CSE248
Spring 2011

Linear Placement,
Spectral Partitioning
Project Discussions

- “Benchmark Construction with Known Optimal Solution”
- Partitioning of Test Vectors
- Many-core Placement (global + detailed)
- Planar-BST (improved WL, error bound)
- Generic requirements
  - Problem statement (supporting notation, terminology, defs)
  - Motivation (application, potential impact)
  - Literature review (algorithmic “open direction”)
  - “New Idea” (algorithmic approach? formal analysis? …) + what directions are not chosen
  - Implementation (code/scripts (external/internal), test suite, validation or experimental plan, qualitative goals (10x, 20%, what are deliverables, …), timeline)
  - Set of checkpoints (agree on problem statement; agree on literature review and “open” status; test data; first working flow/numbers; draft/skeleton results tables, charts; …)
  - At any given moment: What is blocking, what is next step
- ABK style: “If it isn’t in writing, it doesn’t exist” / “don’t assume it’s done unless you have positive confirmation” / (Samsung: “no data, no result”)
Many-Core Placement (discussion #1)

- Problem statement = ?
- Qualitative (novel) goal: $\Theta(P)$ speedup on $P$ processors, avoid “pollution” from previous works / think out of the box!
- Key background (~3-4 pp. tex)
  - Relevant methods for Partitioning and clustering; Global placement (not legal, but spread); Detailed placement $\rightarrow$ which are relevant, which are not, and why? (e.g., is “hierarchical” a priori off the table?)
  - Many-core architecture and HW-SW platform options
- New ideas = ?
  - How to combine non-disjoint solutions of subproblems?
  - Are changes to the traditional cluster-GP-DP flow warranted?
- Techniques to develop
- Schedule
  - To the extent that “DP” is orthogonal and smaller in space of possibilities, implementation can proceed in parallel with GP development?
- Testbed
  - Problem instances, quality metrics
  - Hardware (SDSC BlueGene clusters)
Many-Core Placement (discussion #2)

- Alpert / Nam (IBM) clustering method is in Aplace3.0 code
- Still suggest: need a “religious” stance here of “no serial computation” (cf. “New Ideas” from previous discussion)
  - “Absolute scalability” is your stake in the ground; everything else is built around it
  - What does this break in your approach? E.g., no top-down decomposition, no guarantees of disjoint clusters, …
  - Another stake in the ground: don’t use P processors to solve the same problem P times (bounded-discrepancy search, GWTW). And, assume that symmetry is broken somehow.
  - Can you place, say, 10*P clusters using P processors, where each cell belongs on average to 10 clusters? (P ~ sqrt(N)) ?
  - How to keep clusters from overlapping too much in any area?
  - How to deduce a cell placement from the cluster placement?
  - How to combine non-disjoint solutions of subproblems?
  - Are changes to the traditional cluster-GP-DP flow warranted?

- Details of how shared memory is actually accessed?
  - How are the clustering, the clusters, the original netlist, the induced netlist(s) over clusters, the placement (locations of clusters and cells): stored, accessed, updated?
Known Optimal Solutions (discussion #1)

- Problem Statement = ?
- Arena: **sizing**, placement, partitioning, routing, …
- Related Works (constructions with known optima; “scaling”)
  - Sizing: DAC10 “eye charts”
  - Placement/Partitioning: HHK95, PEKO/PEKU, “Planted Partition” Bui, Garbers, etc.
  - Routing: SLIP11 scaling
  - “Synthetic Benchmark” netlist constructions (Ghent ?, UCSC ?, Toronto circ/gen, …)
- Qualitative goals: “scalable” (= composable?), “realistic”, “general conclusions vs. tractability”
- Issues
  - How to simplify “sizing” so that it is tractable (e.g., DAC10 paper eliminated slew-dependence in its testbed)?
- **New** and/or basic ideas = ?
  - Metric of “realism” (see the “synthetic benchmark” literature; note that realism and tractability to analysis are opposing goals)
  - Maybe “known optimal” can be in a probabilistic sense (cf. planted partition idea) or a bounded sense (cf. HHK95 scaling suboptimality)
  - Always try to optimally solve as large a “small case” as possible (e.g., B&B or SA running for weeks gives a proof point later)
Planar Bounded-Skew Tree (discussion #1)

- Literature review (“annotated bibliography” kind of summary ~1 page) (5/4)
  - On Optimal Interconnections; Edahiro93 Greedy-DME, Planar (Zhu-Dai92, KahngT96); BST-DME; Charikar; Zelikovsky  
    [Note: everyone should have such a summary ~5/4]

- Motivations / “fundamental questions to address”
  - Why single-layer / planar required?  (Interposer in 3DIC; clock distribution on MCM substrate; avoid vias using “dedicated clock layer”?)
  - How does “planar” change the BST problem?  E.g., how would you change KahngT96 when “Z” becomes “B”? (5/6)
  - Ignoring planarity, “BST” gives continuum between “ZST” and “SMT”.  Then, “pBST” gives continuum between “pZST” (PlanarDME) and “SMT” → should achieve such behavior as skew bound B is varied (in particular, should “match” other methods for B = 0, ∞)
  - What intuitions have you built so far?  Pencil and paper: points on a line, on a circle, on a 2-D lattice (5/6)

- Code
  - BST-DME code from Bookshelf (need to get it to compile/run – 5/2)
  - Implement Charikar, Zelikovsky methods (what are “planar”, “bounded” versions?)

- Scope (are you still comfortable with PBST as the topic?)
  - Linear delay model (ignore Elmore)
  - No buffering (no clock gating, no sink polarity, no obstacles, …)
  - Topology design
    - Open: minimum-cost Planar BST consistent with given topology → “topology given” is unclear
    - Charikar: Is there a “terminal ordering” (e.g., as in k-center heuristic) that gives a good topology somehow?
Outline

- Hur-Lillis linear placement – global techniques
- Spectral ratio cut partitioning
- Spectral partitioning: more eigenvectors is better
Input: $m$ and $c$

```
while $k > 0$
    Extract set of $m$ mobile nodes
    Find optimal relaxed placement
    $P \leftarrow$ Legalized placement
    Improved
    Yes: $P \leftarrow$ New placement
         $k \leftarrow c$
    No: $k \leftarrow k - 1$
```

by net-based
“BFS” clustering
Subcircuit Concept

- Given a set of mobile nodes of size $m$
- Determine active nets and fixed node set $\rightarrow$ induces a subcircuit

Example

- Mobile node
- Fixed node
- Active net

J. Lillis, UIC
LP Formulation of Relaxed Placement

- Exact model of hyper-edges (no clique needed)
- Easily models true (linear) wire length rather than squared wire length
- $x_v$: x-coordinate of node $v$
- $F_v$: x-coordinate of fixed node $v$ \( i.e., F_v = P^{-1}[v] \)
  
  // $P$ is the assignment of vertices to locations
- $E_A$: set of active nets
- $r_e$, $l_e$: two extreme locations of net $e$

LP Formulation:

$$\min \sum_{e \in E_A} (r_e - l_e) \quad \text{s.t.}$$

$$l_e \leq x_v \leq r_e, \quad \forall v \in e,$$

$$x_v = F_v, \quad \forall v \in F$$
Example of Relaxed Placement

Initial Placement

\[ len = 20 \]

Relaxed Placement

\[ len = 14 \]

Note: Not clear why E had to move.
LP Impractical

- **Problem**: Using an LP-Solver in inner loop may not be practical
- **Solution**: Fast network flow based algorithm solves exactly the same LP-formulation
- **Theorem**: Network flow based algorithm finds an optimal relaxed placement
Net Modeling for Flow Based Algorithm

2-pin net

multi-pin net
Incremental Flow Based Algorithm

- Fixed nodes are arranged
- Find min-cut

Example at $i^{th}$ step
Relaxed Placement by Incremental Flow

Initial Placement

\[ \text{len} = 20 \]

\[
\begin{array}{cccccccc}
A & B & C & D & E & F & G & H & I \\
1 & 2 & 3 & 4 & 5 & 6 & & & \\
\end{array}
\]

Relaxed Placement

\[ \text{len} = 14 \]

\[
\begin{array}{cccccccc}
A & B & C & D & E & F & G & H & I \\
1 & 2 & 3 & 4 & 5 & 6 & & & \\
\end{array}
\]

- Mobile node
- Fixed node

J. Lillis, UIC
### LP vs. Flow

**CPU-time (set of 5000 mobile nodes)**

<table>
<thead>
<tr>
<th>LP-Solver</th>
<th>Network-Flow algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>55 min</td>
<td>8 sec</td>
</tr>
</tbody>
</table>
Placement Legalization

- Calculate \textit{force-values} for coincident nodes
  \begin{equation}
  \text{Force-value} = \# \text{ connections to right} - \# \text{ connections to left}
  \end{equation}

- Relative order determined by using force-values

\begin{itemize}
  \item Mobile node
  \item Fixed node
\end{itemize}

J. Lillis, UIC
Legalized Placement

Relaxed Placement with force-values

$len = 14$

Legalized Placement

$len = 18$

- Mobile node
- Fixed node

CSE 248 Linear Placement / Spectral 18

J. Lillis, UIC

Kahng, UCSD
Outline

- Hur-Lillis linear placement
- Spectral ratio cut partitioning
- Spectral partitioning: more eigenvectors is better
Philosophy of Constraint Relaxation

Relax constraints (e.g., slot constraints)
Solve relaxed formulation optimally (and efficiently)
Resolve infeasibility (cell overlap) heuristically (e.g., by recursive partitioning)

- **Advantages**
  - Adopts a global view of optimization problem
  - Generally fairly efficient

- **Disadvantages**
  - Tends to be deterministic (can’t exploit additional CPU)
  - “Over-relaxation”: Because of cell overlap, final solution tends to be far from relaxed

- See Hur-Lillis MONGREL paper: [http://portal.acm.org/citation.cfm?id=602940](http://portal.acm.org/citation.cfm?id=602940)
Philosophy of Local Search

Repeat perturbation until solution is converged

- Advantages
  - Non-deterministic
  - Easy use of additional CPU

- Disadvantages
  - “Moves” traditionally very simple (e.g., swapping)
  - May not scale well with problem size
  - Lillis et al.: several “big yet efficient” local search methods for interconnect tree design and standard-cell placement
Philosophy of Netlist Clustering

Cluster netlists based on some criteria
Find solution for the clustered circuit
Flatten the clustered circuit

- Advantages
  - Scale down problem size (reduce CPU usage)

- Disadvantages
  - Clustering is usually done a priori based on netlist characteristics only
  - ‘Good’ solution for the clustered circuit may not be ‘good’ for the flattened circuit
Interleaving Approach

- Search mechanism for *linear arrangement*
- Effective to optimize each row in placement
- Idea:
  - Given window size W, find a subsequence A within W from current linear arrangement of cells
  - Let B = A preserving relative order in original sequence
  - Interleave A and B to get an optimal arrangement
  - Repeat above procedures
- Solution space: all possible ways of interleaving is $C(n+m,n)$ – i.e., the size of solution space is exponential.
Optimal Interleaving

Window

Partitioning

Interleaving

A1  B1  A2  B2  B3  A3  B4

A1  A2  A3

B1  B2  B3  B4

B1  B2  A1  B3  A2  B4  A3
Interleaving by Dynamic Programming

- Suppose $|w| = n+m$
- Suppose $A = a_1, a_2, \ldots, a_n$ and $B = b_1, b_2, \ldots b_m$
- $S_{i,j}$: an optimal arrangement with $a_1, a_2, \ldots, a_i$ ($i \leq n$) and $b_1, b_2, \ldots b_j$ ($j \leq m$)
- $C(S_{i,j})$: cost of $S_{i,j}$
- Goal: $S_{n,m}$
Recurrence Relation

- $C(S_{00}) = 0$

- $S_{i,j} = \begin{cases} S_{i-1,j} a_i, & \text{if } C(S_{i-1,j} a_i) < C(S_{i,j-1} b_j) \\ S_{i,j-1} b_j, & \text{otherwise} \end{cases}$

- **Time complexity:** by using $n \times m$ table, $S_{n,m}$ can be found in $O(nm+p(n+m))$ time
  - $p = \text{total number of pins on incident nets}$
**Two Row Interleaving**

- **Idea:** Enable more flexibility by considering adjacent rows simultaneously.

- **Cells constrained to be in original rows, but enables cells to “move together” (less influence of “anchors” in other rows)**

- **Formulation:**
  - A1, B1: Subsequences of row 1 (relative order obeyed)
  - A2, B2: Subsequences of row 2 (relative order obeyed)
  - Simultaneously find optimal interleavings of (A1,B1) and (A2,B2) s.t. WL is minimized

- **Note:** only x-wire length changes
Two Row Interleaving – Uniform Case

- All cells identical width (e.g., FPGAs)
- Cells (e.g., a, b) can move together
- Solution: Dynamic Programming
DP Sketch for Uniform Case

- For Separability, “scan line” must move L-to-R
- Decomposition:
  - $k$: length of prefix in # cells
  - $i_1 \leq k$: index into subseq $A_1$
  - $[ j_1 = k - i_1 :$ implicit index into $B_1 ]$
  - $i_2 \leq k$: index into subseq $A_2$
  - $[ j_2 = k - i_2 :$ implicit index into $B_2 ]$
  - $S[k,i_1,i_2] = WL$ of opt. Interleaving of:
    - $A_1[1..i_1]$, $B_1[1..j_1]$ and
    - $A_2[1..i_2]$, $B_2[1..j_2]$
Four Ways to Form $S[k,i_1,i_2]$
Uniform Case Discussion

- Fill in table using recurrence
- Take best of four cases by incremental wirelength calculation
- Parameterization by k ensures separability required for DP
- Runtime: $O(N^3)$ for constant degree (more complex expression if pins considered)
- More difficult for variable-width standard cells

Notes

- Partition of window into A, B is done at random: is there a better approach?
  - E.g., splitting window into A, B should be “max cut” rather than “min cut”? Is there a recursive variant of MONGREL? (Think about chain, etc. examples….)
  - Selecting AAA….ABBB…B is worse than ABABABA…B?
  - Question: If dist($\Pi_1,\Pi_2$) = #transpositions (between two linear placements), is one MONGREL implementation (#interleaved subsequences, how subsequences chosen) “stronger” than others in ability to recover an optimal linear placement from a starting linear placement (at distance d)?
- Is there any point to partitioning window into A, B, C instead of A, B?
- What is DP for interleaving of 3 rows?
- MONGREL seems to dominate (or be an extension of) *CORR !
Outline

- Hur-Lillis linear placement
- Spectral ratio cut partitioning
- Spectral partitioning: more eigenvectors is better
Spectral properties of graphs

- **G = (V,E):** undirected graph, \( V = \{v_1, v_2, \ldots, v_n\} \)

- **A = A(G):** adjacency matrix
  - \( A_{ij} = 1 \) if \((v_i, v_j) \in E\), \( A_{ij} = 0 \) otherwise

- **D:** degree matrix
  - \( D_{ii} = \text{deg}(v_i), D_{ij} = 0 \) for \( i \neq j \)

- **Q = D - A:** Laplacian matrix
  - Symmetric \( \Rightarrow \) real eigenvalues
  - Non-negative definite: \( x^T Q x = \sum_{ij} Q_{ij} x_i x_j \geq 0 \) for every vector \( x \)
    \( \Rightarrow \) all eigenvalues are \( \geq 0 \)

- **N.B.:** Any square matrix \( M \) has at least one nonzero vector \( \mathbf{v} \) which satisfies \( M \mathbf{v} = \lambda \mathbf{v} \)
  - \( \mathbf{v} \) is called an *eigenvector* of \( M \) with *eigenvalue* \( \lambda \)
K. M. Hall’s Result (1970)

- Observe: $x^TQx$ corresponds to “squared wirelength”
  
  $$x^TQx = x^TDx - x^TAx$$
  
  $$= \sum_i \text{deg}(v_i) x_i^2 - \sum_{i,j} A_{ij} x_i x_j$$
  
  $$= \sum_i \text{deg}(v_i) x_i^2 - 2\sum_{(vi,vj) \in E} x_i x_j$$
  
  $$= (\text{writing as complete square}) \sum_{(vi,vj) \in E} (x_i - x_j)^2$$

- 1-dimensional quadratic placement problem

  Find $x = (x_1, x_2, \ldots, x_n)$ with unit norm (i.e., $|x|=(x^T x)^{1/2} = 1$) such that $z := x^TQx$ is minimum
K. M. Hall’s Result (1970)

- To minimize $z$, form the Lagrangian
  \[ L = x^TQx - \lambda(x^Tx - 1) \]
- Taking first partial derivative of $L$ w.r.t. $x$ and setting it to zero yields
  \[ 2Qx - 2\lambda x = 0 \iff Qx = \lambda x \]
  
  Nontrivial 1-dimensional quadratic placement solutions are eigenvectors of $Q$

- Eigenvector $(1/\sqrt{n}, 1/\sqrt{n}, \ldots, 1/\sqrt{n})$ corresponding to smallest eigenvalue $\lambda=0$ gives trivial placement solution (all vertices placed on top of each other)

- Hall used eigenvectors corresponding to second and third eigenvalues to derive 2-dimensional placement
Minimum Ratio Cut Problem

- Given undirected graph $G=(V,E)$, partition $V$ into disjoint $U$ and $W$ s.t. the cut ratio, $|e(U,W)| / (|U| \cdot |W|)$, is minimum

- Captures “natural” partitions (cf. Leighton-Rao “sparse cuts”)

- Min cut $a|bcdef$ has cutsize=18 but is very imbalanced

- Optimum bisection $abd|cef$ has cutsize = 300

- Optimum ratio cut is $ab|cdef$
The optimum cut ratio is at least $\lambda/n$, where $\lambda$ is the second eigenvalue of the Laplacian matrix.

Proof: Consider the optimum cut $(U,W)$ and let $p=|U|/n$ and $q=|W|/n$ (note that $p+q=1$).

Define $x_i=q$ if $v_i \in U$ and $x_i=-p$ if $v_i \in W$.

- $x$ is orthogonal to $(1,1,\ldots,1)$
- $x_i - x_j = q - (-p) = 1$ for every edge $(v_i,v_j)$ crossing the cut $(U,W)$
  \[ x^TQx = |e(U,W)| \]
- $|x|^2 = q^2pn + p^2qn = pqn (p+q) = pqn = (|U| \cdot |W|)/n$
- Fact (Courant-Fischer):
  \[ \lambda = \min \{ y^TQy / |y|^2 : y \neq 0 \text{ orthogonal to } (1,1,\ldots,1) \} \]

Hence, $\lambda \leq x^TQx / |x|^2 = n |e(U,W)| / (|U| \cdot |W|)$, i.e.,

\[ |e(U,W)| / (|U| \cdot |W|) \geq \lambda / n \]
Summary

- **Spectral partitioning heuristic**
  - Make graph model for given VLSI circuit netlist
  - Find second eigenvector of Laplacian
  - Sort coordinates to induce vertex ordering
  - Find splitting point of ordering with minimum ratio cut (or, net cut)
  - See also: Pothen, Simon and Liou, Partitioning sparse matrices with eigenvectors of graphs, SIAM J. Matrix Anal. Appl. 11 (1990), pp. 430-452

- **Issues**
  - Fixed vertices, vertex weights, numerical stability, … (many)
  - 1990s: Some persistent efforts to fix spectral partitioning
    - UCSC (Chan/Schlag/Zien)
    - Waterloo (Rendl/Wolkowicz)
Outline

- Hur-Lillis linear placement
- Spectral ratio cut partitioning
- Spectral partitioning: more eigenvectors is better
Main Idea

Spectral Bipartitioning:
Split single best eigenvector into two clusters
But: many other “good” eigenvectors

Main Result: when all eigenvectors available,
k-way min-cut graph partitioning reduces to
k-way vector partitioning

The more eigenvectors we have, the closer
we are to solving the actual partitioning problem
Many Previous Spectral Methods

- Barnes (1982): Spectral bipartitioning
- Alpert/Kahng (1993-4): Geometric embeddings
- Chan et al. (1993): Directional cosines
- Frankle/Karp (1986): Linear probes
Min-Cut Graph Partitioning

Instance: Weighted graph $G(V,E)$ with $V = \{v_1, v_2, \ldots, v_n\}$, adjacency matrix $A = (a_{ij})$, number of clusters $k$

Definition: A $k$-way partitioning of $G$ is a set of clusters $P^k = \{C_1, C_2, \ldots, C_k\}$ such that for each $v_i \in V$, there is a unique $C_h \in P^k$ with $v_i \in C_h$

Objective: Minimize

$$f(P^k) = \sum_{h=1}^{k} E_h$$

where

$$E_h = \sum_{v_i \in C_h} \sum_{v_j \not\in C_h} a_{ij}$$
Key Concept: Assignment Matrix X for $P^k$

The assignment matrix $X = (x_{ih})$ is the $n \times k$ matrix with $x_{ih} = 1$ if $v_i \in C_h$ and $x_{ih} = 0$ if $v_i \notin C_h$.

$\overrightarrow{X}_h$ (column $h$ of $X$) is indicator vector for cluster $C_h$.

$$X = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
1 & 0 & 0 \\
0 & 0 & 1 \\
0 & 0 & 1 \\
0 & 0 & 1 
\end{bmatrix}$$
Min-Cuts and the Laplacian

Definition: \( D = \text{degree matrix} \),
\[ Q = \text{Laplacian matrix} = D - A \]

Theorem 1: \( f(P^k) = \text{trace}(X^T Q X) \), i.e., \( E_h = X_h^T Q X_h \)

Why: \( X_h Q \) can have nonzero entries only for vertices of \( C_h \). An entry is 0 for a vertex in \( C_h \) if all its edges are internal to the cluster; otherwise, it counts the external degree of the vertex. Then, \( X_h^T Q X_h \) sums the external degrees of vertices of \( C_h \).

\[ \begin{align*}
E_1 &= 2 \\
E_2 &= 3 \\
E_3 &= 3 \\
f(P^3) &= 8
\end{align*} \]
Example

\[
Q \equiv \begin{bmatrix}
2 & -1 & -1 & 0 & 0 \\
-1 & 2 & -1 & 0 & 0 \\
-1 & -1 & 3 & -1 & 0 \\
0 & 0 & -1 & 2 & -1 \\
0 & 0 & 0 & -1 & 1
\end{bmatrix}
\]

\[
X_1^TQ = [0 \ 0 \ 1 \ -1 \ 0]
\]

\[
X_1^TQX_1 = [1]
\]

\[
X = \begin{bmatrix}
\overrightarrow{X}_1 \\
\overrightarrow{X}_2
\end{bmatrix} = \begin{bmatrix}
1 & 0 \\
1 & 0 \\
1 & 0 \\
0 & 1 \\
0 & 1
\end{bmatrix}
\]

\[
E_1 = 1 \\
E_2 = 2
\]
The Flaw of Spectral Bipartitioning

Definition: \( \vec{\mu}_1, \vec{\mu}_2, \ldots, \vec{\mu}_n \) orthonormal eigenvectors of \( Q \) with eigenvalues \( 0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n \)

Hall: \( \vec{X}_1 = \vec{\mu}_2 \) is the optimum solution to \( E_1 = \vec{X}_1^T Q \vec{X}_1 \) such that \( ||\vec{X}_1||^2 = 1 \) and entries sum to 0

Heuristic indicator vector \( \vec{X}_1 \) that approximates \( \vec{\mu}_2 \):
- sort \( \vec{\mu}_2 \) entries and assign to 0 or 1

Unfortunately, in general no \( \vec{X}_1 \) is “close” to \( \vec{\mu}_2 \).
**Need More Eigenvectors**

Eigenvectors $\vec{\mu}_1, \vec{\mu}_2, \ldots, \vec{\mu}_n$ form a basis in $n$ dimensions

Observe: $\vec{X}_h = (\vec{X}_h^T \vec{\mu}_1) \vec{\mu}_1 + (\vec{X}_h^T \vec{\mu}_2) \vec{\mu}_2 + \ldots + (\vec{X}_h^T \vec{\mu}_n) \vec{\mu}_n$

**Theorem 2:** $f(P^k) = \sum_{h=1}^{k} \alpha_{2h}^2 \lambda_2 + \alpha_{3h}^2 \lambda_3 + \ldots + \alpha_{nh}^2 \lambda_n$

$= \sum_{h=1}^{k} \sum_{j=2}^{n} \alpha_{jh}^2 \lambda_j$ where $\alpha_{jh} = \vec{X}_h^T \vec{\mu}_j$

Eigenvectors $\mu_1, \mu_2, \ldots, \mu_n$ are mutually orthogonal $n$-vectors

$U_d = n \times d$ eigenvector matrix ($U = n \times n$ matrix of all eigenvectors)

$\Delta_d = d \times d$ diagonal matrix of eigenvalues

Orthonormal eigenvectors $\Rightarrow U^T U = UU^T = I$, so any vector

$x = UU^T x = U(U^T x) \Rightarrow x = \sum_{j=1..n} (\mu_j^T x) \mu_j$. 

Magnitude of projection of an indicator vector $X_h$ onto $\mu_j$ is $\alpha_{jh}$.

$n \times k$ projection matrix $\Gamma = (\alpha_{jh}) = U^T X$, where $\Gamma_h$ is the indicator vector for cluster $C_h$ using the eigenvector basis.
Eigenvector Basis

\[ X_h = U \Gamma_h = \sum_{j=1..n} \alpha_{jh} \mu_j \]

\[ |C_h| = \|X_h\|^2 = \sum_{j=1..n} \alpha_{jh}^2 = \|\Gamma_h\|^2 \]

**Definition**: For each \( \mu_j \), \( Q \mu_j = \lambda_j \mu_j \)

\[ \Rightarrow QU = U \Delta \]

\[ \Rightarrow \text{tr}(X^TQX) = \text{tr}(X^TU\Delta U^TX) = \text{tr}((U^TX)^T\Delta(U^TX)) = \text{tr}(\Gamma^T\Delta\Gamma) \]
Vector Partitioning

**Instance:** Set of vectors $Y = \{\vec{y}_1, \vec{y}_2, \ldots, \vec{y}_n\}$

**Definition:** A vector partitioning of $Y$ is a set of disjoint subsets $S^k = \{S_1, S_2, \ldots, S_k\}$ such that for each $\vec{y}_i \in Y$, there is a unique $S_h \in S^k$ with $\vec{y}_i \in S_h$

**Objective:** Maximize

$$g(S^k) = \sum_{h=1}^{k} ||\vec{Y}_h||^2$$

where $\vec{Y}_h = \sum_{\vec{y}_i \in S_h} \vec{y}_i$
An Example

\[\vec{Y}_1, \vec{y}_1, \vec{y}_3, \vec{y}_4, \vec{y}_5, \vec{Y}_2\]
# Graph vs. Vector Partitioning

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Graph</th>
<th>Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance</td>
<td>G(V,E), k</td>
<td>Y, k</td>
</tr>
<tr>
<td>Basic element</td>
<td>Vertex v&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Vector 〈y&lt;sub&gt;i&lt;/sub&gt;</td>
</tr>
<tr>
<td>Find</td>
<td>C&lt;sub&gt;1&lt;/sub&gt;, C&lt;sub&gt;2&lt;/sub&gt;, ..., C&lt;sub&gt;k&lt;/sub&gt;</td>
<td>S&lt;sub&gt;1&lt;/sub&gt;, S&lt;sub&gt;2&lt;/sub&gt;, ..., S&lt;sub&gt;k&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>Each v&lt;sub&gt;i&lt;/sub&gt; in a C&lt;sub&gt;h&lt;/sub&gt;</td>
<td>Each 〈y&lt;sub&gt;i&lt;/sub&gt; in a S&lt;sub&gt;h&lt;/sub&gt;</td>
</tr>
<tr>
<td>Objective</td>
<td>E&lt;sub&gt;1&lt;/sub&gt; + ... + E&lt;sub&gt;h&lt;/sub&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Min cut edges</td>
<td>Max squared sums</td>
</tr>
</tbody>
</table>
Construct vector partitioning instance \( \mathbf{Y} \) with \( j^{\text{th}} \) entry of \( \vec{y}_i \) equal to \( \mu_{ij} \sqrt{H - \lambda_j} \), where \( \mu_{ij} \) is \( i^{\text{th}} \) entry in eigenvector \( \vec{\mu}_j \).

Given solution \( S^k \): if \( \vec{y}_i \in S_h \) then assign \( \mathbf{v}_i \) to \( C_h \).

**Theorem 3:** \( g(S^k) = nH - f(P^k) \)

i.e., graph partitioning reduces to vector partitioning.

**Proof:** Show:

\[
\|Y_h\|^2 = \sum_{j=1}^{n} \alpha_{jh}^2 (H - \lambda_j)
\]

Apply Theorem 2:

\[
f(P^k) = \sum_{h=1}^{k} \sum_{j=1}^{n} \alpha_{jh}^2 \lambda_j
\]
Simple Heuristic: MELO

Observe: \( nH - f(P^k) = \sum_{h=1}^{k} \sum_{j=1}^{n} \alpha_{jh}^2 (H-\lambda_j) \)

→ smaller eigenvalue means more important eigenvector

1. Construct \( Y = \{\vec{y}_1, \vec{y}_2, \ldots, \vec{y}_n\} \) with as many eigenvectors as practical, where entry \( j \) of \( \vec{y}_i \) is \( \mu_{ij} \sqrt{H-\lambda_j} \). Set \( S = \emptyset \).

2. Iteratively add \( \vec{y}_i \in Y \) to \( S \) to maximize \( \| \sum_{y \in S} \vec{y} + \vec{y}_i \|^2 \), remove \( \vec{y}_i \) from \( Y \).

3. Induce vertex ordering from Step 2, split into clusters.
An Example

Order: $y_4 \ y_2 \ y_5 \ y_1 \ y_3$
Summary

- Spectral bipartitioning: $\mu_2$ is not close to $X_1$

- Graph partitioning reduces to vector partitioning (more eigenvectors = better)

- MELO ordering heuristic (allows application of DP-RP)

- Still open: appropriate applications, objectives (FPGAs, Scaled Cost)

- Still open: “direct” vector partitioning heuristics
EXTRA (MELO experimental data)
Multi-way Experiments I

Scaled Cost Results

MELO % Improvement

Benchmark

19ks p1 p2 t2 t3 t4 t5 t6 str bio

-20 -10 0 10 20 30 40

KP SB SFC MELO(10)
Multi-way Experiments II

Scaled Cost Results

Number of clusters

MELO % Improvement

KP
SB
SFC
MELO(10)
Bipartitioning Results

Ratio cut \( \times 10^5 \)

Benchmark

SB
PARABOLI
MELO

p1  balu  str  p2  bio  9234  13207  15850  in2
EXTRA (CSE 101 Network Flow slides – see website)