УДК 004.93'1; 004.932

Д.В. Прадун, Б.А. Залесский

БЛОЧНО-ПАРАЛЛЕЛЬНАЯ КЛАСТЕРИЗАЦИЯ МУЛЬТИСПЕКТРАЛЬНЫХ ИЗОБРАЖЕНИЙ С ПОМОЩЬЮ АЛГОРИТМА МАКСИМАЛЬНОГО ПОТОКА В СЕТИ

Рассматривается алгоритм максимального потока в сети для кластеризации мультиспектральных изображений, в том числе большого размера, в блочно-параллельном режиме. Описываются общая схема и основные этапы реализации блочно-параллельной модификации алгоритма максимального потока. Дается сравнение результатов кластеризации космических изображений алгоритмом максимального потока в последовательном и параллельном режимах.

Введение

Широкое использование алгоритма максимального потока в сети обусловлено его эффективностью при решении задач широкого диапазона: от расчета пропускных способностей электросетей до задач кластеризации полутоновых и мультиспектральных изображений [1–4].

В настоящее время параллельно с увеличением вычислительных возможностей современных персональных компьютеров и рабочих станций возрастает сложность возникающих задач, решение которых требует увеличения вычислительных ресурсов и ускорения процессов обработки информации. Нередко эти задачи не могут быть выполнены имеющимися на данный момент методами и алгоритмами на современных вычислительных системах. Это же касается и алгоритма максимального потока в сети, который использует информацию о всех вершинах сети и ее дугах. В случае вычисления максимального потока для космических изображений, где количество цветовых каналов может достигать несколько десятков, а количество пикселов – сотни миллионов, можно с уверенностью утверждать, что объема оперативной памяти вычислительной машины не хватит.

В данной работе предлагается блочно-параллельная модификация алгоритма максимального потока сети, позволяющая разбивать каждый цветовой канал на блоки подходящего размера и вычислять максимальный поток в параллельном режиме, что дает, например, возможность выполнять кластеризацию изображений больших размеров, которые не могут быть обработаны известными алгоритмами максимального потока.

1. Вычисление максимального потока сети в последовательном режиме

1.1. Определение максимального потока

Обозначим через s (источник) вершину сети $G = (V, E)$, не имеющую входящих дуг, а через t (сток) – вершину, не имеющую исходящих дуг [3]. Кроме того, зададим для каждой дуги $e = (i, j), i, j \in V,$ неотрицательно число $c(e)$, называемое пропускной способностью дуги. Величиной f потока сети G будет называться функция, сопоставляющая каждой дуге $e = (i, j)$ неотрицательное вещественное число $f(e) = f(i, j)$. Для каждой величины f верны следующие условия [4, 5]:

$$f(i, j) \le c(i, j), \; i, j \in V; \tag{1}$$

для всех i из $V \setminus \{s, t\}$

$$\sum_{j \in V} f(i, j) - \sum_{j \in V} f(j, i) = 0. \tag{2}$$

Тогда величина потока f сети представляет собой сумму потоков по всем дугам, исходящим из истока либо входящим в сток:

$$val(f) = \sum_{j \in V} f(s, j) = \sum_{i \in V} f(i, t) . \quad (3)$$

Распределим все вершины сети по двум непересекающимся множествам S и $\overline{S}$, $S \bigcup \overline{S} = V$, так, чтобы $s \in S$ и $t \in \overline{S}$. Тогда $s-t$ -разрез сети – это такой разрез $K =< S, \overline{S} >$, который содержит дуги, соединяющие S и $\overline{S}$. В этом случае пропускная способность $c(K) = c(S, \overline{S})$ разреза определяется выражением

$$c(K) = \sum_{\substack{i \in S \\ j \in \overline{S}}} c(i, j) . \quad (4)$$

Существует доказательство того [5], что для любого потока f и любого разреза $< S, \overline{S} >$ в сети верно выражение

$$val(f) = f(S, \overline{S}) - f(\overline{S}, S) . \quad (5)$$

Отсюда следует, что согласно условию (1)

$$val(f) \le c(S, \overline{S}) .$$

Таким образом, если найти такой поток f и разрез $K =< S, \overline{S} >$, для которых $val(f) = c(K)$, то поток f окажется максимальным, а разрез – минимальным. Другими словами, величина максимального потока в сети равна пропускной способности минимального разреза [5].

1.2. Кластеризация изображения

1.2.1. Кластеризация бинарного изображения. Любое бинарное изображение $Y = (y_i)_{i \in P}$ с пикселами P можно представить в виде сети. В этом случае каждый пиксел $i \in P$ – это вершина сети, которая соединяется дугой с соседним пикселом j из выбранной окрестности $O(i)$. Чаще всего выбираются стандартные окрестности $O(i)$ из четырех или восьми пикселов. Тогда множество дуг E представляет собой объединение множеств:

$$E = \{(i, j), i \sim j\} \cup \{(s, i),\ y_i = 1\} \cup \{(i, t),\ y_i = 0\} ,$$

где $i \sim j$ – соседние пикселы изображения для выбранной системы окрестностей $O(i)$; y_i – значение яркости бинарного изображения в пикселе i . Нетрудно заметить, что каждая обычная вершина этой сети $i \in P$ соединена либо только с источником, либо только со стоком.

Для произвольного $s-t$ -разреза $< W, B > (s \in W,\ t \in B)$ построенной сети бинарное изображение X будет иметь следующие значения яркостей [6]:

$$x_i = \begin{cases} 1, & \text{если } i \in W; \\ 0, & \text{если } i \in B. \end{cases}$$

Пропускная способность $s-t$ -разреза $< W, B >$ будет следующей:

$$C(W, B) = \sum_{\substack{i \in W \\ j \in B}} c(i, j) . \quad (6)$$

Пусть задана квадратичная функция от бинарного изображения X с помощью формулы

$$F(X) = \sum_{\substack{i \in W, \\ y_i = 0}} \lambda_i + \sum_{\substack{i \in B, \\ y_i = 1}} \lambda_i + \sum_{\substack{i \in V, \\ j = V}} \beta_{i,j} (x_i - x_j) \cdot x_i , \quad (7)$$

где λ_i – пропускная способность дуги, выходящей из вершины s либо входящей в вершину t, а $\beta_{i,j}$ – пропускные способности всех остальных дуг [2]. Можно убедиться, что $C(W,B)=F(X)$.

Теперь зададим функцию вида

$$U(X)=\sum_{i\in V}\lambda_i\cdot(1-2y_i)\cdot x_i+\sum_{\substack{i\in V,\\ j=\overline{V}}}\beta_{i,j}(x_i-x_j)\cdot x_i\,. \tag{8}$$

Тогда $F(X)=U(X)+\sum_{i\in S}\lambda_i y_i$, вследствие чего множества $\left\{X^*\,|\,F(X^*)=\min_X F(X)\right\}$ и $\left\{X^{**}\,|\,U(X^{**})=\min_X U(X)\right\}$ равны, т. е. множество векторов, удовлетворяющих равенству $X^*=\arg\min_X U(X)$, задает минимальный разрез сети, построенной для искомого бинарного изображения Y [6].

1.2.2. Кластеризация полутонового изображения. Любое полутоновое изображение X представляет собой сумму бинарных изображений-слоев $X(l)$:

$$X=m(0)\cdot X(0)+\sum_{l=1}^{L-1}(m(l)-m(l-1))\cdot X(l)\,,$$

где $m(0),m(1),...,m(L-1)$ – заданная последовательность неотрицательных чисел, $0<l\le L-1$, $2<L\le 256$. Координаты каждого вектора $x(l)$ равны 1, если $x_i(l)\ge m(l)$, и равны 0 в противном случае. Полученные бинарные изображения задают монотонно убывающую последовательность $X(1)\ge X(2)\ge ...\ge X(L-1)$ [7]. Следовательно, для любого полутонового изображения функцию $U(X)$ можно представить как

$$U(X)=\sum_{l=1}^{L-1}U_l(X(l))\,, \tag{9}$$

где

$$U_l(X(l))=\lambda\sum_{i\in S}\left|y_i(l)-x_i(l)\right|+\sum_{(i,j)\in S}\beta_{i,j}\left|x_i(l)-x_j(l)\right|\,. \tag{10}$$

При этом $y_i(l)$ – координаты вектора исходного полутонового изображения, полученные на слое l. Из формул (9), (10) следует, что

$$U(X)=\sum_{i\in S}\lambda_i\left|y_i-\sum_{l=1}^{L-1}x_i(l)\right|+\sum_{(i,j)\in S}\beta_{i,j}\sum_{l=1}^{L-1}\left|x_i(l)-x_j(l)\right|\,. \tag{11}$$

Найдя невозрастающую последовательность таких бинарных изображений-слоев $\widetilde{X}(l)$, каждое из которых удовлетворяет условию

$$\widetilde{X}(l)=\arg\min_{X_{bin}}U_l(X_{bin}),\quad l\in\{1,...,L-1\}\,, \tag{12}$$

получим сегментированное изображение [7], которое будет представлено как

$$\widetilde{X}=m(0)\cdot\widetilde{X}(0)+\sum_{l=1}^{L-1}(m(l)-m(l-1))\cdot\widetilde{X}(l)\,.\;\widetilde{X}(l)=\arg\min_{X_{bin}}U_l(X_{bin}),\quad l\in\{1,...,L-1\}\,. \tag{13}$$

При этом коэффициенты λ_i определяют пропускную способность дуг, выходящих из источника и входящих в сток, а отношение $\dfrac{\beta_{i,j}}{\lambda_i}$ – степень сглаживания изображения.

1.3. Вычисление пропускных способностей дуг по градиенту изображения

Пусть $Gr_{i,j}(I)$ – значение градиента изображения между пикселами i и j, вычисленное с помощью вертикальной, горизонтальной и двух диагональных масок 45 и 135° [8]:

1	2	1
−1	−2	−1

1	−1
2	−2
1	−1

1	2	1	0
0	−1	−2	−1

0	−1	−2	1
1	2	1	0

Тогда для $\beta_{i,j}$ можно определить различные функциональные зависимости от $Gr_{i,j}(I)$:

$$\beta_{i,j} = \frac{\beta_{i,j}}{1 + \Phi(Gr_{i,j}(I))}; \tag{14}$$

$$\beta_{i,j} = \frac{\beta_{i,j}}{1 + \Phi(Gr^2_{i,j}(I))}; \tag{15}$$

$$\beta_{i,j} = \frac{\beta_{i,j}}{\Phi(\exp(Gr_{i,j}(I)))}, \tag{16}$$

где $\Phi(Gr_{i,j}(I))$ – некая функция, зависящая от значения градиента. При этом значения $\beta_{i,j}$ вычисляются во всех направлениях, задаваемых окрестностью $O(i)$ пиксела i, тем самым обеспечивается связь центрального пиксела с каждым соседним пикселом из окрестности $O(i)$.

2. Вычисление максимального потока в сети в параллельном режиме

Алгоритм вычисления максимального потока в сети можно представить следующим образом.

Шаг 1. Пусть для заданного слоя l полутонового изображения либо отдельного цветового канала мультиспектрального снимка выделен блок пикселов размером $M \times N$. Перед тем как вычислять максимальный поток для данного блока данных, построим сеть по следующему правилу. Если пиксел i лежит на границе решетки $M \times N$, пропускные способности дуг, которые входят в данный пиксел от соседних пикселов из окрестности $O(i)$, лежащих вне решетки, добавляются к пропускной способности дуги (s,i):

$$\lambda_{s,i} = \lambda_{s,i} + \sum_{j \in O(i)} \beta_{j,i}. \tag{17}$$

Определим минимальный разрез для построенной сети. Затем выберем из него только вершины, принадлежащие множеству B, т. е. множеству, которое содержит сток t. Остальные вершины помечаем как неопределенные.

Шаг 2. Для решетки $M \times N$ строим сеть по следующему правилу. Если пиксел i лежит на границе решетки $M \times N$, то пропускные способности дуг, которые исходят из данного пиксела в соседние пикселы из окрестности $O(i)$, лежащие вне решетки, добавим к пропускной способности дуги (i,t):

$$\lambda_{i,t} = \lambda_{i,t} + \sum_{j \in O(i)} \beta_{i,j}. \tag{18}$$

Определив минимальный разрез для построенной сети, выбираем из него только те вершины, которые принадлежат множеству W, т. е. множеству, которое содержит исток s. Остальные вершины пометим как неопределенные, кроме тех, которые на предыдущем шаге были определены как принадлежащие множеству B.

В результате описанных шагов, выполненных для всех блоков, на которые было разбито исходное изображение, получим изображение с пикселами следующего вида:

$$x_i(l) = \begin{cases} m(l) - m(l-1), & \text{если } i \in W; \\ 0, & \text{если } i \in B; \\ \text{остальные.} \end{cases}$$

Шаг 3. Для пикселов, не определенных на предыдущих шагах алгоритма, построим сеть и найдем ее максимальный поток. При этом если в окрестности пиксела i есть пиксел j, определенный как принадлежащий множеству B на шаге 1, значение пропускной способности $\beta_{i,j}$ соответствующей дуги добавим к значению пропускной способности $\lambda_{i,t}$ дуги (i,t). Если в окрестности пиксела i есть пиксел j, определенный как принадлежащий множеству W на шаге 2, то значение пропускной способности $\beta_{i,j}$ соответствующей дуги добавим к значению пропускной способности $\lambda_{s,i}$ дуги (s,i).

Шаг 4. Шаги 1–3 повторяем для всех слоев L, на которые разбивается исходное изображение.

3. Сравнение результатов работы алгоритма максимального потока в последовательном и параллельном режимах

Результаты кластеризации мультиспектральных изображений с помощью алгоритма максимального потока в последовательном и параллельном режимах показаны на рисунке. При этом использовалось $L = 10$ слоев. Характеристики исходных снимков представлены в табл. 1. Для сравнения объема памяти, необходимой для выполнения алгоритма максимального потока, зададим формулу

$$K = 3 \cdot M \cdot N + n_{\text{int}} \cdot n_{proc} \cdot [gr_h \cdot gr_v + 2 \cdot ((gr_h - 1) \cdot gr_v + (gr_v - 1) \cdot gr_h + (gr_h - 1) \cdot (gr_v - 1))], \quad (19)$$

где M, N – ширина и высота исходного изображения соответственно; n_{int} – размер блока памяти, необходимый для хранения одного целого числа, определяющего значение пропускной способности дуги сети; n_{proc} – количество вычислительных процессоров, выполняющих вычисления; gr_h и gr_v – ширина и высота блока изображения, для которого строится сеть. Для последовательного режима $n_{proc} = 1$, $gr_h = M$, $gr_v = N$. Для предложенного блочно-параллельного режима работы значения gr_h и gr_v желательно выбирать значительно меньше ширины и высоты самого изображения. Например, при тестировании использовались $gr_h = 64$ и $gr_v = 64$, т. е. для каждого вычислительного процессора выделялось $n_{\text{int}} \cdot (64 \cdot 64 + 2 \cdot (63 \cdot 64 + 63 \cdot 64 + 63 \cdot 63)) = 28182 \cdot n_{\text{int}}$ байт памяти. При $n_{proc} = 4$ и $n_{\text{int}} = 4$ общее количество памяти, необходимой для создания сетей на всех вычислительных процессорах в блочно-параллельном режиме, равно $28182 \cdot 4 \cdot 4 \approx 450$ КБ. Таким образом, для изображений из табл. 1 необходимый объем оперативной памяти составляет:

1) для последовательного режима $K_{посл} = 32{,}7 \div 78{,}8$ МБ;

2) для предложенного блочно-параллельного режима $K_{пар} = 3{,}4 \div 7{,}3$ МБ.

Поэтому обработка больших и сверхбольших мультиспектральных изображений в последовательном режиме физически невозможна при использовании известных алгоритмов по-

строения сети и определения ее максимального потока. Так, для снимка размером $15\,000 \times 15\,000$ пикселов понадобится около $K_{посл} \approx 7{,}8$ ГБ. При этом в параллельном режиме для аналогичного изображения нужно $K_{пар} \approx 675$ МБ, хотя в принципе можно выбирать блоки еще меньшего размера, чтобы свести загрузку памяти к минимуму.

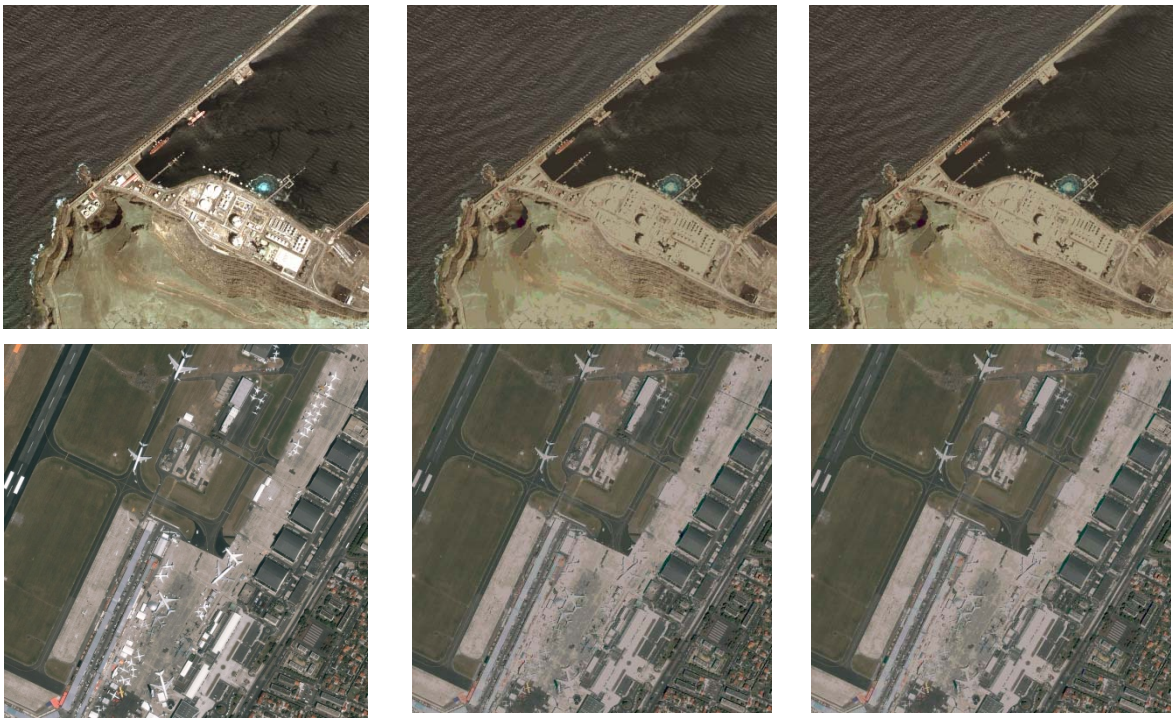
Таблица 1

Основные характеристики изображений

Размер	Количество каналов	Формат изображения
1030×907	3	BMP
1481×1521	3	JPG

В связи с тем что классический метод построения сети не работает в случае больших изображений, следует разбить изображение на большие блоки, которые могут быть загружены в память. При таком разбиении можно выполнять вычисление максимального потока в сети для каждого блока аналогично описанному блочно-параллельному алгоритму. Однако при этом скорость выполнения кластеризации в последовательном режиме значительно уступает блочно-параллельной версии (табл. 2). Результаты были получены при обработке изображений из табл. 1 на четырехъядерном процессоре Intel с частотой 2,66 ГГц и оперативной памятью 3,24 ГБ. Видно, что даже для сравнительно небольших космоснимков время выполнения алгоритма в предложенном блочно-параллельном режиме превосходит в $1{,}5 \div 4$ раза быстродействие алгоритма в последовательном режиме. При этом объем необходимой памяти при последовательной реализации в несколько раз превосходит блочно-параллельную версию.

Использование оперативной памяти при работе блочно-параллельного алгоритма можно уменьшить за счет загрузки с жесткого диска не всего цветового канала изображения размером $M \times N$, а лишь его части для каждого вычислительного процессора размером $gr_h \times gr_v$. Однако из-за частых процедур чтения файла время выполнения алгоритма значительно увеличивается даже для небольшого изображения и скорость выполнения кластеризации уменьшается в несколько раз.



а) *б)* *в)*

Рис. 1. Результаты кластеризации алгоритмом максимального потока: *а)* исходные снимки; *б)* кластеризация в последовательном режиме; *в)* кластеризация в параллельном режиме

Таблица 2

Время выполнения алгоритма максимального потока, с

Исходные данные	Последовательный режим	Параллельный режим
Изображение 1 (табл. 1)	33,1	8,3
Изображение 2 (табл. 1)	34,7	20,6

Если при относительно небольших изображениях разница в 15–30 с не слишком заметна, то при кластеризации больших мультиспектральных изображений преимущество блочно-параллельного режима в быстродействии очевидна (табл. 3). Кроме того, за счет разбиения изображения на небольшие блоки, например размером 64×64 пикселов, происходит экономия памяти, необходимой для построения сетей на всех вычислительных процессорах.

Таблица 3

Показатели работы последовательного и блочно-параллельного алгоритмов

Режимы	Характеристики изображения	Время выполнения	Объем выделяемой памяти, МБ
Последовательный (блок 1338×1338)	Формат BMP, 9372×9372 пикселов, три цветовых канала	27 мин 45 с	314
Параллельный		12 мин 8 с	264

При увеличении значения L разность в быстродействии последовательного и параллельного режимов алгоритма максимального потока становится более ощутимой. Кроме того, параллельный режим можно усовершенствовать за счет того факта, что получаемые бинарные слои исходного изображения задают невозрастающую последовательность $X(1) \geq X(2) \geq \ldots \geq X(L-1)$. Другими словами, если на предыдущем бинарном слое пикселы принадлежат множеству B, эти же пикселы будут принадлежать тому же множеству и на всех последующих слоях. Данное утверждение позволяет значительно сократить время выполнения алгоритма максимального потока (табл. 4).

Таблица 4

Сравнение показателей быстродействия последовательного и параллельного алгоритмов

Параметры, используемые при работе алгоритма	Режимы		
	Последовательный (блок 1338×1338)	Параллельный	Параллельный с учетом предыдущих слоев
Характеристики изображения	Формат JPEG, 9372×9372 пикселов, три цветовых канала		
Количество бинарных слоев $L = 10$	27 мин 45 с	12 мин 8 с	8 мин 25 с
Количество бинарных слоев $L = 256$	9 ч 46 мин	3 ч 47 мин	3 ч 8 мин

Заключение

Результаты кластеризации мультиспектральных изображений показывают, что использование блочно-параллельного алгоритма максимального потока предпочтительно в случаях ограниченности вычислительных ресурсов, в первую очередь оперативной памяти. При этом объем выделяемой памяти зависит только от размеров самого исходного изображения. Так, при разбиении изображения на блоки размером 64×64 пикселов на четырехъядерном процессоре понадобится всего лишь около 450 КБ памяти для создания сетей на каждом ядре. Такой объем памяти прак-

тически не влияет на общий объем необходимой памяти при обработке изображения размером 1000×1000 пикселов, где для хранения результатов и исходных данных необходимо около 270 МБ. Результаты работы алгоритма показали, что чтение отдельного сегмента цветового канала изображения, а не всего канала, неоптимально с точки зрения быстродействия, хотя позволяет уменьшить объем выделяемой памяти почти в 1,5 раза. Увеличение времени выполнения задачи связано в этом случае с большим числом обращений к HDD.

Быстродействие блочно-параллельного алгоритма максимального потока зависит в первую очередь от количества вычислительных процессоров ЭВМ. Так как при параллельной реализации количество итераций по сравнению с классическим алгоритмом максимального потока увеличилось в три раза, во избежание значительных временных потерь при работе алгоритма требуется минимум три процессора. С увеличением количества процессоров, а также объема оперативной памяти скорость выполнения блочно-параллельного алгоритма возрастает.

Время выполнения параллельного алгоритма максимального потока значительно сокращается при учете результатов кластеризации на более низких бинарных слоях, на которые разбивается исходное изображение. Если учитывать тот факт, что пикселы, определенные на нижнем бинарном слое как принадлежащие множеству B, будут также принадлежать на всех последующих слоях данному множеству, быстродействие алгоритма максимального потока увеличивается дополнительно в 1,2–1,5 раза по сравнению с быстродействием блочно-параллельного алгоритма. Для дальнейшего увеличения скорости выполнения параллельного алгоритма можно использовать результаты кластеризации на верхних слоях. Тогда пикселы, определенные на верхнем бинарном слое как принадлежащие множеству W, будут принадлежать этому множеству и на всех остальных более низких слоях.

Список литературы

1. Гонсалес, Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс. – М. : Техносфера, 2005. – 1072 с.

2. Boykov, Y. An experimental comparison of min-cut/max-flow algorithms of energy minimization in vision / Y. Boykov, V. Kolmogorov // IEEE Transactions on PAMI. – 2004. – Vol. 26, № 9. – P. 1124–1137.

3. Boykov, Y. Graph Cuts and Efficient N-D image Segmentation / Y. Boykov, G. Funka-Lea // Intern. J. of Computer Vision. – 2006. – Vol. 70 (2). – P. 109–131.

4. Прадун, Д.В. Использование алгоритма максимального потока графа для фильтрации мультиспектральных изображений / Д.В. Прадун, Б.А. Залесский // Информатика. – 2009. – № 4 (24). – С. 18–27.

5. Свами, М. Графы, сети и алгоритмы / М. Свами, К. Тхуласираман. – М. : Мир, 1984. – 454 с.

6. Picard, J.C. Minimum cuts and related problems / J.C. Picard, H.D. Ratliff // Networks. – 1975. – Vol. 5, № 4. – P. 357–370.

7. Zalesky, B.A. Network flow optimization for restoration of images / B.A. Zalesky // Journal of Applied Mathematics. – 2002. – Vol. 2, № 4. – P. 199–218.

8. Залесский, Б.А. Алгоритм адаптивной фильтрации мультиспектральных изображений / Б.А. Залесский, Д.В. Прадун // Информатика. – 2009. – № 2 (22). – С. 31–38.

Поступила 13.09.10

Объединенный институт проблем
информатики НАН Беларуси,
Минск, Сурганова, 6
e-mail: pradundv@gmail.com

D.V. Pradun, B.A. Zalesky

BLOCK-PARALLEL MULTISPECTRAL IMAGE CLUSTERING USING MAXIMUM NETWORK FLOW ALGORITHM

The algorithm of the maximum network flow for multispectral image clustering, including large images, in a block-parallel mode is examined. A general scheme and the basic stages of block-parallel modification of maximum flow algorithm implementation are described. Comparison of clustering results of space images by means of maximum network flow algorithm in consecutive and parallel modes is provided.