Minimum maximal matching

Павле Савић

Математички факултет

Увод

Matching неусмереног графа $(G = V, E)$ јесте његов подграф за који важи да никоје две гране нису коинцидентне, односно важи $degV \leq 1 \; \forall \; V \in G$, при чему се за чворове чији је степен 1 каже да су повезани. Maximal Matching је matching такав да ниједна грана не може бити додата а да остане задовољен претходно наведени услов. Minimum maximal matching комбинаторни проблем, дакле, представља одређивање управо оваквог подграфа минималне кардиналости скупа грана (или минималне укупне тежине грана у тежинском случају) за произвољни задати неусмерени граф. Овај проблем деценијама је предмет обимног изучавања у контексту теорије графова. Раних 1980-тих година Yannakakis и Gavril показали су да се ради о NP-тешком проблему у општем случају. У њиховом раду такође је показана еквивалентност овог проблема и проблема Minimum Edge Dominating Set, где је циљ пронаћи минималну кардиналност подскупа грана F таквог да је свака грана у графу G суседна некој грани из F. Сваки maximal matching већ јесте и edge dominating set, и било који edge dominating set може се лако трансформисати у maximal matching мање или једнаке кардиналости. Ова еквиваленција не важи за пондерисане варијанте проблема. Највећи део постојеће литературе разматра полиномијалне егзактне или апроксимативне алгоримте за неке класе графова. Овај рад се бави најопштијом варијантом проблема и применом различитих варијанти бинарне оптимизације ројем честица (BPSO), као и поређењем експериментално добијених резултата са резултатима brute-force алгоритма, 2-апроксимативног похлепног алгоритма као и резултатима алгоритма из литературе (Integer Programming Formulations for the Minimum Weighted Maximal Matching Problem - аутора Z. Caner Taskin и Tinaz Ekim) заснованог на целобројном програмирању.

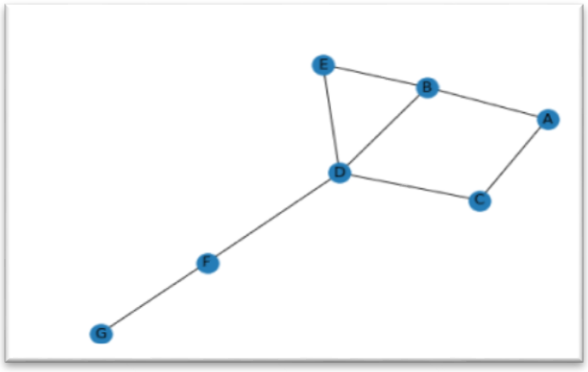
Укратко о brute-force и апроксимацији

У експоненцијалном егзактном алгоритму одређује се партитивни скуп скупа грана (скуп свих подскупова). Затим се редом пролази кроз подскупове почевши од оних најмање кардиналости, алгоритам се прекида чим се пронађе неки који испуњава услове за maximal matching (врши се провера да ли је неки чвор искоришћен у већем броју грана - у том случају није matching, и провера да ли за нека два чвора која нису повезана, у полазном графу постоји грана инцидентна са оба чвора - у том случају није maximal јер би та грана могла бити додата). Ради побољшања временске и просторне сложености (свакако не асимптотски) искоришћен је концепт генератора који су on-demand (lazy), односно подскупови ће се генерисати како итерирамо кроз партитивни скуп, а не унапред. До 2 -апроксимативног алгоритма (гарантује да решење које врати није више од два пута лошије од оптималног у контексту ф-је циља, односно кардиналости грана у нашем случају) долази се тривијално. Ово је последица чињенице да у графу не могу постојати два maximal matching-а таква да је један више од два пута већи од другог у поменутом контексту. Дакле, довољно је конструисати било какав maximal matching. У анализи је коришћена библиотечка ф-ја која налази произвољан maximal matching (принцип рада: иницијализује скуп грана на празан , а потом рекурзивно додаје грану инцидентну са два неповезана чвора и брише та два чвора из листе неповезаних). Развијање бољих апроксимативних алгоритама за произвољан граф показало се као изазовно, Gotthilf, Lewenstein и Rainschmidt дошли су до $2 - c * \log n/n$ апроксимације. За поједине типове графова постоје бољи апроксимативни алгоритми.

Важно је напоменути да је класа Graph имплементирана тако да чува матрицу суседства (биће симетрична јер радимо са неусмереним графовима), игнорисаће

вишеструка додавања исте гране у граф јер нам по дефиницији проблема овакве гране не могу бити од интереса, петље могу постојати у графу али никада неће бити разматране као потенцијалне гране у matching-у.

	A	B	C	D	E	F	G
A	0	1	1	0	0	0	0
B	1	0	0	1	1	0	0
C	1	0	0	1	0	0	0
D	0	1	1	0	1	1	0
E	0	1	0	1	0	0	0
F	0	0	0	1	0	0	1
G	0	0	0	0	0	1	0



Бинарна оптимизација појем честица (BPSO)

Ову методу увели су Kennedy и Eberhart са циљем да омогуће Particle swarm optimization алгоритму, чији су такође аутори, да ради у бинарном простору проблема - честа примена постала је feature selection као алтернатива другим методама. Главна идеја алгоритма је да се брзина (velocity) схвата као вероватноћа да ће бит позиције узети вредност 1 или 0. Ажурирање брзине остаје непромењено у односу на основни алгоритам, док је ажурирање позиције редефинисано тако да не узима у обзир претходну позицију већ само тренутну брзину, што у комбинацији са додатним псеудослучајним фактором (rand() из [0.0, 1.0], униформна расподела) уз случајне векторе R1 и R2 који постоје и у основном PSO може у неким случајевима угрозити конвергенцију алгоритма о чему ће више речи бити у наставку. S јесте сигмоидна функција која трансформише брзину у вероватноћу.

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1R_1(p_{\text{best},i,j} - x_{i,j}(t)) + c_2R_2(g_{\text{best},i,j} - x_{i,j}(t))$$

$$x_{i,j}(t+1) = \begin{cases} 0 & \text{if rand}() \geqslant S(v_{i,j}(t+1)) \\ 1 & \text{if rand}() < S(v_{i,j}(t+1)) \end{cases} \qquad S(v_{i,j}(t+1)) = \frac{1}{1 + e^{-v_{i,j}(t+1)}}$$

Треба напоменути да је BPSO подложан сатурацији сигмоидне функције, што се дешава када су вредности брзине превелике или премале. У таквим случајевима, вероватноћа промене вредности бита се приближава нули чиме се угрожава експлорација. За вредност брзине 0 сигмоидна ф-ја враћа 0.5, дакле 50% шансе да ће се бит инвертовати. Овај проблем решаваћемо velocity clamping поступком, односно поставићемо $[-V_{max}, V_{max}]$ за интервал могућих вредности брзине. Морамо бити пажљиви приликом постављања овог параметра, премала вредност може условити спору конвергенцију.

Као и у основној формулацији параметри c1 и c2 представљају редом утицај когнитивне и социјалне компоненте (личне и глобалне најбоље позиције) и одређују се емпиријски, углавном је пракса да буду једнаки. Такође параметар w је непромењеног значења, представља инерцију, односно утицај претходне брзине. У раду Clerc M, Kenney J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space показано је да се добро понашање алгоритма може обезбедити тако што се ове константе учине међусобно зависним увођењем посредног параметра φ из препорученог интервала [2.01, 2.4], при чему:

$$\begin{cases} w = \dfrac{1}{\varphi - 1 + \sqrt{\varphi^2 - 2\varphi}} \\ c_1 = c_2 = \varphi w \end{cases}$$

Кодирање проблема у BPSO контекст

Свака инстанца Particle класе у посматраном року садржи референцу на Graph објекат чији MMM желимо да одредимо. Graph објекти имају имплементирану методу која одређује скуп (листу) свих грана на основу матрице повезаности. Приликом иницијализације позиције честице врши се пресликавање те колекције у бинарни вектор псеудослучајних вредности при чему вредност 1 означава да се та грана налази у почетном скупу грана те честице. Имплементирано је и декодирање бинарног вектора у листу грана које се користи у фитнес ф-ји (делу који се бави одржавањем коректности), као и за испис решења.

Честице такође имају референцу на фитнес ф-ју која израчунава вредност ф-је циља коју минимизујемо (број јединица у бинарном вектору). Експериментално је утврђено за овај проблем да додељивање пенала (позитивног фактора) у случају

недопустивог решења (позиције честице) доводи до велике експлорације простора али често и неуспешног проналажења било каквог допустивог решења у случају великих димензија проблема. Као боља опција показало се одржавање допустивости кроз итерације. Пре израчунавања вредности ф-је циља позиција честице се своди на maximal matching инвертовањем неких битова пратећи већ описану шему за проверу да ли је подскуп грана неког графа његов maximal matching. Ипак, основна варијанта, која је подразумевала да се (лексикографским) редом пролази кроз скуп грана и елиминишу оне у којима је неки чвор већ искоришћен, а потом за чворове који остану неповезани редом испитују његови суседи (у оригиналном графу) и додаје грана ка првом неповезаном који је пронађен (наравно, ако се неки пронађе) имала је велику позициону пристрасност, односно гране које су лексикографски испред су фаворизоване. Из тог разлога, код друге варијанте, која се боље показала при тестирању, листа грана се прво промеша - shuffle (наравно, претходно се сачувају индекси битова који одговарају гранама) па се потом пролази кроз њу и бришу гране по истом принципу, такође након тога се за чворове који остану у скупу неповезаних псеудослучајно узима неповезани сусед ка коме се додаје грана. Позивање ове ф-је обавезно иде након иницијализацијее и сваког ажурирања позиције честице након чега се у случају да је добијена вредност мања од до сада личне најбоље или глобалне најбоље (чува се као класна променљива) те позиције ажурирају. Овако ригидан однос има и своје недостатке по питању експлорације простора решења, потребно је балансирати са другим елементима алгоритма.

Мутација

У оригиналном BPSO може доћи до заглављивања у локалном оптимуму када брзине конвергирају близу Vmax или -Vmax (сигмоидна ф-ја имаће вредност блиску 1,

односно 0), мале промене у брзини тешко ће променити одговарајући бит позиције. У циљу да се избегне ова нежељена ситуација, неопходна је велика промена брзине некорелисана са личном и глобалном најбољом позицијом. Један од начина да се ово уради је уметање оператора мутације између ажурирања брзине и ажурирања позиције. Свака вредност у вектору брзине мења знак са вероватноћом r_{mu}, што ће приликом наредног ажурирања позиције честице довести до инвертовања вероватноћа за избор 1 или 0. Дакле, ако се ова операција изврши када је брзина приближна Vmax или -Vmax са вероватноћом блиској 1 доћи ће до инвертовања одговарајућег бита позиције. Као и у генетским алгоритмима, одакле је концепт и преузет, мутација доприноси у диверзификацији и експлорацији простора решења - Lee S, Park H, Jeon M. Binary particle swarm optimization with bit change mutation.

$$\textbf{for}(i = 1; i < n; i = i + 1)\{$$
$$\textbf{if}(\text{rand}() < r_{\text{mu}})$$
$$\textbf{then} v_{i,j_r}(t+1) = -v_{i,j_r}(t+1)\},$$

Генотип-фенотип концепт

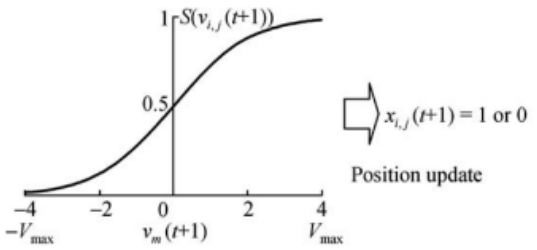
Као што је већ поменуто, главна разлика BPSO у односу на основни PSO је ажурирање позиције честице. За ажурирање позиције код BPSO претходна позиција у бинарном простору претраге је потпуно ирелевантна, једино што је значајно је вектор брзине. Дакле, у BPSO непрекидну брзину и бинарну позицију можемо схватити редом као инстанцу решења и инстанцу решења трансформисану сигмоидном ф-јом на претходно описани начин. Овај концепт је заснован на имитацији генотип-фенотип концепта из природе.

Овим начином размишљања можемо модификовати основни BPSO и укључити информацију о претходној позицији честице приликом одређивања нове - Sangwook Lee,

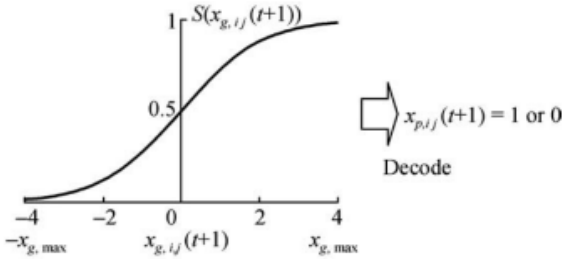
Sangmoon Soak, Sanghoun Oh, Witold Pedrycz, Moongu Jeon Modified binary particle swarm optimization. Брзина остаје непрекидан вектор који се ажурира на исти начин. Уместо позиције за сваку честицу уводимо два нова вектора x_g - генотип и x_p - фенотип. Генотип је непрекидни вектор који се ажурира на начин на који се у основном PSO ажурира вектор позиције (узима у обзир претходни генотип и вектор брзине) и преузима улогу брзине приликом ажурирања позиције у бинарном простору проблема - фенотипа.

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1R_1(p_{\text{best},i,j} - x_{p,i,j}(t)) + c_2R_2(g_{\text{best},i,j} - x_{p,i,j}(t))$$
$$x_{g,i,j}(t+1) = x_{g,i,j}(t) + v_{i,j}(t+1)$$
$$x_{p,i,j}(t+1) = \begin{cases} 0 & \text{if rand}() \geqslant S(x_{g,i,j}(t+1)) \\ 1 & \text{if rand}() < S(x_{g,i,j}(t+1)) \end{cases}$$

$$S(x_{g,i,j}(t+1)) = \frac{1}{1 + e^{-x_{g,i,j}(t+1)}}$$



основни BPSO



модификовани BPSO

Важно је обратити пажњу на неколико ствари:

- Информације о глобалној и личној најбољој позицији које се користе приликом ажурирања брзине се односе на фенотип, а не на генотип
- Модификовани BPSO идентичан је основном са инертивним фактором 1.0 ако је w = 0 у ф-ји ажурирања брзине
- Код основног BPSO имали смо velocity clamping поступак, односно $[-V_{max}, V_{max}]$ за интервал могућих вредности брзине, код модификованог уместо тога радимо clamping вредности из вектора генотипа на интервал $[-X_{g,max}, X_{g,max}]$

- Ако се одлучимо да користимо оператор мутације у комбинацији са модификованим BPSO то радимо тако што мутирамо битове вектора генотипа са вероватноћом r_{mu} уместо битове вектора брзине

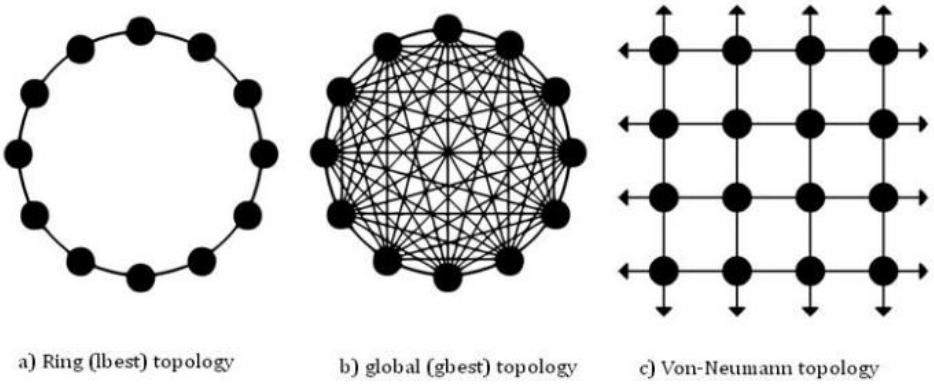
$$\textbf{for}(i = 1; i < n; i = i + 1)\{$$
$$\textbf{if}(\text{rand}() < r_{\text{mu}})$$
$$\textbf{then} x_{g,i,j_r}(t+1) = -x_{g,i,j_r}(t+1)\}$$

Топологије утицаја

До сада приказане варијанте BPSO употребљавале су, први осмишљени, gbest модел где је свака честица под утицајем најбоље јединке из читавог роја, односно свака честица је суседна свакој. Последица тих директних међусобних веза међу честицама је да овај модел често узрокује конвергенцију честица ка локалном оптимуму. Касније је показано да lbest модели у којима се дефинишу суседства за сваку честицу дају боље резултате - Clerc M и Kenney J. Суседства се иницијализију на основу индекса честица у рoју и свака честица је под утицајем најбоље честице у свом суседству. Постоји мноштво различитих lbest топологија од којих су овде имплементиране топологија прстена и Фон Нојманова топологија која се сматра најбољом - према Kennedy J и Mendes R.

Имплементационо гледано, направљена је посебна класа LocalBPSO која је свесна свих честица у рoју и има метод који је задужен да иницијализује вектор суседства за сваку честицу у зависности од тражене топологије. Дакле, i-ти елемент дводимензионалног neighbourhoods вектора је вектор индекса свих честица које су суседне i-тој честици у рoју (наравно свака честица се налази у сопственом суседству). Ако је тражена топологија прстена свака честица имаће левог и десног суседа, Фон Нојманова топологија распоређује честице у дводимензионалну решетку (уз услов да корен броја честица у рoју мора бити цео број) која омогућава честици да дели

информације у више праваца, односно са своја четири суседа: левим, десним, горњим и доњим (у имплементацији није експлицитно прављена решетка, математиком по модулу над индексима постигнуто да свака честица има сва четири суседа). У овој класи имплементиран је и метод који проналази вектор најбољих честица (парова позиција и вредност ф-је циља у њој) у сваком суседству, овај метод биће позван када честице иницијализују своје позиције као и након сваке итерације у којој све честице ажурирају своје позиције. Приликом ажурирања брзине i-та честица у рoју користиће i-ти елемент овог вектора у социјалној компоненти (свака честица свесна свог индекса у рoју - прослеђује јој се из LocalBPSO класе). Метод optimize прати стандардну шему PSO алгоритма, на крају враћа најбољу инстанцу решења пронађену у свим суседствима, коју декодирамо у листу грана.



Приступ заснован на целобројном програмирању

Овај приступ решавању преузет је ради компарације из литературе:

- c - матрица тежина грана (концепт ради и за тежинску варијанту проблема)
- x - бинарна матрица, $x_{ij} = 1$ значи да је грана $(i, j) \in E$ присутна у решењу
- A(i) - скуп суседних чворова чвору i

$$\text{Minimize} \quad \sum_{(i,j)\in E} c_{ij}x_{ij}$$

$$\text{subject to:} \quad \sum_{j\in A(i)} x_{ij} \leq 1 \quad \forall i \in N$$

$$\sum_{\substack{k\in A(i)\\ k\neq j}} x_{ik} + \sum_{\substack{k\in A(j)\\ k\neq i}} x_{kj} + x_{ij} \geq 1 \quad \forall (i,j) \in E$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in E.$$

За имплементацију искоришћена библиотека docplex која имплементира cplex.

Експерименти

У експериментима тестирани су основни gbest BPSO, модификовани gbest BPSO, lbest BPSO са топологијом прстена и lbest BPSO са Фон Нојмановом топологијом и упоређивани са brute-force алгоритмом (за мање инстанце проблема), 2-апроксимативним алгоритмом као и експерименталним резултатима алгоритма из научног рада заснованог на целобројном програмирању за minimum maximal matching проблем у произвољном графу. За тестирање алгоритама имплементирана је random graph generator ф-ја.

Тестирање је вршено на процесору Intel(R) Core(TM) i5-9400F CPU @ 2.90GHz, 8.00 GB RAM меморије, на 64-битном оперативном систему Windows 10. За имплементацију је коришћен програмски језик Python и интегрисано развојно окружење Jupyter Notebook.

Параметри BPSO

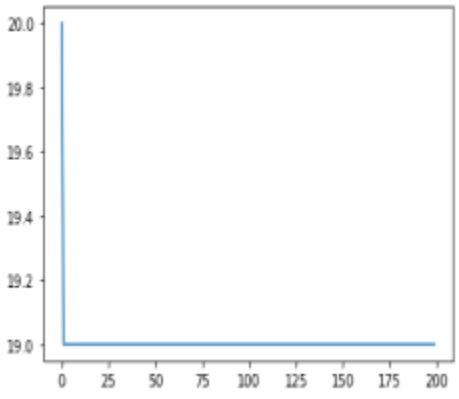
Одабир параметара једна је од кључних ствари које утицу на перформансе алгоритма. За генерисање псеудо-случајних бројева коришћена је Python-ова random библиотека. Што се тиче инертивног, когнитивног и социјалног параметра (w, c1, c2) испоштована је Clerc-ова препорука, уведен је посредни параметар φ, за његову вредност

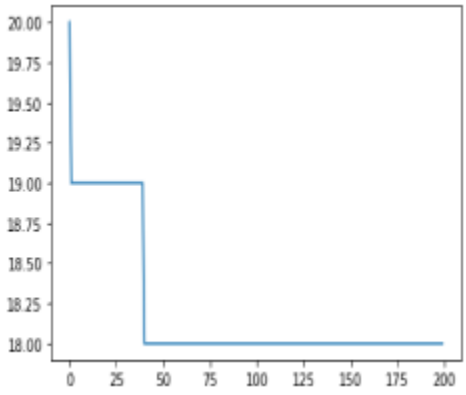
је узето 2.07 (w = 0.689343, c1 = c2 = 1.42694). Поред овог испробан је и стандардни приступ где су c1 = c2 = 2, а w иницијално има вредност 0.9, а потом линеарно опада кроз итерације до 0.4, испробано је и случајно варирање c1 и c2 у интервалу [1.2, 2], ипак ови приступи су узроковали лошије перформансе. За интервал дозвољених вредности вектора брзине код основног, одосно интервал дозвољених вредности вектора генотипа код модификованог BPSO узето је $V_{max} = X_{gmax} = 4$, односно [-4, 4]. Сви алгоритми покретани су за величину роја 50 (49 код lbest Фон Нојманове топологије) и 200 итерација за мање, а 500 за веће инстанце проблема, са вероватноћом мутације $r_{mu} = 0.1$.

Резултати

Мале инстанце: |V| <= 20	**2-approx**	**BPSO**	**mBPSO**	**Ring BPSO**	**VonNeumman BPSO**
Edge density ≈ 0.3	0% / 22.49%	100% / 0%	100% / 0%	100% / 0%	100% / 0%
Edge density ≈ 0.5	0% / 30.71%	80% / 3.1%	90% / 1.42%	80% / 3.1%	90% / 1.42%
Edge density ≈ 0.7	0% /22.17%	80% / 2.68%	80% /2.68%	80% / 2.68%	80% / 2.68%

- *Свака ћелија представља проценат инстанци за које је пронађено оптимално решење (добијено исцрпном претрагом) / просечна процентуална разлика у односу на оптимално решење*

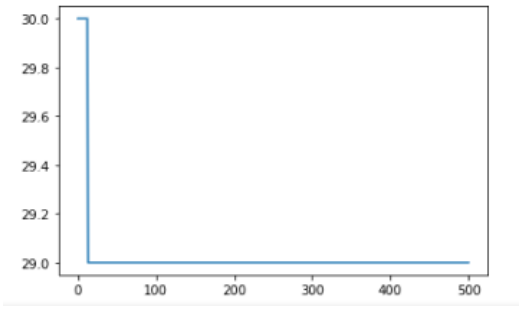




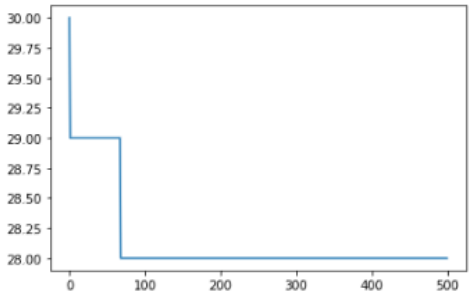
BPSO *модификовани BPSO*

Веће инстанце: 30 <= |V| <= 70	**BPSO**	**mBPSO**	**Ring BPSO**	**VonNeumman BPSO**	**ILP**
Edge density ≈ 0.3	*8.75%*	*10.43%*	*8.15%*	*9.83%*	*12.56%*
Edge density ≈ 0.5	*7.18%*	*8.18%*	*8.18%*	*8.18%*	*10.64%*
Edge density ≈ 0.7	*3.06%*	*3.83%*	*3.06%*	*3.06%*	*6.90%*

- *Свака ћелија представља просечан проценат за колико су решења добијена оптимизационом методом боља од решења добијених 2-апроксимативним алгоритмом*
- *Cplex-у дато лимитирано време за рад (200 s)*



Прстен



Фон Нојман

Број чворова / Број грана	**2-approx**	**BPSO**	**mBPSO**	**Ring BPSO**	**VonNeumman BPSO**	**ILP - 100s**	**ILP - 500s**
47/1039	*23*	*22*	*22*	*22*	*22*	*22*	*22*
58/1588	*29*	*28*	*28*	*28*	*28*	*28*	*28*
56/1475	*28*	*27*	*27*	*27*	*27*	*27*	*27*
62/1768	*30*	*29*	*29*	*29*	*29*	*30*	*29*
49/1122	*24*	*23*	*22*	*22*	*22*	*23*	*22*
35/702	*17*	*16*	*15*	*16*	*15*	*15*	*15*
54/897	*27*	*25*	*25*	*25*	*25*	*25*	*25*
63/601	*31*	*29*	*29*	*29*	*28*	*28*	*28*
53/642	*26*	*24*	*23*	*24*	*24*	*24*	*23*

66/788	*33*	*30*	*30*	*31*	*30*	*30*	*30*
70/1059	*34*	*32*	*32*	*32*	*32*	*31*	*31*
56/1117	*27*	*26*	*26*	*26*	*26*	*26*	*26*
67/815	*32*	*30*	*29*	*29*	*29*	*29*	*29*
66/1328	*32*	*31*	*31*	*30*	*30*	*30*	*30*
46/665	*22*	*19*	*18*	*19*	*18*	*18*	*18*
55/963	*26*	*26*	*25*	*26*	*25*	*25*	*25*
48/770	*23*	*21*	*21*	*21*	*21*	*21*	*21*
69/950	*34*	*33*	*32*	*32*	*32*	*32*	*32*
39/518	*18*	*17*	*17*	*17*	*16*	*17*	*16*

Закључак

У овом раду описан је начин како се један проблем теорије графова може бинарно кодирати. За оптимизацију је коришћена p-метахеуристика BPSO која је потом унапређена увођењем оператора мутације из генетских алгоритама са циљем повећања диверзитета роја. Концепт генотип-фенотип нам је омогућио да посматрамо природу ажурирања брзине и позиције честице из другачије перспективе што је даље довело до модификације алгоритма тако да задржимо добре аспекте основног PSO-а који ради у непрекидном простору решења што је довело до бољих експерименталнх резултата. Поред тога у раду смо се осврнули и на ефекат различитих топологија утицаја на перформансе алгоритма. Експериментални резултати су показали да различити алгоритми који у основи имају BPSO уз, наравно, даља унапређења по питању оператора и параметара, могу бити конкурентни алгоритмима који се традиционално више користе у сфери бинарне оптимизације.

Литература

Sangwook Lee, Sangmoon Soak, Sanghoun Oh, Witold Pedrycz, Moongu Jeon. Modified binary particle swarm optimization

Clerc M, Kenney J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space

Lee S, Park H, Jeon M. Binary particle swarm optimization with bit change mutation

Z. Caner Taşkin, Tınaz Ekim. Integer Programming Formulations for the Minimum Weighted Maximal Matching Problem