УДК 519.682.1

С.В. Листровой, д-р техн. наук
Украинский госуниверситет железнодорожного транспорта
(Украина, 61050, Харьков, пл. Фейербаха, 7,
тел. (050) 9355042, e-mail: om1sergeyvladimirovih@gmail.com),
А.В. Сидоренко
Samsung Electronics Ukraine Company,
LLC Samsung R&D Institute Ukraine
(Украина, 01302, Киев, ул. Льва Толстого, 57,
тел. +380509800852, e-mail: cdandrey@gmail.com),
Е.С. Листровая, канд. техн. наук
Национальный аэрокосмический университет им. Н.Е. Жуковского
(Украина, 61070, Харьков, ул. Чкалова, 17,
e-mail: listravkina@gmail.com)

Метод поиска наибольших максимальных независимых множеств вершин неориентированного графа

Предложен метод поиска наибольших максимальных независимых множеств неориентированного связного графа, позволяющий при числе вершин в графе, не превышающем 120, и плотностях ребер в диапазоне от 0,067 до 0,9, решать задачу определения наибольших максимальных независимых множеств за полиномиальное время. При дальнейшем увеличении числа вершин и уменьшении плотности ребер в графе алгоритм имеет экспоненциальную сложность, в среднем не превышающую $O(2^{0,4n})$, которая имеет тенденцию к уменьшению при увеличении плотности ребер в графе, где n — число вершин графа.

Ключевые слова: максимальное независимое множество, клика, вершинное покрытие.

Запропоновано метод пошуку найбільших максимальних незалежних множин неорієнтованого зв'язкового графа, який дозволяє при числі вершин в графі, що не перевищує 120, і щільності ребер у діапазоні від 0,067 до 0,9, вирішувати задачу визначення найбільших максимальних незалежних множин за поліноміальний час. При подальшому збільшенні числа вершин і зменшенні щільності ребер в графі алгоритм має експоненціальну складність, що в середньому не перевищує $O(2^{0,4n})$, яка має тенденцію до зменшення при збільшенні щільності ребер в графі, де n — число вершин графа.

Ключові слова: максимальна незалежна безліч, кліка, вершинне покриття.

Поиск наибольших максимальных независимых множеств (НМНМ) вершин неориентированного графа — одна из важнейших задач дискретной математики. Алгоритмы решения этой задачи используются в широких спектрах прикладных проблем, в том числе в социометрии [1], химии

[2—4], компьютерных технологиях [5—8] и биоинформатике [9—15]. Это объясняется тем, что, представив объект исследования в виде модели на графе, множество задач из указанных областей науки можно свести к задаче поиска клики в неориентированном графе или к задаче о минимальном вершинном покрытии в графе, что, в свою очередь, легко трансформируется в задачу нахождения максимальных независимых множеств графа.

Проблема поиска наименьшего покрытия и соответственно НМНМ в произвольном графе — одна из первых задач, выделенных в категорию *NP*-полных [1]. Сделано множество попыток разработки методов, позволяющих найти точное решение за полиномиальное время. Однако до настоящего времени как в теории, так и на практике существуют лишь алгоритмы [2], дающие приближенный результат при построенинии такого вершинного покрытия, в котором число вершин не более чем в k раз превосходит минимально возможное (k — точность приближенного алгоритма).

В последние годы проблема нахождения оптимального алгоритма решения задачи о наименьшем покрытии рассматривалась с учетом фиксированной параметрической сложности [16]. В основе параметрического моделирования лежит идея разделения проблемы на две составляющие. С одной стороны, имеется совокупность множества входных данных, а с другой, — множество различных параметров, так или иначе влияющих на общую вычислительную сложность исследуемого алгоритма. Это позволяет сформировать более гибкую классификацию *NP*-сложных проблем по сравнению с классическим подходом, когда сложность алгоритма зависит только от входного потока.

С начала 50-х годов прошлого века было предложено много точных алгоритмов для решения задач, связанных с поиском максимальных независимых множеств [1—20], но построить эффективный алгоритм, имеющий полиномиальную оценку сложности, до сих пор не удалось. Если для решения задачи о перечислении всех клик графа разработка такого алгоритма представляется мало вероятной (в силу экспоненциальной зависимости числа клик от размерности графа [21]), то для поиска одного НМНМ невозможность построения подобного алгоритма остается недоказанной в силу нерешенной проблемы взаимосвязи классов P и NP. Это позволяет надеяться на возможность создания полиномиального алгоритма для решения данной задачи.

Исходя из того, что все эти алгоритмы имеют экспоненциальную оценку сложности $O(2^{\alpha n})$, разработчики стараются модифицировать свои алгоритмы с целью сокращения дерева поиска, что, в свою очередь, приводит к уменьшению показателя α. Наиболее полный обзор алгоритмов решения данной задачи сделан в работе [22], в которой подробно описана история развития алгоритмов поиска максимальных независимых множеств.

В общем случае все алгоритмы решения задачи о максимальном независимом множестве можно разделить на две группы: алгоритмы перечисления всех максимальных независимых множеств и алгоритмы определения ННМН. Среди алгоритмов нахождения ННМН можно выделить алгоритмы Робсона [19] с временной сложностью $O(2^{0,276n})$ и алгоритм MaxIS [23] и Плотникова [24], позволившие улучшить временные характеристики алгоритма Робсона.

Рассмотрим эффективный алгоритм поиска ННМН и перечисления ННМН неориентированного графа.

Основные понятия и определения. Во многих прикладных задачах синтеза и анализа вычислительных систем и сетей, а также при разработке специального математического обеспечения для их функционирования требуется найти в конечном множестве объектов максимальную систему объектов, попарно не связанных друг с другом, или выбрать минимальную систему объектов, связанных со всеми другими. Формулировки подобных задач на языке теории графов приводят к понятиям независимости и покрытия.

Множество вершин U графа $G(V, E)$ называется независимым (внутренне устойчивым), если никакие две вершины из этого множества не смежные, т.е. если $U \subseteq V$ и U независимо в графе G, то порожденный подграф $G(U)$ является пустым. Очевидно, что если при этом $U^* \subseteq U$, то U^* — также независимое множество. Внутренне устойчивое множество называется максимальным, если оно не является собственным подмножеством некоторого другого независимого множества.

Введем следующие обозначения: X_i^r — множество независимых вершин графа $G(V, E)$, мощность которого равна r; Y_i^k — множество вершин, с которыми связаны вершины, принадлежащие i-му независимому множеству X_i^r.

Рассмотрим множество Ω всех пар независимых $X_i^{r=2}$ вершин в графе $G(V, E)$ и множество вершин Y_i^k, состоящее из k вершин, с которыми связаны вершины из $X_i^{r=2}$ в графе $G(V, E)$. Мощность множества Ω обозначим $\omega = |\Omega|$. Множества X_i^r максимальной мощности могут быть построены из исходного графа $G(V, E)$ посредством объединения всех пар вершин, которые в графе $G(V, E)$ не связаны между собой.

В общем случае условие $X_i^r \cup Y_i^k = V$ означает, что сформированное множество является максимально независимым, поскольку в этом случае подмножество Y_i^k представляет вершинное покрытие в графе $G(V, E)$, а подмножество X_i^r — максимальное независимое множество вершин графа $G(V, E)$, дополняющее это подмножество до n. Если $X_i^r \cup Y_i^k \neq V$, то

очевидно, существуют вершины, которые можно добавить в подмножество X_i^r и оно останется независимым, и число таких вершин не превосходит $n - r$. Если $X_i^r \cup Y_i^k = V$, то таких вершин в графе $G(V, E)$ не существует.

Важной характеристикой анализируемых графов является плотность ρ ребер m в графе $G(V, E)$. Под плотностью ребер графа будем понимать отношение $\rho = m / E_{\max}$, где m — число ребер в графе $G(V, E)$, $E_{\max} = \frac{n(n-1)}{2}$ — максимально возможное число ребер в графе $G(V, E)$, содержащем n вершин.

Формализация и постановка задачи. В качестве исходных данных для работы алгоритма будем использовать все пары независимых $X_i^{r=2}$ вершин в графе $G(V, E)$. При этом, если есть подмножества $(X_i^r | Y_i^k)$, $(X_j^r | Y_j^k) \ldots (X_p^r | Y_p^k)$, являющиеся максимальными, то в множество U заносим наибольшие по мощности максимальные независимые множества. Следует заметить, что при внесении новых множеств в U дублирующие множества удаляются и в множестве U остаются только множества максимальной мощности. Если множество $(X_i^r | Y_i^k)$ является максимальным, то будем помечать его одной звездочкой. Рассмотрим следующую процедуру $\boldsymbol{A_0}$ для перечисления НМНМ на основе введенных правил преобразования множеств.

Процедура A_0.

Ш а г 1. Формируем множество Ω всех пар независимых множеств $(X_i^{r=2} | Y_i^k)$ вершин в графе $G(V, E)$ и проверяем, есть ли среди них множества, являющиеся максимальными независимыми, и если есть, то заносим их в множество U, исключая их из множества Ω.

Ш а г 2. Выбираем из Ω множество $Q = (X_{i=t}^r | Y_{i=t}^k)$ и на его основе начинаем формировать НМНМ. Для этого в оставшихся множествах в Ω удаляем вершины, принадлежащие выбранным множествам X_t^r и Y_t^k, и переходим к выполнению следующего шага.

Ш а г 3. Проверяем все ли множества в Ω стали пустыми. Если да, то достроить множество $(X_t^r | Y_t^k)$ до максимального нельзя, поэтому данное множество исключаем из дальнейшего анализа и переходим к выполнению шага 2, иначе переходим к выполнению следующего шага.

Ш а г 4. В Ω после удаления вершин, принадлежащих множествам X_t^r и Y_t^k выбранного множества, выбираем множество $(X_{i=q}^r | Y_{i=q}^k)$ с максимальным значением $| X_{i=q}^r |$. Если таких множеств несколько, то формируем объединение $Q = X_i^r \cup X_{i=q}^r$ и переходим к выполнению следующего шага.

Ш а г 5. Проверяем, является ли Q максимальным независимым множеством. Если нет, то переходим к выполнению шага 2, иначе — заносим

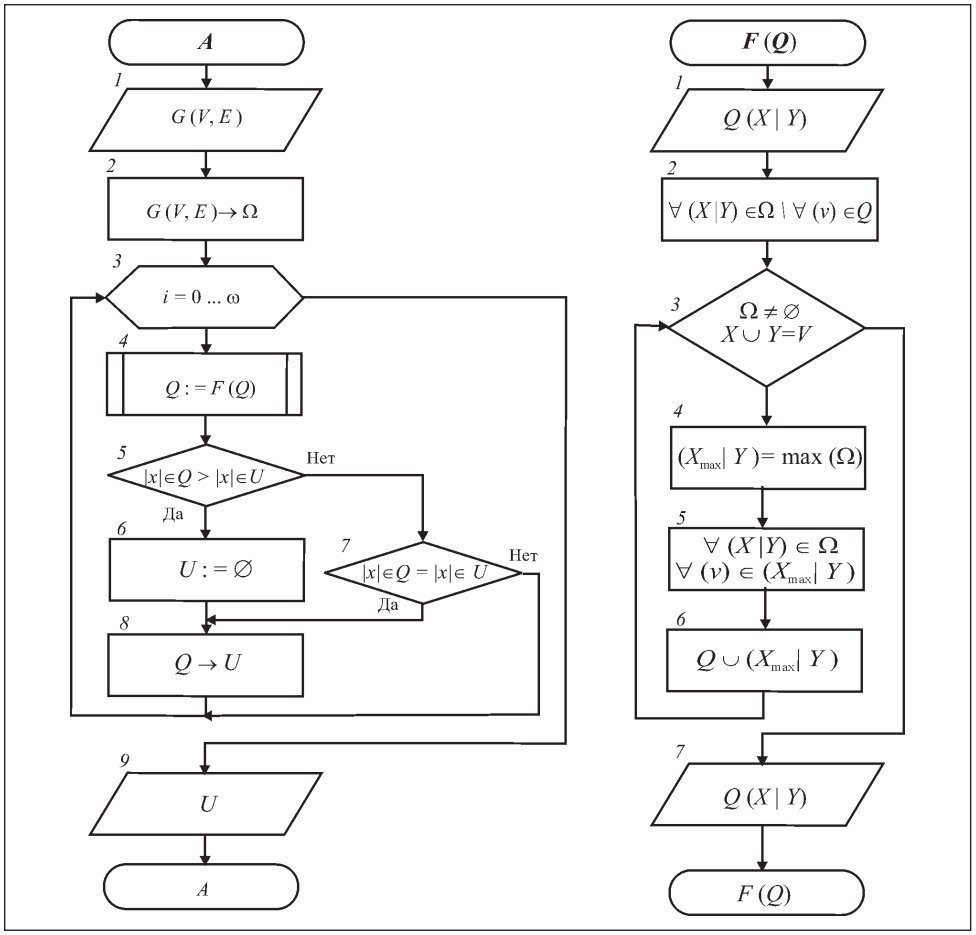

Рис. 1. Блок-схемы процедур ***A*** и ***F*** (***Q***)

его в множество U . При этом в множестве U оставляем только множества с максимальной мощностью и переходим к выполнению следующего шага.

Ш а г 6. Проверяем на основе всех множеств из Ω, сформированы ли максимальные независимые множества. Если нет, то переходим к выполнению шага 2, иначе — процедура заканчивает работу, так как множества, содержащиеся в U, образуют НМНМ исследуемого графа.

Для удобства программной реализации процедуры $\boldsymbol{A_0}$ ее удобно представить в виде двух взаимосвязанных процедур: ***A*** и ***F*** (***Q***) (рис. 1).

В процедуре ***A*** блоки *1*, *2* формируют множество Ω, состоящее из пар $X|Y$; блок *3* соответствует организации цикла перебора всех пар $X|Y$ из Ω, на основе которых строятся НМНМ с запуском работы процедуры ***F*** (***Q***); блоки *5—8* реализуют процесс формирования множества U.

В процедуре $\boldsymbol{F}(\boldsymbol{Q})$ блоки *1*, *2* реализуют выбор пары $Q=(X|Y)$ из всех пар множества Ω и удаление вершин $\{v_i\}$, принадлежащих множествам X и Y из всех пар $X|Y\in\Omega$; блоки *3*—*6* реализуют цикл до тех пор, пока не будут проанализированы все пары $X|Y$ и выполнено равенство $\Omega=\varnothing$, т.е. до того момента, когда множество станет пустым, $\Omega=\varnothing$, а множества Q не могут стать максимальными, т.е. $X\cup Y\neq V$; блок *7* возвращает множество Q в работу процедуры $\boldsymbol{A}$.

Пример работы процедуры A_0. Пусть задан граф $G(V,E)$ с набором вершин i и их связей с остальными вершинами $\{j\}$ (табл. 1). Формируем множество Ω (табл. 2). Максимальные множества вносим в множество $U=\{67;38;18\}$ и удаляем их из табл. 2. Затем формируем множества, начиная с Q = 23 | 1568, которое в табл. 3 помечено двумя звездочками. Для этого удаляем вершины, принадлежащие множеству Q = 23 | 1568, и получаем табл. 4.

Таблица 1

Граф $G(V, E)$								
i	1	2	3	4	5	6	7	8
$\{j\}$	3 6 7	5 6 8	1 5 6	6 8	2 3 6 7 8	1 2 3 4 5 8	1 5 8	2 4 5 6 7

Таблица 2

Множество пар $X_i^{r=2}\|Y_i^k$ множества P				
23 | 1 568	27 | 1 568	34 | 1 568	37 | 1 568	47 | 1 568
15 | 23 678	45 | 23 678	12 | 35 678	14 | 3 678	24 | 568
67 | 123 458*	38 | 124 567*	18 | 234 567*		

Таблица 3

Множество пар $X_i^r\|Y_i^k$ множества Ω				
23 | 1 568**	27 | 1 568	34 | 1 568	37 | 1 568	47 | 1 568
15 | 23 678	45 | 23 678	12 | 35 678	14 | 3 678	24 | 568

Таблица 4

Множество пар $X_i^r\|Y_i^k$ после удаления вершин 1, 2, 3, 5, 6, 8 из множеств в табл. 3				
$\varnothing$	7 | $\varnothing$	4 | $\varnothing$	7 | $\varnothing$	47 | $\varnothing$
$\varnothing$	4 | $\varnothing$	$\varnothing$	4 | $\varnothing$	4 | $\varnothing$

Выбираем множество большей мощности, 47 | ∅, и формируем множества $Q = 23 \mid 1568 \cup 47 \mid \varnothing = 2347 \mid 1568^{*}$ и $U = \{2347\}$. Затем формируем множества, начиная с Q = 27 | 1568, которое в табл. 5 помечено двумя звездочками. Удаляя вершины, принадлежащие множеству $Q = 27 \mid 1568$, получаем табл. 6.

Выбираем множество большей мощности, 34 | ∅, и формируем множества $Q = 27 \mid 1568 \cup 34 \mid \varnothing = 2347 \mid 1568^{*}$ и $U = \{2347\}$. Далее формируем множества, начиная с $Q = 34 \mid 1568$, которое в табл. 7 помечено двумя звездочками. Для этого удаляем вершины, принадлежащие множеству $Q = 34 \mid 1568$, и получаем табл. 8.

Таблица 5

Множество пар $X_i^r \mid Y_i^k$ множества Ω				
23 | 1 568 15 | 23 678	27 | 1 568** 45 | 23 678	34 | 1 568 12 | 35 678	37 | 1 568 14 | 3 678	47 | 1 568 24 | 568

Таблица 6

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 1, 2, 5, 6, 7, 8 из множеств в табл. 5				
3 | ∅ ∅	∅ 4 | ∅	34 | ∅ ∅	3 | ∅ 4 | ∅	4 | ∅ 4 | ∅

Таблица 7

Множество пар $X_i^r \mid Y_i^k$ множества Ω				
23 | 1 568 15 | 23 678	27 | 1 568 45 | 23 678	34 | 1 568** 12 | 35 678	37 | 1 568 14 | 3 678	47 | 1 568 24 | 568

Таблица 8

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 1, 3, 4, 5, 6, 8 из множеств в табл. 7				
2 | ∅ ∅	27 | ∅ 4 | ∅	∅ 2 | ∅	7 | ∅ 4 | ∅	7 | ∅ 2 | ∅

Таблица 9

Множество пар $X_i^r \mid Y_i^k$ множества P				
23 | 1 568 15 | 23 678	27 | 1 568 45 | 23 678	34 | 1 568 12 | 35 678	37 | 1 568** 14 | 3 678	47 | 1 568 24 | 568

Выбираем множество большей мощности, 27 | ∅, и формируем множества $Q = 34|1568 \cup 27 \mid \varnothing = 2347 \mid 1568^{*}$ и $U = \{2347\}$. Далее формируем множества, начиная с Q = 37 | 1568, которое в табл. 9 помечено двумя звездочками. Для этого удаляем вершины, принадлежащие множеству Q = = 37 | 1568, и получаем табл. 10.

Выбираем множество большей мощности, 24 | ∅, и формируем множества $Q = 37 \mid 1568 \cup 24 \mid \varnothing = 2347 \mid 1568^{*}$ и $U = \{2347\}$. Затем формируем множества, начиная с Q = 47 | 1568, которое в табл. 11 помечено двумя звездочками. Удаляем вершины, принадлежащие множеству Q = 47 | 1 568, и получаем табл. 12.

Таблица 10

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 1, 3, 5, 6, 7, 8 из множеств в табл. 9				
2 | ∅	2 | ∅	4 | ∅	∅	4 | ∅
∅	4 | ∅	2 | ∅	4 | ∅	24 | ∅

Таблица 11

Множество пар $X_i^r \mid Y_i^k$ множества Ω				
23 | 1 568	27 | 1 568	34 | 1 568	37 | 1 568	47 | 1 568**
15 | 23 678	45 | 23 678	12 | 35 678	14 | 3 678	24 | 568

Таблица 12

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 1, 4, 5, 6, 7, 8 из множеств в табл. 11				
23 | ∅	2 | ∅	3 | ∅	3 | ∅	∅
∅	∅	2 | ∅	∅	2 | ∅

Таблица 13

Множество пар $X_i^r \mid Y_i^k$ множества Ω				
23 | 1 568	27 | 1 568	34 | 1 568	37 | 1 568	47 | 1 568
15 | 23 678**	45 | 23 678	12 | 35 678	14 | 3 678	24 | 568

Таблица 14

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 1, 2, 3, 5, 6, 7, 8 из множеств в табл. 13				
∅	∅	4 | ∅	∅	4 | ∅
∅	4 | ∅	∅	4 | ∅	4 | ∅

Выбираем множество большей мощности, 23 | ∅, и формируем множества $Q = 47 \mid 1568 \cup 23 \mid \varnothing = 2347 \mid 1568^{*}$ и $U = \{2347\}$. Затем формируем множества, начиная с $Q = 15 \mid 23678$, которое в табл. 13 помечено двумя звездочками. Удаляем вершины, принадлежащие множеству $Q = 15 \mid 23678$, и получаем табл. 14.

Выбираем множество $4 \mid \varnothing$ и формируем множества $Q = 115 \mid 23678 \cup \cup 4 \mid \varnothing = 145 \mid 23678^{*}$ и $U = \{2347\}$. Далее формируем множества, начиная с $Q = 45 \mid 23678$, которое в табл. 15 помечено двумя звездочками. Удаляем вершины, принадлежащие множеству $Q = 45 \mid 23678$, и получаем табл. 16.

Таблица 15

Множество пар $X_i^r \mid Y_i^k$ множества Ω				
23 | 1 568 15 | 23 678	27 | 1 568 45 | 23 678**	34 | 1 568 12 | 35 678	37 | 1 568 14 | 3 678	47 | 1 568 24 | 568

Таблица 16

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 2, 3, 4, 5, 6, 7, 8 из множеств в табл. 15				
∅ 1 | ∅	∅ ∅	∅ 1 | ∅	∅ 1 | ∅	∅ ∅

Таблица 17

Множество пар $X_i^r \mid Y_i^k$ множества Ω				
23 | 1 568 15 | 23 678	27 | 1 568 45 | 23 678	34 | 1 568 12 | 35 678**	37 | 1 568 14 | 3 678	47 | 1 568 24 | 568

Таблица 18

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 1, 2, 3, 5, 6, 7, 8 из множеств в табл. 17				
∅ ∅	∅ 4 | ∅	4 | ∅ ∅	∅ 4 | ∅	4 | ∅ ∅

Таблица 19

Множество пар $X_i^r \mid Y_i^k$ множества Ω				
23 | 1 568 15 | 23 678	27 | 1 568 45 | 23 678	34 | 1 568 12 | 35 678	37 | 1 568 14 | 3 678**	47 | 1 568 24 | 568

Выбираем множество $1 \mid \varnothing$ и формируем множества $Q = 45 \mid 23678 \cup 1 \mid \varnothing = 145 \mid 23678^{*}$ и $U = \{2347\}$. Формируем множества, начиная с $Q = 12 \mid 35678$, которое в табл. 17 помечено двумя звездочками. Удаляем вершины, принадлежащие множеству $Q = 12 \mid 35678$, и получаем табл. 18.

Из оставшегося множества $4 \mid \varnothing$ формируем множества $Q = 12 \mid 35678 \cup 4 \mid \varnothing = 124 \mid 23678^{*}$ и $U = \{2347\}$. Далее формируем множества, начиная с $Q = 14 \mid 3678$, которое в табл. 19 помечено двумя звездочками. Удаляем вершины, принадлежащие множеству $Q = 14 \mid 3678$, и получаем табл. 20.

Выбираем множество $2 \mid 5$ и формируем множества $Q = 14 \mid 3678 \cup 2 \mid 5 = 124 \mid 235678^{*}$ и $U = \{2347\}$. Далее формируем множества, начиная с $Q = 24 \mid 568$, которое в табл. 21 помечено двумя звездочками. Удаляем вершины, принадлежащие множеству $Q = 24 \mid 568$, и получаем табл. 22.

Выбираем множество большей мощности $37 \mid 15$ и формируем множества $Q = 24 \mid 568 \cup 37 \mid 15 = 2347 \mid 1568^{*}$ и $U = \{2347\}$.

Максимальные множества построены на основе всех множеств Ω, приведенных в табл. 2, и при этом осталось одно множество {2347}. Следовательно, число максимальных множеств в анализируемом графе — одно. Решение задачи для графа, заданного табл. 1, методом, перечисляющим все максимальные независимые множества [25], дало следующий результат: {2347;145; 67; 38; 18}, что подтверждает верность процедуры $\boldsymbol{A_0}$.

Таблица 20

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 1, 3, 4, 6, 7, 8 из множеств в табл. 19				
2 | 5 5 | 2	2 | 5 5 | 2	$\varnothing$ 2 | 5	$\varnothing$ $\varnothing$	$\varnothing$ 2 | 5

Таблица 21

Множество пар $X_i^r \mid Y_i^k$ множества Ω				
23 | 1 568 15 | 23 678	27 | 1 568 45 | 23 678	34 | 1 568 12 | 35 678	37 | 1 568 14 | 3 678	47 | 1 568 24 | 568**

Таблица 22

Множество пар $X_i^r \mid Y_i^k$ после удаления вершин 2, 4, 5, 6, 8 из множеств в табл. 21				
3 | 15 1 | 37	7 | 1 $\varnothing$	3 | 15 1 | 37	37 | 15 1 | 37	7 | 15 $\varnothing$

Оценка сложности работы процедуры A_0 и ее экспериментальное исследование. Поскольку исходными данными для работы процедуры являются все пары независимых вершин в анализируемом графе, представляет интерес оценить их число для всех произвольных неориентированных графов, мощность $\omega=|\Omega|$, где Ω — множества всех пар $X_i^{r=2}$ независимых вершин в графе $G(V,E)$. Покажем, что справедливо следующее соотношение:

$$\omega = E_{\max}\alpha(\rho), \tag{1}$$

где $\alpha(\rho)=1-\rho$. Ясно, что число ω несвязанных пар $X_i^{r=2}$ вершин в графе равно числу ребер в графе, являющемся дополнением $\overline{G}$ графа $G(V,E)$, в котором вершины соединены ребрами, если они не соединены в графе $G(V,E)$. Если в графе $G(V,E)$ содержится m ребер, то справедливо равенство

$$\omega = E_{\max} - m = E_{\max}\left(1-\frac{m}{E_{\max}}\right) = E_{\max}(1-\rho) = E_{\max}\alpha(\rho),$$

что и требовалось показать. В общем случае сложность работы процедуры A_0 определяется этапом выполнения поочередного построения максимальных независимых множеств на основе каждого из оставшихся множеств $\{X_i^r | Y_i^k\}$, число которых не может превышать $\omega = E_{\max}\alpha(\rho)$. При этом формирование множеств осуществляется рекурсивно с выбором каждый раз наибольшего по мощности множества.

Для интерпретации работы процедуры A_0 воспользуемся представлением исходного графа G в виде симметричного дерева путей, предложенного в работах [26, 27], в которых подробно рассмотрены идеи рангового подхода к решению задач теории графов и дискретной оптимизации. Смысл такого представления заключается в следующем. Вместо графа $G(V, E)$ с n вершинами будем рассматривать граф D (рис. 2). Этот граф с $(n-1)^2$ вершинами позволяет рассмотреть возможность достижения вершины i в графе $G(V, E)$ путями ранга $r=1$, т.е. используя одно ребро, путями ранга $r=2$, используя два ребра, путями ранга $r=n-1$, используя $n-1$ ребро.

В работах [26, 27] граф D назван стянутым деревом всех путей графа $G(V, E)$. Дерево всех путей D содержит $(n-1)$ горизонтальную линейку и $(n-1)$ ярус. Для прочтения путей на каждой горизонтальной линейке можно быть только один раз. Исходя из стянутого дерева путей, для произвольной вершины j множество путей, ведущих в эту вершину из некоторой вершины s, можно представить в следующем виде:

$$m_s(j) = m_{sj}^{r=1} \cup m_{sj}^{r=2} \cup \ldots \cup m_{sj}^{r=n-1};\ \ j=(\overline{1,n-1}), \tag{2}$$

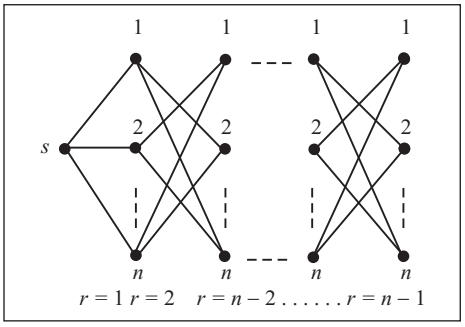

Рис. 2. Стянутое дерево всех путей *D* графа *G* (*V*, *E*)

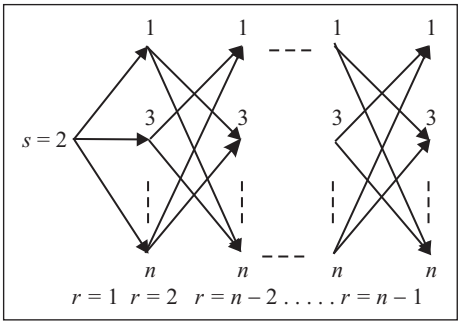

Рис. 3. Стянутое дерево всех путей *D* графа *G* (*V*, *E*) от вершины *s* = 2

где $m_{sj}^r = \{\mu_{sj}^r\}$ — подмножества путей из произвольной вершины *s* в некоторую вершину *j* графа $G(V, E)$ ранга *r*.

Следует заметить, что дерево всех путей *D* может быть построено и от конкретной вершины *i* графа. В этом случае вершина *s* = *i* и *i*-я горизонтальная линейка исключаются в *D*. Например, при *i* = 2 дерево *D* будет иметь вид, представленный на рис. 3. Каждой вершине в графе *D* поставим в соответствие пару $(X_i^r | Y_i^k)$. Поскольку мощность множества *P* всех пар равна ω, общее число вершин *n* в таком графе *D* будет равно ω.

Если рассматривать две пары вершин, $(X_i^r | Y_i^k)$ и $(X_j^{r+1} | Y_j^k)$, расположенных соответственно на ярусах *r* и *r* + 1 графа *D*, то ребро, соединяющее эту пару вершин в графе *D*, будет существовать тогда и только тогда, если объединение $X_i^r \cup X_j^{r+1}$ образует независимое множество вершин. При этом будем полагать, что вершины *i* и *j* удовлетворяют свойству ν.

Если в графе *D* выделить путь $\mu_{ip}^{r=q} = (X_i^r | Y_i^k)(X_j^r | Y_j^k)...(X_p^{r=q} | Y_p^k)$ ранга *r* = *h* из вершины *i* в вершину *p*, то длину этого пути $d(\mu_{ip}^{r=q})$ будем характеризовать мощностью множества независимых вершин $X_i^r \cup X_j^r \cup ... \cup X_p^r$, характеризующих данный путь в графе *D*.

Таким образом, задачу нахождения НМНМ можно рассматривать как задачу определения самого длинного пути $d_{\max}$ в графе *D*, а задачу перечисления всех НМНМ — как задачу перечисления всех путей длины $d_{\max}$ в графе *D*. Решение данной задачи может быть реализовано посредством формирования путей в графе *D* на основе следующего рекуррентного соотношения:

$$\mu_{sp}^{r+1} = \{\mu_{sj}^r\}_{\max} \cup (j, p)\}, \quad j = (\overline{1, \omega}), \quad p = (\overline{1, \omega}), \; j \neq p, \tag{3}$$

где $\{\mu_{sj}^r\}_{\max}$ — путь максимальной длины, выделенный на *r*-м ярусе, т.е. сформированное на текущий момент независимое множество вершин мак-

симальной мощности; (j, p) — ребро графа D, если вершины j, p удовлетворяют свойству ν; ω — число различных вершин в графе D.

Фактически работа процедуры $\boldsymbol{A_0}$ заключается в поочередном нахождении путей максимальной длины в графе D от вершин $s = (1, 2, ..., \omega)$ ко всем остальным вершинам графа на основе рекуррентного соотношения (3). При формировании путей μ_{sp}^{r+1} следующего ранга $r + 1$ на основе путей μ_{sj}^{r} в графе D на каждом шаге их формирования удаляются все вершины, уже вошедшие в путь μ_{sj}^{r}, что соответствует шагу 2 процедуры $\boldsymbol{A_0}$.

Такая интерпретация работы процедуры $\boldsymbol{A_0}$ позволяет получить оценку сверху ее временной сложности, так как на каждом ярусе в любом подмножестве m_{sj}^{r} процедура $\boldsymbol{A_0}$, в случае выделения только одного экстремального пути на ярусе, строит не более одного пути, а на следующем ярусе — соответственно не более ω путей. Поскольку число ярусов в графе D не может превосходить (ω – 1), общее число путей, построенное процедурой $\boldsymbol{A_0}$ до последнего яруса, не превысит величины $(\omega - 1)\,\omega \approx \omega^2$. Следовательно, число элементарных операций сравнения работы процедуры за один проход не превысит ω^2. Следует также учесть число операций, связанных с выделением максимального элемента в массиве из ω чисел, которое не превысит $\omega \log_2 \omega$. Поскольку число таких проходов равно ω, общая сложность работы процедуры не превысит $\boldsymbol{O}(\omega(\omega^2 + \omega \log_2 \omega)) \approx \boldsymbol{O}(\omega^3)$. Учитывая, что

$$\omega = E_{\max}\alpha(\rho) = \frac{n(n-1)}{2}\alpha(\rho)$$

и полагая $\alpha_{\max} = 0{,}9$, получаем временную сложность процедуры $\boldsymbol{A_0}$, в худшем случае равную $\boldsymbol{O}(0{,}09n^6)$.

Данная оценка получена в предположении, что число путей на каждом следующем ярусе уменьшается и не превышает ω^2. Однако при $\rho \to 0$ может наблюдаться экспоненциальное возрастание формируемых путей и следовательно, данная оценка является оценкой снизу. Поэтому представляет интерес сделать оценку работы процедуры $\boldsymbol{A_0}$ в среднем.

Для оценки временной сложности работы процедуры $\boldsymbol{A_0}$ в среднем и точности ее работы было проведено экспериментальное исследование. При этом в процессе тестирования на каждую точку построенных зависимостей генерировалось от 50 до 70 задач заданной размерности и оценивалось среднее число элементарных операций, выполняемых процедурой $\boldsymbol{A_0}$, среднее время решения задачи в миллисекундах, вероятность решения задачи за время, не превышающее заданное допустимое время $T_д$, и проводилось сравнение точности решения с помощью алгоритма перечисления

всех максимальных независимых множеств [25]. При этом число вершин в графе менялось в диапазоне от 10 до 70, а плотность графа — от 10 до 70 %. Все результаты получены с доверительной вероятностью 0,95 и приведены в табл. 23.

Экспериментальное исследование работы процедуры $\boldsymbol{A_0}$ показало, что при малых размерностях исследуемого графа временная сложность процедуры может быть уменьшена посредством объединений множеств в Ω. Если в Ω есть подмножества $(X_i^r | Y_i^k); (X_j^r | Y_j^k) ... (X_p^r | Y_p^k)$, у которых $Y_i^k = Y_j^k = ... = Y_p^k = Y$, то они могут быть объединены в одно подмножество, описываемое подмножествами $(X_{\mu i}^r | Y)$, где $X_{\mu i}^r = X_i^r \cup X_j^r \cup ... \cup X_p^r$. При этом, если $X_{\mu i}^r \cup Y = V$, то $X_{\mu i}^r$ — максимальное независимое множество. В случае, когда множеств $(X_i^r | Y_i^k)$ с одинаковыми множествами Y_i^k нет, то

Таблица 23

ρ, %	Результаты экспериментальных исследований при n						
	10	20	30	40	50	60	70
	Число элементарных операций						
10	1327	48073	331916	1275734	3729554	9011714	20042018
30	861	19323	106211	361107	965756	2190503	4366822
50	342	5942	30698	92633	222313	474830	853169
70	179	1989	8441	23368	50164	90655	151995
	Время решения заданий (мс)						
10	0,0201	0,4936	4,2625	24,9259	77,6657	184,1488	469,4326
30	0,0072	0,2807	2,3260	14,3485	33,3954	94,4739	195,3890
50	0,0021	0,1199	0,9901	6,0142	13,3891	53,3381	75,8878
70	0,0002	0,0136	0,2585	1,9057	4,4737	13,4991	25,2988
	Вероятность выполнения заданий за время $T_д \leq 30$ с						
10	1	1	0,9991	0,6999	0,3204	0,1503	0,0619
30	1	1	1	0,8764	0,5928	0,2721	0,1423
50	1	1	1	0,9932	0,8936	0,4302	0,3265
70	1	1	1	1	0,9988	0,8916	0,6945
	Оценка сложности выполнения заданий						
10	$n^{3,5}$	$n^{3,7}$	$n^{3,8}$	$n^{3,85}$	$n^{3,9}$	$n^{3,9}$	n^4
30	$n^{3,3}$	$n^{3,4}$	$n^{3,5}$	$n^{3,55}$	$n^{3,6}$	$n^{3,6}$	$n^{3,6}$
50	$n^{3,1}$	$n^{3,3}$	$n^{3,4}$	$n^{3,45}$	$n^{3,5}$	$n^{3,5}$	$n^{3,25}$
70	$n^{2,3}$	$n^{2,6}$	$n^{2,7}$	$n^{2,75}$	$n^{2,8}$	$n^{2,8}$	$n^{2,9}$

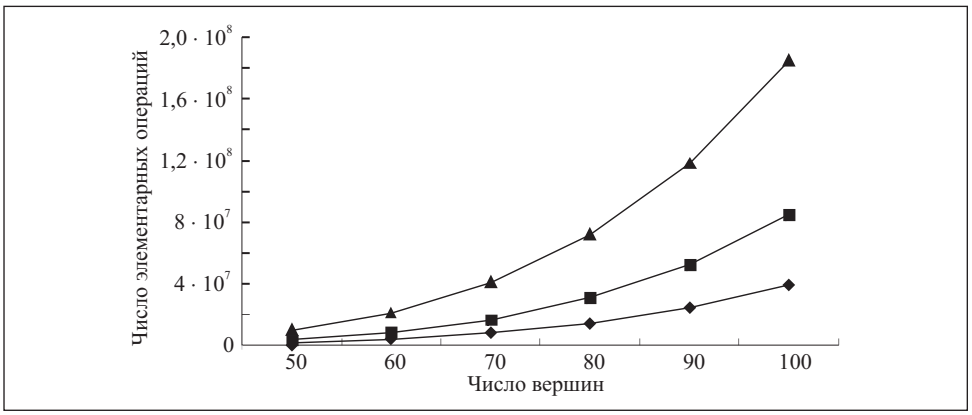

Рис. 4. Зависимость числа элементарных операций, выполняемых процедурой $\boldsymbol{A_0}$, от числа вершин в графе при $\rho = 0{,}3$: ◆ — плотность 30; ■ — плотность 30 с поглощением; ▲ — n^4

два множества, $(X_i^r \mid Y_i^k)$ и $(X_j^r \mid Y_j^k)$, можно объединить при $Y_i^k \subset Y_j^k$ либо $Y_j^k \subset Y_i^k$. Операции объединения позволяют поместить значительное число подмножеств $(X_i^r \mid Y_i^k)$ в Ω, но при этом возрастает число операций сравнения, необходимых для проверки наличия одинаковых множеств Y_i^k и проверки условий $Y_i^k \subset Y_j^k$ и $Y_j^k \subset Y_i^k$.

С увеличением размерности число операций сравнения начинает возрастать пропорционально n^2. Поэтому с увеличением размерности задачи начинает возрастать сложность процедуры $\boldsymbol{A_0}$. На рис. 4 приведены зависимости числа элементарных операций, выполняемых процедурой $\boldsymbol{A_0}$, от размерности решаемой задачи с использованием поглощения подмножеств в Ω и без поглощения. Как видно из рис. 4, выигрыш в 1,1—1,4 раза в быстродействии работы процедуры $\boldsymbol{A_0}$ наблюдается при размерностях $n \le 40$. Дальнейшее увеличение размерности приводит к ухудшению временной сложности процедуры $\boldsymbol{A_0}$ при использовании операции объединения в процессе ее работы. При тестовых проверках графов с числом вершин 100, 200 и 300 при $\rho = 0{,}5$ установлено, что временная сложность работы процедуры не превышает $\boldsymbol{O}\ (n^{5{,}5})$.

Выводы

Предложенная процедура $\boldsymbol{A_0}$ позволяет перечислять НМНМ, однако при этом часть их может быть утеряна. Это обусловлено тем, что в процессе работы процедуры при выборе множеств максимальной мощности может оказаться несколько одинаковых максимальных множеств, из которых будет выбрано одно. Экспериментально установлено, что теряется в среднем не более трех-четырех НМНМ. Однако в случае, когда есть одно

максимальное множество, то оно всегда будет построено, поскольку на всех шагах работы процедуры множество Q, построенное на основе пар вершин, ему принадлежащих, будет доминировать. Поскольку процедура проверяет возможность построения НМНМ на основе всех пар независимых множеств вершин, принадлежащих ему, оно гарантировано будет построено.

Таким образом, предложенная процедура решения задачи при числе вершин, не превышающем 120, и плотностях ребер в диапазоне от 0,067 до 0,9 позволяет решать задачу определения НМНМ за полиномиальное время, и задача может быть решена в масштабе реального времени, что важно для современных систем управления.

СПИСОК ЛИТЕРАТУРЫ

1. *Harary F., Ross I.C.* A Procedure for Clique Detection Using the Group Matrix // Sociometry. — 1957. — Vol. 20. — P. 205—215.
2. *Butenko S., Wilhelm W.E.* Clique-detection models in computational biochemistry and genomics // European Journal of Operational Research. — 2006. — Vol. 173. — P. 1—17.
3. *Raymond J.W., Willett P.* Maximum common subgraph isomorphism algorithms matching chemical structures // Journal of Computer-Aided Molecular Design. — 2002. — Vol. 16. — P. 521—533.
4. *Varmuza K., Penchev P.N., Scsibrany H.* Maximum common substructures of organic compounds exhibiting similar infrared spectra // J. Chem. Inf. Comput. Sci. — 1998. —Vol. 38. — P. 420—427.
5. *Horaud R., Skordas T.* Stereo correspondence through feature grouping and maximal cliques // IEEE Trans. Pattern Anal. Mach. Intell. — 1989. — Vol. 11, № 11.
6. *Pelillo M., Siddiqi K., Zucker S.W.* Matching hierarchical structures using association graphs // IEEE Trans. Pattern Anal. Mach. Intell. — 1999. — Vol. 21, № 11.
7. *Shearer K., Bunke H., Venkatesh S.* Video indexing and similarity retrieval by largest common subgraph detection using decision trees. IDIAP-RR 00-15, Dalle Molle Institute for Perceptual Artificial Intelligence, Martigny, Valais, Switzerland, 2000.
8. *Shirinivas S.G., Vetrivel S., Elango N.M.* Application of graph theory in computer science an overview // International Journal of Engineering Science and Technology. — 2010. — Vol. 2, № 9. — P. 4610—4621.
9. *Селиверстов А.В., Любецкий В.А.* Алгоритм поиска консервативных участков нуклеотидных последовательностей // Информационные процессы. — 2006. — **6**, № 1. — С. 33—36.
10. *Bahadur D.K.C., Akutsu T., Tomita E. et al.* Point matching under non-uniform distortions and protein side chain packing based on efficient maximum clique algorithms // Genome Inform. — 2002. — Vol. 13. — P. 143—152.
11. *Carr R.D., Lancia G., Istrail S.* Branch-and-cut algorithms for independent set problems: integrality gap and application to protein structure alignment. Technical report, Sandia National Laboratories, Albuquerque, NM (US); Sandia National Laboratories, Livermore, CA (US), September 2000.
12. *Harley E., Bonner A., Goodman N.* Uniform integration of genome mapping data using intersection graphs // Bioinformatics. — 2001. — Vol. 17. — P. 487—494.

13. *Samudrala R., Moult J.* A graph-theoretic algorithm for comparative modeling of protein structure // J. Mol. Biol. — 1998. — Vol. 279. — P. 287—302.
14. *Tomita E., Akutsu T., Hayashida M. et al.* Algorithms for computing an optimal protein threading with profiles and distance restraints // Genome Informatics. — 2003. — Vol. 14. — P. 480—481.
15. *Bahadur D.K.C., Tomita E., Suzuki J. et al.* Protein sidechain packing problem: a maximum edge-weight clique algorithmic approach // J. Bioinform Comput. Biol. — 2005. — Vol. 3. — P. 103—126.
16. *Jianer C., Iyad K.A., Ge X.* Improved Parameterized Upper Bounds for Vertex Cover. — Elsevier: Theoretical Computer Science. — 2010. —Vol. 411. — P. 3736—3756.
17. *Bron C., Kerbosch J.* Algorithm 457: Finding All Cliques of an Undirected Graph // Comm. of ACM. — 1973. — Vol. 16. — P. 575—577.
18. *Fomin F.V., Grandoni F., Kratsch D.* Measure and conquer: a simple O(20.288n) independent set algorithm//Proc. of the 17th Annual ACM-SIAM Symp. on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22- 26, 2006, P. 18—25. ACM Press, 2006.
19. *Robson J.M.* Algorithms for maximum independent set // Journal of Algorithms. — 1986. — Vol. 7, No. 3. — P. 425—440.
20. *Tarjan R.E., Trojanowski A.E*. Finding a Maximum Independent Set // SIAM Journal on Computing. — 1977. — Vol. 6. —P. 537—546.
21. *Moon J.W., Moser L.* On cliques in graphs // Israel J. Math. — 1965. — Vol. 3. — P. 23—28.
22. *Pardalos P.M., Xue J.* The maximum clique problem // Journal of Global Optimization. — 1994. — Vol. 4. — P. 301—328.
23. *Олемской И.В., Фирюлина О.С.* Алгоритм поиска наибольшего независимого множества // Вестн. С.-Пб. ун-та. Сер. 10: Прикладная математика, информатика, процессы управления. — 2014. — Вып. 1. — С. 81—91.
24. *Плотников А.Д.* Эвристический алгоритм для поиска наибольшего независимого множества // Кибернетика и системный анализ. — 2012. — № 5. — С. 41—48.
25. *Листровой С.В.* Метод перечисления максимальних независимых множеств в произвольных неориентированных графах // Электрон. моделирование. — 2014. — **36**, № 1. — С. 3—17.
26. *Listrovoy S.V., Minukhin S.V.* General Approach to Solving Optimization Problems in Distributed Computing Systems and Theory of Intelligence Systems Construction // Journal of automation and information sciences. — 2010. — Vol. 42, No. 3. — P. 30—46.
27. *Листровой С.В., Минухин С.В.* Общий подход к решению задач оптимизации в распределенных вычислительных системах и теории построения интеллектуальных систем // Проблемы управления и информатика. — 2010. — № 2. — С. 65—82.

Поступила 10.01.17
после доработки 22.03.17

REFERENCES

1. Harary, F. and Ross, I.C. (1957), “A procedure for clique detection using the group matrix”, *Sociometry*, Vol. 20, pp. 205-215.
2. Butenko, S. and Wilhelm, W.E. (2006), “Clique-detection models in computational biochemistry and genomics”, *European Journal of Operational Research*, Vol. 173, pp. 1-17.
3. Raymond, J.W. and Willett, P. (2002), “Maximum common subgraph isomorphism algorithms matching chemical structures”, *Journal of Computer-Aided Molecular Design*, Vol. 16, pp. 521-533.

4. Varmuza, K., Penchev, P.N. and Scsibrany, H. (1998), "Maximum common substructures of organic compounds exhibiting similar infrared spectra", *J. Chem. Inf. Comput. Sci*., Vol. 38, pp. 420-427.
5. Horaud, R. and Skordas, T. (1989), "Stereo correspondence through feature grouping and maximal cliques", *IEEE Trans. Pattern Anal. Mach. Intell.,* Vol. 11, no. 11.
6. Pelillo, M., Siddiqi, K. and Zucker, S.W. (1999), "Matching hierarchical structures using association graphs", *IEEE Trans. Pattern Anal. Mach. Intell*., Vol. 21, no. 11.
7. Shearer, K., Bunke, H. and Venkatesh, S. (2000), Video indexing and similarity retrieval by largest common subgraph detection using decision trees, IDIAP-RR 00-15, Dalle Molle Institute for Perceptual Artificial Intelligence, Martigny, Valais, Switzerland.
8. Shirinivas, S.G., Vetrivel, S. and Elango, N.M. (2010), "Application of graph theory in computer science an overview", *International Journal of Engineering Science and Technology*, Vol. 2, no. 9, pp. 4610-4621.
9. Seliverstov, A.V. and Lyubetskiy, V.A. (2006), "Algorithm for the search for conservative segments of nucleotide sequences", *Informatsionnyie protsessy*, Vol. 6, no. 1, pp. 33-36.
10. Bahadur, D.K.C., Akutsu, T., Tomita, E. and et al. (2002), "Point matching under non-uniform distortions and protein side chain packing based on efficient maximum clique algorithms", *Genome Inform*., Vol. 13, pp. 143-152.
11. Carr, R.D., Lancia, G. and Istrail, S. (2000), Branch-and-cut algorithms for independent set problems: integrality gap and application to protein structure alignment. Technical report, Sandia National Laboratories, Albuquerque, NM (US); Sandia National Laboratories, Livermore, CA (US).
12. Harley, E., Bonner, A. and Goodman, N. (2001), "Uniform integration of genome mapping data using intersection graphs", *Bioinformatics*, Vol. 17, pp. 487-494.
13. Samudrala, R. and Moult, J. (1998), "A graph-theoretic algorithm for comparative modeling of protein structure", *J. Mol. Biol*., Vol. 279, pp. 287-302.
14. Tomita, E., Akutsu, T., Hayashida, M. and et al. (2003), "Algorithms for computing an optimal protein threading with profiles and distance restraints", *Genome Informatics*, Vol. 14, pp. 480-481.
15. Bahadur, D.K.C., Tomita, E., Suzuki, J. and et al. (2005), "Protein sidechain packing problem: a maximum edge-weight clique algorithmic approach", *J. Bioinform Comput. Biol*., Vol. 3, pp. 103-126.
16. Jianer, C., Iyad, K.A. and Ge, X. (2010), "Improved parameterized upper bounds for vertex cover", *Elsevier: Theoretical Computer Science*, Vol. 411, pp. 3736-3756.
17. Bron, C. and Kerbosch, J. (1973), "Algorithm 457: Finding all cliques of an undirected graph", *Comm. of ACM*, Vol. 16, pp. 575-577.
18. Fomin, F.V., Grandoni, F. and Kratsch, D. (2006), "Measure and conquer: a simple O(20.288n) independent set algorithm", *Proceedings of the 17th Annual ACM-SIAM Symp. on Discrete Algorithms*, SODA 2006, Miami, Florida, USA, January 22-26, 2006.
19. Robson, J.M. (1986), "Algorithms for maximum independent set", *Journal of Algorithms*, Vol. 7, no. 3, pp. 425-440.
20. Tarjan, R.E. and Trojanowski, A.E. (1977), "Finding a maximum independent set", *SIAM Journal on Computing*, Vol. 6, pp. 537-546.
21. Moon, J.W. and Moser, L. (1965), "On cliques in graphs", *Israel J. Math*., Vol. 3, pp. 23-28.
22. Pardalos, P.M. and Xue, J. (1994), "The maximum clique problem", *Journal of Global Optimization*, Vol. 4, pp. 301-328.
23. Olemskoy, I.V. and Firyulina, O.S. (2014), "The algorithm for finding the largest independent set", *Vestnik, S.Pb. Universiteta*, *Ser. 10: Prikladnaya matematika, informatika, protsessy upravleniya*, Iss. 1, pp. 81-91.

24. Plotnikov, A.D. (2012), “Heuristic algorithm for searching for the largest independent set”, *Kibernetika i sistemnyi analiz*, no. 5, pp. 41-48.
25. Listrovoy, S.V. (2014), “The method of enumeration of maximal independent sets in arbitrary non-oriented graphs”, *Elektronnoe modelirovanie*, Vol. 36, no. 1, pp. 3-17.
26. Listrovoy, S.V. and Minukhin, S.V. (2010), “General approach to solving optimization problems in distributed computing systems and theory of intelligence systems construction”, *Journal of Automation and Information Sciences*, Vol. 42, no. 3, pp. 30-46.
27. Listrovoy, S.V. and Minukhin, S.V. (2010), “A general approach to solving optimization problems in distributed computing systems and the theory of constructing intelligent systems”, *Problemy upravleniya i informatiki*, no. 2, pp.65-82.

Received 10.01.17;
after revision 22.03.17

S.V. Listrovoy, A.V. Sidorenko, E.S. Listrovaya

THE METHOD OF DETERMINING THE LARGEST MAXIMAL INDEPENDENT SETS OF VERTICES OF UNDIRECTED GRAPH

A method of search for the largest maximal independent sets of an undirected connected graph is proposed that allows one to solve the problem of determining the largest maximal independent sets in polynomial time with the number of vertices in the graph not exceeding 120 and the density of edges in the range from 0.067 to 0.9. with a further increase in the number of vertices and a decrease in the density of edges in the graph, the algorithm has an exponential complexity that does not exceed at an average $O(2^{0,4n})$, which tends to the decrease with increasing the edge density in the graph, where n is the number of vertices in the graph.

K e y w o r d s: maximal independent set, click, the top cover.

ЛИСТРОВОЙ Сергей Владимирович, д-р техн. наук, профессор Украинского государственного университета железнодорожного транспорта (г. Харьков). В 1972 г. окончил Харьковское высшее военное командно-инженерное училище. Область научных исследований — задачи дискретной оптимизации и теории графов и их приложения к анализу вычислительных систем и сетей.

СИДОРЕНКО Андрей Владимирович, вед. инженер-программист фирмы Samsung Electronics Ukraine Company, LLC Samsung R&D Institute Ukraine (г. Киев). В 2001 г. окончил Харьковский военный университет. Область научных исследований — задачи дискретной оптимизации и теории графов и их приложения к анализу вычислительных систем и сетей.

ЛИСТРОВАЯ Елена Сергеевна, канд. техн. наук, доцент кафедры экономики и маркетинга Национального аэрокосмического университета им. Н.Е. Жуковского (г. Харьков), который окончила в 1998 г. Область научных исследований — применение информационных систем в экономической сфере деятельности.