УДК 519.658

АЛГОРИТМ ВЫДЕЛЕНИЯ БЛОЧНО-ДРЕВОВИДНОЙ СТРУКТУРЫ В РАЗРЕЖЕННЫХ ЗАДАЧАХ ДИСКРЕТНОЙ ОПТИМИЗАЦИИ

© Д. В. Лемтюжникова, А. В. Свириденко, О. А. Щербина

Таврический национальный университет им. В. И. Вернадского
факультет математики и информатики
пр-т Вернадского, 4, г. Симферополь, 95007, Украина
e-mail: *darabbt@gmail.com, oleks.sviridenko@gmail.com, oshcherbina@gmail.com*

Abstract. Authors propose an algorithm of computing block-tree structure for sparse matrices. Finkelshtein's algorithm for constructing quasiblock structures is modified and implemented. Preliminary benchmarking with test problems was done. Quality of computed quasiblock structures is investigated.

Введение

Многие задачи дискретной оптимизации (ДО), возникающие на практике, содержат огромное число переменных и/или ограничений, что создает сложности при попытке решения этих задач с помощью современных решателей. Эти задачи зачастую имеют разреженные матрицы ограничений [5], причем блочная структура в них не всегда явно выделена. Использование разреженности матриц ограничений в задачах ДО может весьма эффективно организовать процесс поиска решения задачи ДО. *Ф. Гловер* заметил в [1], что «... разработка специальных вычислительных методов, ориентированных на операции с такими разреженными матрицами ... может в ближайшем будущем в определенной степени изменить существующие представления о непреодолимых сложностях решения отдельных классов задач целочисленного программирования, имеющих большую размерность».

Перспективными декомпозиционными методами, использующими разреженность матрицы ограничений задач ДО, являются локальные элиминационные алгоритмы [10], включающие локальные алгоритмы декомпозиции [7], алгоритмы несериального динамического программирования (НСДП) [14], [24], алгоритмы сегментной элиминации [16]. Локальные элиминационные алгоритмы относятся к классу локальных алгоритмов, общая теория которых для решения ряда важных задач дискретной математики была развита Ю. И. Журавлевым [2].

Теоретико-графовые методы и интерпретации [3] играют важную роль при выделении блочных структур в разреженных задачах ДО. Для выделения специальных структур задач ДО представляются перспективными такие графовые декомпозиционные подходы, как методы древовидной декомпозиции (ДД) [9].

Задача поиска наилучшей ДД состоит в нахождении *триангуляции* графа (триангуляция графа — его вложение в хордальный граф, т.е. в граф, каждый цикл которого длиной ≥ 4 обладает хордой[1]) с минимальным размером наибольшей клики. Актуальность решения этой задачи обусловлена тем, что многие NP-полные задачи разрешимы за полиномиальное время, если они рассматриваются на графах с ограниченной древовидной шириной [15]. К сожалению, как задача поиска триангуляции с минимальным пополнением, так и задача поиска наилучшей ДД являются NP-полными [12, 27]. Однако для обеих задач имеются полиномиально вычислимые альтернативы нахождения так называемой *минимальной триангуляции* [20].

Определение 1. Триангуляция H данного графа G минимальна, если ни одна триангуляция G не является подграфом H.

Для известных алгоритмов LexBFS[2] [23] и MCS[3] [26] разработаны модификации, позволяющие находить минимальные триангуляции: Lex-M [23] и MCS-M [13]. Результаты экспериментального исследования алгоритмов LexBFS, MCS и Lex-M описаны в работе [21].

Выделение древовидной декомпозиции может быть осуществлено и с помощью элиминационной игры [22].

Частным случаем ДД является *блочно-древовидная структура*, введенная в работе [8] и подробно рассматриваемая ниже.

Финкельштейн [6] предложил алгоритм выделения *квазиблочной* структуры в разреженной матрице. Отметим, что алгоритм Финкельштейна тесно связан с *корневыми уровневыми структурами* [11, 19].

При применении локального алгоритма для решения [8] разреженных задач ДО необходимо выделение блочно-древовидной структуры, алгоритм построения которой для разреженных матриц до настоящего времени был неизвестен. Алгоритм Финкельштейна выделения квазиблочных структур в разреженных матрицах также не использовался, авторам неизвестны попытки его программной реализации, исследование качества полученных с его помощью квазиблочных структур не производилось.

Настоящая работа посвящена разработке алгоритма построения блочно-древовидных структур для разреженных матриц, программной реализации алгоритма Финкельштейна и исследованию качества полученных с его помощью квазиблочных структур.

[1] хорда — ребро, соединяющее 2 несмежные вершины цикла

[2] LEX-BFS — лексикографический поиск в ширину

[3] MCS (Maximum Cardinality Search) — поиск по максимальной степени

1. Структура задачи дискретной оптимизации

Рассмотрим задачу ДО с ограничениями

$$F(x_1, x_1, \ldots, x_n) = \sum_{k \in K} f_k(Y^k) \to \max \tag{1}$$

при ограничениях

$$g_i(X^i) \sim 0,\ i \in M = \{1, 2, \ldots, m\}, \tag{2}$$

$$x_j \in D_j,\ j \in N = \{1, 2, \ldots, n\}, \tag{3}$$

где $Y^k \subseteq \{x_1, x_2, \ldots, x_n\}$, $K = \{1, \ldots, t\}$, $X^i \subseteq \{x_1, x_2, \ldots, x_n\}$, $i \in M$; — одно из отношений $\leq$, $\geq$, $=$, D_j — конечное множество возможных значений переменной $x_j \in D_j$, $j \in N = \{1, 2, \ldots, n\}$. Функции $f_k (k \in K)$ называются *компонентами целевой функции.*

Введем необходимые для дальнейшего изложения понятия.

Определение 2. Две переменные x и y *взаимосвязаны* в задаче ДО с ограничениями, если они появляются вместе в одной компоненте ЦФ или в одном и том же ограничении (другими словами, если переменные x, y входят одновременно во множество X^i или во множество Y^k.

Определение 3. *Графом взаимосвязей* задачи ДО называется неориентированный граф $G = (X, E)$, для которого:

- множество вершин X соответствует множеству переменных X задачи ДО;
- две вершины графа соединены ребром тогда и только тогда, когда соответствующие им переменные взаимосвязаны.

Граф взаимосвязей переменных в литературе называется также *графом ограничений* [24].

В дальнейшем будем отождествлять переменные задачи ДО с вершинами графа взаимосвязей. Основным понятием в теории локальных алгоритмов является понятие *близости элементов*, определяемое понятием *окрестности.*

Определение 4. Две вершины x и y в графе $G = (X, E)$ называются *соседними*, если $x, y \in E$.

Определение 5. Множество переменных, соседних с переменной $x \in X$ в графе $G = (X, E)$, обозначается $Nb_G(x)$(или $Nb(x)$) и называется *окрестностью* переменной x.

Пусть $\Omega_1 = (S_1, U_1),\ \Omega_2 = (S_2, U_2), \ldots,\ \Omega_k = (S_k, U_k)$ — система окрестностей некоторых переменных $x_{j_1}, \ldots, x_{j_k}$ задачи ДО (1)–(3), где S_r, U_r — соответственно множества индексов переменных и ограничений для r-й окрестности, $r = 1, \ldots, k$, причём

$$\bigcup_{r=1}^{k} U_r = M = \{1, \ldots, m\}, \tag{4}$$

$$\bigcup_{r=1}^{k} S_r = N = \{1, \ldots, n\}, \tag{5}$$

$$U_{r_1} \bigcap U_{r_2} = \emptyset, r_1 \neq r_2, \tag{6}$$

$$S_{r_1} \bigcap S_{r_2} \bigcap S_{r_3} = \varnothing, \tag{7}$$

для любой тройки индексов r_1, r_2, r_3.

Введем понятие графа инцидентности (или графа пересечений) G_Ω окрестностей. Граф пересечений окрестностей G_Ω имеет вершины $\nu_1, \nu_2, \ldots, \nu_k$, соответствующие выделенным окрестностям $\Omega_1,\ \Omega_2, \ldots,\ \Omega_k$, причем две вершины графа ν_{r_1} и ν_{r_2}, отвечающие окрестностям Ω_{r_1} и Ω_{r_2}, соединены ребром $(r_1,\ r_2)$, если $S_{r_1} \bigcap S_{r_2} \neq \varnothing$.

Определение 6. *Блочно-древовидной* (БД) назовем задачу ДО[4], для которой можно выделить систему окрестностей $\Omega_1,\ \Omega_2, \ldots,\ \Omega_k$, удовлетворяющую свойствам (4)–(7), причем граф пересечений G_Ω окрестностей является деревом.

Замечание. Частным случаем БД структуры является квазиблочная структура [7], для которой граф пересечений окрестностей (или блоков) является цепью. Отметим, что блочно-древовидная структура является частным случаем древовидной декомпозиции.

1.1. **Гиперграф и двойственный граф.** Введем гиперграфовое представление задачи ДО, которое является наиболее адекватным представлением структуры задачи ДО. В гиперграфовом представлении множество вершин гиперграфа H соответствует множеству переменных X задачи ДО, а гиперребра гиперграфа образованы подмножествами взаимосвязанных переменных.

Часто вместо гиперграфового представления используется более наглядное представление структуры задачи ДО с помощью *двойственного* и *первичного* графов. Первичный граф — это описанный выше *граф взаимосвязей* задачи ДО.

[4]Будем говорить, что описанная задача ДО имеет блочно-древовидную структуру.

Определение 7. *Двойственным графом* гиперграфа $H = (V, S)$ называется граф, вершины которого соответствуют гиперребрам гиперграфа, причем пара таких вершин соединяется ребром в двойственном графе, если они имеют общие вершины из V.

2. Алгоритм Финкельштейна выделения квазиблочной структуры

Финкельштейн [6] предложил следующий алгоритм выделения квазиблочной структуры в разреженной матрице. Пусть $H_{m\times n}$ — разреженная матрица, p — индекс некоторой переменной $x_p, p \in \{1, 2, \dots, n\}$. Найдем окрестности переменной x_p последовательно возрастающих порядков, т.е.

$$S_0^{'} = \{x_p\},\, U_{r+1}^{'} = U(S_r^{'}(x_p)), \quad r \geq 0; \tag{8}$$

$$S_{r+1}^{'} = S(U_r^{'}(p)), \quad r \geq 1 \tag{9}$$

причем процесс выделения окрестностей продолжаем до тех пор, пока при некотором $r = k$ не выполнится условие $S_{k+1}^{'}(x_p) = S_k^{'}(x_p)$. Далее находится разбиение множества индексов ограничений, входящих в блоки:

$$U_r = \begin{cases} U_1^{'}(x_p), & r = 1, \\ U_r^{'}(x_p) \backslash U_{r-1}^{'}(x_p), & 1 < r < k, \\ U_k^{'}(x_p) \bigcup U_{k+1}^{'}(x_p) \backslash U_{k-1}^{'}(x_p), & r = k, \end{cases}$$

и находятся следующие множества переменных:

$$S_{r,r+1} = S(U_r(x_p)) \bigcap S(U_{r+1}(x_p)),$$

$$S_r = \begin{cases} S\left(U_1\left(x_p\right)\right) \backslash S_{12}, & r = 1, \\ S\left(U_1\left(x_p\right)\right) \backslash \left(S_{r-1,r} \cup S_{r,r+1}\right), & 1 < r < k, \\ S\left(U_k\left(x_p\right)\right) \backslash S_{k-1,k}, & r = k. \end{cases}$$

Описанный алгоритм находит n разбиений матрицы H на квазиблоки (не обязательно различных), хотя возможных разбиений существует гораздо больше.

Стоит отметить, что в рамках данного исследования алгоритм Финкельштейна был усовершенствован. Так, если находились вырожденные блоки[5], то они сливались вместе, т.е. вырожденный блок становился частью впереди идущего блока.

[5]Блок B_r считается вырожденным, если $S_r = \varnothing$.

Также был установлен максимальный размер сепаратора. При превышении допустимого размера сепаратора в блоке, выполнялась вышеупомянутая процедура слияния блоков. Предусмотрена также возможная несвязность графа ограничений.

3. Метод построения блочно-древовидной структуры

Как показано в [8], переход от блочно-древовидной структуры к квазиблочной путем объединения «слоев» дерева нецелесообразен. В связи с этим представляет интерес выделение блочно-древовидной структуры в разреженном графе с помощью модификации описанного выше алгоритма Финкельштейна.

Структура задачи ДО, как отмечено выше, может быть описана с помощью *двойственного* (или *реберного* [3]) графа. Полученная с помощью алгоритма Финкельштейна (см. раздел 2) квазиблочная структура $\{U_1, \ldots, U_k\}$ задачи ДО на двойственном графе D имеет вид цепи. Далее рассмотрим граф $D_s = D\left(\bigcup_{r=s}^{k} U_r\right)$, индуцируемый множеством $\bigcup_{r=s}^{k} U_r$ в двойственном графе D $(s = 2, \ldots, k)$ и выделим в нем компоненты связности $C_1, \ldots, C_p$. Далее каждый блок U_s заменяется на подблоки $U_s \bigcap C_1, \ldots, U_s \bigcap C_p$. В результате получается блочно-древовидная структура.

Рассмотрим алгоритм выделения компонент связности [17] в неориентированном графе согласно [4]. В основу рассматриваемого алгоритма выделения компонент связности положен предложенный R. Tarjan [25] алгоритм поиска в глубину на графе $G(X, E)$. Алгоритм находит вектор $Mark$[x] меток вершин $x \in X$ графа. Элементу $Mark$[x] присваивается общий номер той компоненты, которой принадлежит вершина $x \in X$. Сложность алгоритма, как и алгоритма поиска в глубину, составляет $O(|X| + |E|)$.

Пример 1. Рассмотрим построение блочно-древовидной структуры для задачи ДО, состоящей из следующих ограничений:

$$\begin{aligned}
B_1 &: x_4 + x_5 + 4x_6 + 2x_7 \le 4,\\
B_2 &: 3x_1 + 2x_2 + 2x_3 + 3x_4 \le 6,\\
B_3 &: 3x_7 + 8x_8 + 3x_9 + x_{10} \le 8,\\
B_4 &: x_9 + 2x_{11} + 3x_{12} \le 4,\\
B_5 &: x_{10} + x_{13} + 2x_{14} \le 2,\\
&\quad x_j = 0, 1, \quad j = 1, 2, \ldots, 14.
\end{aligned}$$

Построим двойственный граф этой задачи ДО и выделим квазиблочную структуру $\{U_1, U_2, U_3\}$ с помощью алгоритма Финкельштейна (рис. 2).

```
Цикл пока v ∈ X
    Mark[v] = 0;    // Начальная установка
end Цикл пока
count = 0;    // Счетчик числа компонент
Цикл пока v ∈ X
    Если Mark[v] = 0 то
        count = count + 1;
        Component(v, count);
    end Если
end Цикл пока
procedure COMPONENT(x, count)
    Mark[x] = count;
    Цикл пока v ∈ Nb(x)
        Если Mark[v] = 0 то
            Component(v, count);
        end Если
    end Цикл пока
end procedure
```

Рис. 1. Алгоритм выделения связных компонент неориентированного графа.

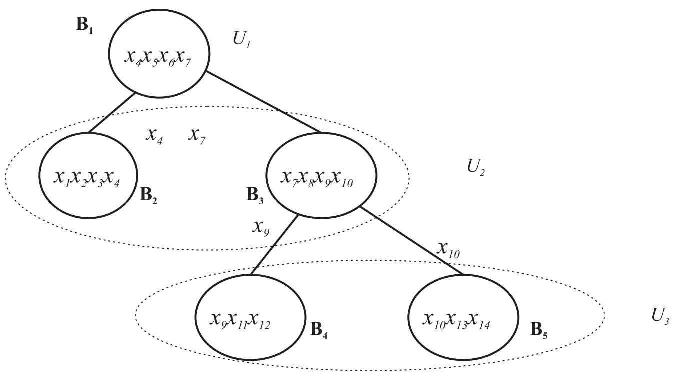

Рис. 2. Выделение квазиблочной структуры $\{U_1, U_2, U_3\}$ с помощью алгоритма Финкельштейна

Найдем компоненты связности с помощью описанного выше алгоритма в графе $D_2 = D(U_2 \bigcap U_3)$, индуцируемом множеством $U_2 \bigcap U_3$, в двойственном графе D (рис. 3). Далее выделим компоненты связности C_1' и C_2' в графе $D_3 = D(U_3)$, индуцируемом множеством U_3 в двойственном графе D (рис. 4).

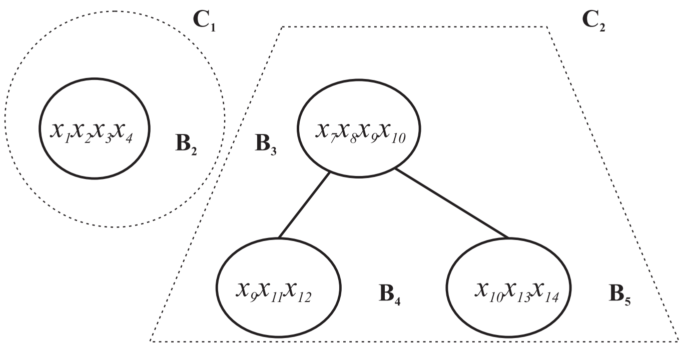

Рис. 3. Выделение компонент связности C_1 и C_2 в графе $D_2 = D(U_2 \bigcup U_3)$, индуцируемом множеством $U_2 \bigcup U_3$ в двойственном графе D.

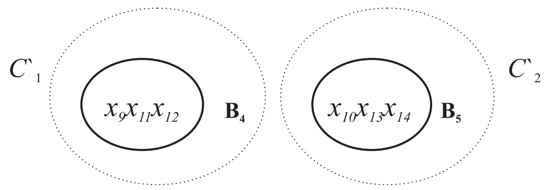

Рис. 4. Выделение компонент связности C_1' и C_2' в графе $D_3 = D(U_3)$, индуцируемом множеством U_3 в двойственном графе D.

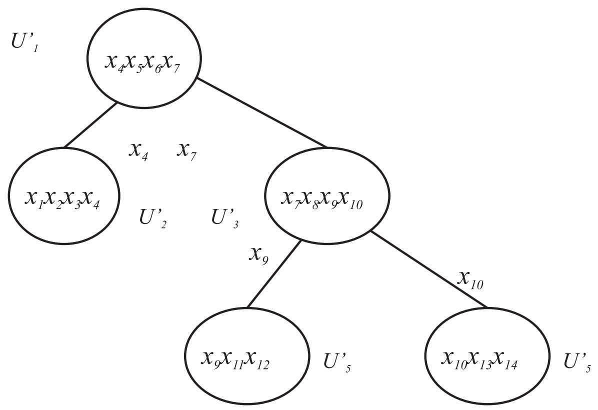

Рис. 5. Блочно-древовидная структура $\{U_1', U_2', U_3', U_4', U_5'\}$.

Заменяя каждый блок U_s на подблоки $U_s \bigcap C_1, \ldots, U_s \bigcap C_p$ для каждого разбиения на связные компоненты, получим в результате блочно-древовидную структуру (рис. 5).

4. Описание вычислительного эксперимента

Алгоритм Финкельштейна реализован с помощью класса FinkelsteinQBDecomposition на языке программирования C++ и является частью библиотеки LES[6].

Все вычисления проводились на базе процессора Intel Core 2 Duo @2.66 GHz, 2 Gb ОЗУ и операционной системы Linux, версия ядра 2.6.35-24-generic.

Для проведения вычислительного эксперимента были построены пять групп тестовых задач 'dubois', 'bridge', 'adder', 'pret' и 'grid' для 17 гиперграфов из библиотеки CSP[7] [18], причем каждая тестовая задача решалась с помощью исходного и модифицированного алгоритмов Финкельштейна. Результаты вычислительного эксперимента приведены в таблице 1.

Таблица 1. Результаты декомпозиции задач ДО с помощью исходного и модифицированного алгоритма Финкельштейна.

Задача	Исходный алгоритм		Модифицированный алгоритм	
	Кол-во блоков	Максимальный сепаратор	Кол-во блоков	Максимальный сепаратор
adder_15	31	4	16	4
adder_50	61	6	51	6
adder_99	121	6	101	6
dubois23	23	2	23	2
dubois30	30	2	30	2
dubois50	50	2	50	2
dubois100	98	6	2	4
pret60_25	8	10	7	10
pret60_60	8	10	7	10
pret150_25	15	12	11	10
pret150_75	15	12	11	10
grid2d_10	8	10	2	3
grid10	16	10	3	9
grid3d_4	5	12	2	6
bridge_15	21	10	4	10
bridge_50	56	10	5	8
bridge_75	78	9	4	9

[6]Local Elimination Solver https://github.com/d2rk/les/

[7]CSP (Constraint Satisfaction Problem) — задача удовлетворения ограничений.

Следует отметить, что в качестве первой переменной, используемой алгоритмом Финкельштейна, для которой находилась окрестность, выбиралась x_1, т.е. $p = \{1\}$.

Анализируя результаты, полученные в таблице 1, нетрудно увидеть преимущества модифицированного алгоритма Финкельштейна по сравнению с исходным алгоритмом Финкельштейна. Так, во многих задачах были существенно уменьшены количество построенных блоков и размеры сепараторов.

Рассмотрим некоторые результаты для тестовых задач более подробно. Можно отметить задачу 'dubois100', в которой существенно сократилось количество построенных блоков и максимальный сепаратор уменьшился на 2. Для задач 'pret150_25' и 'pret150_75' существенно уменьшилось количество построенных блоков.

Заключение

В статье предложен алгоритм выделения блочно-древовидной структуры для разреженных матриц. Реализован в виде программы на C++ и протестирован алгоритм Финкельштейна для выделения квазиблочных структур в разрежённых матрицах. В качестве тестовых задач взяты пять групп задач: 'dubois','bridge','adder','pret' и 'grid'. Произведен сравнительный эксперимент для модифицированной и исходной версий алгоритма, показавший существенное уменьшение количества построенных блоков и размеров сепараторов для модифицированного алгоритма Финкельштейна. В дальнейшем планируется проведение сравнительного эксперимента для алгоритмов нахождения блочно-древовидной структуры и древовидной декомпозиции.

Список літератури

1. Гловер Ф. Целочисленное программирование и комбинаторика / Ф. Гловер // Исследование операций. Т.1: Методологические основы и математические методы. — М.: Мир, 1981. — С. 122–182.
2. Журавлев Ю. И. Избранные научные труды / Ю. И. Журавлёв. — М.: Магистр, 1998. — 420 с.
3. Лекции по теории графов / [Емеличев В. А., Мельников О. И., Сарванов В. И., Тышкевич Р. И.] / Изд. 2-е, испр. — М.: Книжный дом «ЛИБРОКОМ», 2009. — 392 с.
4. Иванов Б. Н. Дискретная математика. Алгоритмы и программы: Учеб. пособие / Б. Н. Иванов. — М.: Лаборатория Базовых Знаний, 2003. — 288 с.
5. Писсанецки С. Технология разреженных матриц / С. Писсанецки. — М: Мир, 1988. — 356 с.
6. Финкельштейн Ю. Ю. Методы решения некоторых дискретных задач математического программирования: Дисс. … канд. физ.-матем. наук / Ю. Ю. Финкельштейн. — М., 1966. — 110 с.

7. Щербина О. А. О локальных алгоритмах решения квазиблочных задач дискретного программирования / О. А. Щербина // Проблемы кибернетики. — М.: Наука, 1983. — Вып. 40. — С. 171–200.
8. Щербина О. А. Локальные алгоритмы для блочно-древовидных задач дискретного программирования / О. А. Щербина // Журн. вычисл. математики и матем. физики. — 1985. — Т. 25, N 8. — С. 1143–1154.
9. Щербина О. А. Древовидная декомпозиция и задачи дискретной оптимизации (обзор) / О. А. Щербина // Кибернетика и системный анализ. — 2007. — № 4. — С. 102–118.
10. Щербина О. А. Локальные элиминационные алгоритмы решения разреженных дискретных задач / О. А. Щербина // Журнал вычислительной математики и математической физики. — 2008. — Т. 48, N 1. — С. 161—177.
11. Arany I. An improved method for reducing the bandwidth of sparse symmetric matrices / I. Arany, W. F. Smyth, L. Szoda // Proc. of IFIP Congress "Information Processing 71", North-Holland, Amsterdam, 1972. — P. 1246–1250.
12. Arnborg S. Complexity of finding embeddings in a k-tree / S. Arnborg, D. G. Corneil, A. Proskurowski // SIAM J. Alg. Disc. Meth. — 1987. — V. 8. — P. 277–284.
13. Maximum cardinality search for computing minimal triangulations of graphs / [A. Berry, J. Blair, P. Heggernes, B. Peyton] // Algorithmica. — 2004. — V. 39. — P. 287–298.
14. Bertele U. Nonserial Dynamic Programming / U. Bertele, F. Brioschi. — New York: Academic Press, 1972. — 235 p.
15. Bodlaender H. L. A tourist guide through treewidth / H. L. Bodlaender // Acta Cybernetica. — 1993. — V. 11. — P. 1-21.
16. Dechter R. Constraint Processing / R. Dechter. — Morgan Kaufmann, 2003. — 481 p.
17. Even S. Graph Algorithms / S. Even. — Rockville: Computer Science Press, 1983. — 249 p.
18. A CSP hypergraph library: Technical Report / [T. Ganzow, G. Gottlob, N. Musliu, M. Samer]. — Vienna: Technische Universität Wien, 2005. — 10 p. — DBAI-TR-2005-50.
19. George J. A. Computer solution of large sparse positive definite systems / J. A. George, J. W. H. Liu. — Englewood Cliffs: Prentice-Hall Inc., 1981. — 324 p.
20. Heggernes P. Minimal triangulations of graphs: A survey / P. Heggernes // Discrete Mathematics. — 2006. — V. 306, № 3. — P. 297-317.
21. Koster A. M. C. A. Treewidth: Computational experiments / A. M. C. A. Koster, H. L. Bodlaender, S. P. M. van Hoesel // Electronic Notes in Discrete Mathematics. — 2001. — V. 8. — P. 54–57.
22. Parter S. The use of linear graphs in Gauss elimination / S. Parter // SIAM Review. — 1961. — V.3. — P. 119–130.
23. Rose D. Algorithmic aspects of vertex elimination on graphs / D. Rose, R. E. Tarjan, G. Lueker // SIAM J. Comput. — 1976. — V. 5. — P. 266–283.
24. Shcherbina O. Nonserial dynamic programming and tree decomposition in discrete optimization / O. Shcherbina // Proceedings of Int. Conference on Operations Research "Operations Research 2006" (Karlsruhe, 6-8 September, 2006). — Berlin: Springer Verlag, 2007. — P. 155–160.

25. Tarjan R. E. Depth first search and linear graph algorithms / R. E. Tarjan // SIAM J. Comput. — 1972. — V. 1. — P. 146–160.
26. Tarjan R.E. Simple linear-time algorithms to test chordality of graphs, test acyclity of hypergraphs, and selectively reduce acyclic hypergraphs / R.E. Tarjan, M. Yannakakis // SIAM J. Comput. — 1984. — V. 13. — P. 566–579.
27. Yannakakis M. Computing the minimum fill-in is NP-complete / M. Yannakakis // SIAM J. Alg. Disc. Meth. — 1981. — V. 2. — P. 77—79.

Статья поступила в редакцию 02.06.2012