

Разбиение графов

Чернов Андрей, 371 группа

Разбиение графа (graph partition)

Представление графа G в виде непересекающихся множеств подмножеств его вершин.

Разрез графа (graph-cut)

Разбиение графа на 2 подграфа так, что сумма удаленных ребер наименьшая.

Данная проблема эквивалентна нахождению максимального потока (по теореме Форда-Фалкерсона).

Бисекция (bisection)

Разбиение графа на два подграфа с выполнением следующих условий:

- 1) Число вершин в подграфах равны* – для графов без весов вершин; сумма весов вершин подграфов равна* – для графов с весами
- 2) Сумма весов удаленных ребер минимальна

* При невозможности равенства, наиболее близкое к нему положение.

к-разбиение графа (k-partition)

Аналог бисекции для разбиения на k подграфов.

Кластеризация (clusterization)

Разбиение графа на подграфы с учетом некоторой метрики, например модульности (modularity).

Алгоритм Кернигана-Лина (Kernighan-Lin)

В простейшем виде алгоритм предназначен для бисекции графа с $2n$ вершинами, однако он может быть адаптирован для решения еще некоторого ряда сходных задач.

Алгоритм Кернигана-Лина

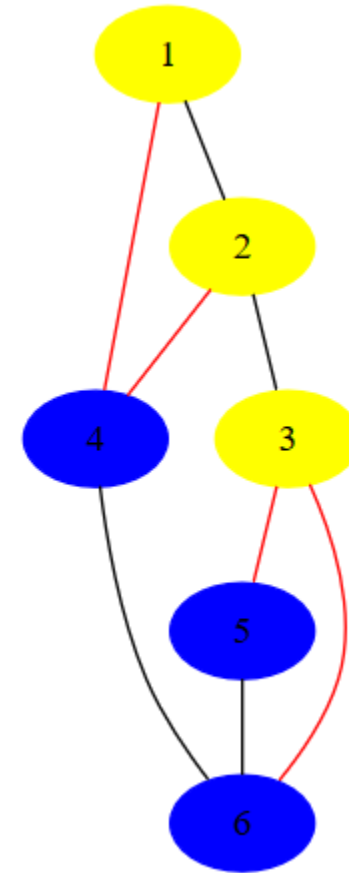
$D(v) = \sum e_i - \sum e_n$, где e_i – веса убранных инцидентных вершине ребер, e_n – веса не убранных инцидентных вершине ребер.

$g(v_1, v_2) = D(v_1) + D(v_2) - 2 * e(v_1, v_2)$, где $e(v_1, v_2)$ – вес ребра между вершинами v_1 и v_2 .

$$G_k = g_1 + g_2 + \dots + g_k$$

Алгоритм Кернигана-Лина

1) Выполнить произвольное разбиение графа на 2 равных подграфа.



Алгоритм Кернигана-Лина

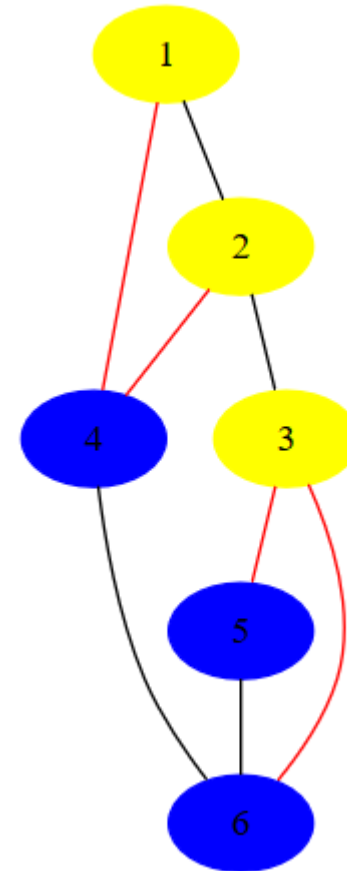
2) Посчитать $D(v)$ для вершин:

- $v(1) = 1 - 1 = 0$
- $v(2) = 1 - 2 = -1$
- $v(3) = 2 - 1 = 1$
- $v(4) = 2 - 1 = 1$
- $v(5) = 1 - 1 = 0$
- $v(6) = 1 - 2 = -1$

3) Выбрать v_1 и v_2 из разных подграфов с максимальной g :

$$g_1 = g(3, 4) = 1 + 1 - 2 \cdot 0 = 2$$

4) Поменять местами 3 и 4, зафиксировать их



Алгоритм Кернигана-Лина

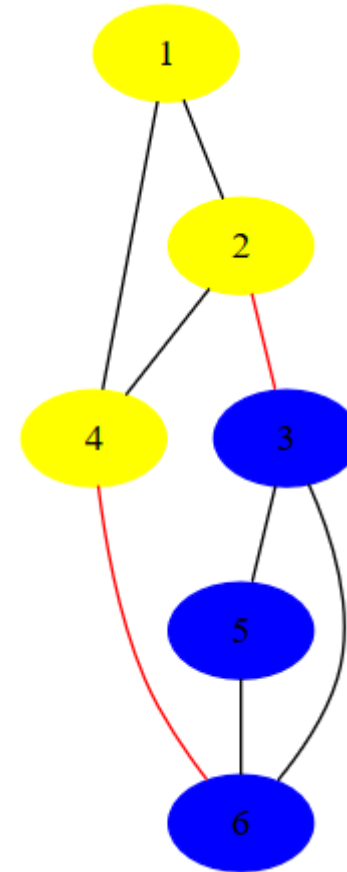
5) Посчитать $D(v)$ для незафиксированных вершин:

- $v(1) = 0 - 2 = -2$ • $v(5) = 0 - 2 = 0$
- $v(2) = 1 - 2 = -1$ • $v(6) = 1 - 2 = -1$

6) Выбрать v_1 и v_2 из разных подграфов с максимальной g :

$$g_2 = g(2, 6) = -1 + (-1) - 2 \cdot 0 = -2$$

7) Поменять местами 2 и 6, зафиксировать их



Алгоритм Кернигана-Лина

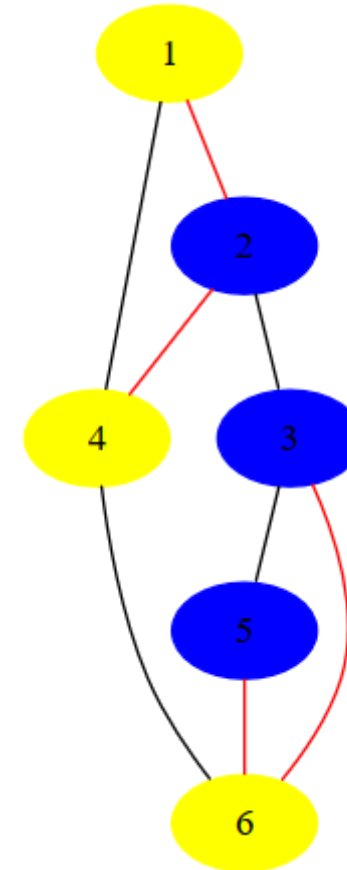
8) Посчитать $D(v)$ для незафиксированных вершин:

- $v(1) = 1 - 1 = 0$ • $v(5) = 1 - 1 = 0$

9) Выбрать v_1 и v_2 из разных подграфов с максимальной g :

$$g_3 = g(1, 5) = 0 + 0 - 2 * 0 = -2$$

10) Поменять местами 1 и 5, зафиксировать их

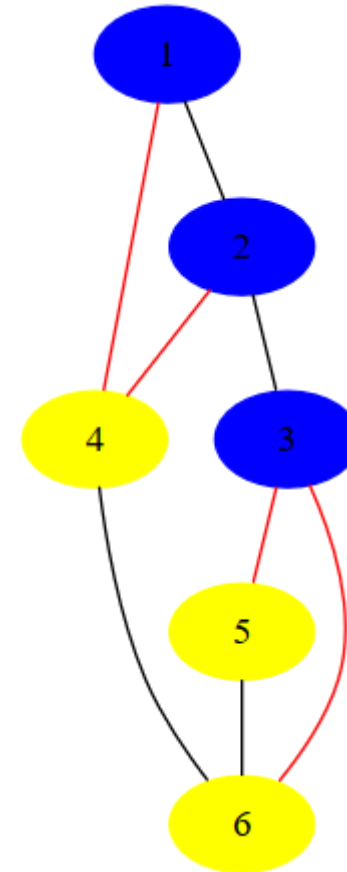


Алгоритм Кернигана-Лина

11) Когда зафиксированных вершин не осталось, посчитать все G :

- $G_1 = g_1 = 2$
- $G_2 = g_1 + g_2 = 2 - 2 = 0$
- $G_3 = g_1 + g_2 + g_3 = 2 - 2 + 0 = 0$

12) Выбрать максимальное G . Так как оно больше 0, то продолжаем шаг, взяв граф полученный после выбранного G .

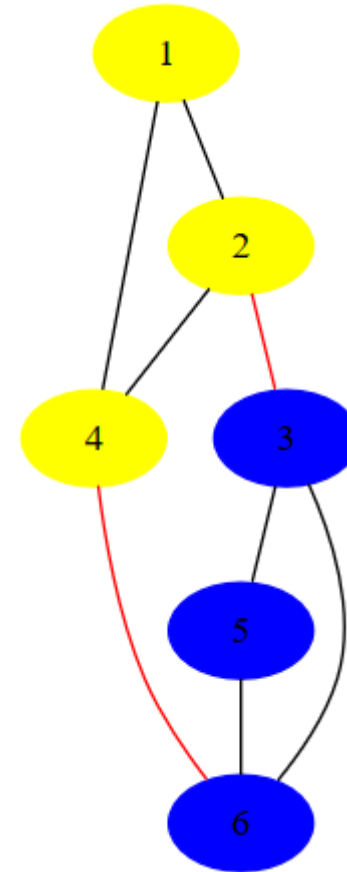


Алгоритм Кернигана-Лина

13) Все вершины снова незафиксированы. Повторяем предыдущие шаги.

...

Останавливаемся либо когда максимальная $G \leq 0$, либо разбиение после максимального G то же, что и начальное на данном шаге.



Модификации алгоритма Кернигана-Лина

- Разбиение графов с нечетным числом вершин

Модификации алгоритма Кернигана-Лина

- Разбиение графов с нечетным числом вершин
 - Добавить «фальшивую» вершину, не соединенную ни с одной вершиной графа

Модификации алгоритма Кернигана-Лина

- Разбиение графов с нечетным числом вершин
 - Добавить «фальшивую» вершину, не соединенную ни с одной вершиной графа
- Разбиение графа с весами вершин

Модификации алгоритма Кернигана-Лина

- Разбиение графов с нечетным числом вершин
 - Добавить «фальшивую» вершину, не соединенную ни с одной вершиной графа
- Разбиение графа с весами вершин
 - Вершина с весом n превращается в клику из n вершин, связи между ними имеют очень большой вес

Модификации алгоритма Кернигана-Лина

- Разбиение графов с нечетным числом вершин
 - Добавить «фальшивую» вершину, не соединенную ни с одной вершиной графа
- Разбиение графа с весами вершин
 - Вершина с весом n превращается в клику из n вершин, связи между ними имеют очень большой вес
- Несбалансированное разбиение графа

Модификации алгоритма Кернигана-Лина

- Разбиение графов с нечетным числом вершин
 - Добавить «фальшивую» вершину, не соединенную ни с одной вершиной графа
- Разбиение графа с весами вершин
 - Вершина с весом n превращается в клику из n вершин, связи между ними имеют очень большой вес
- Несбалансированное разбиение графа
 - Добавить «фальшивые» соединенные вершины с большими весами ребер между ними

Алгоритма Кернигана-Лина для k -разбиения

1. k – степень двойки

- Рекурсивно разбиваем каждый из подграфов, пока их число не достигнет необходимого

Алгоритма Кернигана-Лина для k -разбиения

1. k – степень двойки

- Рекурсивно разбиваем каждый из подграфов, пока их число не достигнет необходимого

2. k – не степень двойки

- Производим случайное разбиение графа на k подграфов. Далее, применяем обычный алгоритм Кернигана-Лина для всех пар.

Параллельный алгоритм Кернигана-Лина

Шаги, выполняемые в параллели:

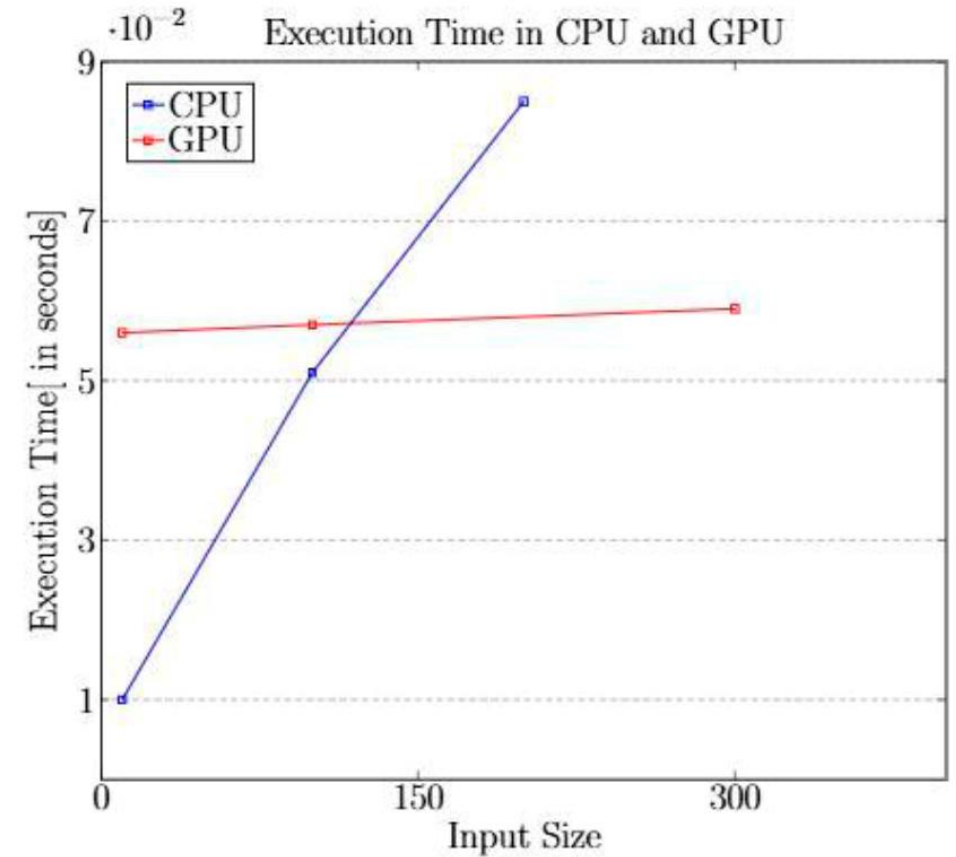
- Просчет V для незафиксированных вершин
- Просчет g для всех незафиксированных вершин из разных подграфов

Шаги, выполняемые последовательно:

- Начальное разбиение
- Выбор максимального G и переход на следующий шаг

Сравнение алгоритма на CPU и GPU

Time Complexity of KL algorithm		
Input Size	Execution Time in CPU (seconds)	Execution Time In GPU (seconds)
10*10	0.01	0.056
100*100	0.076	0.058
300*300	0.828	0.059
500*500	2.852	0.059
800*800	10.003	0.060
1000*1000	18.76	0.060
2000*2000	24.36	0.062



Другие варианты

Другие варианты

- (1) **Until** graph is small enough
 graph := coarsen(graph)
- (2) Partition graph
- (3) **Until** graph = original graph
 graph := uncoarsen(graph)
 partition := uncoarsen(partition)
 locally refine partition if desired

Укрупнение графа

Укрупнение графа $G = (V, E)$ с весами ребер ω и весами вершин ζ – отображение $\pi: G \rightarrow G'$ такое, что:

1. $G' = (V', E')$ с весами ребер ω' и весами вершин ζ' :
2. $\pi(V) = V'$
3. $\pi(E) = \{\{\pi(u), \pi(v)\} \mid \{u, v\} \in E\} = E'$
4. Для $v' \in V'$: $\zeta'(v') = \sum_{v \in V, \pi(v) = v'} \zeta(v)$
5. Для $e' \in E'$: $\omega'(e') = \sum_{\substack{\{u, v\} \in E, \\ \{\pi(u), \pi(v)\} = e'}} \omega(\{u, v\})$

Параллельный алгоритм укрупнения

v	1	2	3	4	5	6	7	8	9	10
μ	7	2	4	7	4	15	4	2	15	9

Параллельный алгоритм укрупнения

v	2	8	3	5	7	1	4	10	6	9
μ	2	2	4	4	4	7	7	9	15	15

Параллельный алгоритм укрупнения

v	2	8	3	5	7	1	4	10	6	9
μ	2	2	4	4	4	7	7	9	15	15
i	1	0	1	0	0	1	0	1	1	0

Параллельный алгоритм укрупнения

v	2	8	3	5	7	1	4	10	6	9
μ	2	2	4	4	4	7	7	9	15	15
i	1	0	1	0	0	1	0	1	1	0

v'	1'	2'	3'	4'	5'
π^{-1}	1	3	6	8	9

Параллельный алгоритм укрупнения

Algorithm 2 Parallel coarsening algorithm on the GPU, given a graph G with $V = (1, 2, \dots, |V|)$ and a map $\mu : V \rightarrow \mathbf{N}$, this algorithm creates a graph coarsening π , G' compatible with μ .

```
1:  $\rho \leftarrow V$ 
2:  $(\rho, \mu) \leftarrow \text{parallel\_sort\_by\_key}(\rho, \mu)$ 
3:  $\mu \leftarrow \text{parallel\_adjacent\_not\_equal}(\mu)$ 
4:  $\pi^{-1} \leftarrow \text{parallel\_copy\_index\_if\_nonzero}(\mu)$ 
5:  $V' \leftarrow (1, 2, \dots, |\pi^{-1}|)$ 
6: append $(\pi^{-1}, |V| + 1)$ 
7:  $\mu \leftarrow \text{parallel\_inclusive\_scan}(\mu)$ 
8:  $\pi \leftarrow \text{parallel\_scatter}(\rho, \mu)$ 
9: for  $v' \in V'$  parallel do {Sum vertex weights.}
10:    $\zeta'(v') \leftarrow 0$ 
11:   for  $i = \pi^{-1}(v')$  to  $\pi^{-1}(v' + 1) - 1$  do
12:      $\zeta'(v') \leftarrow \zeta'(v') + \zeta(\rho(i))$ 
13: for  $v' \in V'$  parallel do {Copy neighbours.}
14:    $V_{v'}^{\omega'} \leftarrow \emptyset$ 
15:   for  $i = \pi^{-1}(v')$  to  $\pi^{-1}(v' + 1) - 1$  do
16:     for  $(u, \omega) \in V_{\rho(i)}^\omega$  do
17:       append $(V_{v'}^{\omega'}, (\pi(u), \omega))$ 
18: for  $v' \in V'$  parallel do {Compress neighbours.}
19:    $V_{v'}^{\omega'} \leftarrow \text{compress\_neighbours}(V_{v'}^{\omega'})$ 
```


Параллельный алгоритм укрупнения

Для новых вершин v' вычисляем их вес — складываем веса вершин, из которых они состоят.

Для всех вершин v и их соседей u составляем список соседей, состоящий из пар $(\pi(u), \omega)$, добавляя его в v' .

Для укрупненного графа и вершины v' сжимаем список соседей, у которых первый элемент пары совпадает:

$$(u_1, \omega_1), (u_1, \omega_2), (u_1, \omega_3), \dots \rightarrow (u_1, \omega_1 + \omega_2 + \omega_3 + \dots)$$

Разбиение укрупненного графа

У укрупненного графа есть веса вершин – алгоритм Кернигана-Лина не подойдет.

Разбиение укрупненного графа

У укрупненного графа есть веса вершин – алгоритм Кернигана-Лина не подойдет.

Использовать другой алгоритм разбиения:

1. Алгоритм Фидуччи-Маттейсеса
2. Алгоритм спектрального разбиения
3. ...

Алгоритм Фидуччи-Маттейсеса (Fiduccia-Mattheyses)

Вводим критерий баланса.

Вместо замены используем перемещение одной вершины в противоположный подграф – выполняем, если после этого будет выполняться условие баланса. Какую ячейку передвигать выбираем по максимальной $D(V)$.

Список литературы

- B. W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs
- C. M. Fiduccia, R. M. Mattheyses, A Linear-Time Heuristic for Improving Network Partitions
- B. O. Fagginger Auer, R. H. Bisseling, Graph Coarsening and Clustering on the GPU
- B. Hendrickson, R. Leland, A multilevel algorithm for partitioning graphs
- A. K. Rajan, D. Bhaiya, VLSI Partitioning using Parallel Kernighan-Lin Algorithm
- Y. Wang, J. Owens, Large-Scale Graph Processing Algorithms on the GPU