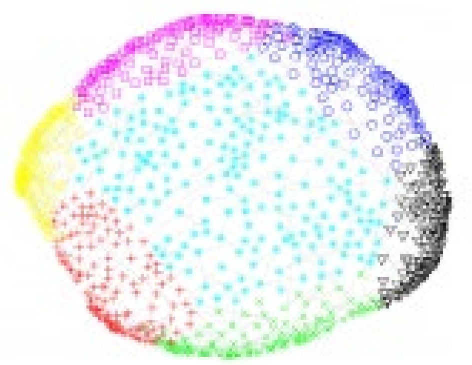
Graph Partition

Hanqing Zhu
07/15/2019

Graph Partition Problem



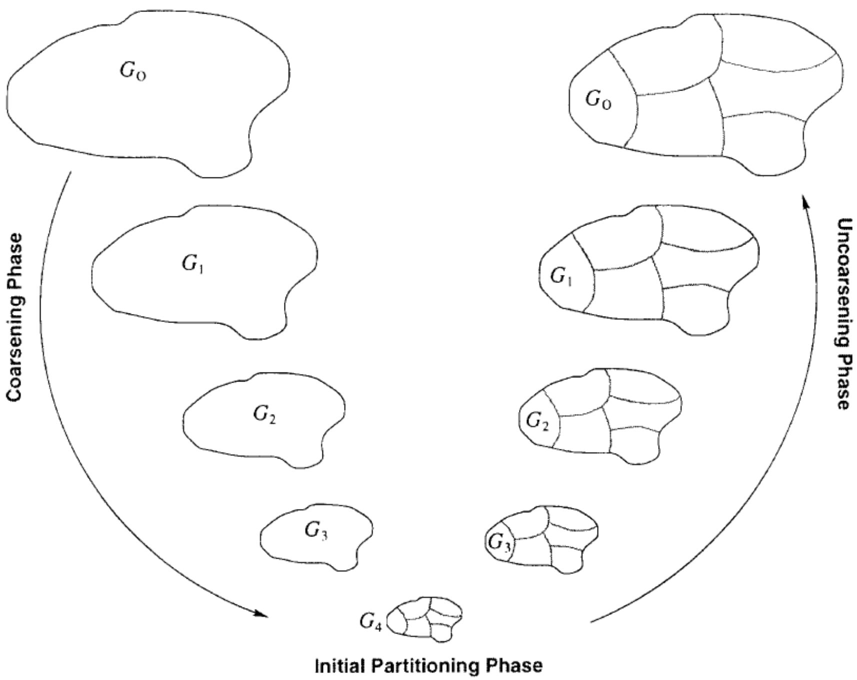
Example: Graph Partition Result

- Given
  - Graph G=(V,E), n=|V| vertices, E=edges
  - Possibly weights (W_V, W_E)
  - Edge e = (u , v)
- Objective: For a (k, v) partition problem , choose a partition $\mathrm{V} = V_1 \cup V_2 \cup \cdots \cup V_\mathrm{k}$
  - The sum of vertices in each V_j is "about the same"
  - The sum of all edge weights of edges connecting all different pairs V_i and V_j is minimized

Graph Partition Problem

- Graph Partition Problem Complexity
  - Choose optimal partition : $NP - hard\ Problem$
  - Heuristics and approximation algorithms are generally used

- METIS/ hMETIS →graph/hypergraph → edge-cut minimization
  - Multi-level methods
  - Idea:



Analogy to k-means Algorithm

- Target: To group the data into a few cohesive “cluster”

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence: {

   For every i, set
   $$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

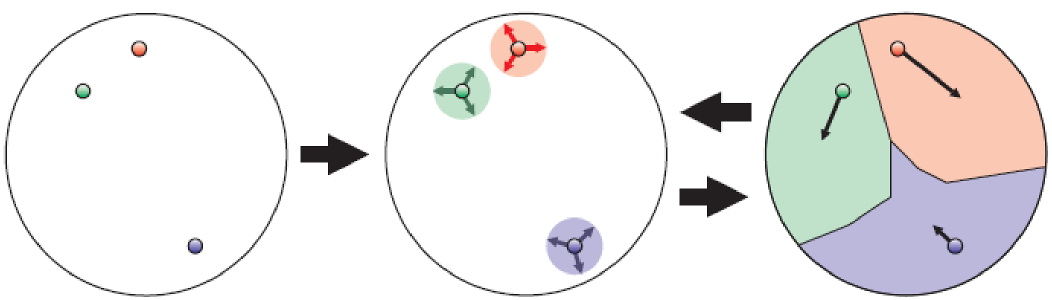
   For each j, set
   $$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$$

   }

- For Graph Partition Problem:
  - How to define the **centroid** in graph?
  - How to define the **“distance”** between centroid and other node ?

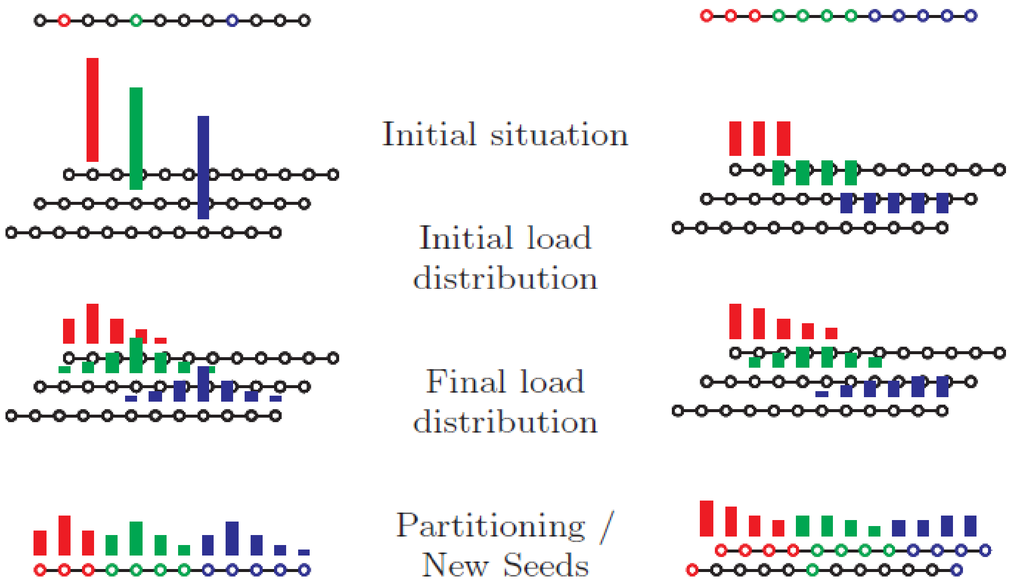
Diffusive Framework

- To start with an initial, often randomly chosen vertex(seed) <u>per partition</u>
- Then, all sub-domains grow simultaneously
- Last, all vertices of the graph→ a partition, each component computes its new center
- Several ways can be used to implement the three steps
  - Randomly choose initial vertices?
  - Distribute initial evenly by using BFS to find a farthest seed from prior seed (serially search)



Learning bubble framework

Diffusive Mechanisms



The growth and seed determination process

- Realize the growth process via iterations of a diffusive process
  - Load $\xrightarrow{\text{diffuse into}}$ densely connected regions,
- Two important properties required
  - “hill - like” load contribution
  - Connectivity also affects load distribution (not only distance)

Diffusion scheme

- Given $G = (V, E)$ an unweighted , connected graph with $n = |V|$ vertices
  - $\boldsymbol{A}$: the adjacency matrix of the graph
  - $\boldsymbol{D}$: the diagonal matrix $\mathrm{D} = diag(\boldsymbol{Ae}), where\ vector\ e = (1, \dots, 1)^T$
  - $\boldsymbol{L}$: the Laplacian matrix $\mathrm{L} = D - A$
  - $\boldsymbol{M}$: the diffusion matrix $M = \mathrm{I} - \alpha L, (0 < \alpha < 1, a\ constant\ param)$
- FOS scheme (first order scheme)
  - For each

$$
\begin{aligned}
f_{e=(u,v)}^{(i)} &= \alpha \cdot \left(w_u^{(i)} - w_v^{(i)} \right) \\
w_v^{(i+1)} &= w_v^{(i)} - \sum_{e=(v,*)} f_e^{(i)}
\end{aligned}
$$

$$\Downarrow$$

$$w^{(i+1)} = \mathbf{M} w^{(i)}$$

Diffusion with Constant Draining

- FOS/C scheme performs two operation
  - Origin diffusion step
  - Shift a small load amount $\delta > 0$ from all vertices of the graph to some selected source vertices (seed) $S \subset V$

$$d_v = \begin{cases} -\delta & : \ v \notin S \\ \delta \cdot |V|/|S| - \delta & : \ \text{otherwise} \end{cases}$$

- FOS/C scheme

$$\begin{aligned} f^{(i)}_{e=(u,v)} &= \alpha \cdot \left(w^{(i)}_u - w^{(i)}_v \right) \\ w^{(i+1)}_v &= w^{(i)}_v - \sum_{e=(v,*)} f^{(i)}_e + d_v \end{aligned} \Rightarrow w^{(i+1)} = \mathbf{M} w^{(i)} + d.$$

- FOS/C Convergence state
  - Solve the linear system :

Lw* = *d

$$\begin{aligned} w^{(*)} &= \mathbf{M} w^{(*)} + d \\ \Leftrightarrow (\mathbf{I} - \mathbf{M}) w^{(*)} &= d \\ \Leftrightarrow \alpha \mathbf{L} w^{(*)} &= d \end{aligned}$$

Result

- During the learning process, the partition centers are directed into areas which are dense regions with more load diffusing into → the partition boundaries tend to be in sparser regions → reduce the number of boundary vertices → improve the partition quality
- Run time is too long

Graph	Metis			Jostle			Bubble-FOS/C		
	Time	Cut	Boundary	Time	Cut	Boundary	Time	Cut	Boundary
biplane9	0.03 s	670	1142	0.15 s	647	1104	7.10 s	672	955
crack	0.02 s	1041	1030	0.06 s	1031	1018	1.59 s	1017	1004
crack (dual)	0.02 s	466	919	0.08 s	450	893	6.00 s	447	865
grid100x100	0.03 s	584	1006	0.09 s	549	992	2.26 s	575	949
stufe10	0.02 s	570	948	0.15 s	546	919	8.73 s	574	725
shock9	0.07 s	1010	1663	0.19 s	909	1665	15.21 s	961	1480
whitacker	0.01 s	1005	992	0.11 s	966	953	1.60 s	966	957
whitacker (dual)	0.01 s	528	1048	0.11 s	515	1027	5.43 s	493	973

Thank you