Graph Partitioning Algorithms

Leonid E. Zhukov

School of Applied Mathematics and Information Science
National Research University Higher School of Economics

03.03.2014
Minimum Cut

- **Graph**: $G(E, V)$, $|E| = m$, $|V| = n$,
- **Graph cut** - partition vertices in two disjoint subsets: $V = V_1 + V_2$
- **Cut-set** of the cut is a set of edges with endpoints in different partitions.
- Cut size - the number of edges in cut-set

$$\text{cut}(V_1, V_2) = \sum_{i \in V_1, j \in V_2} e_{ij}$$

- **Min cut** is the smallest possible cut in the graph: $\min \text{cut}(V_1, V_2)$
Randomized Min Cut

Vertex contraction: replace connected \((v_1, v_2) \rightarrow v\), edges \(E_v = E_{v_1} \cup E_{v_2}\)
Randomized Min Cut

David Karger, 1993

Algorithm: Randomized Min-Cut

Input: Graph $G(V,E)$

Output: Graph min cut-set

repeat
  pick a random edge $e$ in $G$;
  contract its endpoints, $G' \leftarrow G \setminus e$;
until two vertices remain;
Randomized Min Cut

- Vertex contraction: $G' \leftarrow G \setminus e$
- For any cut in $G'$ there is a cut in $G$ with the same size (converse not true)
- Collapsing an edge can not decrease minimum cut size,

\[ \min \text{cut}(G') \geq \min \text{cut}(G) \]
Randomized Min Cut

Diagram of a graph showing different cuts and their corresponding components.
Randomized Min Cut

Graph $G(m, n)$, $m$-edges, $n$-nodes
Let $\text{min}(\text{cut}) = k$, then every node has degree $k_i \geq k$
and there $m = \frac{1}{2} \sum_i k_i \geq \frac{nk}{2}$ edges in the graph
Probability randomly select an edge from the min cut $P(e \in \text{min cut}) = \frac{k}{m}$
Let $E_i$ event that on $i$-th step selected edge is not in min cut.

\[
P(E_1) \geq 1 - \frac{k}{m} \geq 1 - \frac{2}{n} = \frac{n - 2}{n}
\]
\[
P(E_2|E_1) \geq 1 - \frac{2}{n-1} = \frac{n - 3}{n - 1}
\]
\[
P(E_i|E_1 \cup E_2 \cup ...E_{i-1}) \geq 1 - \frac{2}{n - i + 1} = \frac{n - i - 1}{n - i + 1}
\]

\[
P(E_1 \cup ..E_{n-2}) = P(E_1)P(E_2|E_1)....P(E_{n-2}|E_1 \cup E_2 \cup ...E_{n-3}) \geq \frac{n - 2}{n} \cdot \frac{n - 3}{n - 1} \cdot \frac{n - 4}{n - 2} \cdot \frac{n - 5}{n - 3} \ldots \frac{2}{4} = \prod_{i=1}^{n-2} \frac{n - i - 1}{n - i + 1} = \frac{2}{n(n - 1)}
\]
Randomized Min Cut

- Probability of success - all selected edges are not in min cut (what’s left is min cut)
  \[ P(\text{success}) \geq \frac{2}{n(n-1)} \]

- Probability of not finding min cut after \( N \sim n^2/2 \) independent runs:
  \[ P(\text{error}) \leq \left(1 - \frac{2}{n(n-1)}\right)^{n^2/2} \sim \frac{1}{e} = 0.37 \]

- With \( N = c \frac{n(n-1)}{2} \log n \) independent runs
  \[ P(\text{error}) \leq \frac{1}{n^c} \]
Multilevel Graph Partitioning

G. Karypis, 1998
- Matching - independent edge set, i.e. set of edges without common vertices
- Maximal matching - if one more edge added, it is no longer a matching
Coarsening schemes

- random matching, heavy-edge matching, light-edge matching
- multinode/hypergraph
- heavy clique matching (max density)
Multilevel Graph Partitioning

George Karypis, 1998

Algorithm: Multilevel graph partitioning

Input: Graph $G(V, E)$

Output: Graph partition

1. Coarsening: $G_0 \rightarrow G_1 \rightarrow G_2 \ldots \rightarrow G_m$, such that $|V_0| > |V_1| > |V_2| > \ldots > |V_m|$

2. Partition: $P_m$

3. Uncoarsening: $P_m \rightarrow P_{m-1} \rightarrow P_{m-2} \ldots \rightarrow P_0$

- coarsening is done by randomized maximal matching
- partition on coarse graph can be done by greedy or advanced algorithms
## Local clustering algorithm

- **Conductance of the vertex set** $S$

  $$\phi(S) = \frac{\text{cut}(S, V\setminus S)}{\min(\text{vol}(S), \text{vol}(S\setminus V))}$$

  where $\text{vol}(S) = \sum_{i \in S} k_i$ - sum of all node degrees in the set

- Conductance is a probability of picking up an edge from the smaller set that crosses the cut. It is also probability that one-step random walk starting in the cluster will leave the cluster

- **Example:** $\text{cut}(S) = 7$, $\text{vol}(S) = 33$, $\text{vol}(V\setminus S) = 11$, $\phi(S) = 7/11$
• Cheeger inequality

\[
\frac{1}{2}\lambda_2 \leq \min_{S \in V} \phi(S) \leq \sqrt{2}\lambda_2
\]

• \(\lambda_2\) - second smallest eigenvalue of normalizes graph Laplacian

\[
L = D^{-1/2}(D - A)D^{-1/2}, \quad D = \text{diag}(d(i))
\]
Local clustering algorithm
Spielman, 2003/2008

Algorithm: Nibble

Input: Graph $G$, $q_0(v_0), \phi_0$
Output: Graph partition $S$

\begin{algorithm}
\For{t = 1 : t_{last}}{
    $q_t = Mr_{t-1}$;
    $r_t(i) = q_t(i)$ \text{ if } q_t(i)/d(i) > \epsilon, \text{ else } 0$;
    sort $i$ descending based on $r_{t_{last}}(i)/d(i)$;
    sweep from top $\phi(S\{i = 1..j\}) < \phi_0$ or $\phi(S\{i = j + 1..n\}) < \phi_0$;
}\end{algorithm}

- Random walk:

\[ M = (AD^{-1} + I)/2, \quad D = \text{diag}(d(i)) \]
Global min-cuts in RNC, and other ramifications of a simple min-cut algorithm, D.R. Karger, SODA ’93, pp 21-30, 1993

Multilevel algorithms for partitioning power-law graphs, A. Abou-Rjeili, G. Karypis. IPDPS ’06, p 124, 2006

Daniel A. Spielman, Shang-Hua Teng: Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. STOC 2004: 81-90
