ходит его. Алгоритм QWC может быть довольно легко перенесен на недорогую аппаратную платформу, способную в режиме реального времени обрабатывать видео-поток телевидения высокой четкости.

Разработка алгоритма сжатия поддерживается по программе «Участник молодежного научно-инновационного конкурса» («У.М.Н.И.К.») Фондом содействия развитию малых форм предприятий в научно-технической сфере.

СПИСОК ЛИТЕРАТУРЫ

1. Ватолин Д., Москвин А., Петров О. Сравнение кодеков изображений стандарта JPEG2000 и Windows Media Photo (новое название: Microsoft HD photo) [Электронный ресурс]. – режим доступа: http://compression.ru/video/codec_comparison/wmp_codecs_comparison.html (20.06.2007).
2. Ватолин Д., Ратушняк А., Смирнов М., Юкин В. Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео. – М.: ДИАЛОГ-МИФИ, 2002. – 384 с.
3. Ватолин Д. Программный продукт оценки качества восстановленного изображения MSU Quality Measure [Электронный ресурс]. – режим доступа: http://compression.ru/video/quality_measure/video_measurement_tool.html (20.06.2007).
4. Сидоров Д.В., Осокин А.Н. Аппаратно-ориентированный субполосный квантователь для вейвлет-сжатия полутоновых изображений // Молодежь и современные информационные технологии: Сб. трудов V Всеросс. научно-практ. конф. студентов, аспирантов и молодых ученых. Томск, 27 февраля – 1 марта 2007 г. – Томск: Изд-во ТПУ, 2007. – С. 446–448.
5. Уэлстид С. Фракталы и вейвлеты для сжатия изображений в действии. – М.: Триумф, 2003. – 320 с.
6. Rabbani M., Santa Cruz D. The JPEG2000 Still-Image Compression Standart. – Lausanne: Swiss Federal Institute of Technology (EPFL), 2002. – 246 p.
7. Sagetong P., Ortega A. Analytical model-based bit allocation for wavelet coding with applications to multiple description coding and region of interest coding // Proc. IEEE Int. Conf. on Multimedia and Expo (ICME). Tokyo, Japan August 2001. – Los Angeles: SIPI Department of EES University of Southern California, 2001. – P. 300–304.
8. Штойер Р. Многокритериальная оптимизация: теория, вычисления и приложения: пер. с англ. / Р. Штойер; под ред. А.В. Лотова. – М.: Радио и связь, 1992. – 504 с.

Поступила 12.10.2007 г.

УДК 681.3.06

МАТРИЧНЫЙ АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ РАЗРЕЗАНИЯ ГРАФОВ

В.К. Погребной

Институт «Кибернетический центр» ТПУ
E-mail: vk@ad.cctpu.edu.ru

Предложен матричный алгоритм решения задачи разрезания графов. Рассмотрены основные положения алгоритма, базирующиеся на матричном представлении графов. Приведена формализация основных процедур алгоритма – определение оценок для выбора перемещаемых элементов матрицы и преобразование матрицы путем взаимного переноса столбцов и строк. Работа алгоритма рассмотрена на примере графа передач данных между станциями локальной сети вычислительной системы.

Введение

Задачу разрезания графа на минимально связанные части можно отнести к числу классических задач теории графов, широко используемых в практических приложениях. Среди наиболее ранних и хорошо разработанных приложений выделяются задача разбиения схемы вычислительного устройства, представленного в виде графа [1] или в виде более адекватной модели – гиперграфа [2]. Для обыкновенного графа [3] задача разрезания формулируется следующим образом. Требуется разрезать граф $G=(S,V)$ на части $G_f=(S_f,V_f)$, $f=1,2,...,F$, где F – число частей, на которые разрезается граф; S_f – множество вершин, принадлежащих f-ой части; V_f – множество ребер инцидентных вершинам S_f.

Совокупность частей $B(G_f)$ называется разбиением графа G, если

$$\forall G_f \in B(G_f)[G_f \neq \varnothing \,\&\, \bigcup G_f = G];$$

$$\forall G_f, G_q \in B(G_f)[G_f \neq G_q \,\&\, S_f \bigcap S_q = \varnothing \,\&\, V_f \bigcap V_q = V_{fq}],\ \ f,q = 1,2,...,F.$$

Здесь V_{fq} – множество ребер, связывающих пары вершин, одна из которых принадлежит части G_f, а другая – G_q.

Обозначим $|V_{fq}|=g_{fq}$ и назовем его числом реберного соединения частей G_f и G_q. Тогда число реберного соединения разрезания графа G определяется величиной g:

$$g = \sum_{f=1}^{F}\sum_{q=1}^{F} q_{fq},\ \ f \neq q.$$

Традиционным критерием решения задачи разрезания графа G является минимизация числа реберного соединения g при ограничении на число вершин в частях G_f.

Среди известных алгоритмов разрезания графов существуют точные алгоритмы, использующие методы решения задач дискретного программирования, и приближенные. Более широкое распространение получили приближенные алгоритмы, среди которых выделяются последовательные, итерационные и смешанные.

В приближенных алгоритмах последовательного типа сначала по определенному критерию выбирается вершина графа, затем к ней присоединяются другие вершины до получения первой части. Затем из оставшихся вершин графа формируется вторая часть и последующие части до полного разрезания.

Итерационные алгоритмы в качестве исходного берут некоторое разрезание, полученное, например, с помощью одного из последовательных алгоритмов, а затем последовательно в связанных парах частей производятся перестановки вершин из одной части в другую так, чтобы улучшалось значение критерия качества.

Анализ данных алгоритмов приводит к следующим выводам:

- наличие большого разнообразия алгоритмов, которое порождается стремлением учесть специфику исследуемых объектов и используемых для их описания графов;
- ориентация алгоритмов на работу непосредственно со списками вершин, ребер и их атрибутов;
- выполнение операций по оптимизации реберного соединения между двумя частями графа носит локальный характер и «не видит», что происходит с реберными соединениями между другими частями разрезания.

Первые два вывода отражают как положительные, так и отрицательные свойства алгоритмов. Так, учет специфики приводит к уникальности алгоритмов, но позволяет повышать их эффективность. Использование списков является предпочтительным для графов с большим числом вершин и малым числом ребер. Что касается третьего вывода, то он отражает трудно преодолимый недостаток таких алгоритмов, когда оптимизация осуществляется локальными улучшениями реберных соединений внутри очередной пары частей разрезания.

Предлагаемый в данной работе алгоритм разрезания основан на представлении графа матрицей связности вершин и более полном анализе оценок эффективности назначения вершин в подграфы разрезания.

Основные положения матричного алгоритма

Решение задачи разрезания рассматривается для обыкновенного взвешенного графа $G=(S,V,\mathbf{R})$, представленного матрицей $\mathrm{R}=\|r_{ij}\|$, где r_{ij} – вес ребра $v_{ij}\in V$, $i,j=1,2,...,n$. Таким графом является, например, граф передач данных [4] между вершинами $s_i\in S$, которые в данном случае соответствуют станциям локальной вычислительной сети, а веса r_{ij} – объемам данных, передаваемых между станциями s_i и s_j в сети. Матрица **R** является симметричной, так как вес r_{ij} включает объем данных, который передается от станции s_i к s_j и от s_j к s_i.

При построении локальной сети вычислительной системы на основе нескольких магистралей возникает задача распределения станций по магистралям так, чтобы объем данных, передаваемых между станциями, подключенными к разным магистралям, был минимальным. Такая задача соответствует задаче разрезания графа на минимально связанные подграфы. Вершины, входящие в один подграф соответствуют станциям, подключенным к одной магистрали.

Результат решения задачи разрезания можно представить матрицей **R**, разбитой на блоки, как это показано на рис. 1. Каждый f-й блок является матрицей связности вершин подграфа G_f. Элементы матрицы **R**, расположенные вне блоков, определяют число реберного соединения разрезания графа G и образуют область сечения. Матрицу, изображенную на рис. 1, относительно матрицы **R** можно рассматривать в качестве матрицы-шаблона, задающего условия разрезания – размерность графа, состав блоков с указанием размерности каждого из них, привязку блоков к номерам строк и столбцов шаблона. Блоки при этом рассматриваются как потенциальные места назначения вершин графа при включении их в соответствующий подграф.

Если такой шаблон наложить на матрицу **R** графа G, то мы получим исходное разрезание, в котором номера вершин графа совпадают с номерами строк и столбцов шаблона. На рис. 1 номера строк и столбцов шаблона показаны слева и сверху соответственно, а номера вершин – справа и снизу. При этом индексные множества J_f и J_q включают номера строк и столбцов шаблона, а J_f^* и J_q^* – матрицы **R**. Для исходного разрезания $J_f=J_f^*$, $J_q=J_q^*$.

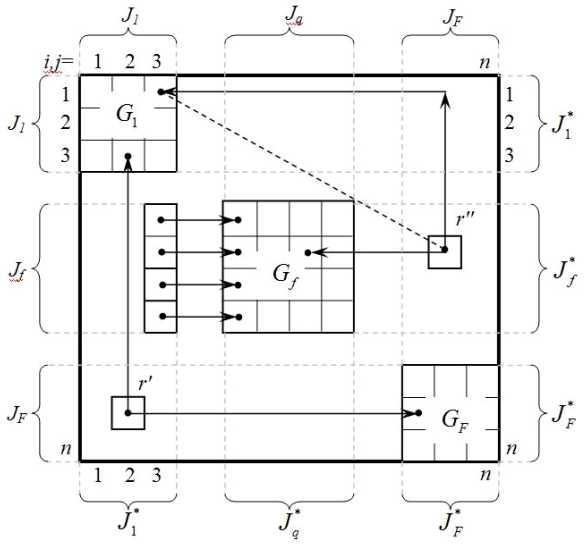

Рис. 1. *Матрица – шаблон для задачи разрезания*

Качество разрезания для рассматриваемого приложения определяется суммой весов реберного соединения разрезания, что соответствует сумме элементов матрицы **R** расположенных в области сечения шаблона. Сумму весов обозначим величиной *r*, а множество элементов области сечения величиной V_c. Из этого следует, что $r = \sum_{v_{ij} \in V_c} r_{ij}$.

Уменьшить величину *r* можно путем эквивалентных преобразований матрицы **R**, перемещая элементы из множества V_c с большими весами в область блоков шаблона. Матрица **R** и матрица $\mathbf{R}_k$, полученная в результате *k*-го преобразования, эквивалентны, если каждая из них является матрицей смежности вершин одного и того же графа *G*. Исходя из этого, задача разрезания заключается в том, чтобы, перемещая элементы из области сечения в область блоков, и сохраняя при этом эквивалентность матрицы **R**, преобразовать ее к виду, в котором сумма весов элементов области сечения была бы минимальной, а сумма весов элементов области блоков, соответственно – максимальной.

Для реализации данного подхода к решению задачи разрезания необходимо разработать стратегию выбора и перемещения элементов матрицы **R** с соблюдением условия эквивалентности. Соответствующий алгоритм назван матричным и включает две основные процедуры:

- определение оценок для выбора перемещаемых элементов;
- эквивалентное преобразование матрицы **R**, реализующее выбранное перемещение.

При реализации данной стратегии выделяются два варианта матричного алгоритма. В первом из них блоки шаблона рассматриваются как свободные места для назначения выбранных элементов. Второй вариант алгоритма использует наложение шаблона на матрицу **R**. Полученный при этом исходный вариант разрезания улучшается взаимным перемещением элементов между областями сечения и блоков. Ниже предлагается второй более простой вариант алгоритма.

Перемещение элементов матрицы

Перемещение выполняется над элементами матрицы **R**, на которую наложена матрица – шаблон. Необходимость перемещения элементов обусловлена критерием решения задачи разрезания, в соответствии с которой в области сечения должна остаться минимальная сумма весов элементов. При наложении шаблона на матрицу **R** мы получаем исходный вариант разрезания. Поэтому операции по перемещению элементов осуществляются в условиях, когда все места в блоках заняты. Следовательно, чтобы выполнить перемещение, необходимо освободить место назначения для перемещаемых элементов. Если освобождаемые элементы не выносятся для хранения за пределы матрицы, а размещаются на места перемещаемых элементов, то имеет место взаимное перемещение элементов, такое правило перемещения соответствует предлагаемому варианту матричного алгоритма.

Элемент матрицы может перемещаться по строке или столбцу как это показано на рис. 1 для элемента *r′*, расположенного во 2-м столбце матрицы. Если элемент перемещается с изменением строки и столбца, как показано для элемента *r″*, расположенного в (*n*–1)-м столбце, то выполняется два перемещения, например, вначале по столбцу, а затем по строке или наоборот.

Для сохранения эквивалентности матрицы вместе с перемещением элемента необходимо переносить соответствующие столбец и строку. Если перемещение осуществляется по строке, то переносится столбец и соответствующая строка. Так перемещение элемента *r′* в блок G_F влечет перенос 2-го столбца в (*n*–2)-й столбец и 2-й строки в (*n*–2)-ю строку. Аналогично, при перемещении элемента *r′* по столбцу (*n*–1)-я строка переносится в 3-ю строку и (*n*–1)-й столбец переносится в 3-й столбец.

Объектом перемещения часто является не один элемент, а несколько. Как правило, число элементов определяется размерностью блока, в который они перемещаются. На рис. 1 показано перемещение 4-х элементов 3-го столбца в 1-й столбец блока G_f размерностью 4×4. Один из перемещаемых элементов при этом попадает на диагональный элемент блока, в данном случае это 1-й элемент сверху. Если данный элемент имеет вес $r_{ij}>0$, то он переносится на место диагонального элемента перемещаемого столбца. Аналогично, если при перемещении строки ненулевой элемент попадает на диагональный элемент, то он переносится на диагональный элемент перемещаемой строки. Такое правило взаимодействия ненулевого элемента и диагонального элемента используется также при переносе строк и столбцов с освобождаемыми элементами.

Выбор элементов для перемещения

Исходя из стремления переместить из области сечения в область блоков элементы с максимальным суммарным весом введем оценку предпочтительности элементов для их взаимного переноса. С этой целью каждый столбец матрицы разобьем на группы элементов в соответствии с их принадлежностью строкам одного блока, так, что *f*-ая группа элементов соответствует строкам J_f, принадлежащим блоку G_f. Например, для матрицы на рис. 1, у всех столбцов 1-я группа будет состоять из первых трех элементов, а последняя – из элементов {*n*–2, *n*–1, *n*}. При таком разбиении у каждого столбца одна из групп будет принадлежать блоку, а все остальные области сечения.

Введем следующие обозначения. Индексами *i* и *j* будем обозначать номера строк и столбцов матрицы-шаблона: *j* – номер столбца или строки, перемещаемых в блок; *i* – номер столбца или строки, исключаемых из блока. Индексами *f* и *q* будем

обозначать номера блоков матрицы-шаблона: q – номер блока, в который перемещается столбец j; f – номер блока, из которого исключается столбец j; J_q – множество номеров строк блока q; J_f – множество номеров строк блока f.

Оценки предпочтительности будем вычислять относительно групп элементов расположенных в области сечения. Сумму весов элементов группы J_q у столбца j, который перемещается в блок q, обозначим величиной $\tilde{\alpha}_{qj}$,

$$\tilde{\alpha}_{qj} = \sum_{i \in J_q} r_{ij}, \quad \alpha_{qji} = \tilde{\alpha}_{qj} - \xi_{qji}. \qquad (1)$$

Здесь ξ_{qji} – вес r_{ij} элемента (i,j) из группы J_q, который при перемещении столбца j в блок q на место столбца i, попадает на его диагональный элемент; α_{qji} – сумма весов элементов столбца j, после его перемещения в блок q на место столбца i.

Аналогично для столбца i, который перемещается в блок f на место столбца j, вычисляются величины $\tilde{\alpha}_{fi}$ и α_{fij}:

$$\tilde{\alpha}_{fi} = \sum_{i \in J_f} r_{ij}, \quad \alpha_{fij} = \tilde{\alpha}_{fi} - \xi_{fij}. \qquad (2)$$

Снижение сумм весов элементов, которое происходит в блоках q и f при исключении из них соответствующих столбцов i и j, определяется величинами β_{qi} и β_{fj}:

$$\beta_{qi} = \sum_{i \in J_q} r_{ij}, \quad \beta_{fj} = \sum_{i \in J_f} r_{ij}. \qquad (3)$$

В принятых обозначениях схема взаимного переноса столбцов j и i и определения соответствующих оценок приведена на рис. 2. Диагональные элементы столбцов i и j в блоках q и f заштрихованы. Выделены также элементы столбцов i и j, которые при переносе попадают на диагональные элементы в блоках. Перед переносом столбцов выделенные в них элементы, перемещаются внутри столбца на место диагонального элемента, что на рис. 2 показано стрелками. Места выделенных элементов после переноса столбцов принимаются в качестве диагональных элементов.

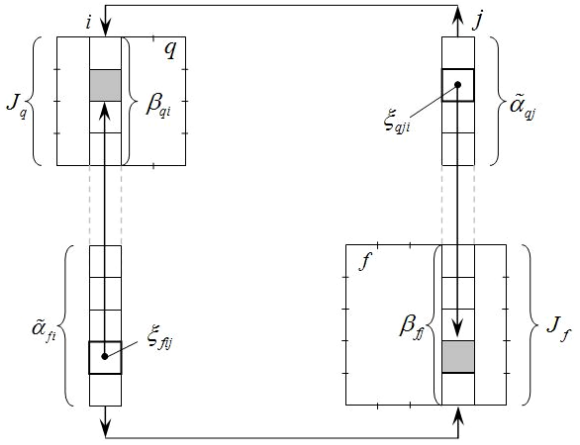

Рис. 2. *Схема взаимного переноса столбцов*

Введенные в (1)–(3) оценки позволяют получить суммарные оценки предпочтительности μ_{ij} для пар столбцов i и j относительно их взаимного переноса,

$$\mu_{ij} = \alpha_{qji} - \beta_{fj} + \alpha_{fij} - \beta_{qi}. \qquad (4)$$

Оценки $\mu_{ij}>0$ показывают, на сколько увеличится суммарный вес элементов в блоках после взаимного переноса столбцов j и i или, соответственно, на сколько уменьшится суммарный вес элементов области сечения.

Оценки μ_{ij} вычисляются только для элементов области сечения шаблона, так как перемещение элементов внутри блоков не меняет сумму их весов. Если при этом матрица **R** симметрична, то оценки μ_{ij} вычисляются для одной части области сечения – верхней или нижней.

Матричный алгоритм и пример его применения

После представления графа матрицей смежности **R** работа алгоритма включает выполнение следующих шагов.

1. Формирование шаблона и наложение его на матрицу **R**. Шаблон формируется на основе сведений о числе блоков разрезания, размерности каждого из них и расположении в шаблоне. Места расположения блоков в шаблоне устанавливаются в произвольной последовательности независимо от размерности блоков. В результате наложения шаблона на матрицу **R** получается исходный вариант разрезания, в котором $J_f=J_f^*$; $J_q=J_q^*$; $f,q=1,2,...,F$.
2. Вычисление оценок μ_{ij}. Оценки вычисляются по выражениям (1)–(4). Если матрица **R** симметрична, то вычисление производится для одной из частей области сечения симметричных относительно диагонали матрицы – верхней (над диагональю) или нижней (под диагональю). Среди оценок $\mu_{ij}>0$ выбирается максимальная и соответствующим столбцам i и j отдается предпочтение для их взаимного переноса. В этом случае осуществляется переход к выполнению шага 3. При отсутствии оценок $\mu_{ij}>0$ алгоритм заканчивает работу и фиксирует полученный вариант разрезания.
3. Взаимный перенос столбцов i и j. Перенос столбцов и соответствующих строк производится по изложенным выше правилам и схеме, приведенной на рис. 2. Взаимный перенос строк и столбцов сопровождается корректировкой индексных множеств J_f^* и J_q^*. После этого производится переход к выполнению 2-го шага алгоритма.

Работу алгоритма покажем на примере взвешенного обыкновенного графа, содержащего 10 вершин. Граф необходимо разрезать на 3 подграфа, один из которых содержит 4 вершины, а два других по 3 вершины. Матрица **R** данного графа и совмещенная с ней матрица – шаблон, построен-

ная для указанных условий разрезания, представлены на рис. 3, *а*. Полученный при этом исходный вариант разрезания характеризуется суммами весов элементов в блоках (числитель) и в области сечения (знаменатель) равными 15/50.

Для улучшения соотношения 15/50 вычисляются оценки μ_{ij}, среди которых выбирается оценка с максимальным значением. В данном случае это оценки $\mu_{5,3}=\mu_{9,3}=9$. Выбираем любую из них, например $\mu_{5,3}$, что предполагает перемещение элемента (2,5) с весом $r_{2,5}=2$ в блок J_1 и элемента (4,3) с весом $r_{4,3}=7$ в блок J_2. Для перемещения данных элементов выполняется взаимный перенос столбцов $j=5$ и $i=3$ и соответствующих строк. Результат переноса представлен на рис. 3, *б*. Соотношение сумм весов при этом становится равным 24/41.

Для данного варианта разрезания максимальной оценкой μ_{ij} является $\mu_{9,1}=14$. После взаимного переноса столбцов $j=9$ и $i=1$ получается вариант разрезания, представленный на рис. 3, *в*, с соотношением сумы весов 38/27. Максимальной оценкой μ_{ij} в данном случае является $\mu_{8,6}=3$. В результате взаимного переноса столбцов $j=8$ и $i=6$ получаем разрезание, представленное на рис. 3, *г*, для которого оценок $\mu_{ij}>0$ нет. Следовательно, алгоритм закончил работу с вариантом разрезания $J_1^*=\{0,9,2,5\}$, $J_2^*=\{4,3,8\}$, $J_3^*=\{7,6,1\}$ и соотношением сумм весов 41/24.

Если, используемый в качестве примера, граф интерпретировать как граф передачи данных [4], вершинами в котором являются станции локальной сети s_i, $i=0,1,2,...,9$, а веса ребер r_{ij} соответствуют объемам данных, передаваемых между станциями s_i и s_j, то в результате решения задачи разрезания получаем три подмножества станций $\{s_0,s_9,s_2,s_5\}$, $\{s_4,s_3,s_8\}$, $\{s_7,s_6,s_1\}$, каждое из которых подключается к одной из трех магистралей сети. При таком варианте построения сети большая часть данных, в данном примере это 41 единица, передается между

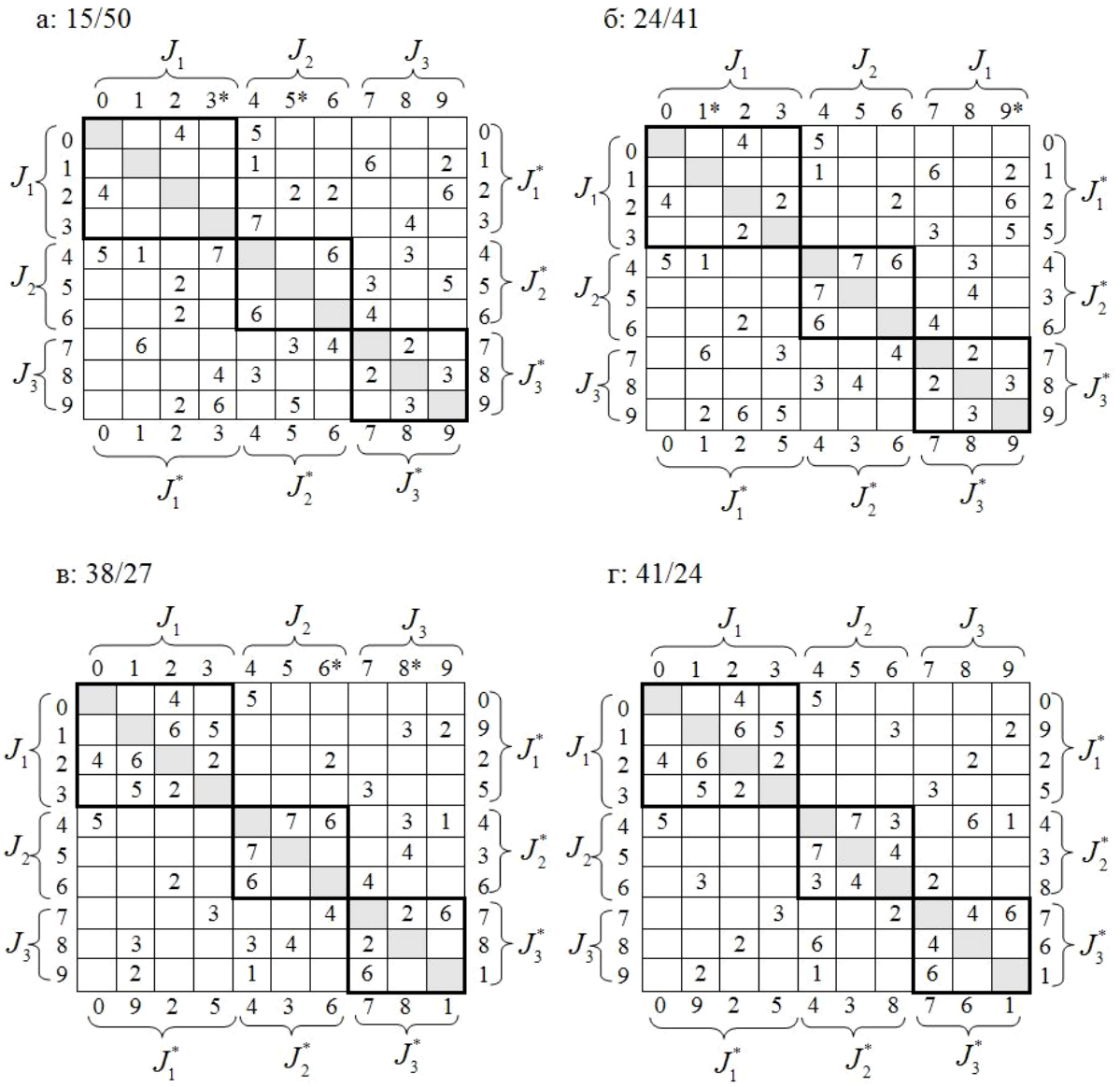

Рис. 3. *Пример решения задачи разрезания графа*

станциями внутри множеств, загружая соответствующую магистраль. При этом передача данных во всех трех магистралях может происходить одновременно (параллельно). Другие данные в размере 24 единицы передаются между станциями, подключенными к разным магистралям. При таких передачах одновременно загружается две или все три магистрали, что приводит к увеличению загрузки сети и, соответственно, времени на передачу данных.

Заключение

Преимущества предложенного алгоритма в сравнении с известными, работающими со списками ребер, обусловлены представлением графов в матричной форме, при которой область сечения задается в явном виде. Это позволяет легко наблюдать («видеть») изменения, происходящие в области сечения в процессе преобразования матриц, а оценки предпочтительности для выполнения очередного преобразования вычислять на основе анализа текущего состояния всей области сечения.

В алгоритме реализована самая простая схема перемещения элементов из области сечения в область блоков, основанная на взаимном переносе столбцов и соответствующих строк. Другие более сложные схемы связанные, например, с формированием замкнутых цепей перемещений, являются предметом дальнейших исследований и в данной статье не рассматривались.

Матричный алгоритм применим для графов с несимметричной матрицей, в том числе и не взвешенной. Очевидно, что использование данного алгоритма оказывается более предпочтительным для графов с высокой степенью связности вершин, так как объем вычислений, выполняемых алгоритмом, не зависит от коэффициента заполнения матрицы ненулевыми элементами.

Наряду с достижением основной цели – формализации матричного алгоритма решения задачи разрезания, матричный метод имеет другое, не менее важное значение. Оно заключается в том, что матричная форма представления задачи разрезания создает возможности для более глубокого понимания ее природы, совершенствования методов анализа области сечения в процессе оптимизации, изучения зависимости алгоритма от различных стратегий определения оценок для принятия решений.

Матричный метод может быть положен в основу разработки алгоритмов для различных типов графов и условий разрезания. В частности, интерес представляют двудольные графы, матричные представления которых, в сравнении с рассмотренными, предполагают более эффективные операции преобразования матриц при решении задачи разрезания.

СПИСОК ЛИТЕРАТУРЫ

1. Штейн М.Е., Штейн Б.Е, Методы машинного проектирования цифровой аппаратуры. – М.: Советское радио, 1973. – 296 с.
2. Корниенко А.В., Погребной В.К. Модель и алгоритм для разбиения цифровых вычислительных устройств на функциональные блоки // Управляющие системы и машины. – 1976. – № 5. – С. 94–98.
3. Кристофидес Н. Теория графов. Алгоритмический подход. – М.: Мир, 1978. – 432 с.
4. Погребной А.В. Определение объемов передач данных в сети вычислительной системы для заданной модели программной нагрузки // Известия Томского политехнического университета. – 2007. – Т. 310. – № 3. – С. 103–107.

Поступила 31.10.2007 г.