GRAF

Algoritam je testiran na kompletnom bipartitnom grafu.

Jasno je da će na ovakvim grafovima genetski algoritam uvek naći najbolje rešenje, zahvaljujući rešenju greedy algoritma koje je uključeno u početnu populaciju i elitizmu koji će to rešenje propagirati u naredne generacije.



CIKLIČNI GRAF

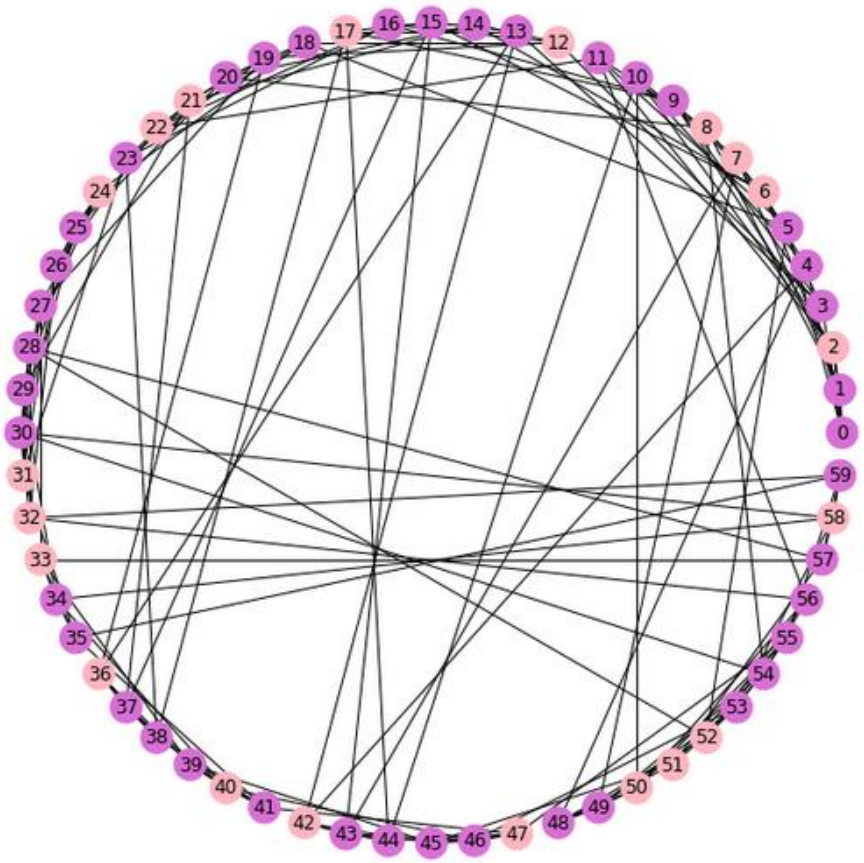
Dužina minimum vertex cover-a kod cikličnog grafa sa n čvorova iznosi n/2 ako je n parno, a (n+1)/2 ako je n neparno.

I na ovakvim grafovima genetski algoritam će uvek naći najbolje rešenje, zahvaljujući rešenju greedy algoritma.

THE BERGE GRAPH

Neka je G graf dodekaedra i H=K3 graf trougla. Berge graf GxH se definiše kao graf čiji je skup čvorova V(G)xV(H), sa granom koja spaja čvor (u1, v1) sa čvorom (u2, v2) akko je u1=u2 i {v1, v2} je grana u H, ili je v1=v2 i {u1, u2} je grana u G. [4]

Ovaj graf ima 60 čvorova, a njegov minimum vertex cover se sastoji od 40 čvorova. Implementirani genetski algoritam uvek nalazi minimalno rešenje.

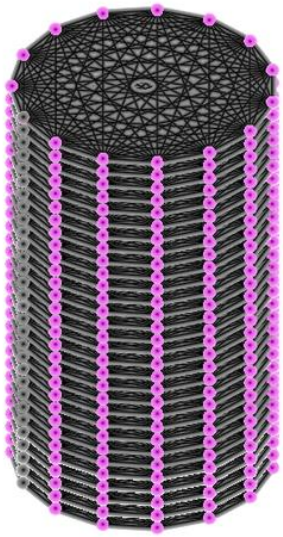


The Berge graph with a minimum vertex cover (n = 60, k = 40).

THE WITZEL GRAPH

Graf sa 450 čvorova koji se konstruiše na sledeći način: Konstruisati 30 medjusobno nepovezanih kompletnih grafova sa po 15 čvorova, a zatim spojiti slučajne parove takvih grafova nasumičnim granama. Promešati oznake čvorova. Takav graf će imati minimum vertex cover sa najviše 420 čvorova (svi osim po jednog iz svakog početnog grafa). [5]

Koristeći dvopoziciono ukrštanje, veličinu populacije 150 i stopu mutacije 0.007, implementirani genetski algoritam našao je vertex cover dužine 420.



The Witzel Graph

THE DIMACS BENCHMARK SET

Dimacs benchmark skup grafova uzet je iz "Second DIMACS Implementation Challenge" [7] *, takmičenja čiji je fokus bio na maksimalnoj kliki, bojenju grafa i zadovoljivosti. Ti grafovi se od tada koriste kao referentna tačka za nove algoritme.*

Referentni skup obuhvata 80 problema iz raznih domena. Na primer, C-fat porodica grafova je motivisana dijagnostikom kvara, johnson i hamming grafovi teorijom kodiranja, keller grupa zasniva se na Kelerovoj pretpostavci o popločavanju pomoću hiperkocki, dok MANN grafovi potiču iz Štajnerovog trostrukog problema.

Implementirani genetski algoritam pokrenut je na nekoliko grafova iz ovog skupa po 5 puta sa različitim stopama mutacije. Grafovi su preuzeti sa interneta u DIMACS formatu [10]. Rezultati su uporedjeni sa rešenjima dobijenim algoritmima NOVCA[8] i COVER[9]. Rezultati se nalaze u narednoj tabeli:

graf	Broj čvorova	Minimum vertex cover	Genetski algoritam	NOVCA	COVER
brock200-1	200	179	182	181	179
brock800-4	800	774	787	782	775
c-fat200-5	200	142	142	142	142
c-fat500-10	500	374	374	374	374
hamming10-2	1024	512	512	512	512
hamming10-4	1024	984	992	988	986
keller4	171	160	163	164	160
keller5	776	749	758	761	749
MANN-a27	378	252	253	253	252
san200-0-7-1	200	170	184	183	170
sanr-400-0-7	400	379	383	382	380

ZAKLJUČAK

Za rešavanje minimum vertex cover problema posmatrane su metode brute force, greedy i genetski algoritam. Iako je najtačniji, brute force algoritam se ne može koristiti za veće instance problema. Zato postoji potreba za razvijanjem dobrih metaheuristika za njihovo rešavanje.

Implementirani genetski algoritam se pokazao veoma dobro na malim grafovima i prevazišao je probleme koje je imao greedy algoritam. Ipak, na velikim instancama iz Dimacs benchmark skupa grafova, on radi slično kao i greedy algoritam.

Kako bi se poboljšao rad genetskog algoritma u ovim slučajevima moguće ga je kombinovati sa drugim metaheuristikama, tako što nakon svake generacije unapredimo najbolju jedinku nekim drugim algoritmom. Takodje, treba razmisliti o inicijalizaciji početne populacije. Ona pored rešenja greedy algoritma može sadržati još neko rešenje dobijeno na sličan način.

Ideja koja može biti razmatrana radi bržeg izvršavanja algoritma u nekim slučajevima je provera da li je rešenje problema trivijalno, tj. da li je graf bipartitan, ciklični i sl.

LITERATURA

[1] 1972.Richard Karp: "Reducibility Among Combinatorial Problems".

[2] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 3rd edition. The MIT Press, 2009.

[3] K. Steiglitz, Christos H. Papadimitriou, Combinatorial Optimization: Algorithms and Complexity. Dover Publications Inc., 1982.

[4] C. Berge, Graphes et Hypergraphes, Dunod, 1970.

[5] Klaus D. Witzel, Personal Communication, 2006.

[6] Ashay Dharwadker : The Vertex Cover Algorithm, Institute of Mathematics H-501 Palam Vihar District Gurgaon Haryana 122017 India

[7] Johnson, D.S., Trick, M.A., eds.: Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge. Volume 26 of DIMACS Series. American Mathematical Society (1996)

[8] Sanjaya Gajurel, Roger Bielefeld: A Simple NOVCA: Near Optimal Vertex Cover Algorithm , International Conference on Computational Science, ICCS 2012

[9] S. Richter, M. Helmert, and C. Gretton: A Stochastic Local Search Approach to Vertex Cover, In Proceedings of the 30th German Conference of Artificial Intelligence (KI), 2007, pp 412-426.

[10] http://archive.dimacs.rutgers.edu/pub/challenge/graph/benchmarks/clique/