Graph Partitioning for Multi-phase and Multi-physics Computations

Vipin Kumar
Army HPC Research Center
Department of Computer Science and Engineering
University of Minnesota

Collaborators: Dr. George Karypis and Dr. Kirk Schloegel

Work supported by: ARL, ARO, DOE, NSF, AHPCRC, MSI, IBM, and SGI/Cray

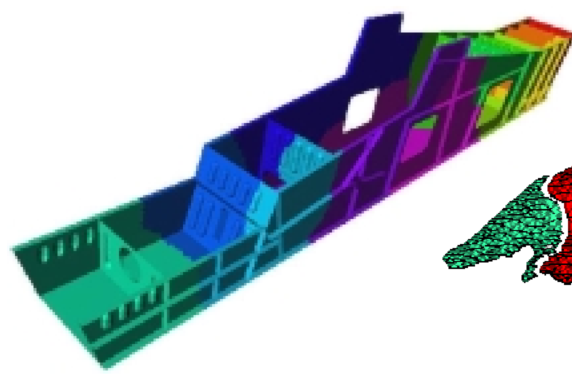
High Performance Scientific Simulation

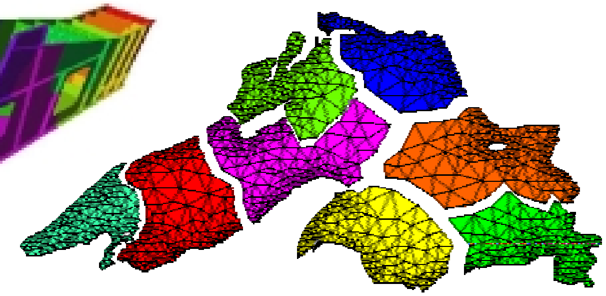
Physical Modeling

Computational Simulation

- **Solvers for large sparse linear systems**
- **Graph partitioning**

Verification & Testing



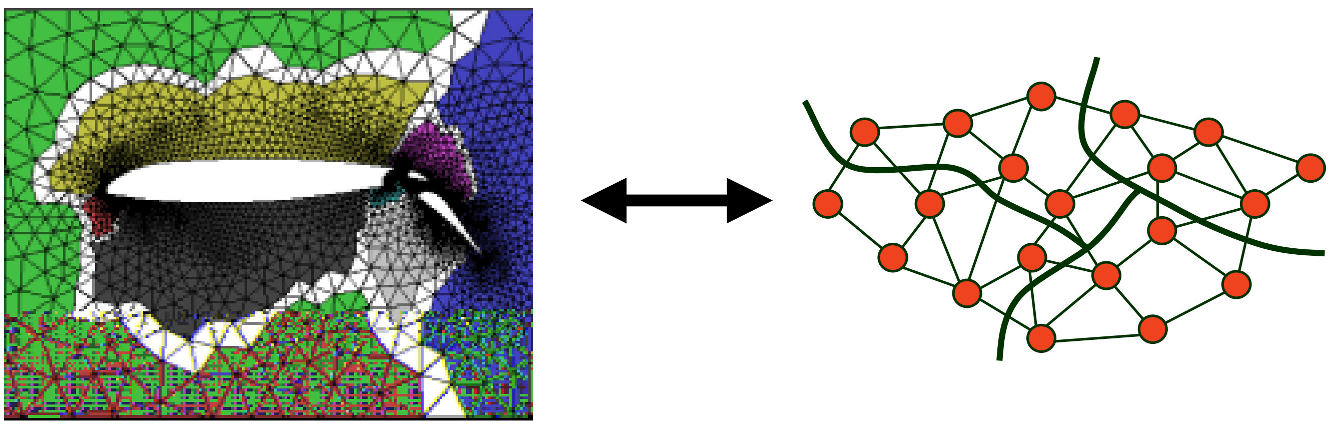


Computational Simulation is now widely accepted as the third path to Science

Scientific Simulation on Parallel Computers

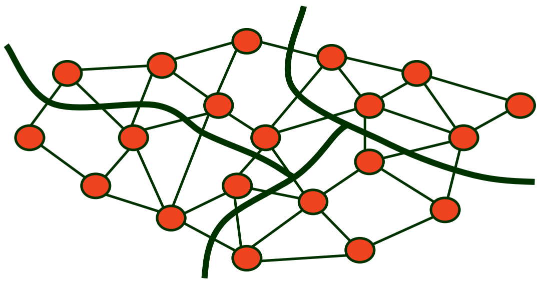
- **Efficient execution of scientific simulations on parallel computers requires a mapping of the computational mesh to the processors such that:**
  - **each processor gets a roughly equal number of mesh elements, and**
  - **the amount of inter-processor communication required to exchange information among adjacent mesh elements is minimized.**

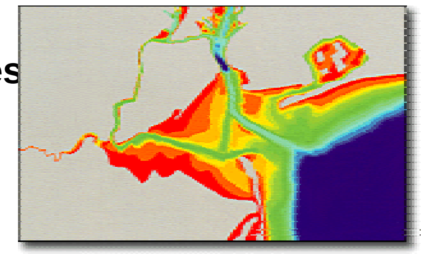
- **This is mapping is typically computed as a pre-processing step by partitioning the graph that models the underlying computation.**

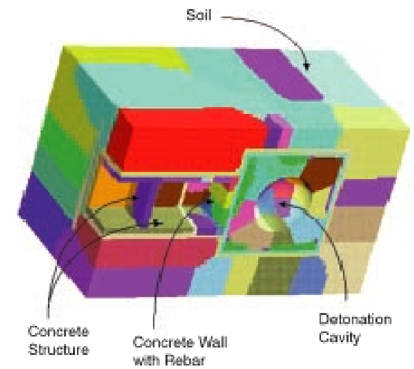


Problem

Given a graph, compute a partitioning so that each subdomain contains a roughly equal number of vertices and the number of edges crossing subdomains is minimized.

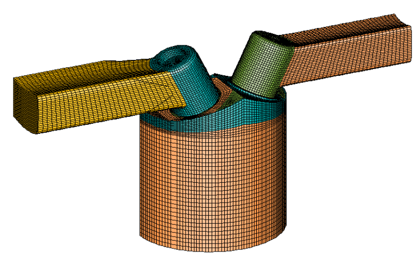


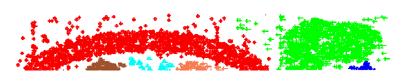




Applications

- **Domain decomposition for executing numerical simulation on highly parallel computers.**
- **Re-ordering matrices to minimize fill during solution of sparse linear systems for scientific simulation.**
- **Data-mining.**
- **VLSI circuit design.**
- **Efficient storage of large databases.**

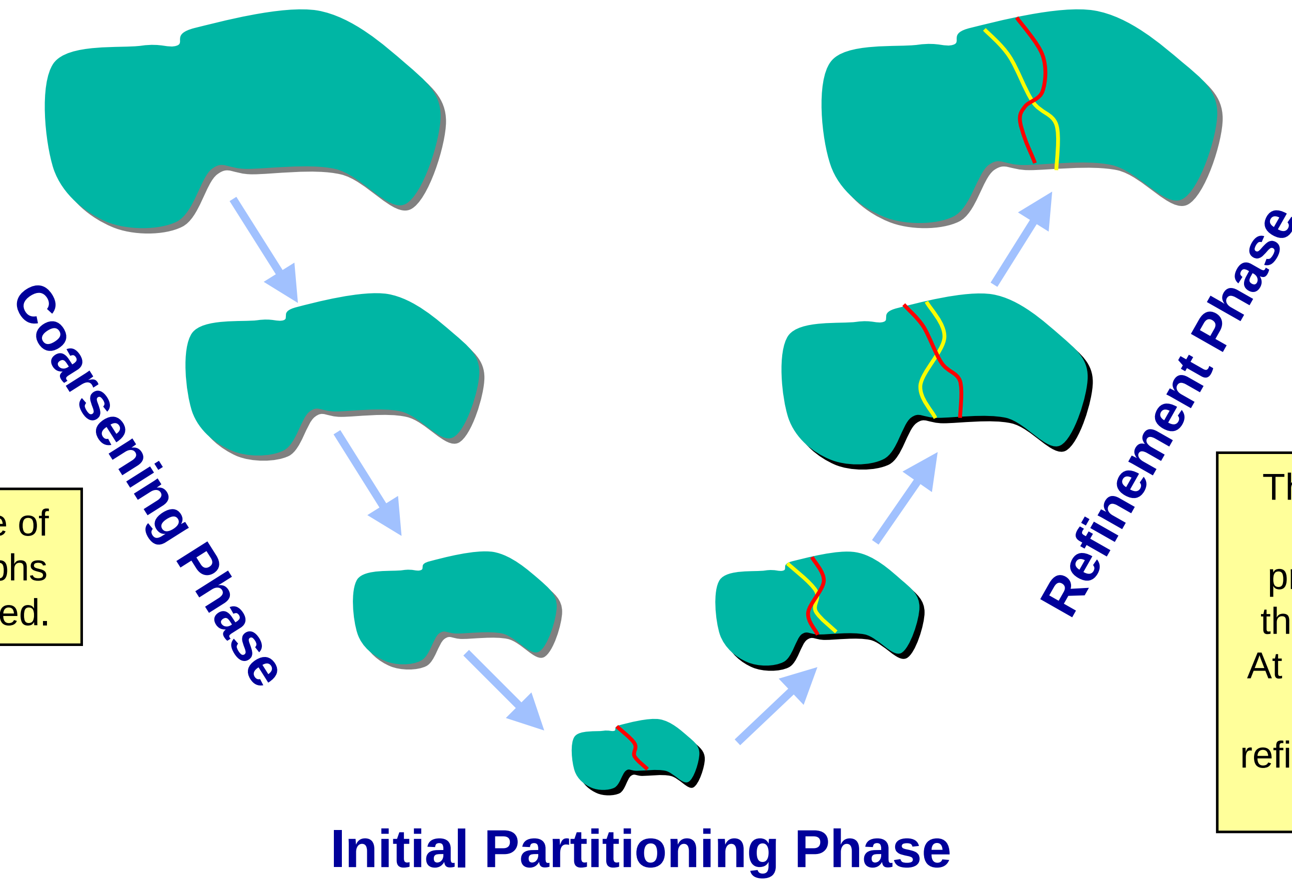




History of Partitioning Algorithms

Multilevel Partitioning Algorithms

A sequence of coarse graphs is constructed.



The partitioning is successively projected back to the original graph. At each finer graph, a KL/FM type refinement algorithm is applied.

Fast!

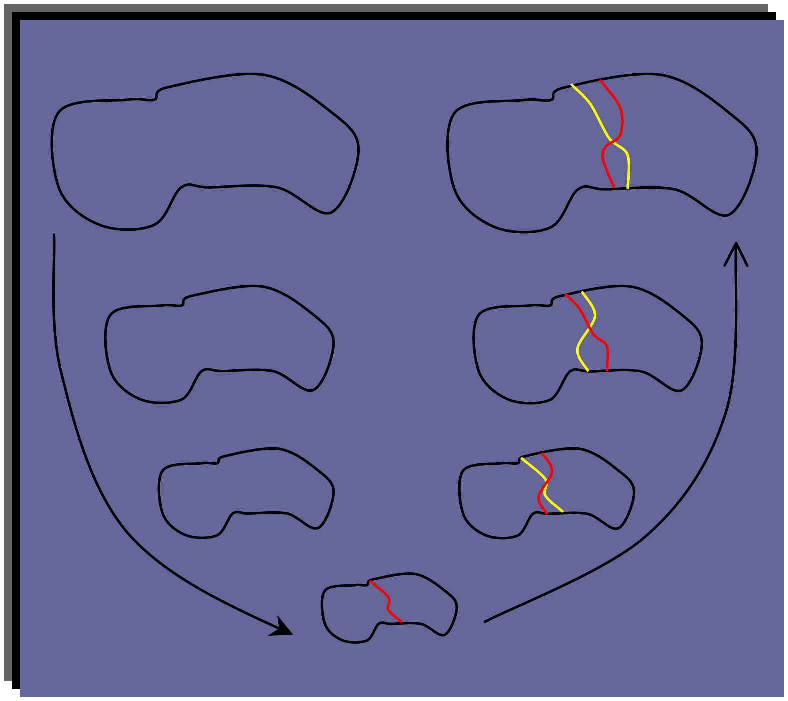
A partitioning of the coarsest graph is computed quickly.

High Quality!

Hendrickson-Leland, Hanck-Borillo, Cory-Smith (1993-95); Karypis-Kumar (1995-96)

METIS and ParMETIS

Serial and Parallel Software Packages for Partitioning Unstructured Graphs and for Computing Fill-reducing Orderings of Sparse Matrices



Fast.

Less than a minute required to partition graphs with millions of vertices on a workstation.

High quality.

Results in substantial reduction in edge-cut or fill compared to other schemes for a variety of graphs.

Parallel.

8M-vertex graph takes under 3 seconds to partition on a 256-processor Cray T3E.

Used Extensively.

National labs, industry, academic institutions worldwide.

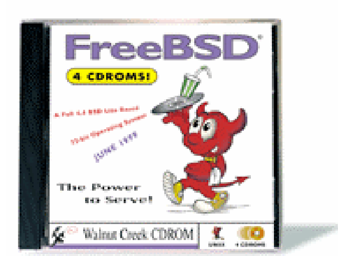
Computations on the largest ever unstructured meshes (over 1 billion elements) have been performed at LLNL and AHPCRC for which the decompositions were computed using ParMETIS.

Karypis, Schloegel, Kumar (1996-2001)

METIS and ParMETIS

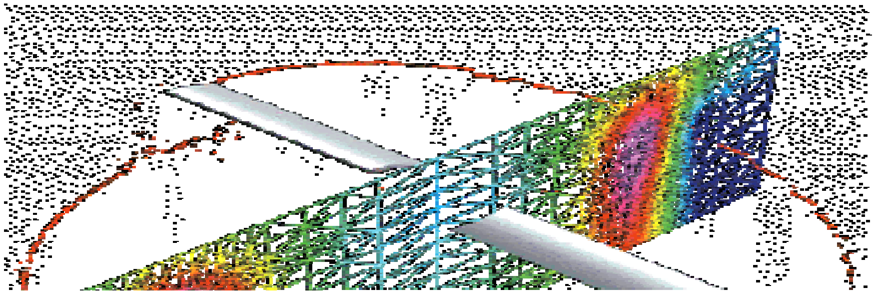
Serial and Parallel Software Packages for Partitioning Unstructured Graphs and for Computing Fill-reducing Orderings of Sparse Matrices

- METIS & ParMETIS have been downloaded by thousands of users worldwide.
- A number of websites worldwide mirror both METIS & ParMETIS.
- METIS 4.0 & ParMETIS 2.0 are supplied with the popular FreeBSD 3.2 distribution.
- Companies that have licensed METIS or ParMETIS
  - SGI, Cray, IBM, Centric, Sun, HP, NEC, Ansys, Boeing, Ford, Rockwell, MCS, HKS, AKA, Adapco, Altair, CSAR, Star Inc., and NAG
- DoD / National Lab Users
  - CE-WES, ARL, NRL, JPL, CAA, NASA, Livermore, Los Alamos, Sandia, Argonne, Oak Ridge, Maui HPC Center, USAF



Adaptive Mesh Computations

- **In adaptive mesh computations, the processor loads can become imbalanced due to refinement and de-refinement of the mesh.**

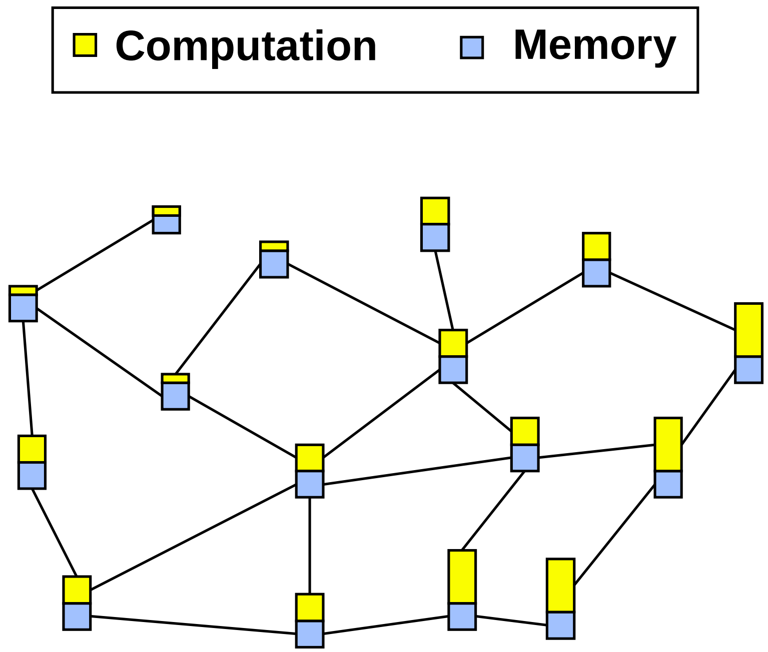


Adaptive Partitioning Work

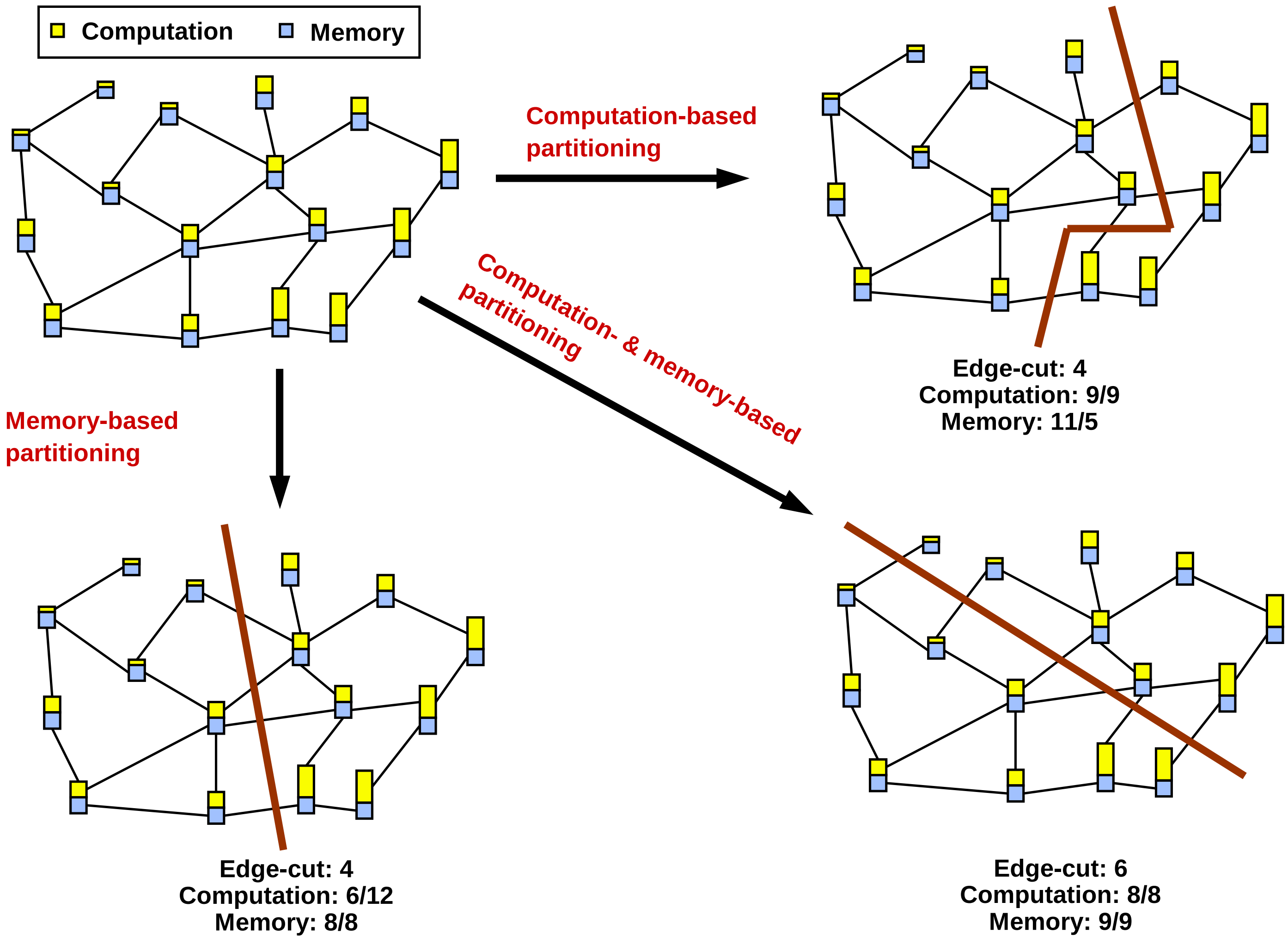
- **P. Diniz, S. Plimpton, B. Hendrickson, and R Leland. Parallel algorithms for dynamically partitioning unstructured grids. Proc. 7th SIAM Conf. Parallel Proc., pages 615-620, 1995.**
-

Partitioning for Computation and Memory

Consider the computation in which the amount of storage required by different mesh elements (*e.g.,* depending on the type of material or computation performed) is different.

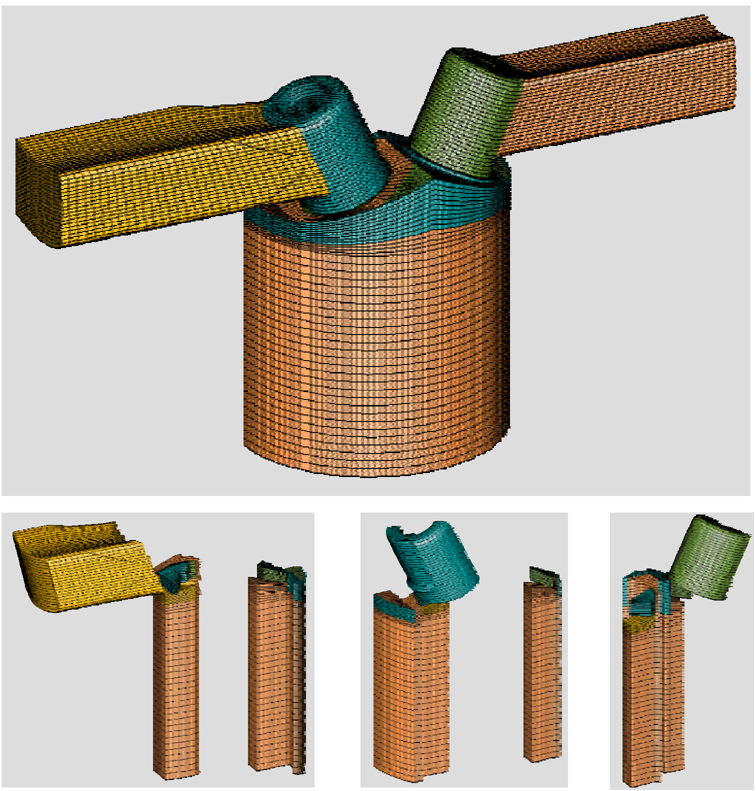


Balancing Computation and Memory



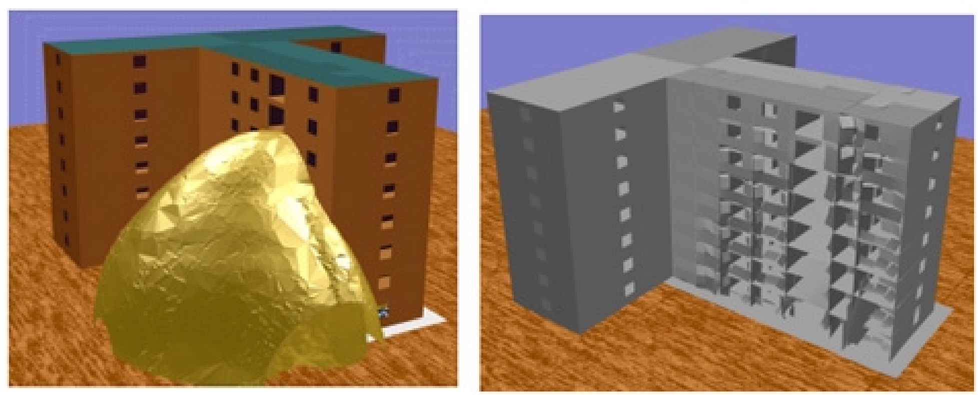
Multi-phase Simulations

- **Consider a parallel multi-phase computation that consists of *m* distinct computational phases such that:**
  - **Each is performed on a different region of the mesh (possibly overlapping).**
  - **They are separated by an explicit synchronization step.**



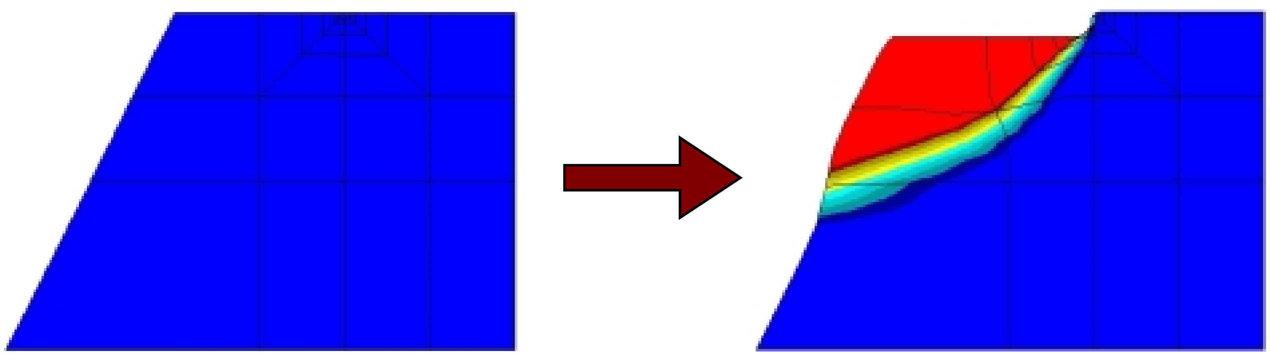
<u>The partitioning must</u>:

- **Simultaneously balance the computations performed at each phase.**
- **Minimize the number of edges that straddle different subdomains.**



Partitioning for Adaptive Multi-physics / Multi-phase Computations

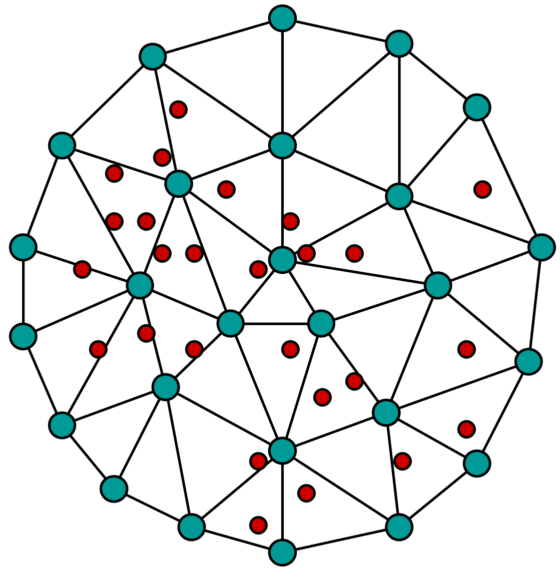
- **Example: combined elastic-plastic simulations for geomaterial solids and structures**
  - **The amount of computation associated with the elastic computation is known in advance.**
  - **The amount of computation associated with the elasto-plastic computation changes dynamically.**



These methods are being used in large-scale, detailed 3D earthquake, soils, rocks, concrete, powders, foams, and bone material simulations.

Particle-in-Mesh Simulations

- **Particle-in-mesh simulations consist of**
  - **a mesh-based computation, and**
  - **a particle tracking phase.**
- **This is a two-phase adaptive computation**
  - **particles can move**
  - **the mesh can adapt**
- **An adaptive multi-phase repartitioner is required:**
  - **to load balance both phases of the computation,**
  - **while minimizing both the inter-processor communications**
  - **and the data redistribution cost.**



These methods are being used in scientific simulations that model such diverse phenomenon as pollution, combustion engines, hydro-planing car tires, and airbags.

Multi-constraint Graph Partitioning Formulation

The traditional graph partitioning formulation has a *single constraint*

(i.e., ensure that each subdomain has a roughly equal amount of vertex weight).

and a *single optimization objective*.

(i.e., minimize the edge-cut)

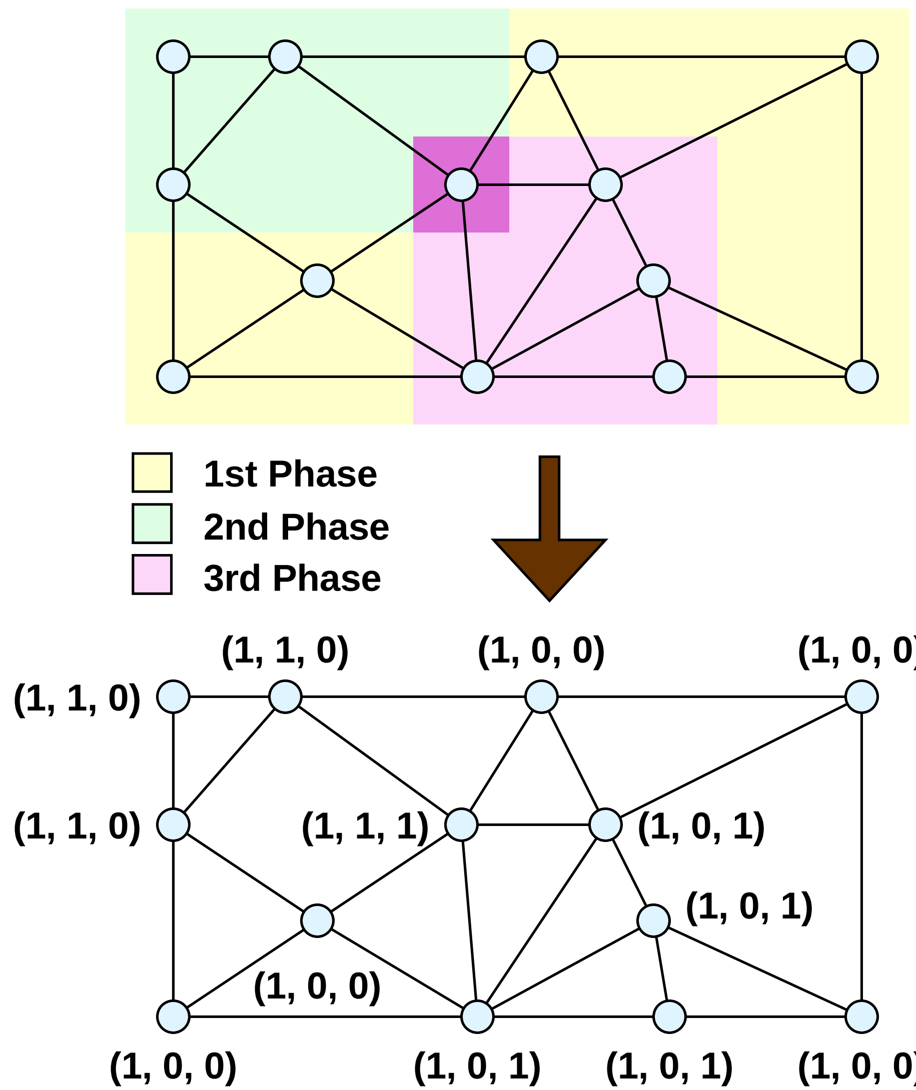
We can generalize the problem by assigning a vector of weights to every vertex.

The new formulation becomes a *multi-constraint* problem

(i.e., ensure that each subdomain has an equal amount of all of the vertex weights).

with a single optimization objective.

(i.e., minimize the edge-cut)

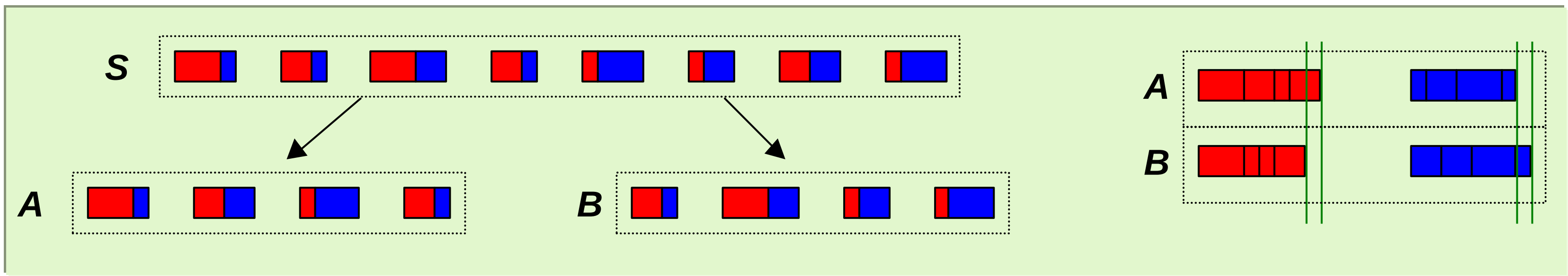


Karypis, Kumar (SC '98)

Multi-Constraint Bin-Packing

Lemma

Consider a set S of n objects with k weights, and let μ be the heaviest weight of any object.
We can partition these objects into two buckets A and B such that $|w_i^A - w_i^B| \leq k\mu$ for $i = 1 \ldots k$.

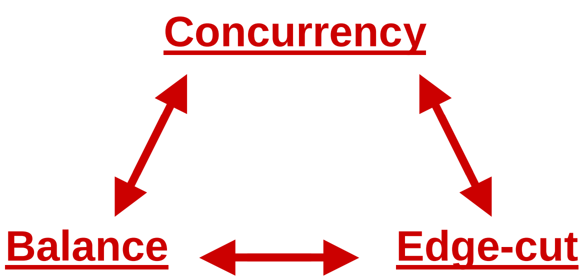


The proof of the lemma leads to an algorithm for constructing the two buckets *A* and *B*.

- **This algorithm is used to compute an initial partitioning of the coarsest graph that:**
  - **Balances the multiple weights & minimizes the edge-cut.**

Challenges in Multi-constraint Graph Partitioning

- **Multi-constraint graph partitioning**
  - **The feasible solution space consists of the intersection of m feasible solution spaces.**
    - **Hard to find a feasible solution.**
    - **Hard to balance the partitioning.**
      - **2-way multi-constraint balancing is NP-complete.**
    - **Hard to refine the partitioning.**



- **Parallel formulation**
  - **An initial feasible solution can be found serially.**
  - **The difficulty of partition refinement with multiple constraints requires sophisticated heuristics in order to be effective.**
    - **These are quite serial in nature.**

- **Adaptive multi-constraint**
  - **All the challenges of parallel multi-constraint plus an additional objective to minimize the data redistribution cost.**
    - **Diffusion is especially difficult**

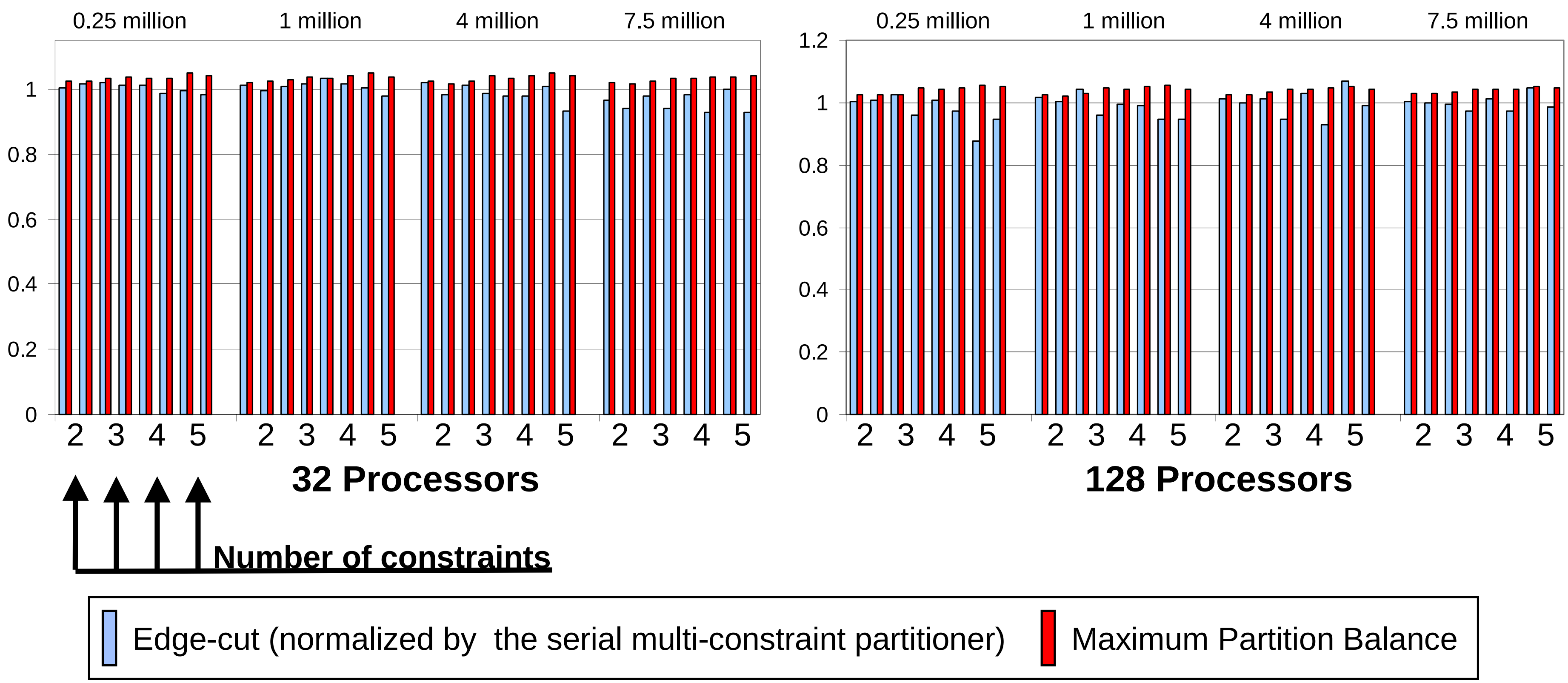
Our Multi-constraint Partitioning Algorithms

- **We have developed a number of parallel static and dynamic multi-constraint partitioning algorithms.**
  - **Our parallel multi-constraint graph partitioning algorithm provides both powerful refinement and a high level of currency while also helping to ensure that the multiple balance constraints are maintained (by use of a reservation scheme during refinement).**
  - **We have also developed a multi-constraint repartitioner based on the unified repartitioning approach using the parallel multi-constraint partitioning algorithm as a key component.**

Schloegel, Karypis, and Kumar (EuroPar2000)

Parallel compared to Serial Multi-constraint Partitioner

Edge-cut and Balance results on 32 and 128 processors



The parallel formulation computes balanced partitionings that are of similar quality to the serial algorithm.

Schloegel, Karypis, Kumar (EuroPar2000)

Parallel Multi-constraint Graph Partitioner

Run Time and Efficiency Results for 3 constraints on 32 and 128 processors

Run Time Results (in seconds) of the Parallel Multi-constraint Partitioner.

Graph size	8-procs	16-procs	32-procs	64-procs	128-procs
1 million	9.8	5.3	3.5	2.5	3.1
4 million	31.8	16.9	9.3	5.7	4.4
7.5 million	out of mem	30.7	16.7	9.2	6.4

Parallel run time: O(nm/p) + O(pm log n)

Efficiencies of the Parallel Multi-constraint Partitioner.

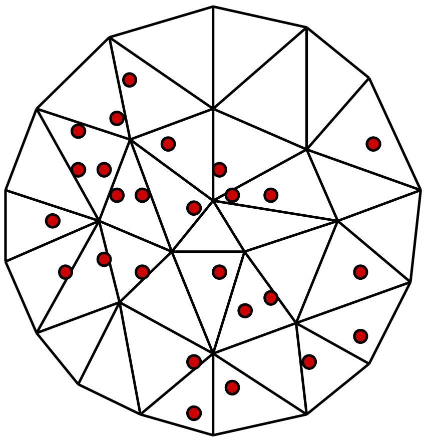
Graph size	8-procs	16-procs	32-procs	64-procs	128-procs
1 million	100%	92%	70%	49%	20%
4 million	100%	94%	85%	70%	45%
7.5 million	out of mem	100%	92%	83%	60%

Isoefficiency: $p^2 \log p$

Schloegel, Karypis, Kumar (EuroPar2000)

Results From a Real Particle-in-mesh Application An adaptive multi-constraint problem

Scheme	Edge-cut	Data Redistribution	Balance	
8-processors				
Naïve SR	9,412	23,171	1.01	1.05
Mc-LMSR	8,188	897	1.01	1.05
Static	8,028	0	1.02	3.47
16-processors				
Naïve SR	17,398	60,364	1.01	1.06
Mc-LMSR	15,073	5,772	1.01	1.07
Static	14,757	0	1.02	8.01
32-processors				
Naïve SR	25,243	75,534	1.04	1.15
Mc-LMSR	22,635	3,200	1.03	1.11
Static	23,327	0	1.02	11.57



Mc-LMSR outperforms the naïve scratch-remap scheme for both Edge-cut and Data Redistribution.

Results were obtained by repartitioning a series of 175,000-vertex graphs derived from a particle-in-mesh simulation of a diesel combustion engine.

Repartitioning occurs every 150 timesteps.

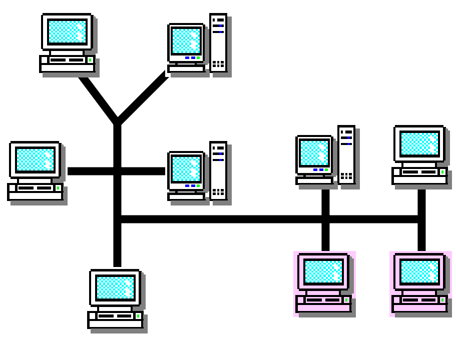
These graphs were provided by Boris Kaludercic, Computational Dynamics, Ltd., London, England.

Schloegel, Karypis, Kumar (To appear CPE, 2001)

Graph Partitioning for Heterogeneous Architectures

- **Partitioning for parallel architectures in which the processors have different speeds (but the network topology is homogeneous)**
  - **Example, cluster of workstations**

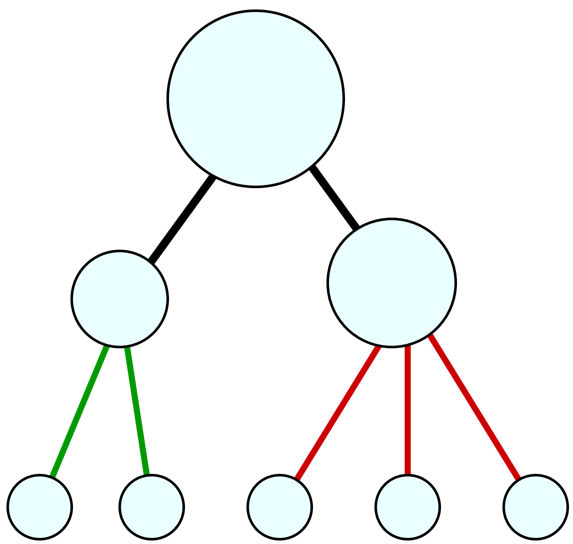
- **Each subdomain of the partitioning can be sized according to the relative speed of the corresponding processor.**
  - **ParMETIS, Version 3.0**

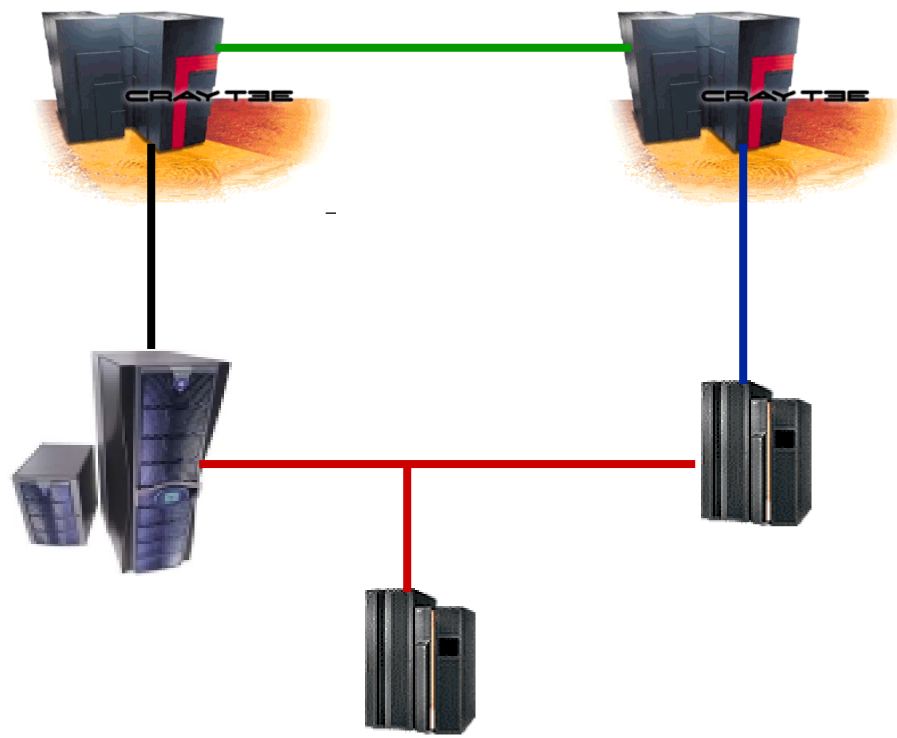


Partitioning for Generic Meta-computing Environments

- An arbitrary heterogeneous architecture can be described by a hierarchical model

- A recursive partitioning scheme scheme can be used.



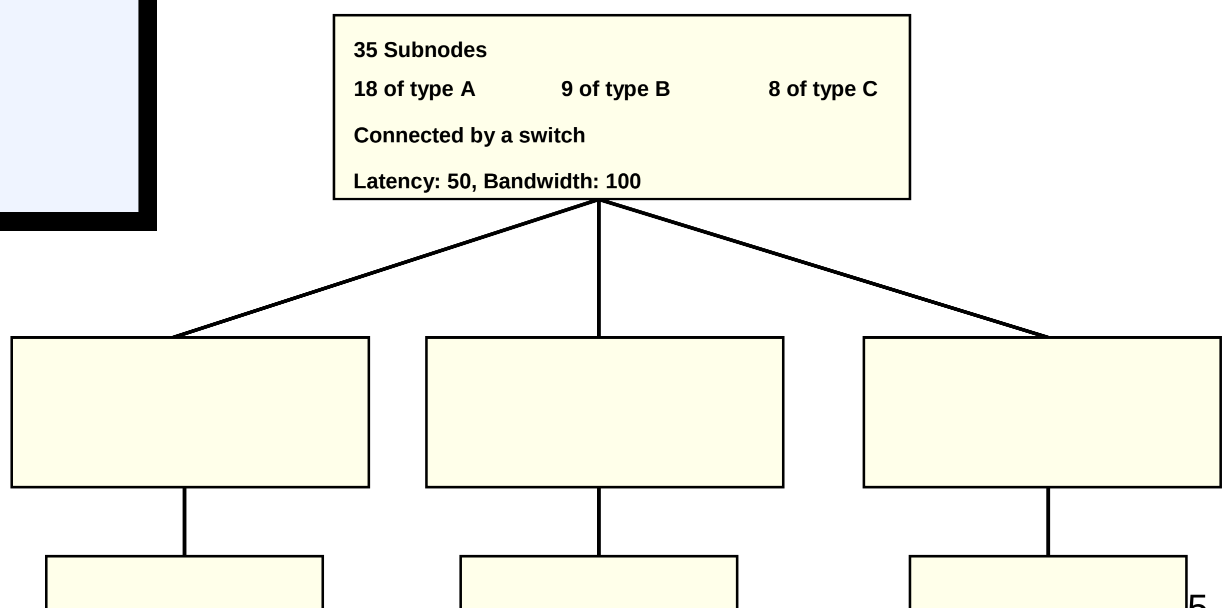


Machine-specification Model

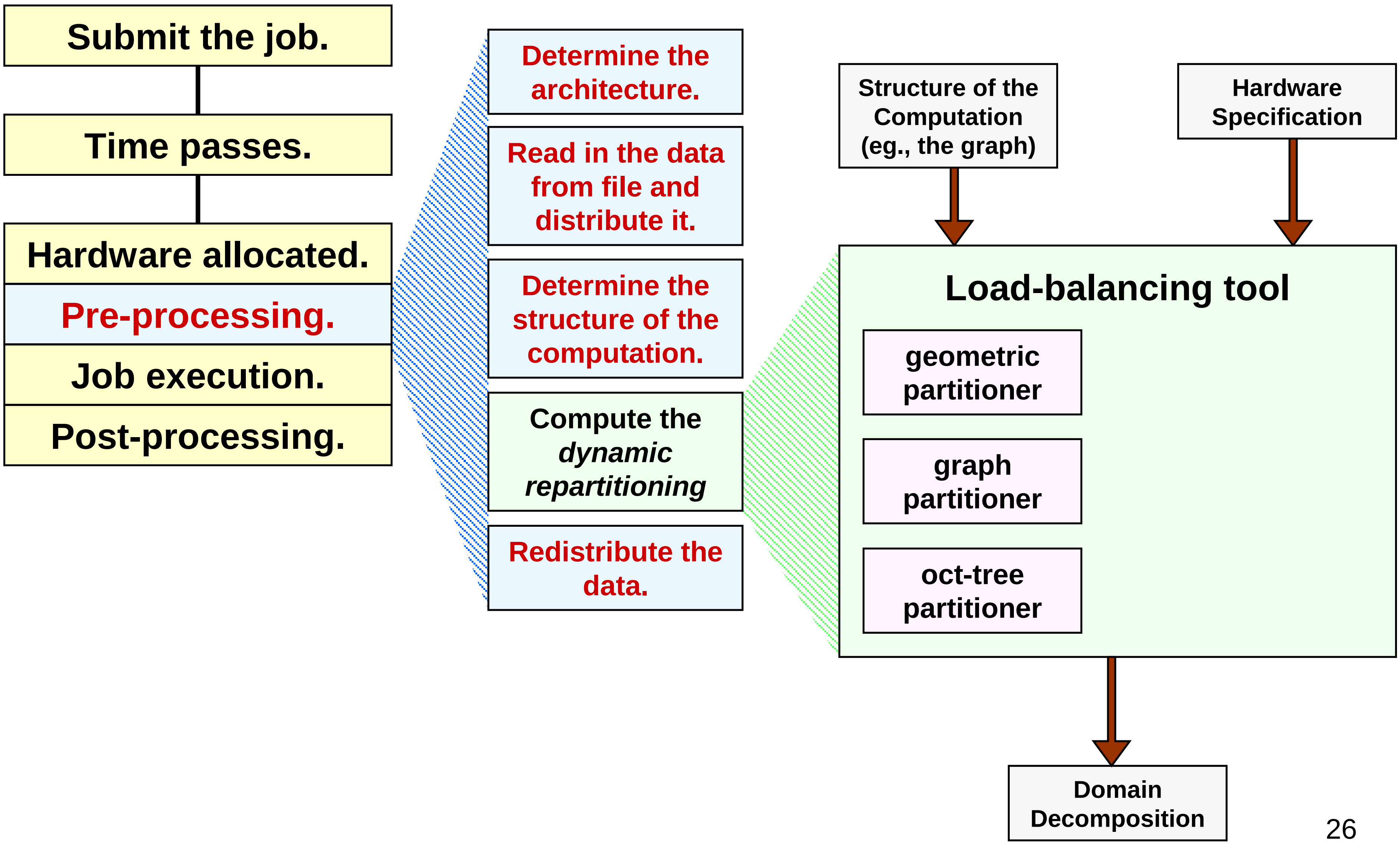
Machine-specification Model

Computing nodes
- number of sub-nodes;
- number of different types of sub-nodes;
- power, memory, and type of each sub-node;

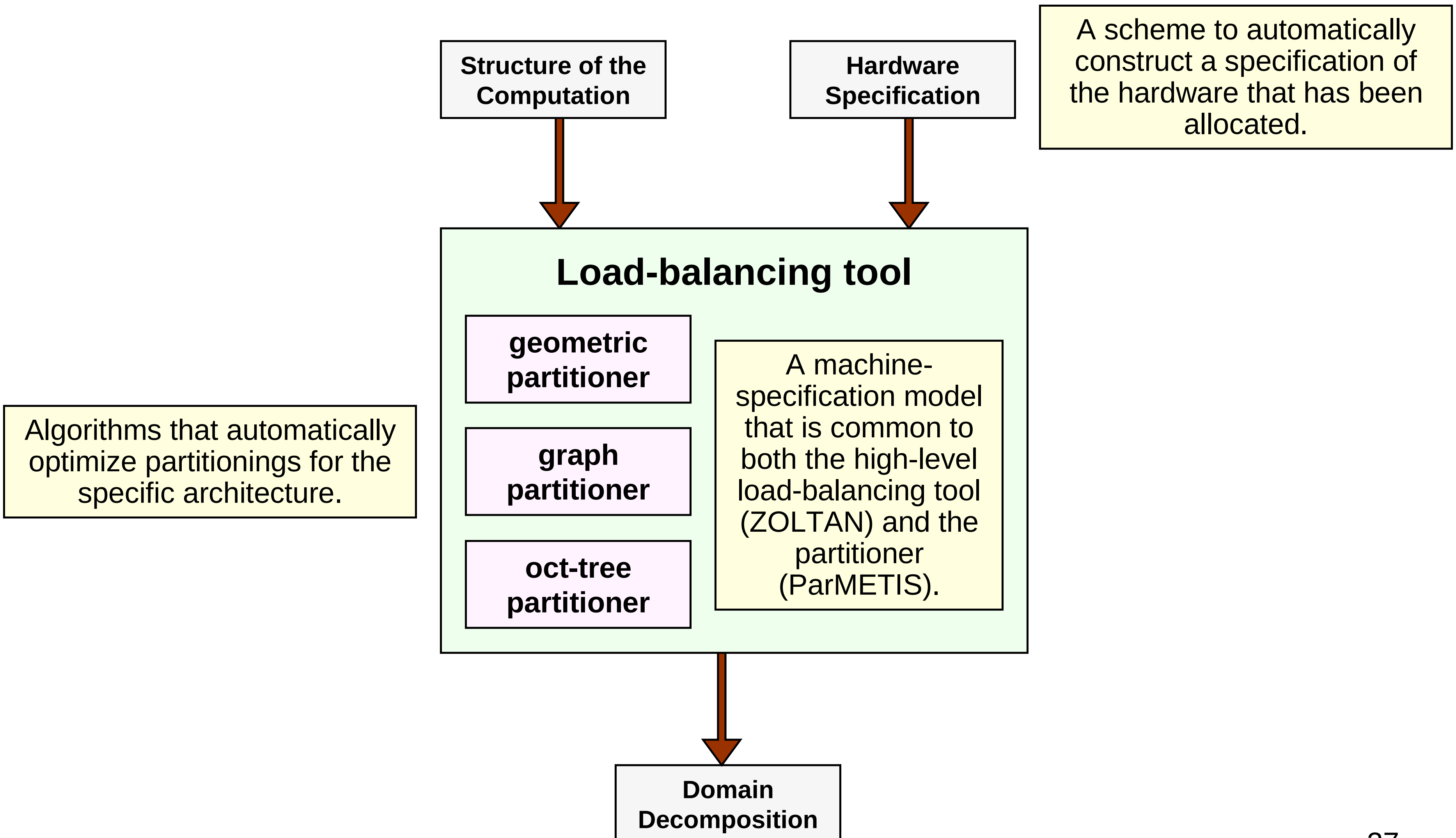
Topology
- bandwidth, latency;
- number of mesh dimensions;
- size of each dimension;
- wrap around in each dimension;
- topology type
  - none or SMP
  - mesh
  - hypercube
  - user-defined



Meta-computing Environment Model



What is Required to Partition Under This Model?



Conclusions

- **It is usually possible to compute good partitionings of sparse graphs that have some inherent structure.**
  - **Example domains: 2D / 3D finite-element meshes, VLSI circuits, linear programming, data mining, storage of geographic information.**
- **There is some theoretical understanding of why multilevel schemes work. However, more work is needed here.**
- **There is a reasonable understanding of repartitioning schemes for adaptive 3D finite-element meshes.**
- **For multi-constraint & multi-objective problems, there is a great deal of work needed for specific problems, as well as for parallel and adaptive formulations.**
-

Talk based on

Graph Partitioning for High Performance Scientific Simulation,
By Schloegel, Karypis, Kumar

Book chapter in

CRPC Parallel Computing Handbook
Editors: Dongarra, Foster, Fox, Kennedy, White
Morgan Kaufmann