MINIMUM MAXIMAL MATCHING

УВОД

- Matching, Maximal Matching
- **Minimum Maximal Matching** - maximal matching минималне кардиналности скупа грана
- Тежинска варијанта - maximal matching минималне суме тежине грана
- Yannakakis и Gavril показали да се ради о NP-тешком проблему
- Сродан проблему Minimum Edge Dominating Set у основној варијанти

УПОРЕДНИ АЛГОРИТМИ

- **Експоненцијални Brute-force алгоритам -** редом пролази кроз партитивни скуп скупа грана почевши од оних кардиналности = 1, чим пронађе неки Maximal Matching враћа га као решење
- **2 - апроксимативни алгоритам** тривијалан, враћа било који maximal matching
- Развијање бољих апроксимативних алгоритама показало се као изазовно
- Gotthilf, Lewenstein и Rainschmidt дошли су до 2 – c * logn/n апроксимације
- За поједине класе графова постоје бољи апроксимативни алгоритми
- **Приступ заснован на целобројном програмирању**

$$
\begin{aligned}
\text{Minimize} \quad & \sum_{(i,j)\in E} c_{ij}x_{ij} \\
\text{subject to:} \quad & \sum_{j\in A(i)} x_{ij} \le 1 \quad \forall i \in N \\
& \sum_{\substack{k\in A(i)\\ k\neq j}} x_{ik} + \sum_{\substack{k\in A(j)\\ k\neq i}} x_{kj} + x_{ij} \ge 1 \quad \forall (i,j) \in E \\
& x_{ij} \in \{0,1\} \quad \forall (i,j) \in E.
\end{aligned}
$$

IBM CPLEX

БИНАРНА ОПТИМИЗАЦИЈА РОЈЕМ ЧЕСТИЦА

- Kennedy и Eberhart аутори методе
- **Главна идеја -** брзина се схвата као вероватноћа да ће бит позиције узети вредност 1 или 0

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

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

- **Важно:** Нова позиција не узима у обзир претходну!
- Опасност од сатурације сигмоидне ф-је
- Решава се **velocity clamping** поступком, интервал могућих вредности брзине је $[-V_{max}, V_{max}]$
- **c1** - утицај когнитивне, **c2** - утицај социјалне компоненте, **w** - инертивни фактор
- Clerc M, Kenney J показали да се добро понашање алгоритма може обезбедити тако што се ове константе учине међусобно зависним, уз увођење посредног параметра **ф** из препорученог интервала [2.01, 2.4]

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

КОДИРАЊЕ ПРОБЛЕМА

- Свака честица има референцу на граф чији MMM желимо да одрадимо
- Граф има методу која на основу матрице повезаности одређује листу грана, позиција честице иницијализује се тако што се та колекција пресликава у **бинарни вектор псеудослучајних вредности**
- **Фитнес ф-ја** рачуна вредност ф-је циља, број јединица у бинарном вектору позиције
- Претходно бави се **одржавањем коректности** решења, тако што своди решење на maximal matching инвертовањем одређених битова
- Редом (лексикографским) пролази кроз скуп грана и елиминише 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)
- Велика **позициона пристрасност**, фаворизоване гране које су лексикографски испред
- **Побољшање:** листа грана се прво промеша, (наравно, претходно се сачувају индекси битова који одговарају грана ма) 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
- Након иницијализације и сваког ажурирања позиције честице проверава се да ли је добијена вредност мања од до сада личне најбоље или глобалне најбоље (чува се као класна променљива), у случају да јесте те позиције се ажурирају

МУТАЦИЈА

- У оригиналном BPSO може доћи до заглављивања у локалном оптимуму када брзине конвергирају близу Vmax или -Vmax (вредност сигмоидне ф-је биће блиска 1 или 0)
- Мале промене у брзини тешко ће променити одговарајући бит позиције
- 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
- Концепт преузет из генетских алгоритама, доприноси диверзификацији роја

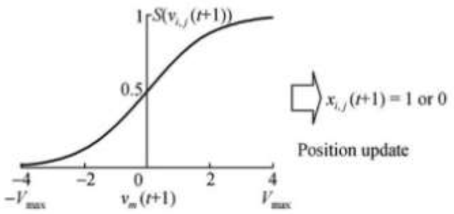
$$
\begin{aligned}
&\mathbf{for}(i = 1; i < n; i = i + 1)\{\\
&\mathbf{if}(\mathrm{rand}() < r_{\mathrm{mu}})\\
&\mathbf{then}\, v_{i,j_r}(t+1) = -v_{i,j_r}(t+1)\},
\end{aligned}
$$

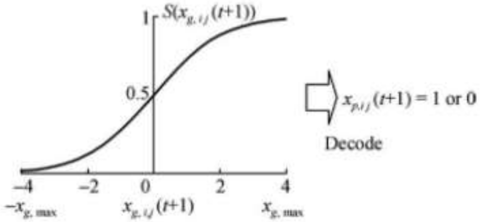
ГЕНОТИП-ФЕНОТИП КОНЦЕПТ

- У BPSO непрекидну брзину и бинарну позицију можемо схватити редом као инстанцу решења и инстанцу решења трансформисану сигмоидном ф-јом на прехтодно описани начин
- Овај концепт заснован је на имитацији **генотип-фенотип** концепта из природе
- Можемо модификовати основни BPSO и укључити информацију о претходној позицији честице приликом одређивања нове
- Брзина остаје непрекидан вектор који се ажурира на исти начин. Уместо позиције за сваку честицу уводимо два нова вектора $\mathbf{x_g}$ **- генотип** и $\mathbf{x_p}$ **- 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

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

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





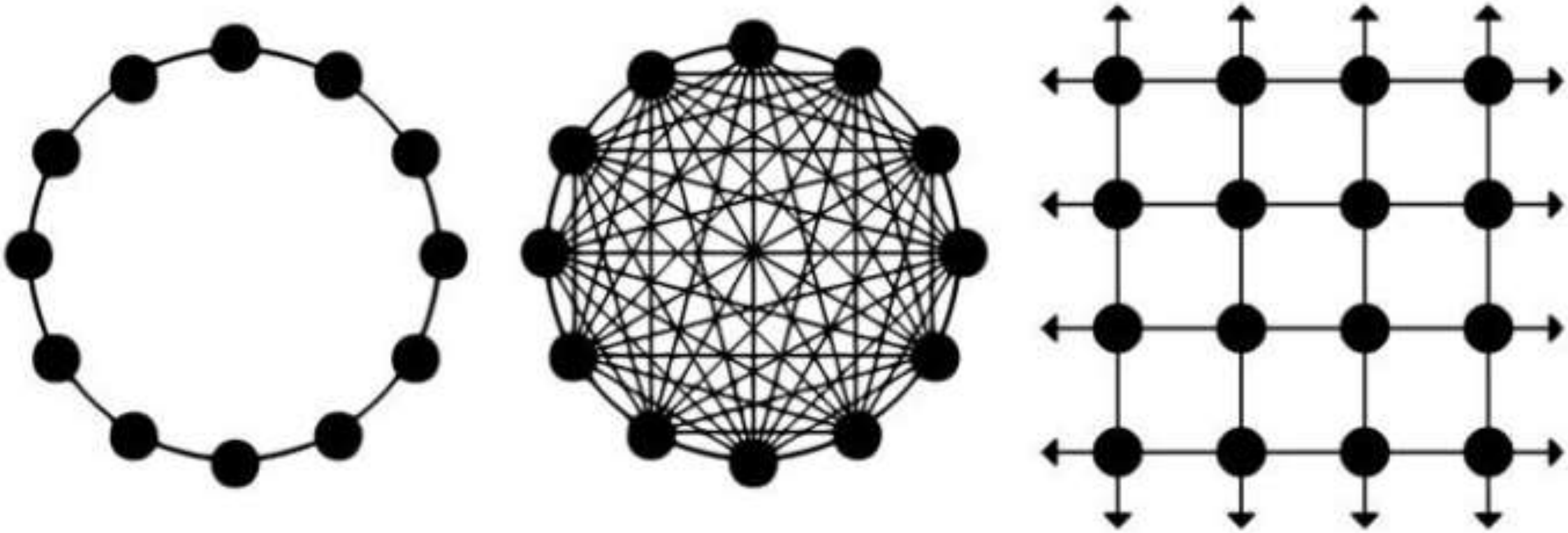
Напомене:

I. *Информације о глобалној и личној најбољој позицији које се користе приликом ажурирања брзине се односе на фенотип, а не на генотип*
II. *Уместо velocity clamping-а радимо clamping вредности из вектора генотипа на интервал [-$X_{g,max}$, $X_{g,max}$]*
III. *Ако се одлучимо да користимо оператор мутације то радимо тако што мутирамо битове вектора генотипа*

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

ТОПОЛОГИЈЕ УТИЦАЈА

- До сада се говорило о **gbest** BPSO, где је свака честица под утицајем најбоље јединке из читавог роја **-** често узрокује конвергенцију ка локалном оптимуму
- Касније показано да **lbest** модели у којима се дефинишу суседства за сваку честицу дају боље резултате **-** Clerc M и Kenney J
- На почетку алгоритамска класа иницијализује **вектор суседства** за сваку честицу у зависности од тражене топологије
- Ако је тражена **топологија прстена** свака честица имаће левог и десног суседа, у случају **Фон Нојманове топологије** честице се распоређују у дводимензионалну решетку (уз услов да корен броја честица у роју мора бити цео број) што омогућују честице да дели информације са четири своја суседа: левим, десним, горњим и доњим
- Алгоритамска класа има метод који израчунава вектор најбољих честица **-** најбољу честицу за свако суседство, овај метод позива се након што све све честице иницијализују или ажурирају своје позиције на крају итерације
- Приликом наредног ажурирања брзине свака честица у роју користиће одговарајући елемент овог вектора у социјалној компоненти
- На крају алгоритам враћа најбољу инстанцу пронађену у свим суседствима



a) Ring (lbest) topology b) global (gbest) topology c) Von-Neumann topology

ОДАБИР ПАРАМЕТАРА

- Једна од кључних ствари које утичу на перформансе алгоритма
- 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, као и случајно варирање c1 и c2 у интервалу [1.2, 2]
- За интервал дозвољених вредности вектора брзине, односно генотипа 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 $r_{mu} = 0.1$

ЕКСПЕРИМЕНТАЛНИ УСЛОВИ

- Процесор: Intel(R) Core(TM) i5-9400F CPU @ 2.90GHz
- RAM: 8.0 GB
- Оперативни систем: Windows 10, 64-bit
- Програмски језик: Python
- Развојно окружење: Jupyter Notebook
- Тест скуп: генерисан Erdős–Rényi моделом

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%

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%*

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 р-метахеуристика BPSO која је потом унапређена увођењем оператора мутације са циљем повећања диверзитета. Концепт генотип-фенотип нам је омогућио да посматрамо природу ажурирања брзине и позиције честице из другачије перспективе што је даље довело до мофикације алгоритма тако да задржимо добре апсекте основног PSO-а који ради у непрекидном простору решење што је довело до бољих експерименталних резултата. Поред тога у раду смо се осврнули и на ефекат различитих топологија на перформансе алгоритма. Експериментални резултати су показали да различити алгоритими 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*

ПАВЛЕ САВИЋ 169/2017