Nested Dissection and Graph Partitioning

Fabiano Oyafuso

CS 497mh

February 25, 1999

Outline

- Nested Dissection
  - basic idea
  - planar graph – example
  - planar graph – optimality

- General Graph Partitioning
  - overview
  - coordinate approach
  - a modified breadth first search
  - kernighan-lin algorithm
  - spectral partitioning

Definitions

- **Edge Separator:** Given a partition of the set of nodes of a graph $\mathcal{G}\,(V, E)$, into sets A and B,

$$\delta(A, B) = \{(x, y) \in E : x \in A, y \in B\}$$

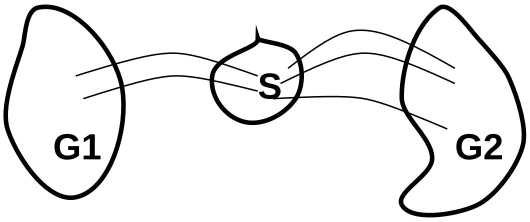
- **Vertex Separator:** A subset $S \subset V$ whose removal disconnects $\mathcal{G}\,(V, E)$ into two components A and B such that

$$(\forall (x, y) \in E)\ x, y \in A \cup S \ \text{ or } \ x, y \in B \cup S$$

- **Bisection:** A partition $V \to A, B$ is a bisection if $|A| \approx |B|$.

Basic Idea of Nested Dissection

- Basic Algorithm
  - Select a vertex separator S that separates G_1 and G_2
  - reorder columns and rows so that indices(G_1) $<$ indices(G_2) $<$ indices(S)
  - iterate



- Remarks
  - reordering produces matrix of form

$$\begin{bmatrix} A_1 & 0 & V_1 \\ 0 & A_2 & V_2 \\ V_1^T & V_2^T & A_3 \end{bmatrix}$$

  - key is that there is no fill in zero-blocks during decomposition
  - $|S|$ should be small to minimize fill
  - $|\mathcal{G}_1| \approx |\mathcal{G}_2|$
  - algorithm is often used for graphs arising from FD, FEM applications

Regular Grid Example

86	87	88	89	90	100	40	39	38	37
81	82	83	84	85	99	36	35	34	33
76	77	78	79	80	98	32	31	30	29
71	72	73	74	75	97	28	27	26	25
66	67	68	69	70	96	24	23	22	21
61	62	63	64	65	95	20	19	18	17
56	57	58	59	60	94	16	15	14	13
51	52	53	54	55	93	12	11	10	9
46	47	48	49	50	92	8	7	6	5
41	42	43	44	45	91	4	3	2	1

Figure 8.2.1: A one-level dissection ordering of a 10 by 10 grid.

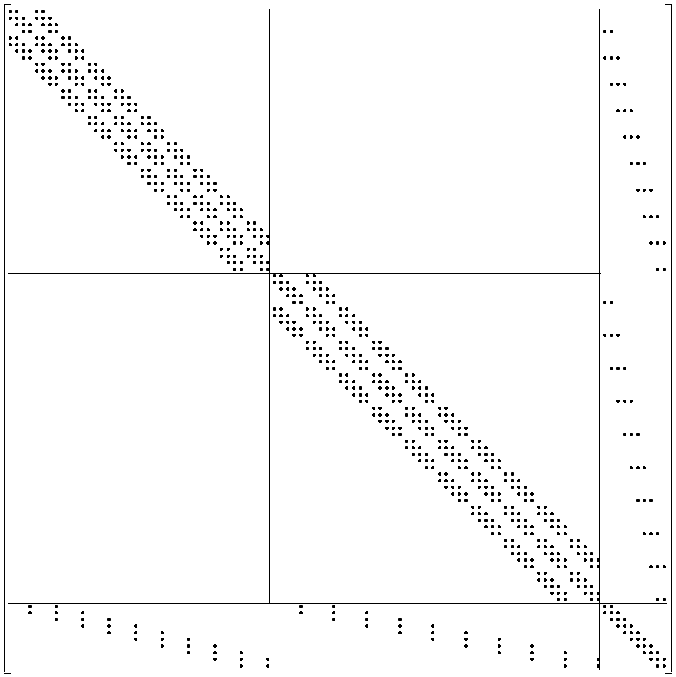
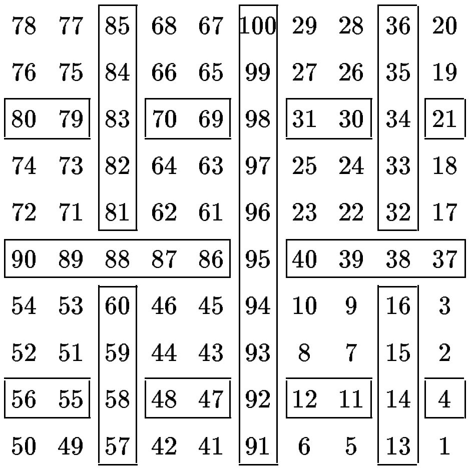


Figure 8.2.2: Matrix structure associated with a one-level dissection ordering.

Regular Grid Example (ctd.)



78	77	85	68	67	100	29	28	36	20
76	75	84	66	65	99	27	26	35	19
80	79	83	70	69	98	31	30	34	21
74	73	82	64	63	97	25	24	33	18
72	71	81	62	61	96	23	22	32	17
90	89	88	87	86	95	40	39	38	37
54	53	60	46	45	94	10	9	16	3
52	51	59	44	43	93	8	7	15	2
56	55	58	48	47	92	12	11	14	4
50	49	57	42	41	91	6	5	13	1

Figure 8.2.3: A nested dissection ordering of a 10 by 10 grid.

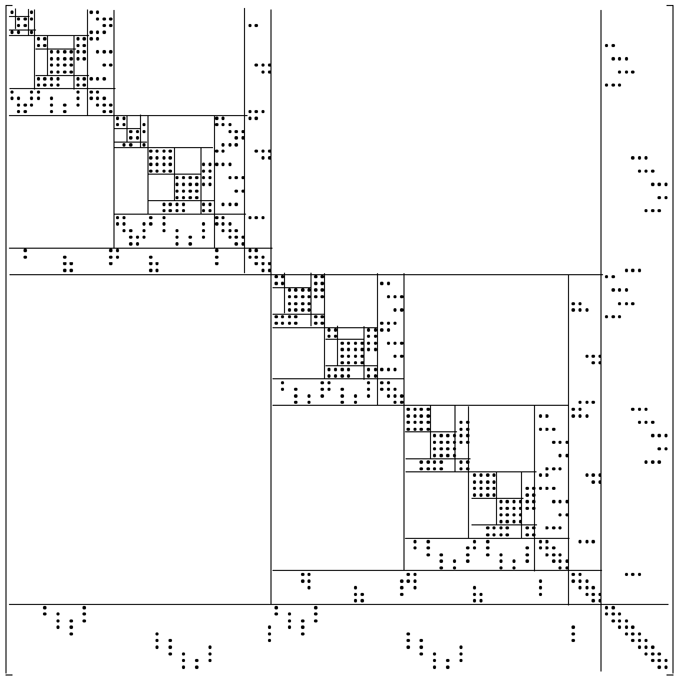


Figure 8.2.4: Matrix structure associated with a nested dissection ordering.

Optimality of Ordering

- Again, consider the LU factorization of an SPD matrix determined by a regular $s \times s$ grid. ND achieves the following:
  - number of nonzeros of L $\sim \mathcal{O}\left(s^2 log_2 s\right)$
  - operation count $\sim \mathcal{O}\left(s^3\right)$
- These values are optimal in an order of magnitude sense.

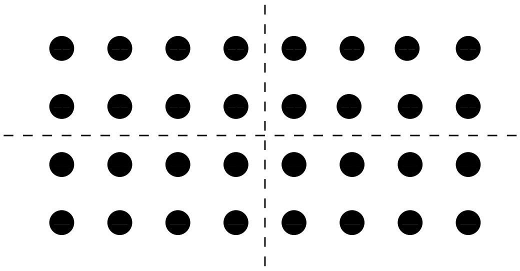
Inertial Partitioning

- Key to ND is selection of vertex partition S.
- There are two classes of partitioning algorithms:
  - coordinate based
  - coordinate free

  Consider the following simple coordinate based heuristic algorithm:
- Inertial Partitioning Algorithm

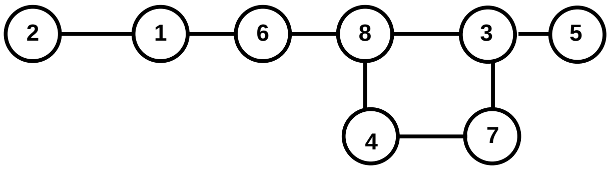
```
1. Pick a line L about which the tensor
   of inertia is small
2. Find the center of mass and project to L
3. The line normal to L passing through the
   center of mass divides G into G1 and G2.
```

- Example

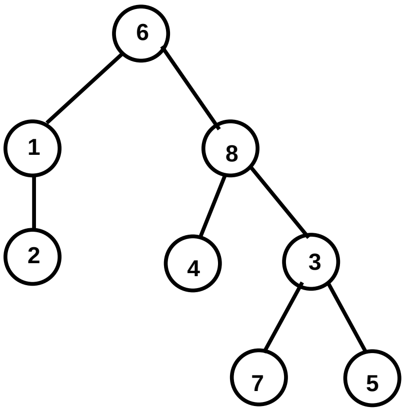


More Definitions

- **Eccentricity:** $\epsilon(x) = sup_{y \in X} \{d(x,y)\}$
- **Diameter:** $\delta(\mathcal{G}) = sup_{x \in X} \epsilon(x)$
- **Peripheral Node:** $x \in X$ is *peripheral* if $\epsilon(x) = \delta(\mathcal{G})$.
- **Level Structure:** a partitioning $\mathcal{L} = \{L_0, ..., L_\ell\}$ such that Adj$(L_i) \subset L_{i+1} \cup L_{i-1}$



An 8-node graph with diameter 5



Level structure rooted at 6

Breadth First Search

- Constructs a subgraph (a level tree) T of G, rooted at r.

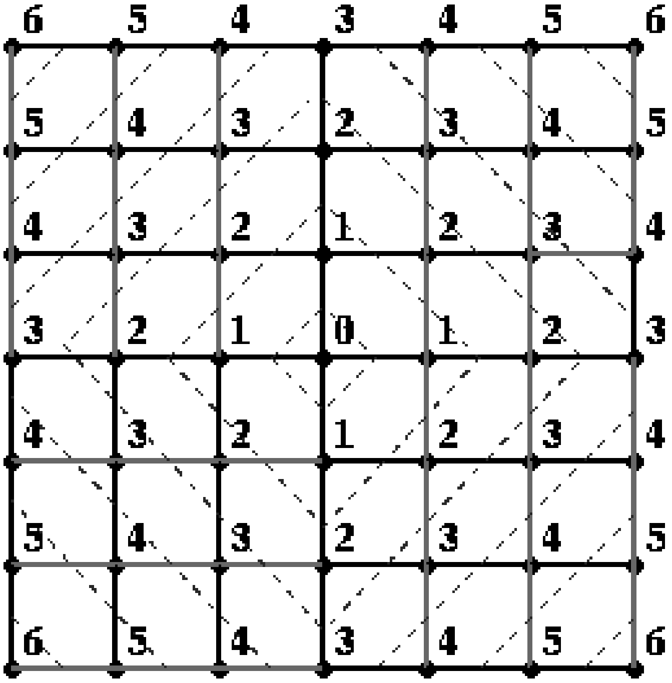
  Basic algorithm:

```
T.N = { (r,0) }; T.E := NULL;
   Q.enqueue(r,0); mark r

while ( Q.not_empty() ) {
   (n,level) = Q.deque();
   for (unmarked child c of n) {
      T.N = T.N U (c,level+1);
      T.E = T.E U (n,c);
      Q.enqueue(c,level+1); mark c;
   }
}
```

- Remarks
  - BFS covers all $n \in N$ (if N is connected), but ...
  - There are no edges connecting nodes in levels differing by more than 1. (e.g. There are no extra edges connecting root to a leaf)

Another Level Structure Example



Breadth First Seach on a 7 by 7 Grid
Starting at the center node
Nodes labeled by level

Finding a pseudo-peripheral node

- Finding peripheral nodes is too expensive. Instead find *pseudo-peripheral nodes* with following algorithm. These nodes tend to have large eccentricities.

Basic algorithm:

1. pick arbitrary node $r \in X$
2. generate a level structure with $\epsilon(r)$ levels
3. choose node x in last level having minimum degree
4. generate a level structure rooted at x
5. if $(\epsilon(x) > \epsilon(r))$ then $r \leftarrow x$, goto 3; else x is pseudo-peripheral node

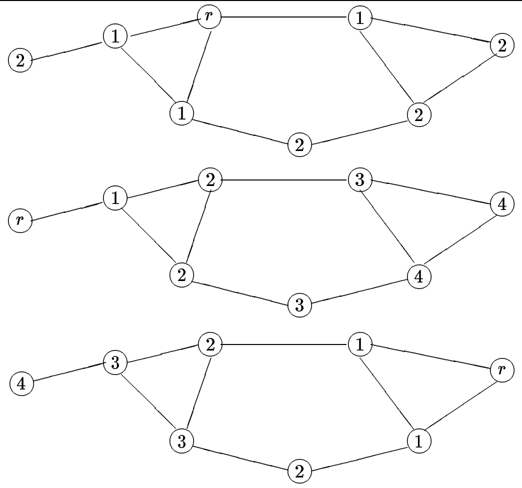


Figure 4.4.9: An example of the application of the pseudo-peripheral node finding algorithm.

ND – a Heuristic Algorithm

1. (Initialize) $R = X$

2. (Construct a level structure). For each connected component C construct a level structure (with ℓ levels).

3. If $\ell \leq 2$, set $S \leftarrow C$ and goto 4.
   Else let $j \leftarrow \lfloor \ell + 1)/2 \rfloor$ and set
   $$S = \{y \in L_j | Adj(y) \cap L_{j+1} \neq \varnothing\}$$

4. Label nodes in S from $|R| - |S| + 1$ to $|R|$. Reset $R \leftarrow R - S$. If $R \neq \varnothing$ goto 2

Kernighan/Lin – introduction

- Original version (ca. 1970) $\sim \mathcal{O}\left(|N|^3\right)$. More efficient algorithm introduced (1982) by Fiduccia ($\sim \mathcal{O}\left(|E|\right)$).
- Today, K/L is used only as part of multi-level approach
- Basic Idea:
  - Assign weights to edges such that $w(e) = 0$ if $(e \notin E)$.
  - For simplicity, take $w(e) = 1$ if $(e \in E)$.
  - Given a partition $\mathcal{G} = A \cup B$, find $X \subset A$ and $Y \subset B$ such that for $A' = A - X \cup Y$ and $B' = B - Y \cup X$, the total cost

    $$T = \sum_{a \in A', b \in B'} w(a, b)$$

    is reduced.
- Definitions:
  - external cost of a: $E(a) = \sum_{b \in B} w(a, b)$
  - internal cost of a: $I(a) = \sum_{a' \in A(\neq a)} w(a, a')$
  - total cost of a: $D(a) = E(a) - I(a)$
  - Swapping $X = \{a\}, Y = \{b\}$ gives

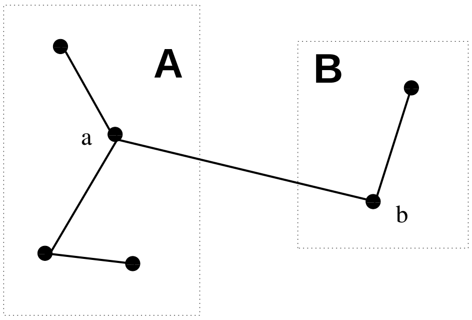
    $$\delta T = -(D(a) + D(b) - 2w(a, b)) \text{ (-gain)}$$

    and

    $$\delta D(a') = 2(w(a', a) - w(a', b))$$

    for $a' \neq a$.

Kernighan/Lin – Example

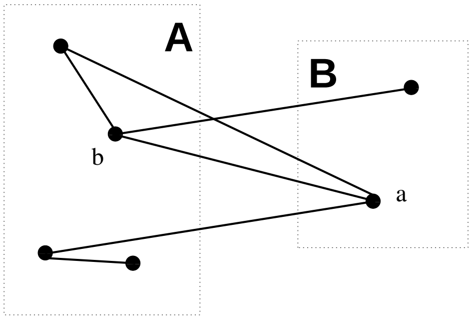


T = 1

E_a = 1; I_a = 2; D_a = -1

E_b = 1; I_b = 1; D_b = 0

δT = 3



T = 4

E_a = 2; I_a = 0; D_a =2

E_b = 2; I_b = 1; D_b = 1

Kernighan/Lin – Basic Algorithm

```
compute T
while ( gain >= 0) {
   compute D(n) for all n in N
   unmark all nodes in G
   while ( unmarked nodes exist ) {
      Find unmarked (a,b) st gain(a,b) is maximized
      mark a,b
      update D(n) for all unmarked n as if a,b
               had been swapped
   }

   // so far we have computed a sequence of pairs
   // (a[1],b[1]), ..., (a[N/2],b[N/2]) and
   // gains g[1], ..., g[N/2]

   form the running sum G[j] = sum_{i=1:j} g[j] and
      select j such that G[j] is maximized
   if G[j_max] > 0 perform swaps and update A,B,T
}
```

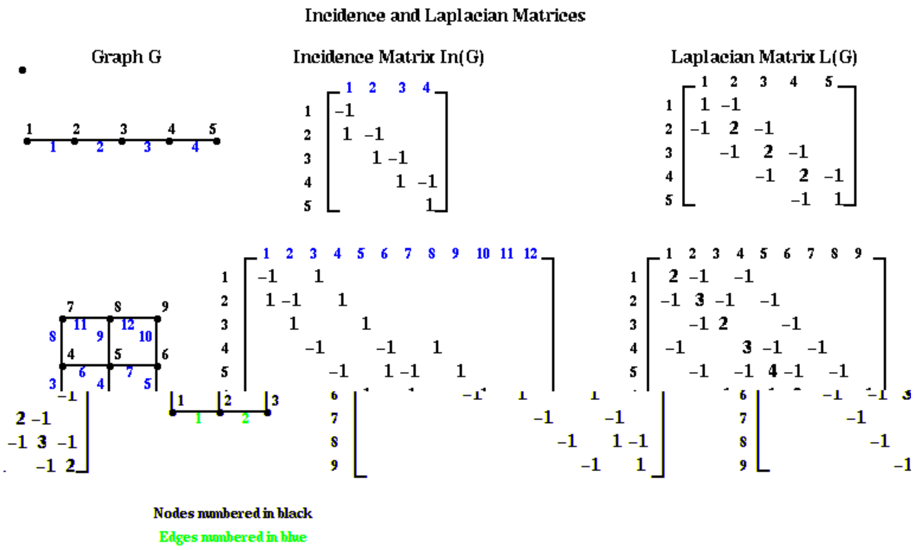
- Note that gain(a,b) may be negative.
- inner while loop runs over $|N|/2$ iterations. Maximizing gain(a,b) is $\mathcal{O}\left(|N|^2\right)$, making algorithm $\sim \mathcal{O}\left(|N|^3\right)$
- empirical testing by K/L on small graphs ($|N| < 360$) show convergence after 2-4 steps

Spectral Partitioning

- Incidence matrix: $In(G)$ is an $|N| \times |E|$ matrix whose columns consist of the edges in E.

- Laplacian matrix: $L(G)$ is an $|N| \times |N|$ matrix of the form

$$
\begin{aligned}
L(G)_{ij} &= deg(i) \quad i = j \\
&= -1 \quad i \neq j \ \text{ and } \ e(i,j) \in E \\
&= 0 \quad \text{else}
\end{aligned}
$$



Spectral Partitioning –ctd.

- $L(G)$ is symmetric, so eigenvalues are real
- sum of rows of $L(G)$ vanishes. Gerschgorin disks all lie in right half of plane. Order eigenvalues by

$$0 \leq \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$$

- The number of connected components equals the multiplicity of the eigenvalue 0.
- λ_2 is the algebraic connectivity of a graph $\mathcal{G}$. If $\mathcal{G}'\,(N, E') \subset \mathcal{G}\,(N, E)$ then $\lambda_2(\mathcal{G}') \leq \lambda_2(\mathcal{G})$, i.e. $\mathcal{G}'$ is "less connected" than $\mathcal{G}$.
- Algorithm:

```
compute v2 corresponding to lambda2

for (n in N) {
   if (v2(n) < 0)
      put n in partition N1
   else
      put n in partition N2
}
```

Spectral Partitioning – more

- Cost of finding eigenvectors is expensive, but we need only find approximate solution. Hint: use an iterative method (Lanczos).

- Note that really this eigenvalue problem is just the Helmholtz equation where the L is just the discretized Laplacian with Neumann BC ($\nabla\psi|\partial\Omega = 0$):
  $$\nabla^2\psi = -k^2\psi$$
  – the fundamental modes of which are just cosines.