sent and received in portions of n walks each, the total of

$$\frac{N}{(P-1)\,n}$$

communications are needed for all of the N experiments to exchange messages between the senior and executive processes. Time c spent on one such exchange is a constant and is consumed for sending one integer value and three floating-point values.

Since work on portions of the tasks does not require any synchronization between the executive processes, the time consumed by the entire group of processors for parallel calculations is P – 1 times smaller than in the case of the sequential implementation. Thus, estimation for the time complexity of the parallel algorithm can be presented in the following form:

$$T(N,P) = \frac{T(N)}{(P-1)} + \frac{N}{(P-1)n}c \qquad (15)$$

This time complexity is quite good and demonstrates high parallelization efficiency, which fully justifies additional resources spent on development, implementation and launch of the respective software.

Література

1. Dekhtiaryuk, E.S. & Syniavsky, O.L. 1978. Solving elasticity theory problems by Monte Carlo method. 8-th International congress for applied mathematics in engineering, Weimar.
2. Muller M. E. 1956. Some continuous Monte Carlo method for the Dirichlet problem. Ann. Math. Stat., 27, (3).
3. Sabelfeld K. K. & Shalimova I. A. 1997. Spherical means for PDEs. VSP, Utrecht.
4. Sabelfeld, K. K. 1991. Monte Carlo Methods in Boundary Value Problems. Springer–Verlag, Berlin.
5. Varvaak, L. P., Dekhtiaryuk, E. S., Reutfarb, I. Z., Syniavsky, O. L. & Khimenko, V. V., 1973. Solving the spatial problem of elasticity theory by the method of statistical experiments. Resistance of materials and structural theory, 20, Kyiv.

Синявський О. Л., Кислоокий В. М., Хоменко О. І.

ПАРАЛЕЛЬНИЙ АЛГОРИТМ РОЗВ'ЯЗАННЯ ПЕРШОЇ КРАЙОВОЇ ЗАДАЧІ ТЕОРІЇ ПРУЖНОСТІ В 3-ВИМІРНОМУ ПРОСТОРІ ЗА МЕТОДОМ МОНТЕ-КАРЛО

Роботу присвячено розв'язанню першої крайової задачі теорії пружності для 3-вимірних тіл довільної форми та зв'язності.

Наш підхід полягає у розробленні методу блукання сферами для знаходження розв'язку системи диференційних рівнянь 6-го порядку, яка відповідає цій задачі. Запропонований алгоритм є паралельним і розрахованим на виконання на високопродуктивних обчислювальних кластерах.

Ключові слова: алгоритм блукання сферами, метод Монте-Карло, перша крайова задача, теорія пружності, паралельний алгоритм.

Матеріал надійшов 14 червня 2011 р.

УДК 519.854

Шило В. П., Градинар І. П., Ляшко В. І.

НАБЛИЖЕНИЙ АЛГОРИТМ ЗНАХОДЖЕННЯ МАКСИМАЛЬНОГО *K*-PLEX (CO-*K*-PLEX) ГРАФУ

У розвідці запропоновано та досліджено наближений алгоритм розв'язання задачі знаходження максимального k-plex (co-k-plex) графу, який дав змогу покращити рекорди для деяких задач.

Ключові слова: граф, k-plex, co-k-plex, незалежна множина, кліка, соціальні мережі, біологічні мережі.

Вступ

Останнім часом зріс інтерес до задач на мережах, які зводяться до задач на графах. Серед них широке практичне застосування [2, 5] має задача знаходження максимального k-plex (co-k-plex) графу, яка виникає, зокрема, при аналізі соціальних, біологічних, фінансових та ін. мереж. Поняття k-plex графу введено 1978 р. у праці [6], а

© *Шило В. П., Градинар І. П., Ляшко В. І.*, 2011

в [2] сформульовано проблему знаходження максимального k-plex графу як задачу цілочислового програмування та доведено її NP-складність.

У цій розвідці представлено наближений алгоритм знаходження максимального k-plex (co-k-plex) графу, в якому враховано переваги запропонованих в [1] наближених алгоритмів дискретної оптимізації. Наведено результати застосування даного алгоритму до DIMACS-графів, для багатьох з них покращено відомі з літератури рекорди для k-plex (co-k-plex) графів.

Позначення та визначення

Нехай задано неорієнтований граф $G(V,E)$, в якому V – множина вершин, E – множина ребер. Позначимо $N(v)$ множину вершин графу G, зв'язних з вершиною $v \in V$, а $deg_G(v)$ – кількість таких вершин. Нехай $G[V'] = (V', E \cap (V' \times V'))$ – підграф графу $G(V,E)$, породжений підмножиною $V' \subset V$.

Клікою в графі G називається підмножина множини V вершин, які всі попарно зв'язні між собою. Незалежною множиною в графі G називається підмножина множини V вершин, які всі попарно не зв'язні між собою [1].

Розглянемо більш загальні поняття: k-plex та co-k-plex [2, 5, 6]. Будемо вважати, що $k \in N$ і $k \geq 1$.

Означення 1. Множина $K \subset V$ називається k-plex в графі $G(V,E)$, якщо

$$\forall v \in K \; deg_{G[K]}(v) \geq |K| - k. \qquad (1)$$

Означення 2. Множина $C \subset V$ називається co-k-plex в графі G(V, E), якщо

$$\forall v \in C \; deg_{G[C]}(v) \leq k - 1. \qquad (2)$$

На рис. 1 наведено приклади k-plex та co-k-plex графів.

1-plex – кліка графу, co-1-plex – незалежна множина. Справедливе твердження, що co-k-plex в графі G є k-plex у його доповненні. Неважко також показати, що для будь-яких $k_1 < k_2$, всякий k_1-plex є k_2-plex-ом, а будь-який co-k_1-plex є co-k_2-plex-ом.

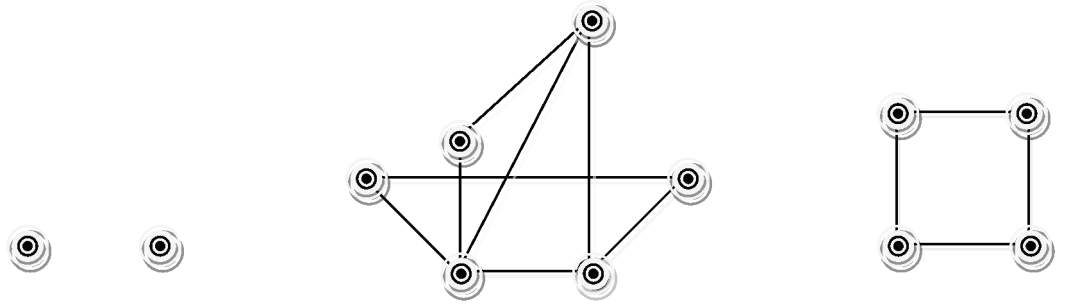
Задача знаходження максимального k-plex (co-k-plex) графу полягає в побудові k-plex (co-k-plex) максимальної потужності.

Переваги застосування *k*-plex графу

Довільну мережу можна представити у вигляді графу, вершинам якого відповідають її учасники (ними можуть бути населені пункти, банки, компанії, люди, нейрони і т. д.), а ребра вказують на зв'язки між ними. Часто у мережі виникає потреба у виділенні підгрупи з великою щільністю. Під щільністю мережі (або її підгрупи) розуміємо відношення кількості зв'язків, що існують між її учасниками до максимально можливої кількості таких зв'язків. Аналогічно, щільність графу – це відношення кількості ребер до максимально можливої їх кількості.

Незважаючи на можливість використання клік, у багатьох задачах на мережах, вони насправді не завжди пасують для визначення підгруп у них з великою щільністю [5]. Це пояснено необхідністю існування зв'язків між всіма парами вершин, що належать до кліки у відповідному графі, та вмотивовує розгляд релаксації (послаблення) кліки. Серед видів релаксації кліки виділяють [2] k-clique, k-club та k-plex, останній з яких розглянуто тут.

Нехай у деякій великій мережі виділено певну підмережу з кількістю учасників $n=2m$ ($m \in N$), причому кожен з них не має зв'язку тільки з одним з інших учасників, а зв'язаний з n–2 іншими учасниками. Щільність відповідного такій підмережі графу дорівнює $(n–2)/(n–1)$ і є близькою до 1 при великих n. Хоча всі вершини перебувають у дуже сильному зв'язку між собою, максимальна кліка (1-plex) буде мати потужність лише $n/2$ незалежно від величини n. Кількість таких максимальних клік – 2^m. Натомість максимальний 2-plex буде містити всі n вершин цього графу (удвічі більше, ніж у максимальній кліці). Наприклад, при n=2000 кожна вершина зв'язна з 1998-ма іншими вершинами, а щільність графу становить 0,9995. Проте максимальна кліка складається тільки з 1000 вершин. Тоді як 2-plex



2-plex та co-1-plex 4-plex та co-5-plex 2-plex та co-3-plex

Рис. 1. Приклади k-plex та co-k-plex

складається з усіх 2000 вершин. Такі графи при $n = 6$ та $n = 8$ представлено на рис. 2.

Приклад демонструє переваги використання *k*-plex графу для виявлення у мережі підгруп з великою щільністю. Задача знаходження максимального *k*-plex (co-*k*-plex) графу має практичне застосування до найрізноманітніших мереж, що існують у реальності.

Алгоритм

Нехай задано граф $G(V,E)$. Потрібно побудувати максимальний *k*-plex (co-*k*-plex). Позначимо його K.

Для $\forall v \in K$ повинна виконуватися умова (1) або (2) належності до *k*-plex чи co-*k*-plex, відповідно.

Будь-яка вершина v, що претендує на вибір в K, повинна задовольняти умові:

$$\forall v' \in K \cup \{v\}\ deg_{G[K\cup\{v\}]}(v') \geq |K \cup \{v\}| - k \quad (\forall v' \in K \cup \{v\}\ deg_{G[K\cup\{v\}]}(v') \leq k - 1). \tag{3}$$

Наближений алгоритм розв'язання задачі побудови максимального *k*-plex (co-*k*-plex) графу полягає в наступному. Спочатку покладемо $K=\varnothing$. Тоді всі вершини графу $G(V, E)$ задовольняють умові (3). Рандомізовано вибираємо для занесення у множину K таку вершину, яка задовольняє умові (3) та порушить її для мінімальної кількості інших вершин. Цей процес повторюємо до тих пір, поки існують вершини, що задовольняють умові (3). Далі задля побудови більшого *k*-plex (co-*k*-plex) видаляємо з множини K декілька випадково вибраних вершин і знову, керуючись вище описаним правилом вибору вершин, поповнюємо K. Останню дію повторюємо до виконання певного критерію (наприклад, часу для розв'язання задачі, досягнення заданої потужності множини K та ін.).

Слід зазначити, що іноді (з малою ймовірністю) може здійснюватися випадковий вибір вершин для занесення в множину K не тільки серед таких, що порушать умову (3) для мінімальної кількості вершин, а й серед усіх, які ще можуть бути занесені в K.

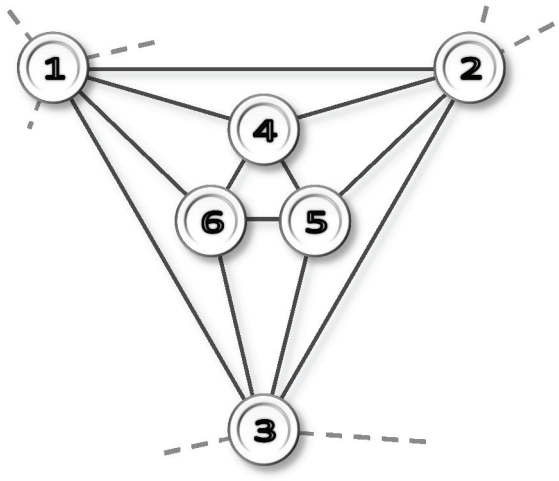
Загальну схему алгоритму можна зобразити:

```
1.  largest_cardinality = 0;
2.  while global_criterion ( )
3.        K = ∅;
4.        filling_K ( );
5.        if largest_cardinality<|K|
6.              largest_cardinality = |K|;
7.        end if
8.        while local_criterion ( )
9.              i = choosing_value ( );
10.             deleting_some_verticies_
                from_K(i);
11.             filling_K ( );
12.             if largest_cardinality<|K|
13.                   largest_cardinality = |K|;
14.             end if
15.       end while
16. end while
17. return largest_cardinality
```

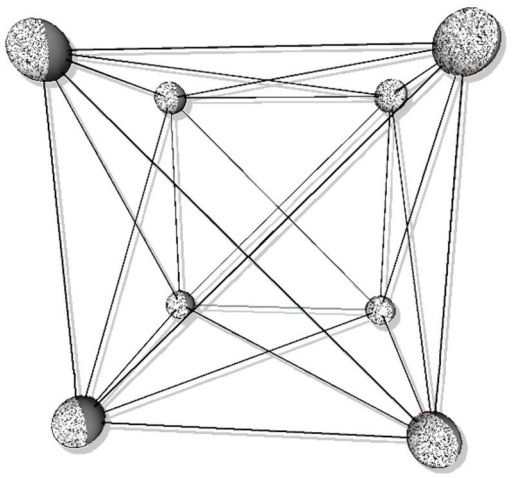
На основі запропонованого алгоритму розроблено програмне забезпечення, налаштоване на знаходження *k*-plex та co-*k*-plex графу для будь-якого *k*. Крім того, розроблено модифікацію алгоритму для знаходження *k*-plex (co-*k*-plex) максимальної ваги ($w_i \in R$, i=1, …, $|V|$) в графі при довільному *k*.

Результати обчислювального експерименту

Для тестування наближеного алгоритму ми вибрали DIMACS-графи [3, 4]. Перелік цих графів, їх розмірності, а також відомі з літератури рекорди *k*-plex для них наведено в табл. 1. Конфігурація комп'ютера, на якому здійснювались обчислювальні експерименти (при завантаженні 25 %): Intel® Core™2 Quad CPU, Q9550 @



n=6



n=8

Рис. 2. Приклади 2-plex

2.83GHz, 8,00GB RAM. Визначений для кожного графу максимальний час за певного значення k – одна година. Результати розв'язання поставленої задачі запропонованим алгоритмом для тестових графів при $1 \le k \le 3$ представлено в табл. 1.

З результатів аналізу розрахунків випливає, що за допомогою наближеного алгоритму покращено відомі нам рекорди 2-plex для 8-ми графів та 3-plex для 13-ти графів. Для 25-ти графів одержано відомі з літератури рекорди 2-plex та для 21-го графу відомі рекорди 3-plex, причому, багато значень є точними. При застосуванні наближеного алгоритму до задачі знаходження максимальної кліки (1-plex), а до неї застосовано вже чимало алгоритмів, одержано відомі значення потужності для 58-ми графів (значна частина цих значень також є точними), а для 8-ми графів їх не досягнуто.

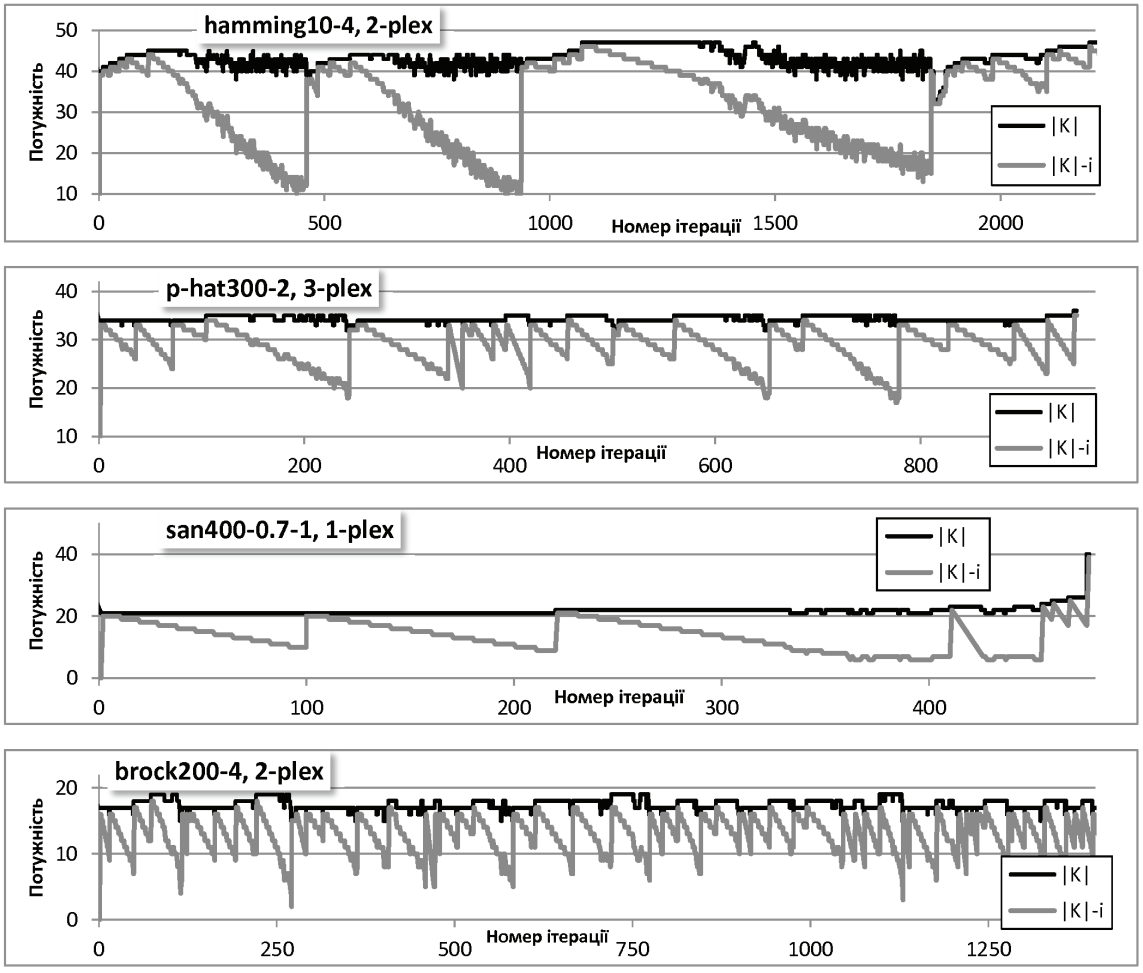


Рис 3. Схематичні ілюстрації роботи алгоритму

Графіки на рис. 3 схематично ілюструють роботу запропонованого алгоритму. Для кожного прикладу верхній графік ($|K|$) відображає зміну потужності максимального по включенню k-plex, а нижній ($|K|$-i) – зміну потужності цієї множини в моменти, коли з неї видалено i вершин.

Висновки

Порівнявши одержані рекорди для k-plex графу та час їх знаходження з відомими з досліджень результатами (напр., з [5]), можна дійти висновку, що запропонований алгоритм знаходження максимального k-plex (со-k-plex) графу проявляє прийнятну ефективність.

Таблиця 1. **Результати обчислювальних експериментів**

№	$G(V,E)$	1-plex				2-plex				3-plex			
		Відома потужність	Одержана потуж-ність	Час, с	Верхня оцінка	Відома потужність	Одержана потуж-ність	Час, с	Верхня оцінка	Відома потужність	Одержана потуж-ність	Час, с	Верхня оцінка
1	brock200-1	**21**	**21**	0,000	*21*	**25**	**26**	0,015	*53*	**28**	**30**	0,000	*139*
2	brock200-2	**12**	**12**	0,688	*12*	**13**	**13**	0,015	*13*	**15**	**16**	6,125	*89*

№	*G*(*V*,*E*)	1-plex				2-plex				3-plex			
		Відома потужність	Одержана потуж-ність	Час, с	Верхня оцінка	Відома потужність	Одержана потуж-ність	Час, с	Верхня оцінка	Відома потужність	Одержана потуж-ність	Час, с	Верхня оцінка
3	brock200-3	**15**	**15**	6,453	*15*		**17**	0,016	–		**20**	0,093	–
4	brock200-4	**17**	**17**	1,657	*17*	**20**	**20**	0,359	*41*	**22**	**23**	0,875	*122*
5	brock400-1	**27**	**25**	0,203	*27*		**30**	0,031	–		**36**	13,547	–
6	brock400-2	**29**	**25**	0,204	*29*	**28**	**30**	0,422	*133*	**31**	**36**	2,343	*267*
7	brock400-3	**31**	**31**	228,094	*31*		**30**	1,172	–		**36**	0,125	–
8	brock400-4	**33**	**33**	742,000	*33*	**33**	**31**	47,032	*133*	**33**	**36**	0,594	*265*
9	brock800-1	**23**	**21**	1,187	*23*		**25**	3,375	–		**29**	42,797	–
10	brock800-2	**24**	**21**	1,593	*24*	**23**	**25**	5,109	*221*	**26**	**30**	1481,391	*401*
11	brock800-3	**25**	**22**	95,235	*25*		**25**	16,562	–		**30**	1899,453	–
12	brock800-4	**26**	**21**	2,172	*26*	**23**	**25**	5,875	*223*	**25**	**29**	11,704	*410*
13	c-fat200-1	**12**	**12**	0,000	*12*	**12**	**12**	0,000	*12*	**12**	**12**	4,562	*12*
14	c-fat200-2	**24**	**24**	0,000	*24*	**24**	**24**	0,000	*24*	**24**	**24**	0,062	*24*
15	c-fat200-5	**58**	**58**	0,000	*58*	**58**	**58**	0,000	*58*	**58**	**58**	0,016	*58*
16	c-fat500-1	**14**	**14**	0,000	*14*	**14**	**14**	0,047	*14*	**14**	**14**	369,860	*14*
17	c-fat500-2	**26**	**26**	0,000	*26*	**26**	**26**	0,031	*26*	**26**	**26**	198,078	*26*
18	c-fat500-5	**64**	**64**	0,000	*64*	**64**	**64**	0,031	*64*	**64**	**64**	0,218	*64*
19	c-fat500-10	**126**	**126**	0,000	*126*	**126**	**126**	0,047	*126*	**126**	**126**	0,156	*126*
20	hamming6-2	**32**	**32**	0,000	*32*	**32**	**32**	0,000	*32*	**32**	**32**	0,000	*32*
21	hamming6-4	**4**	**4**	0,000	*4*	**6**	**6**	0,000	*6*	**8**	**8**	0,000	*8*
22	hamming8-2	**128**	**128**	0,000	*128*	**128**	**128**	0,000	*128*	**128**	**128**	0,000	*128*
23	hamming8-4	**16**	**16**	0,000	*16*	**16**	**16**	0,000	*16*	**18**	**20**	0,015	*48*
24	hamming10-2	**512**	**512**	0,000	*512*	**512**	**512**	0,000	*512*	**512**	**512**	0,000	*768*
25	hamming10-4	**40**	**40**	0,032	*40*	**43**	**48**	0,015	*128*	**64**	**64**	6,282	*192*
26	johnson8-2-4	**4**	**4**	0,000	*4*	**5**	**5**	0,000	*5*	**8**	**8**	0,000	*8*
27	johnson8-4-4	**14**	**14**	0,000	*14*	**14**	**14**	0,000	*14*	**18**	**18**	0,000	*18*
28	johnson16-2-4	**8**	**8**	0,000	*8*		**10**	0,000	–		**16**	0,000	–
29	johnson32-2-4	**16**	**16**	0,000	*16*		**21**	0,000	–		**32**	1,141	–
30	keller4	**11**	**11**	0,000	*11*	**15**	**15**	0,000	*15*	**21**	**21**	0,015	*88*
31	keller5	**27**	**27**	0,015	*27*		**31**	0,000	–		**45**	1,547	–
32	keller6	**59**	**59**	67,032	–		**63**	0,062	–		**93**	82,765	–
33	MANN-a9	**16**	**16**	0,000	*16*	**26**	**26**	0,000	*26*	**36**	**36**	0,000	*36*
34	MANN-a27	**126**	**126**	0,016	*126*	**236**	**236**	0,000	*236*	**351**	**351**	0,000	*366*
35	MANN-a45	**345**	**344**	0,610	*345*	**662**	**662**	0,000	*668*	**990**	**990**	0,000	*108*
36	MANN-a81	**1100**	**1098**	31,016	–		**2162**	0,078	–		**3240**	0,015	–
37	p-hat300-1	**8**	**8**	0,000	*8*	**10**	**10**	0,110	*10*	**12**	**12**	0,328	*12*
38	p-hat300-2	**25**	**25**	0,000	*25*	**30**	**30**	0,016	*30*	**35**	**36**	0,000	*129*
39	p-hat300-3	**36**	**36**	0,000	*36*	**43**	**44**	0,000	*108*	**51**	**52**	0,000	*201*
40	p-hat500-1	**9**	**9**	0,000	*9*		**12**	1,078	–		**14**	0,563	–
41	p-hat500-2	**36**	**36**	0,000	*36*		**42**	0,000	–		**50**	0,094	–
42	p-hat500-3	**50**	**50**	0,000	–		**62**	0,000	–		**72**	0,312	–
43	p-hat700-1	**11**	**11**	0,063	*11*	**13**	**13**	0,187	*13*	**14**	**15**	5,156	*123*
44	p-hat700-2	**44**	**44**	0,000	*208*	**51**	**52**	0,015	*146*	**58**	**62**	0,015	*272*
45	p-hat700-3	**62**	**62**	0,000	*201*	**73**	**76**	0,031	*243*	**84**	**89**	0,031	*428*
46	p-hat1000-1	**10**	**10**	0,016	*10*		**13**	0,578	–		**15**	0,078	–
47	p-hat1000-2	**46**	**46**	0,031	–		**56**	0,203	–		**67**	0,203	–
48	p-hat1000-3	**68**	**68**	0,047	–		**82**	0,063	–		**98**	0,047	–
49	p-hat1500-1	**12**	**12**	13,265	*12*		**14**	4,875	–		**17**	2712,375	–
50	p-hat1500-2	**65**	**65**	0,016	–		**80**	0,125	–		**93**	0,188	–
51	p-hat1500-3	**94**	**94**	0,047	–		**114**	0,047	–		**133**	0,047	–
52	san200-0.7-1	**30**	**30**	0,000	*30*		**31**	0,000	–		**46**	0,078	–
53	san200-0.7-2	**18**	**18**	0,531	*18*	**26**	**26**	0,016	*79*	**36**	**37**	0,016	*140*
54	san200-0.9-1	**70**	**70**	0,000	*70*	**90**	**90**	0,000	*127*	**125**	**125**	0,000	*125*
55	san200-0.9-2	**60**	**60**	0,016	*60*		**71**	0,000	–		**105**	0,000	–
56	san200-0.9-3	**44**	**44**	0,016	*44*		**54**	0,015	–		**73**	0,000	–
57	san400-0.5-1	**13**	**13**	0,047	*13*		**15**	0,016	–		**22**	0,047	–
58	san400-0.7-1	**40**	**40**	0,094	*40*		**41**	0,031	–		**61**	33,531	–
59	san400-0.7-2	**30**	**30**	0,047	*30*		**32**	0,125	–		**46**	0,031	–
60	san400-0.7-3	**22**	**22**	0,156	*22*		**27**	0,078	–		**38**	0,093	–
61	san400-0.9-1	**100**	**100**	0,000	*100*		**102**	0,032			**150**	0,000	–
62	san1000	**15**	**15**	0,718	*15*		**18**	180,672	–		**25**	58,922	–
63	sanr200-0.7	**18**	**18**	0,000	*18*		**22**	0,046	–		**26**	0,031	–
64	sanr200-0.9	**42**	**42**	0,000	*42*		**51**	0,016	–		**61**	0,468	–
65	sanr400-0.5	**13**	**13**	0,500	*13*		**15**	0,188	–		**18**	0,156	–
66	sanr400-0.7	**21**	**21**	0,172	*21*		**26**	0,235	–		**30**	0,843	–

Література

1. Сергиенко И. В. Задачи дискретной оптимизации : проблемы, методы решения, исследования / И. В. Сергиенко, В. П. Шило. – К. : Наук думка, 2003. – 264 с.
2. Balasundaram B. Clique relaxations in social network analysis : The maximum k-plex problem [e-resource]: (B. Balasundaram. Journal publications) / B. Balasundaram, S. Butenko, I. V. Hicks // Operations Research – December 8, 2009. – Режим доступу: http://iem.okstate.edu/baski/files/kplex4web.pdf. – Назва з екрана.
3. Clique Benchmark Instances [e-resource]: (Penn State Harrisburg Math/Computer Sciences home page). – Режим доступу: http://cs.hbg.psu.edu/benchmarks/clique.html. – Назва з екрана.
4. DIMACS: Maximum clique, graph coloring, and satisfiability. Second DIMACS implementation challenge [e-resource]: (The

site of DIMACS Implementation Challenges). – Режим доступу : http://dimacs.rutgers.edu/Challenges/. – Назва з екрана.
5. Hicks I. V. Combinatorial Algorithms for the Maximum k-plex Problem [e-resource]: (I. V. Hicks. Papers) / I. V. Hicks, B. McClosky // Journal of Combinatorial Optimization. – Режим доступу : http://www.caam.rice.edu/~ivhicks/CombiOptPaper-1.pdf. – Назва з екрана.
6. Seidman S. B. A graph theoretic generalization of the clique concept / S. B. Seidman, B. L. Foster // Journal of Mathematical Sociology. – 1978. – № 6. – P. 139–154.

V. Shylo, I. Gradinar, V. Lyashko

AN APPROXIMATE ALGORITHM FOR FINDING MAXIMUM *K*-PLEX (CO-*K*-PLEX) IN A GRAPH

In the paper an approximate algorithm for solving the maximum k-plex (co-k-plex) problem in a graph was proposed and studied. This algorithm improved the records for some benchmarks.

Keywords: graph, k-plex, co-k-plex, independent set, clique, social networks, biological networks.
Матеріал надійшов 5 травня 2011 р.

УДК 004.42:510.69

Шкільняк С. С.

ВІДНОШЕННЯ ЛОГІЧНОГО НАСЛІДКУ ДЛЯ МНОЖИН ФОРМУЛ У КОМПОЗИЦІЙНО-НОМІНАТИВНИХ ЛОГІКАХ

Розглянуто відношення логічного наслідку для композиційно-номінативних логік часткових однозначних, тотальних неоднозначних та часткових неоднозначних квазіарних предикатів. Запропоновано різні формалізації відношення логічного наслідку для множин формул. Досліджено властивості таких формалізацій в різних семантиках для загального випадку логік квазіарних предикатів, для логік еквітонних і логік антитонних предикатів.

Ключові слова: логіка, предикат, композиційно-номінативний підхід, семантика, логічний наслідок.

Поняття логічного наслідку належить до центральних понять логіки. Воно може бути формалізованим за допомогою відношень логічного наслідку. Різноманітні відношення та нестандартні семантики для пропозиційної логіки досліджені О. Д. Смирновою [3]. Подібні семантики та відношення узагальнено [4] для композиційно-номінативних логік квазіарних предикатів.

Композиційно-номінативними названі логіки, які будуються на основі композиційно-номінативного підходу [2]. Цей підхід до побудови моделей програм та орієнтованих на них логік виявився дуже плідним: на його базі розроблено широкий спектр логічних формалізмів різних рівнів абстрактності та загальності. Композиційно-номінативні логіки (КНЛ) базуються на загальних класах часткових відображень, заданих на довільних наборах іменованих значень. Такі відображення названо квазіарними. Передумовою виникнення КНЛ стала необхідність посилення можливостей класичної логіки для розв'язку нових задач інформатики й програмування.

Метою цієї розвідки є дослідження відношень логічного наслідку для множин формул КНЛ часткових однозначних, тотальних неоднозначних і часткових неоднозначних квазіарних предикатів реномінативного та кванторного рівнів. Логіки часткових однозначних предикатів – це логіки з неокласичною семантикою, тотальних неоднозначних – із пересиченою семантикою, часткових неоднозначних – із загальною семантикою.

Поняття, які не визначено у цій розвідці, будемо тлумачити за працями [1, 4].

1. Квазіарні предикати та їх композиції. Мови КНЛ

Під предикатом на множині D розуміємо довільну функцію вигляду $P : D \to \{T, F\}$, де $\{T, F\}$ – множина істиннісних значень.

© *Шкільняк С. С.*, 2011