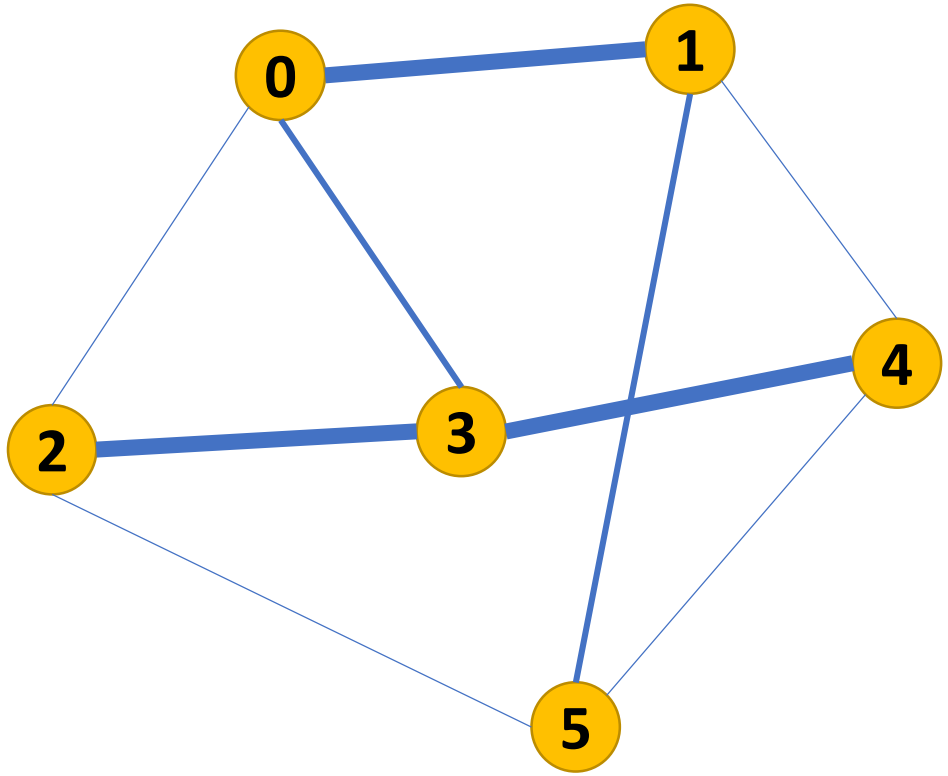
Communication Graph and Process Mapping

Lecture 20

April 6, 2024

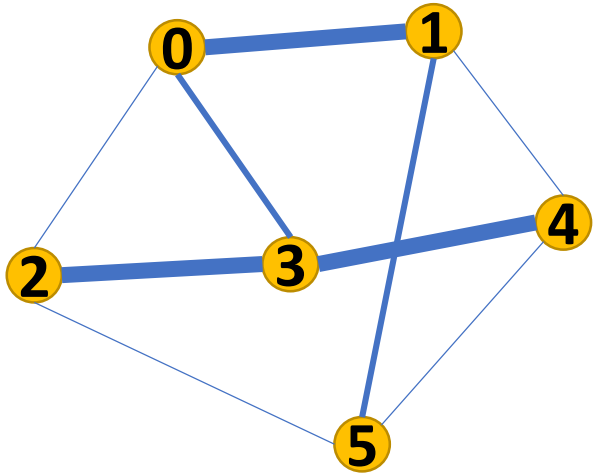
Communication Graph/Matrix

64, 256, 512 bytes (lighter to darker shade)



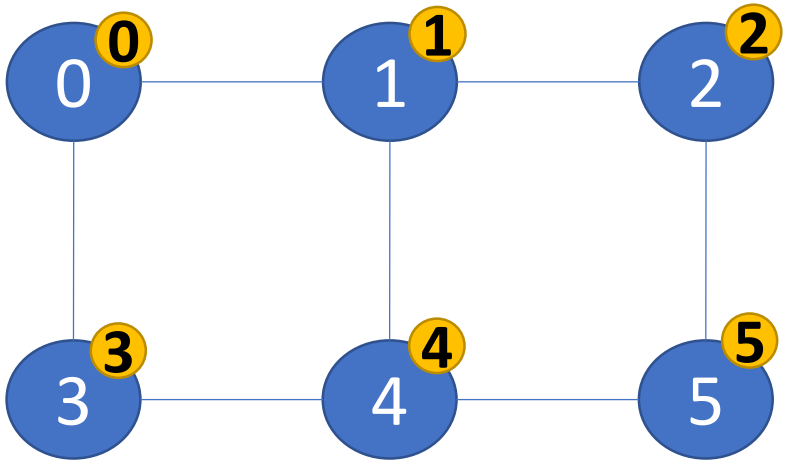
	512	64	256		
512				64	256
64			512		64
256		512		512	
	64		512		64
	256	64		64	

Communication Graph Mapping



	512		256		
512				64	256
			512		64
256		512		512	
	64		512		64
	256	64		64	

Linear mapping



Q1: What are the communicating pairs?

Q2: Distance/hops between the communicating pairs?

Q3: Total hop-bytes?

Process Mapping

- Topology graph
- Task graph
  - c_{ab} → amount of communication (bytes) on e_{ab} between v_a and v_b
- Map: $v_t \in V_t$ placed on $v_p \in Vp$

$$G_p = (V_p, E_p)$$

$$G_t = (V_t, E_t)$$

$$P : V_t \longrightarrow V_p$$

Topology-aware task mapping for reducing communication contention on large parallel machines, Tarun Agarwal, Amit Sharma, Laxmikant V. Kale, IPDPS 2006

Metric 1: Hop-bytes

Total size of inter-processor communication in bytes weighted by distance between the respective end-processors.

The overall hop-bytes is the sum of hop-bytes due to individual nodes in the task graph.

$$HB(G_t, G_p, P) = \frac{1}{2} \sum_{v_a \in V_t} HB(v_a)$$

$$where\ HB(v_a) = \sum_{e_{ab} \in E_t} HB(e_{ab})$$

$$where\ HB(e_{ab}) = c_{ab} \times d_p(P(v_a), P(v_b))$$

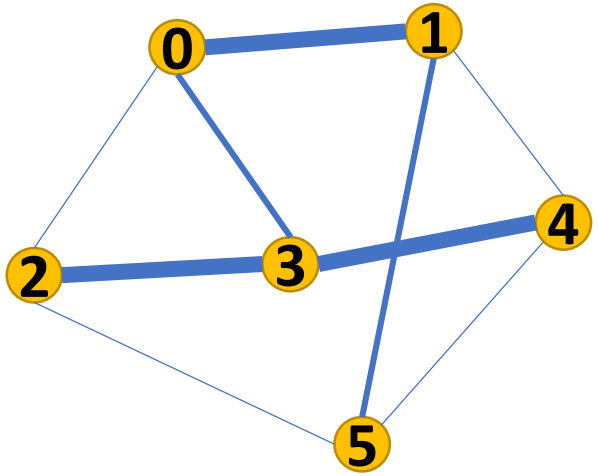
Metric 2: Hops per byte

Average number of network links a byte has to travel under a task mapping

$$Hops\ per\ Byte = \frac{HB(G_t, G_p, P)}{\sum_{e_{ab} \in E_t} c_{ab}}$$

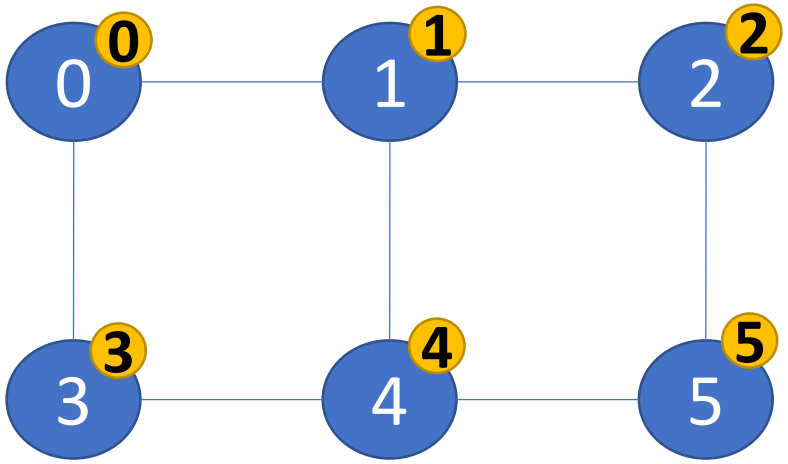
$$Hops\ per\ Byte = \frac{\sum_{e_{ab} \in E_t} c_{ab} \times d_p(P(v_a), P(v_b))}{\sum_{e_{ab} \in E_t} c_{ab}}$$

Communication Graph Mapping



	512	64	256		
512				64	256
64			512		64
256		512		512	
	64		512		64
	256	64		64	

Linear mapping

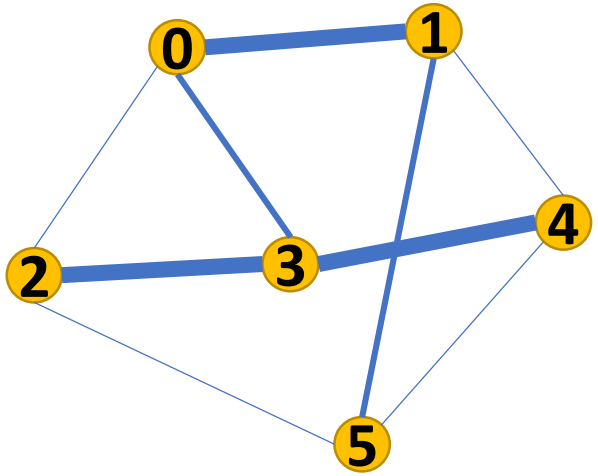


Total hop-bytes?

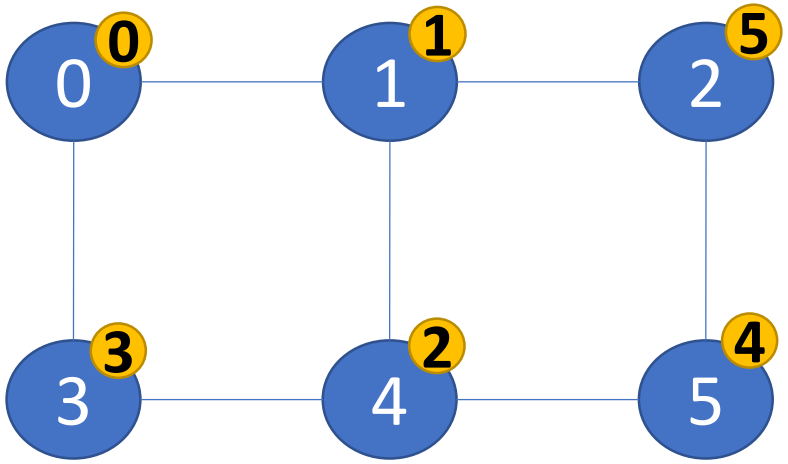
01: 512*1
02: 64*2
03: 256*1
14: 64*1
15: 256*2
23: 512*3
25: 64*1
34: 512*1
45: 64*1

= 3648

Communication Graph Mapping



	512	64	256		
512				64	256
64			512		64
256		512		512	
	64		512		64
	256	64		64	

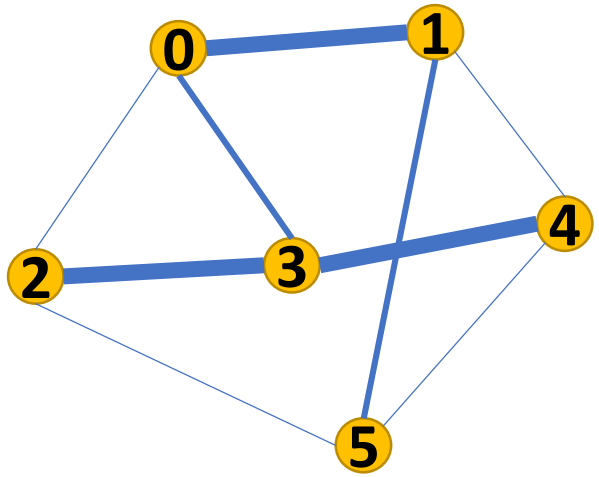


Total hop-bytes?

01: 512*1
02: 64*2
03: 256*1
14: 64*2
15: 256*1
23: 512*1
25: 64*2
34: 512*2
45: 64*1

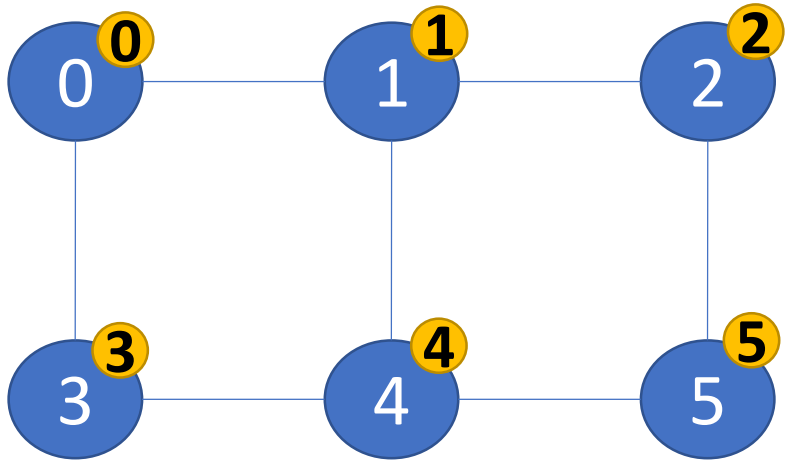
= 3008

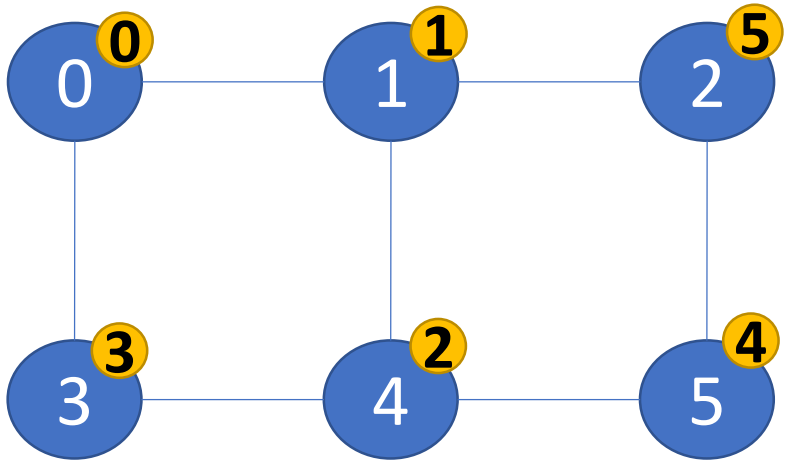
Communication Graph Mapping



	512	64	256		
512				64	256
64			512		64
256		512		512	
	64		512		64
	256	64		64	

Linear mapping





Linear	Towards optimal
01: 512*1	01: 512*1
02: 64*2	02: 64*2
03: 256*1	03: 256*1
14: 64*1	14: 64*2
15: 256*2	15: 256*1
23: 512*3	23: 512*1
25: 64*1	25: 64*2
34: 512*1	34: 512*2
45: 64*1	45: 64*1
= 3648	= 3008

Effect of Mapping

- 16 x 16 x 16 3D torus
  - Diameter: 24
- Average load on links high when #hops large and for large message size

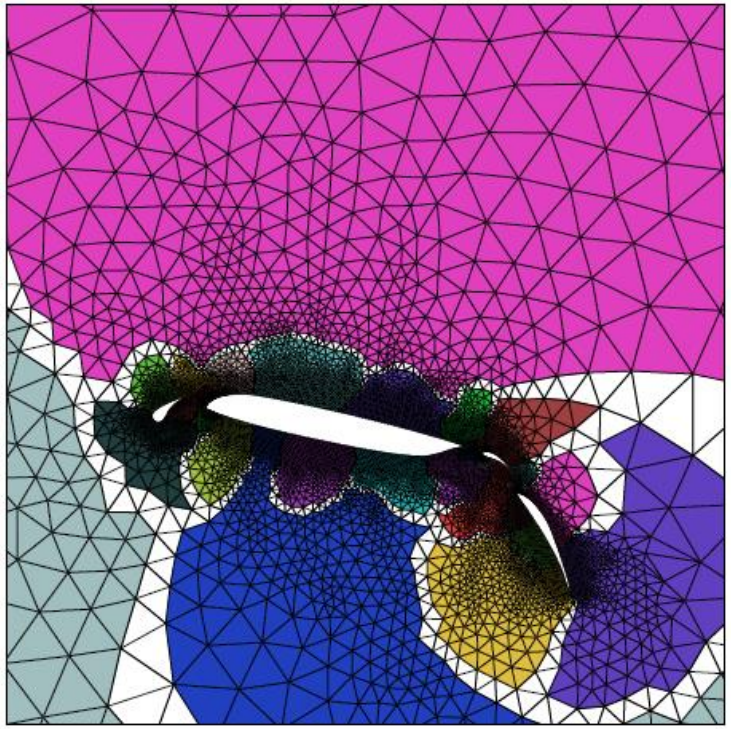
Message Size	Random Mapping	Optimal Mapping	
1KB	56.93ms	46.91ms	17%
10KB	243.64ms	124.56ms	
100KB	2247.75ms	914.72ms	
500KB	11.62s	4.44s	
1MB	23.50s	8.80s	62%

Table 1: Time for 200 iterations of a Jacobi-like program with optimal mapping and random mapping

Topology-aware task mapping for reducing communication contention on large parallel machines, Tarun Agarwal, Amit Sharma, Laxmikant V. Kale, IPDPS 2006

Graph Partitioning

Meshing



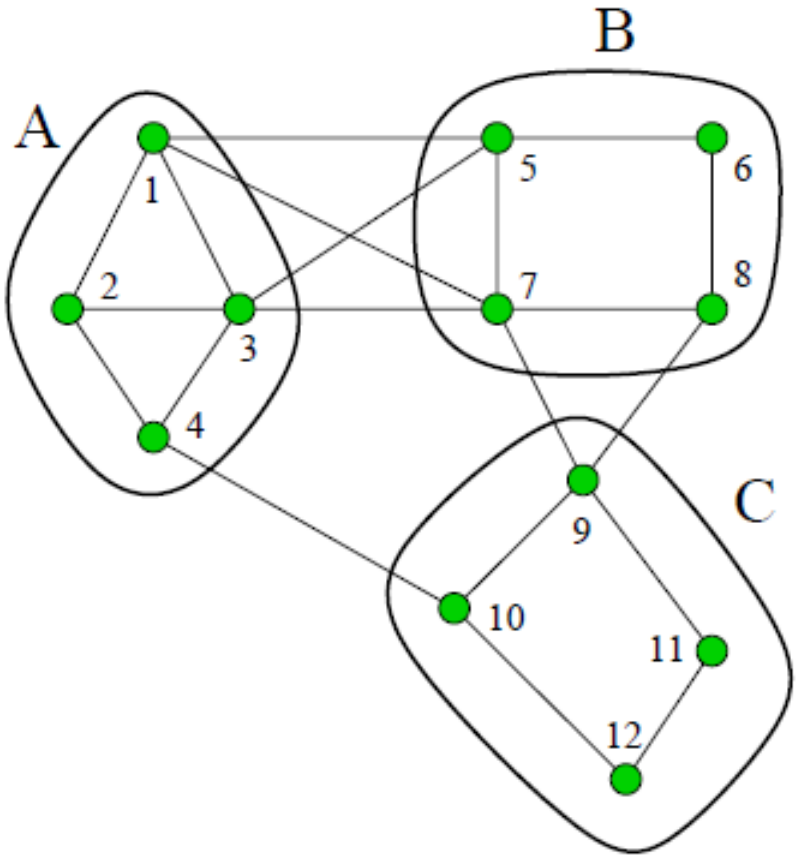
2D irregular mesh of an airfoil *

Q: How can you ensure efficient execution of these simulations?

Q: How do you model irregular mesh computations/communications as graphs?

*Hypergraph-Partitioning-Based Decomposition for Parallel Sparse-Matrix Vector Multiplication

Graph Partitioning



Three subdomains/parts: A, B, C

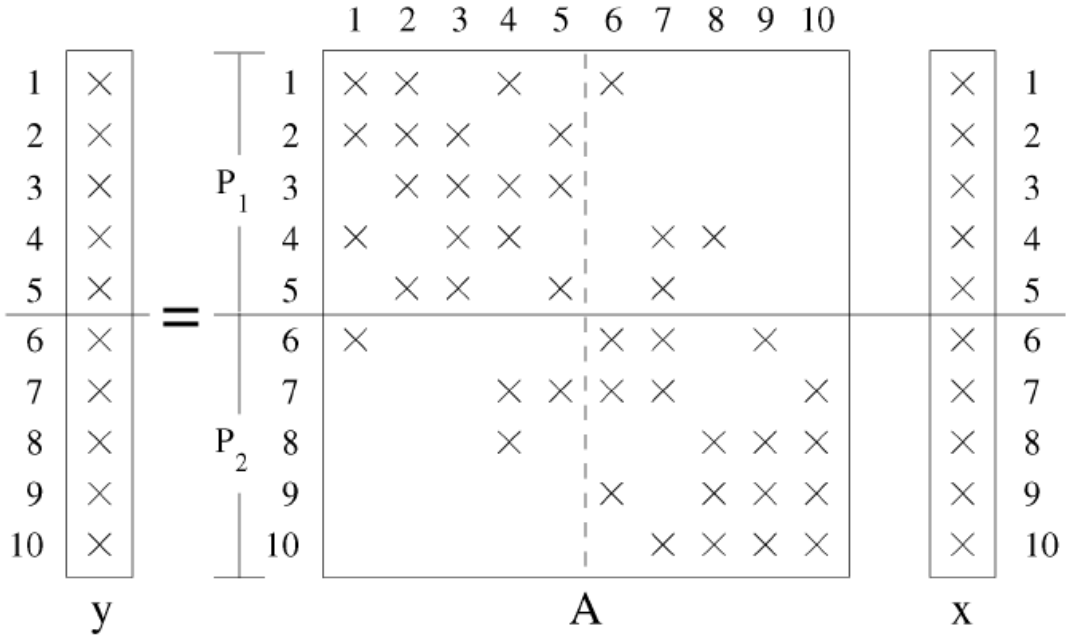
Edge is a *cut* when its vertex pairs belong to two parts

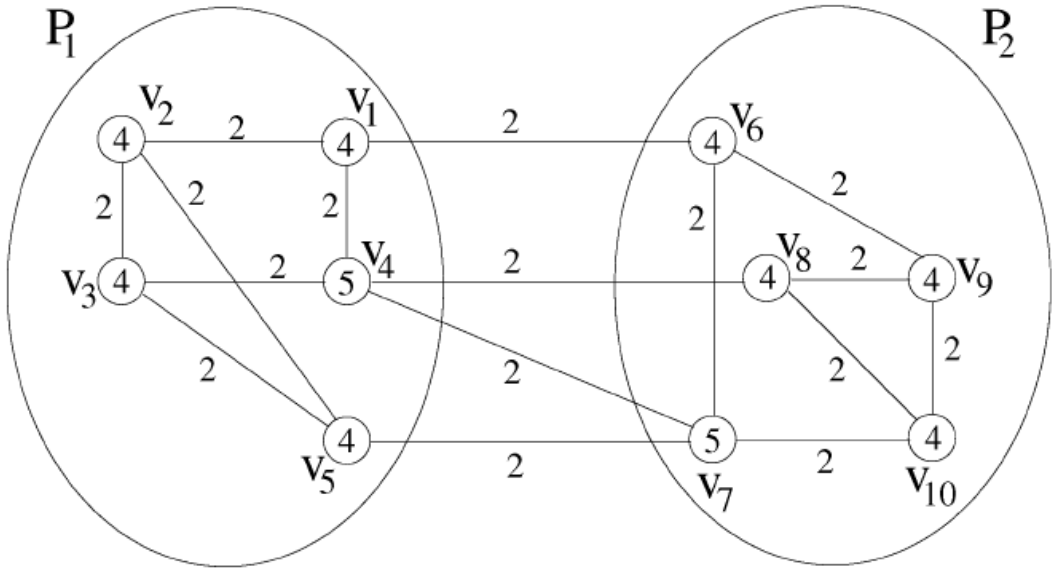
Edge cut = 7

Communication cost ∝ edge-cut

*Hypergraph-Partitioning-Based Decomposition for Parallel Sparse-Matrix Vector Multiplication

Graph Model for SpMV



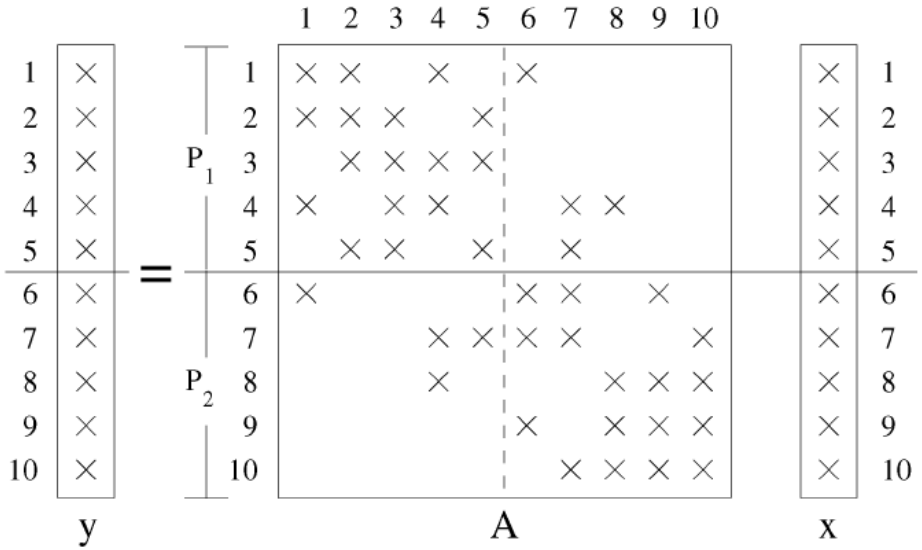


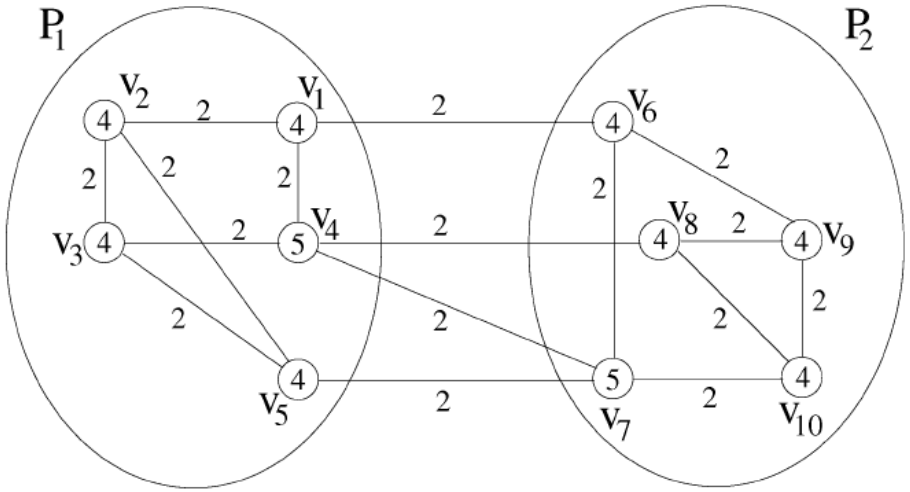
Two-way partitioning of symmetric matrix for SpMV computation

Computational load w_i = Number of non-zero entries in row/column i

Reference: Hypergraph-Partitioning based decomposition for parallel sparse-matrix vector multiplication

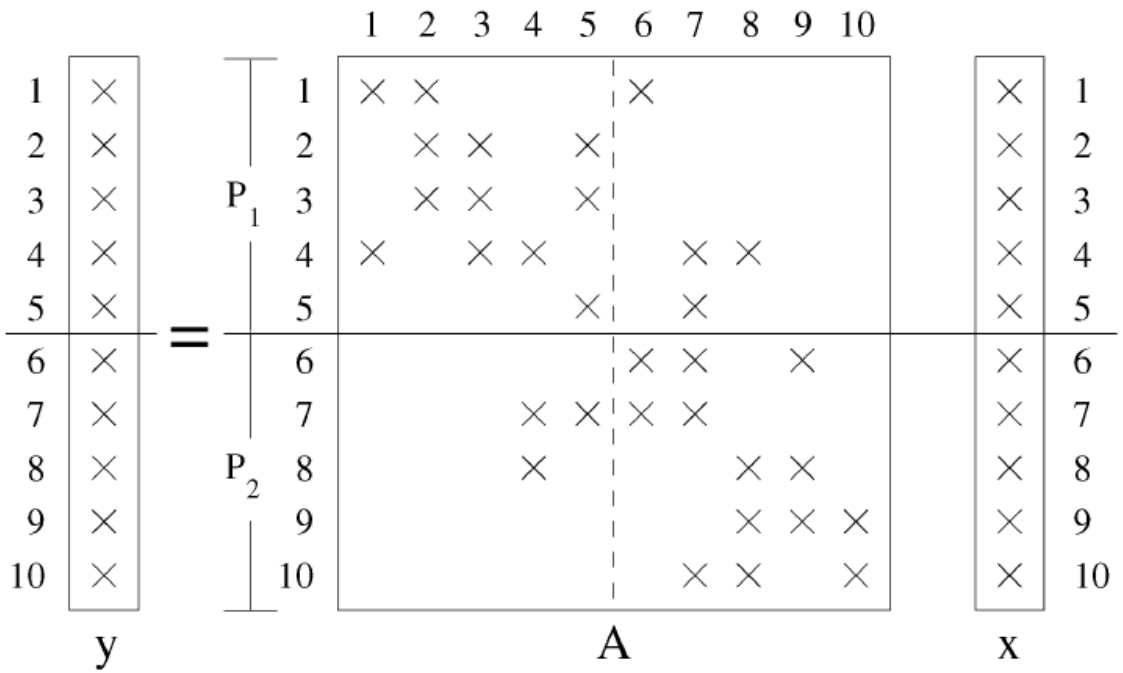
Graph Model for SpMV

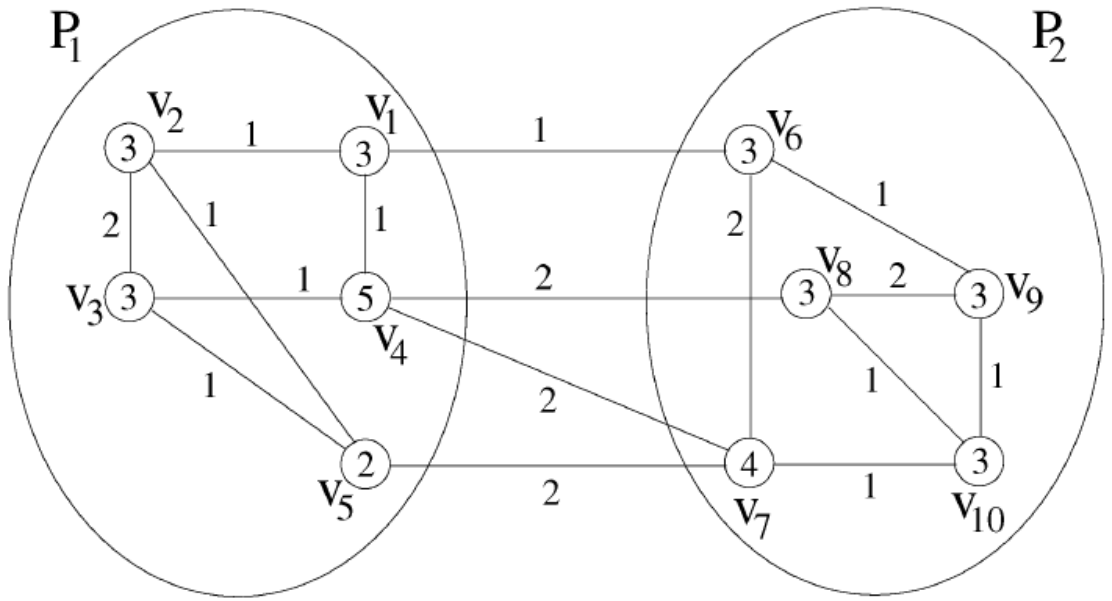




- Computation load?
  - Similar for both processors
- Number of communications?
  - 8 as per this graph
  - Actually 6

Communication Graph





Edge cut = ?

Examples

matrix name	description	number of rows/cols	number of nonzeros									
			total	avg. per row/col	per column				per row			
					min	max	std	cov	min	max	std	cov
Structurally Symmetric Matrices												
SHERMAN3	[9] 3D finite difference grid	5005	20033	4.00	1	7	2.66	0.67	1	7	2.66	0.67
KEN-11	[7] linear programming	14694	82454	5.61	2	243	14.54	2.59	2	243	14.54	2.59
NL	[19] linear programming	7039	105089	14.93	1	361	28.48	1.91	1	361	28.48	1.91
KEN-13	[7] linear programming	28632	161804	5.65	2	339	16.84	2.98	2	339	16.84	2.98
CQ9	[19] linear programming	9278	221590	23.88	1	702	54.46	2.28	1	702	54.46	2.28
CO9	[19] linear programming	10789	249205	23.10	1	707	52.17	2.26	1	707	52.17	2.26
CRE-D	[7] linear programming	8926	372266	41.71	1	845	76.46	1.83	1	845	76.46	1.83
CRE-B	[7] linear programming	9648	398806	41.34	1	904	74.69	1.81	1	904	74.69	1.81
FINAN512	[38] stochastic programming	74752	615774	8.24	3	1449	20.00	2.43	3	1449	20.00	2.43
Structurally Nonsymmetric Matrices												
GEMAT11	[9] optimal power flow	4929	38101	7.73	1	28	2.96	0.38	1	29	3.38	0.44
LHR07	[38] light hydrocarbon recovery	7337	163716	22.31	1	64	26.19	1.17	2	37	16.00	0.72
ONETONE2	[38] nonlinear analog circuit	36057	254595	7.06	2	34	5.13	0.73	2	66	6.67	0.94
LHR14	[38] light hydrocarbon recovery	14270	321988	22.56	1	64	26.26	1.16	2	37	15.98	0.71
ONETONE1	[38] nonlinear analog circuit	36057	368055	10.21	2	82	14.32	1.40	2	162	17.85	1.75
LHR17	[38] light hydrocarbon recovery	17576	399500	22.73	1	64	26.32	1.16	2	37	15.96	0.70
LHR34	[38] light hydrocarbon recovery	35152	799064	22.73	1	64	26.32	1.16	2	37	15.96	0.70
BCSSTK32	[9] 3D stiffness matrix	44609	1029655	23.08	1	141	10.10	0.44	1	192	10.45	0.45
BCSSTK30	[9] 3D stiffness matrix	28924	1036208	35.83	1	159	21.99	0.61	1	104	15.27	0.43

Graph partitioning

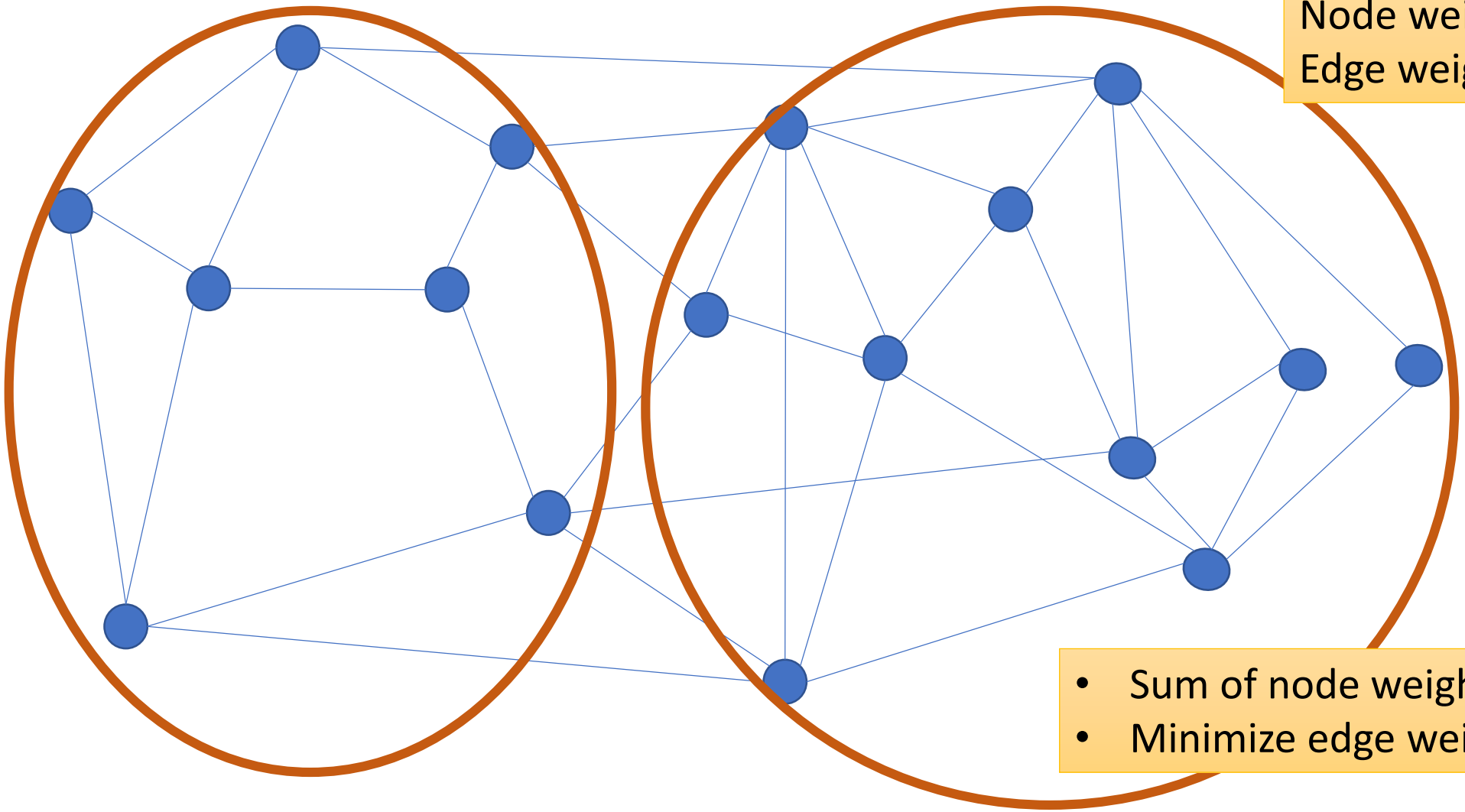
Problem: Partition the graph such that the edge-cut is minimized and partitions are equally-sized.

Techniques

1. Geometric (applicable when coordinate information available)
    - Group together based on neighbourhood
    - Often directly used on meshes

2. Multilevel
    - Group together based on connectivity

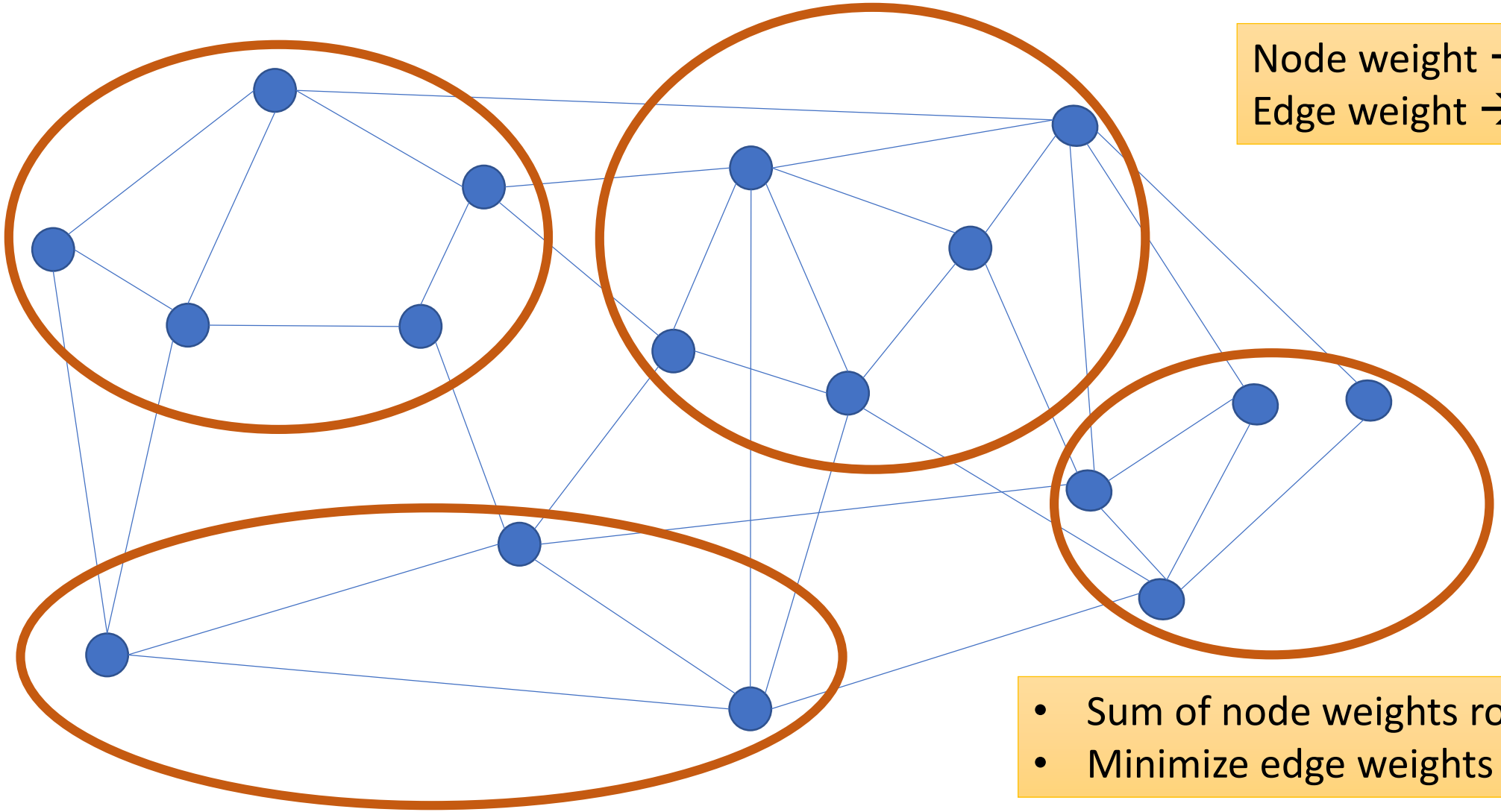
2-way partitioning (NP-hard)



Node weight → Computation
Edge weight → Communication

- Sum of node weights roughly same
- Minimize edge weights across partitions

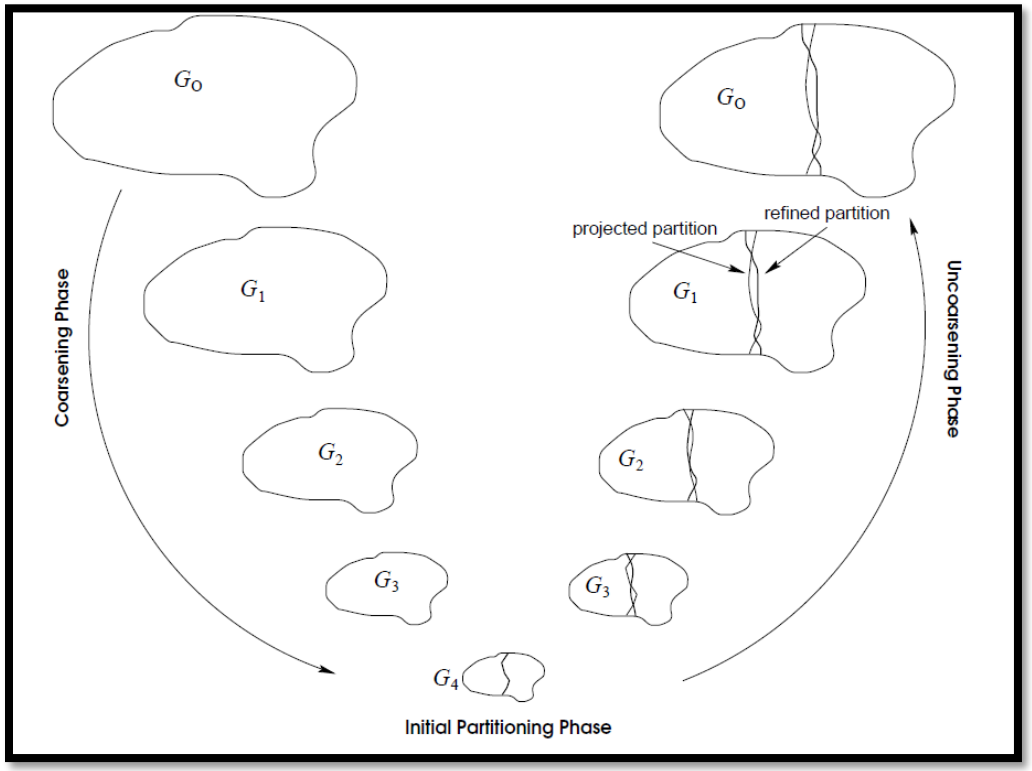
k-way partitioning

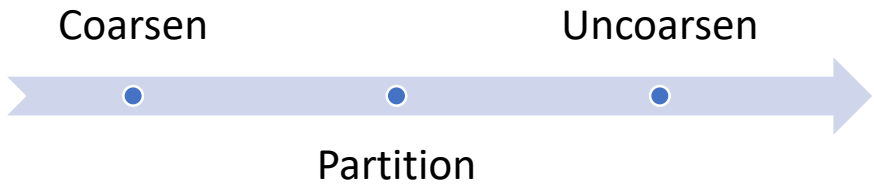


Node weight → Computation
Edge weight → Communication

- Sum of node weights roughly same
- Minimize edge weights across partitions

Multilevel Partitioning for Bisections





RM

HEM

KL

$KL\ refinement$

- G_0 is coarsened to a few hundred vertices
- A set of vertices of G_i is combined to form a single vertex of G_{i+1}
- G_{i+1} is constructed from G_i by finding a matching of G_i
- Maximal matching will ensure large number of edges
- High-quality bisection (2-way partition) of smaller graph is computed
- Smaller graph is uncoarsened – may lead to better edge cuts

REFERENCES:

KARYPIS ET AL. A FAST AND HIGH QUALITY MULTILEVEL SCHEME FOR PARTITIONING IRREGULAR GRAPHS

SCHLOEGEL ET AL., GRAPH PARTITIONING FOR HIGH PERFORMANCE SCIENTIFIC SIMULATIONS

Random and Heavy-edge Matching

1. Visit vertex in any random order
2. If a vertex u is not yet matched, select and unmatched adjacent vertex v
3. Include edge (u, v) in the matching