Load Balancing

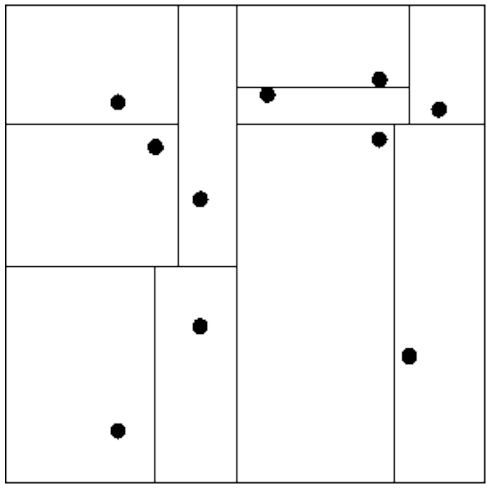
Aiichiro Nakano

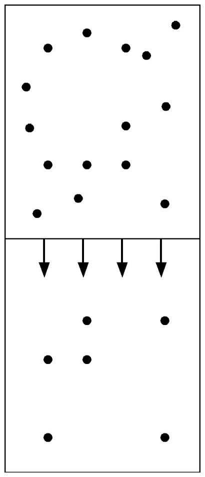
Collaboratory for Advanced Computing & Simulations
Department of Computer Science
Department of Physics & Astronomy
Department of Chemical Engineering & Materials Science
Department of Biological Sciences
University of Southern California

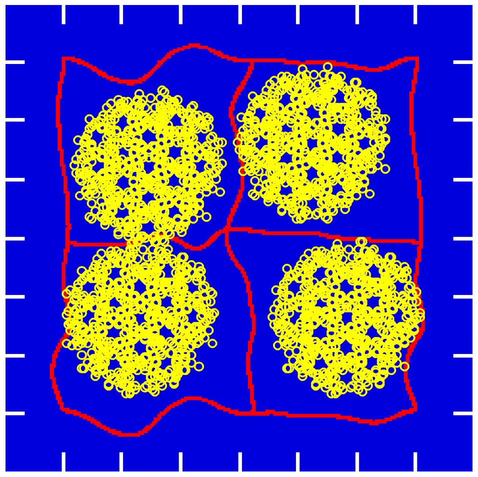
Email: anakano@usc.edu

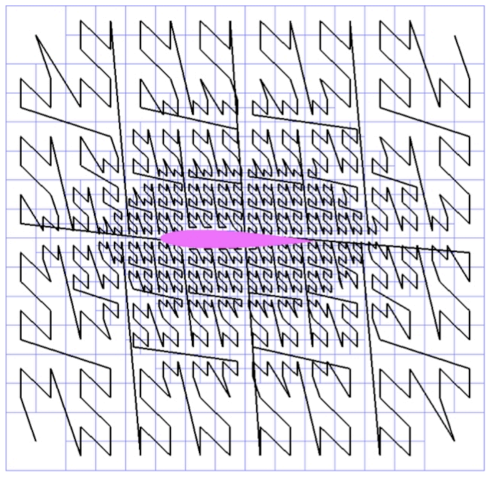
Load Balancing

- **Goal:** Keep all processors equally busy while minimizing inter-processor communication for irregular parallel computations
- **Issues:**
  - Spatial data *vs*. generic graph
  - Static *vs*. adaptive
  - Incremental *vs*. non-incremental
- **Load-balancing schemes:**
  - Recursive bisection
  - Spectral method
  - Spacefilling curve
  - Curved space
  - Load diffusion





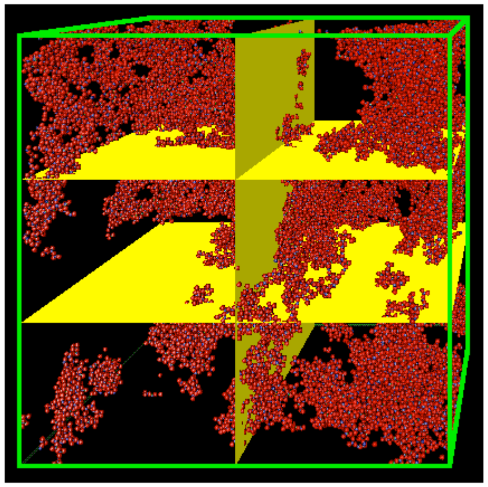




Data Locality in Parallelization

Challenge: Load balancing for irregular data structures

Irregular data-structures/ processor-speed

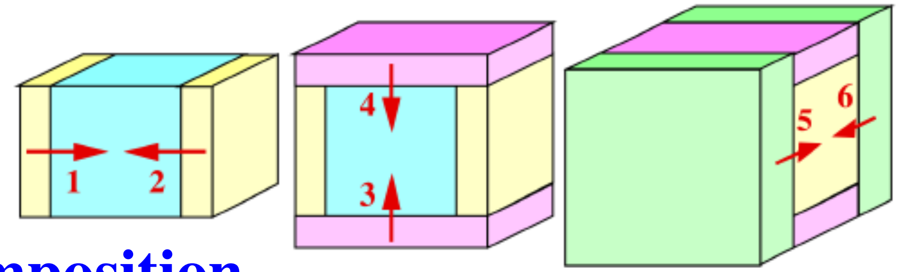


Map

Parallel computer

Optimization problem:

- Minimize the load-imbalance cost
- Minimize the communication cost
- Topology-preserving spatial decomposition
  → structured 6-step message passing minimizes latency



$$E = t_{\text{comp}}\left(\max_p\left|\{i \mid \mathbf{r}_i \in p\}\right|\right) + t_{\text{comm}}\left(\max_p\left|\{i \mid \|\mathbf{r}_i - \partial p\| < r_c\}\right|\right) + t_{\text{latency}}\left(\max_p\left[N_{\text{message}}(p)\right]\right)$$

Computational-Space Decomposition

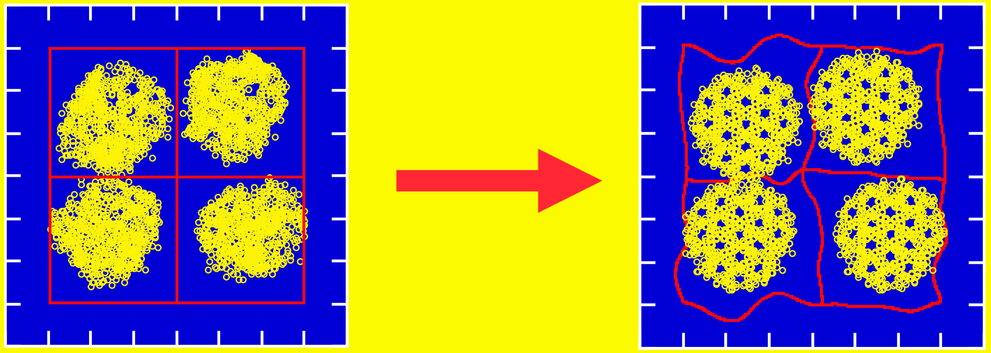
Topology-preserving "computational-space" decomposition in curved space

Curvilinear coordinate transformation

$$\xi = \mathbf{x} + \mathbf{u}(\mathbf{x})$$

Particle-processor mapping: regular 3D mesh topology

$$\begin{cases} p(\xi_i) = p_x(\xi_{ix})P_yP_z + p_y(\xi_{iy})P_z + p_z(\xi_{iz}) \\ p_\alpha(\xi_{i\alpha}) = \lfloor \xi_{i\alpha}P_\alpha / L_\alpha \rfloor \quad (\alpha = x,y,z) \end{cases}$$



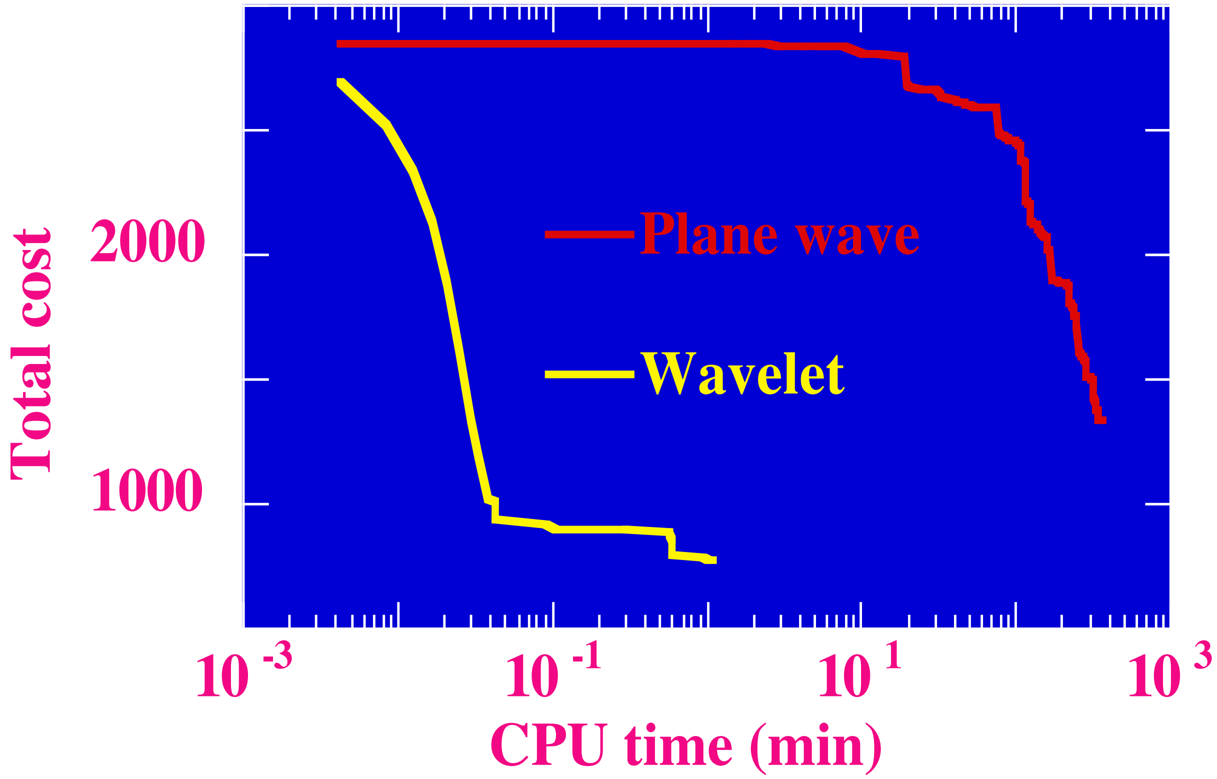
Regular mesh topology in computational space, ξ — **Curved partition in physical space, x**

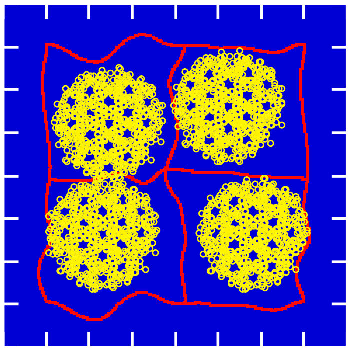
A. Nakano & T. J. Campbell, *Parallel Comput.* **23**, 1461 ('97)

Wavelet-based Adaptive Load Balancing

- Simulated annealing to minimize the load-imbalance & communication costs, $E[\xi(x)]$
- Wavelet representation speeds up the optimization

$$\xi(x) = x + \sum_{l,m} d_{lm}\psi_{lm}(x)$$



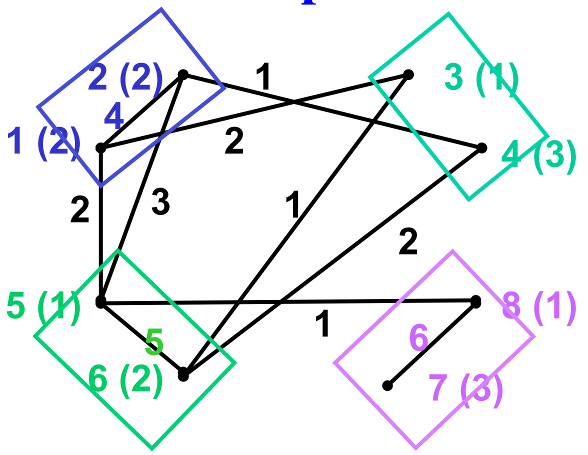


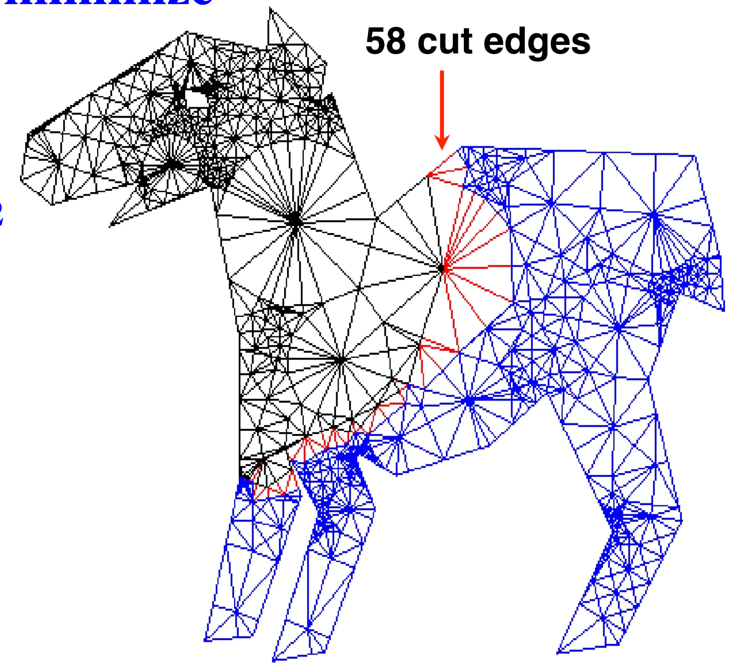
A. Nakano, *Concurrency: Practice and Experience* **11**, 343 ('99)

Load Balancing as Graph Partitioning

- Need: Decompose tasks without spatial indices
- Graph partitioning: Given a graph $G = (N, E, W_N, W_E)$
  - N: node set = $\{j \mid \text{tasks}\}$
  - W_N: node weights = $\{w_N(j)\text{: task costs}\}$
  - E: edge set = $\{(j,k) \mid \text{messages from } j \text{ to } k\}$
  - W_E: edge weights = $\{w_E(j,k)\text{: message sizes}\}$

  choose a partition $N = N_1 \cup N_2 \cup \ldots \cup N_P$ to minimize
  - $\max_p\{\sum_{j \in Np} w_N(j)\}$
  - $\max_{(p,q)}\{\sum_{j \in Np, k \in Nq} w_E(j,k)\}$
- Graph bisection: Special case of $N = N_1 \cup N_2$
- Choosing optimal partitioning is known to be NP-complete → need heuristics



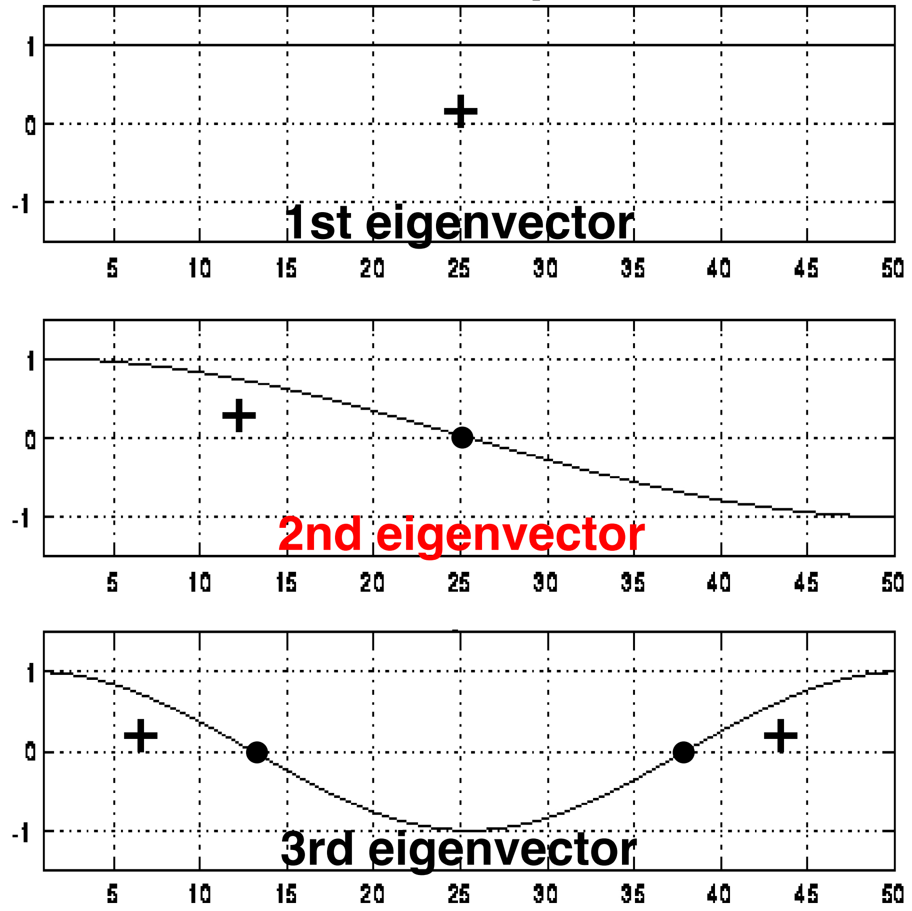


www.cs.berkeley.edu/~demmel/cs267_Spr16
Prof. James Demmel (UC Berkeley)

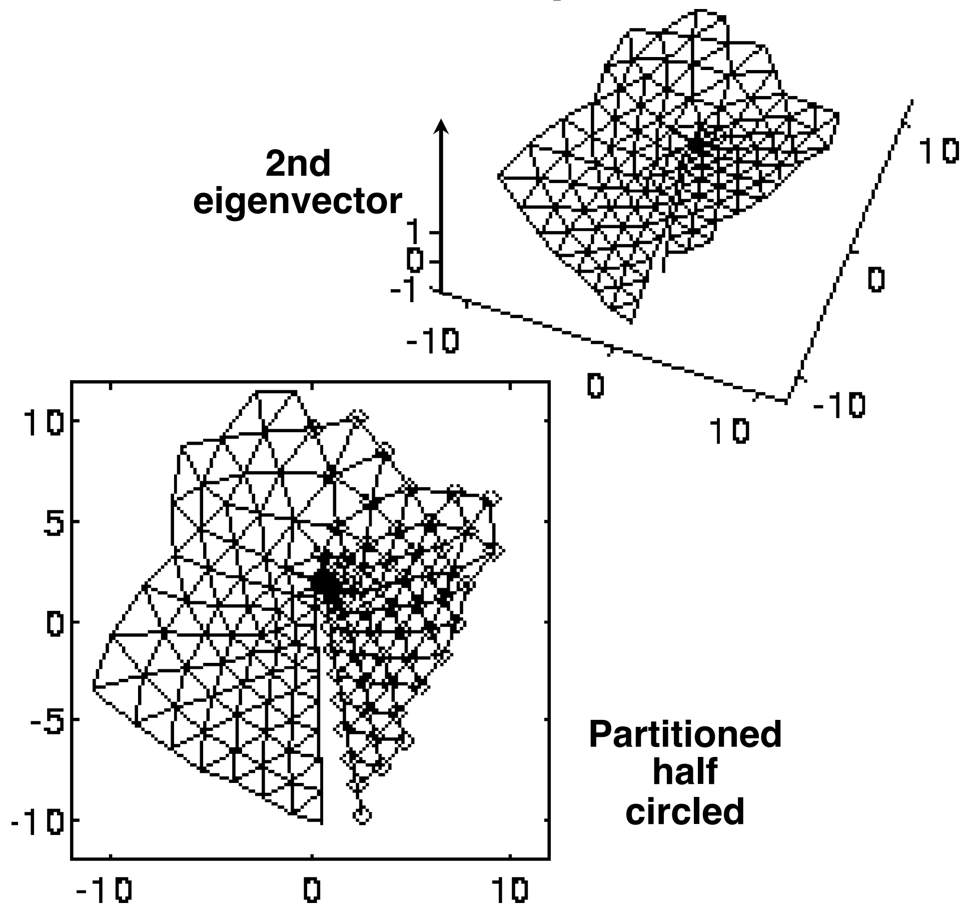
Spectral Bisection: Motivation

1. Graph as point masses connected via harmonic springs
2. The node of the eigenvector of the Hessian matrix, $\partial^2 V/\partial x^2$, corresponding to the 2nd smallest eigenvalue separates the graph into 2

1D example



2D example



Partitioned half circled

Spectral Bisection

Laplacian matrix:

L(G) of a graph $G(N,E)$ is an $|N|$ by $|N|$ symmetric matrix:

- L(G)(i,i) = degree of node i (number of incident edges)
- L(G)(i,j) = -1 if $i \neq j$ and there is an edge (i,j)
- L(G)(i,j) = 0 otherwise

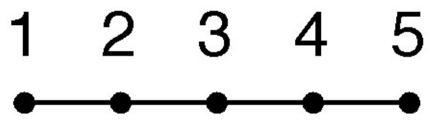
Theorems:

1. The eigenvalues of L(G) are nonnegative: $\lambda_1 = 0 \leq \lambda_2 \leq \cdots \leq \lambda_N$)
2. $\lambda_2(L(G)) \neq 0$ if and only if G is connected

Spectral bisection algorithm:

1. Compute eigenvector v_2 corresponding to $\lambda_2(L(G))$
2. For each node i of G
   - a. if $v_2(i) < 0$, put node i in partition N-
   - b. else put node i in partition N+

Example



$$\begin{array}{c|ccccc|} & 1 & 2 & 3 & 4 & 5 \\ 1 & 1 & -1 & & & \\ 2 & -1 & 2 & -1 & & \\ 3 & & -1 & 2 & -1 & \\ 4 & & & -1 & 2 & -1 \\ 5 & & & & -1 & 1 \end{array}$$

O(*N*) λ_2 Computation

Lanczos algorithm:

- **Given an $N \times N$ symmetric matrix A (*e.g.*, L(*G*)), compute a $K \times K$ "approximation" T by performing K matrix-vector products, where $K << N$**
- **Approximate A's eigenvalues & eigenvectors using T's**

```
Choose an arbitrary starting vector r
b(0) = ||r||
j=0
repeat
  j=j+1
  q(j) = r/b(j-1)
  r = A*q(j)
  r = r - b(j-1)*v(j-1)
  a(j) = v(j)ᵀ * r
  r = r - a(j)*v(j)
  b(j) = ||r||
until convergence
```

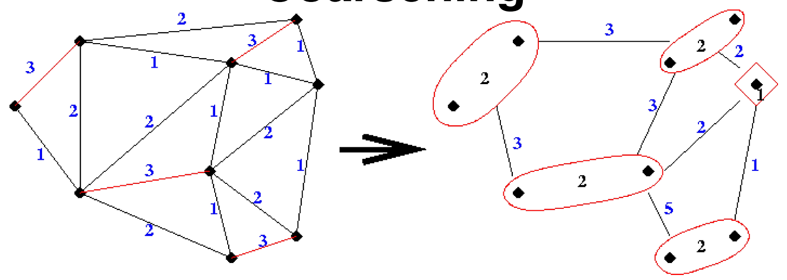
$$\mathbf{T} = \begin{bmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{K-2} & a_{K-1} & b_{K-1} \\ & & & b_{K-1} & a_K \end{bmatrix}$$

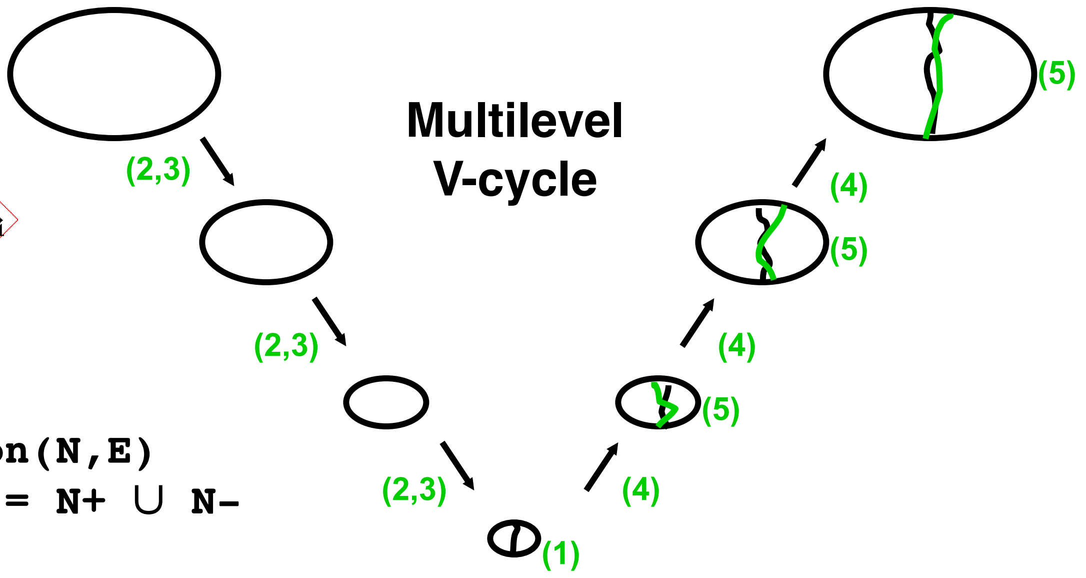
Multilevel Partitioning

Recursively apply:

1. **Replace $G(N,E)$ by a coarse approximation $G_c(N_c,E_c)$, & partition G_c**
2. **Use partition of G_c to obtain a rough partitioning of G, then uncoarsen & iteratively improve it**

Coarsening





```
(N+,N-) = Multilevel_Partition(N,E)
// returns N+ and N- where N = N+ ∪ N-
  if |N| is small
1   Partition G = (N,E) directly to get N = N+ ∪ N-
    Return (N+,N-)
  else
2   Coarsen G to get an approximation G_c = (N_c,E_c)
3   (N_c+,N_c-) = Multilevel_Partition(N_c,E_c)
4   Expand (N_c+,N_c-) to a partition (N+,N-) of N
5   Improve the partition (N+,N-)
    Return (N+,N-)
  endif
```

An Extra Lesson

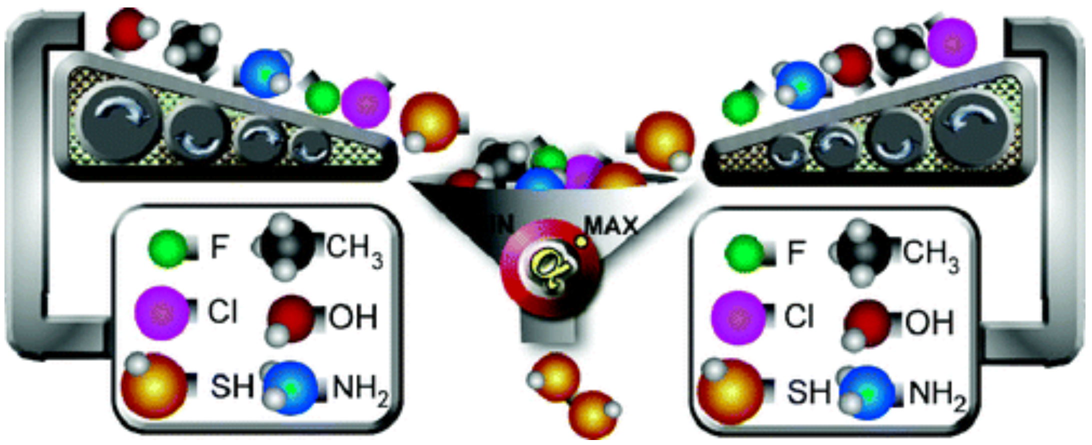
Continuous optimization is easier than discrete combinatorial optimization

cf. • **Linear combination of atomic potentials (LCAP)**

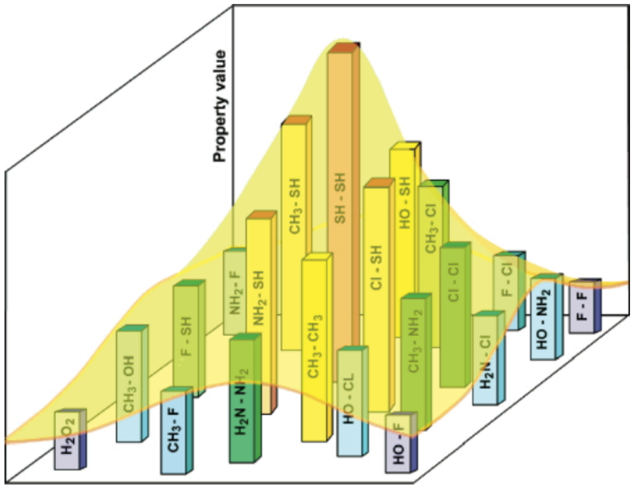
M. Wang *et al*., *J. Amer. Chem. Soc*. **128**, 3228 ('06)

• **Gradient-directed Monte Carlo (DGMC)**

X. Hu, *J. Chem. Phys*. **129**, 064102 ('08)



LCAP: $v(\vec{r}) = \sum_{\vec{R},A} b_A^{\vec{R}} v_A^{\vec{R}}(\vec{r})$


Property value
H_2O_2
CH_3 - OH
CH_3 - F
F - SH
NH_2 - F
H_2N - NH_2
NH_2 - SH
CH_3 - SH
CH_3 - CH_3
SH - SH
HO - CL
Cl - SH
HO - SH
HO - F
CH_3 - NH_2
CH_3 - Cl
Cl - Cl
H_2N - Cl
F - Cl
HO - NH_2
F - F