Minimizing Computation in Stiffness Matrix Assembly

Michael M. Wolf

CS 598LO

December 6, 2005

Introduction

- Work by Kirby, et al., University of Chicago
- Finite element "Compilers"
  - FErari
  - FIAT
- Construction of FE Matrices extremely expensive for large unstructured problems
- Methods for reducing redundant operations in building stiffness matrix in 2D
  - Many local stiffness matrices built
  - Generate code to minimize multiply-add pairs (MAPs) in construction of local stiffness matrix

2D Laplace – Local Stiffness Matrix Assembly

- Element bilinear form:

$$(\nabla u, \nabla v)_e = \det(J)(\nabla u, \nabla v)_{\hat{e}}$$

$$= \det(J)\left[\begin{array}{l} \left(\frac{\partial u}{\partial r}\frac{\partial r}{\partial x}, \frac{\partial v}{\partial r}\frac{\partial r}{\partial x}\right)_{\hat{e}} + \left(\frac{\partial u}{\partial r}\frac{\partial r}{\partial x}, \frac{\partial v}{\partial s}\frac{\partial s}{\partial x}\right)_{\hat{e}} + \left(\frac{\partial u}{\partial s}\frac{\partial s}{\partial x}, \frac{\partial v}{\partial r}\frac{\partial r}{\partial x}\right)_{\hat{e}} + \left(\frac{\partial u}{\partial s}\frac{\partial s}{\partial x}, \frac{\partial v}{\partial s}\frac{\partial s}{\partial x}\right)_{\hat{e}} \\ + \left(\frac{\partial u}{\partial r}\frac{\partial r}{\partial y}, \frac{\partial v}{\partial r}\frac{\partial r}{\partial y}\right)_{\hat{e}} + \left(\frac{\partial u}{\partial r}\frac{\partial r}{\partial y}, \frac{\partial v}{\partial s}\frac{\partial s}{\partial y}\right)_{\hat{e}} + \left(\frac{\partial u}{\partial s}\frac{\partial s}{\partial y}, \frac{\partial v}{\partial r}\frac{\partial r}{\partial y}\right)_{\hat{e}} + \left(\frac{\partial u}{\partial s}\frac{\partial s}{\partial y}, \frac{\partial v}{\partial s}\frac{\partial s}{\partial y}\right)_{\hat{e}} \end{array}\right]$$

$$= \det(J)\left[\begin{array}{l} \frac{\partial r}{\partial x}\left(\frac{\partial u}{\partial r}, \frac{\partial v}{\partial r}\right)_{\hat{e}}\frac{\partial r}{\partial x} + \frac{\partial r}{\partial x}\left(\frac{\partial u}{\partial r}, \frac{\partial v}{\partial s}\right)_{\hat{e}}\frac{\partial s}{\partial x} + \frac{\partial s}{\partial x}\left(\frac{\partial u}{\partial s}, \frac{\partial v}{\partial r}\right)_{\hat{e}}\frac{\partial r}{\partial x} + \frac{\partial s}{\partial x}\left(\frac{\partial u}{\partial s}, \frac{\partial v}{\partial s}\right)_{\hat{e}}\frac{\partial s}{\partial x} \\ + \frac{\partial r}{\partial y}\left(\frac{\partial u}{\partial r}, \frac{\partial v}{\partial r}\right)_{\hat{e}}\frac{\partial r}{\partial y} + \frac{\partial r}{\partial y}\left(\frac{\partial u}{\partial r}, \frac{\partial v}{\partial s}\right)_{\hat{e}}\frac{\partial s}{\partial y} + \frac{\partial s}{\partial y}\left(\frac{\partial u}{\partial s}, \frac{\partial v}{\partial r}\right)_{\hat{e}}\frac{\partial r}{\partial y} + \frac{\partial s}{\partial y}\left(\frac{\partial u}{\partial s}, \frac{\partial v}{\partial s}\right)_{\hat{e}}\frac{\partial s}{\partial y} \end{array}\right]$$

$$\vdots$$

$$= \det(J)\left[\begin{array}{l} \frac{\partial r}{\partial x}\left(u^T D_{rr} v\right)\frac{\partial r}{\partial x} + \frac{\partial r}{\partial x}\left(u^T D_{rs} v\right)\frac{\partial s}{\partial x} + \frac{\partial s}{\partial x}\left(u^T D_{sr} v\right)\frac{\partial r}{\partial x} + \frac{\partial s}{\partial x}\left(u^T D_{ss} v\right)\frac{\partial s}{\partial x} \\ + \frac{\partial r}{\partial y}\left(u^T D_{rr} v\right)\frac{\partial r}{\partial y} + \frac{\partial r}{\partial y}\left(u^T D_{rs} v\right)\frac{\partial s}{\partial y} + \frac{\partial s}{\partial y}\left(u^T D_{sr} v\right)\frac{\partial r}{\partial y} + \frac{\partial s}{\partial y}\left(u^T D_{ss} v\right)\frac{\partial s}{\partial y} \end{array}\right],$$

$$D_{rr}(i,j) = \left(\frac{\partial \phi_i}{\partial r}, \frac{\partial \phi_j}{\partial r}\right)_{\hat{e}} \cdots$$

2D Laplace – Local Stiffness Matrix Assembly

$$\left(\nabla u, \nabla v\right)_e = u^T \left(\det(J) \begin{bmatrix} \frac{\partial r}{\partial x} D_{rr} \frac{\partial r}{\partial x} + \frac{\partial r}{\partial x} D_{rs} \frac{\partial s}{\partial x} + \frac{\partial s}{\partial x} D_{sr} \frac{\partial r}{\partial x} + \frac{\partial s}{\partial x} D_{ss} \frac{\partial s}{\partial x} \\ + \frac{\partial r}{\partial y} D_{rr} \frac{\partial r}{\partial y} + \frac{\partial r}{\partial y} D_{rs} \frac{\partial s}{\partial y} + \frac{\partial s}{\partial y} D_{sr} \frac{\partial r}{\partial y} + \frac{\partial s}{\partial y} D_{ss} \frac{\partial s}{\partial y} \end{bmatrix} \right) v$$

$$S^e = \det(J) \begin{bmatrix} \frac{\partial r}{\partial x} D_{rr} \frac{\partial r}{\partial x} + \frac{\partial r}{\partial x} D_{rs} \frac{\partial s}{\partial x} + \frac{\partial s}{\partial x} D_{sr} \frac{\partial r}{\partial x} + \frac{\partial s}{\partial x} D_{ss} \frac{\partial s}{\partial x} \\ + \frac{\partial r}{\partial y} D_{rr} \frac{\partial r}{\partial y} + \frac{\partial r}{\partial y} D_{rs} \frac{\partial s}{\partial y} + \frac{\partial s}{\partial y} D_{sr} \frac{\partial r}{\partial y} + \frac{\partial s}{\partial y} D_{ss} \frac{\partial s}{\partial y} \end{bmatrix}$$

$$S^e = D_{rr} \left[\det(J) \left(\frac{\partial r}{\partial x} \frac{\partial r}{\partial x} + \frac{\partial r}{\partial y} \frac{\partial r}{\partial y} \right) \right]_e + D_{rs} \left[\det(J) \left(\frac{\partial r}{\partial x} \frac{\partial s}{\partial x} + \frac{\partial r}{\partial y} \frac{\partial s}{\partial y} \right) \right]_e$$
$$+ D_{sr} \left[\det(J) \left(\frac{\partial s}{\partial x} \frac{\partial r}{\partial x} + \frac{\partial s}{\partial y} \frac{\partial r}{\partial y} \right) \right]_e + D_{ss} \left[\det(J) \left(\frac{\partial s}{\partial x} \frac{\partial s}{\partial x} + \frac{\partial s}{\partial y} \frac{\partial s}{\partial y} \right) \right]_e$$

- Used in class 2D code
- Terms not grouped by element dependency
- 8*(# bases)^2 MAPs

"Tensor" K

$$S^e = D_{rr}\left[\det(J)\left(\frac{\partial r}{\partial x}\frac{\partial r}{\partial x}+\frac{\partial r}{\partial y}\frac{\partial r}{\partial y}\right)\right]_e + D_{rs}\left[\det(J)\left(\frac{\partial r}{\partial x}\frac{\partial s}{\partial x}+\frac{\partial r}{\partial y}\frac{\partial s}{\partial y}\right)\right]_e$$

$$+ D_{sr}\left[\det(J)\left(\frac{\partial s}{\partial x}\frac{\partial r}{\partial x}+\frac{\partial s}{\partial y}\frac{\partial r}{\partial y}\right)\right]_e + D_{ss}\left[\det(J)\left(\frac{\partial s}{\partial x}\frac{\partial s}{\partial x}+\frac{\partial s}{\partial y}\frac{\partial s}{\partial y}\right)\right]_e$$

$$S^e{}_{i,j} = \sum_m^2 \sum_n^2 G^e{}_{m,n} K_{i,j,m,n} = K_{i,j} : G^e$$

$$G^e = \begin{bmatrix} \left[\det(J)\left(\frac{\partial r}{\partial x}\frac{\partial r}{\partial x}+\frac{\partial r}{\partial y}\frac{\partial r}{\partial y}\right)\right]_e & \left[\det(J)\left(\frac{\partial r}{\partial x}\frac{\partial s}{\partial x}+\frac{\partial r}{\partial y}\frac{\partial s}{\partial y}\right)_e\right] \\ \left[\det(J)\left(\frac{\partial s}{\partial x}\frac{\partial r}{\partial x}+\frac{\partial s}{\partial y}\frac{\partial r}{\partial y}\right)\right]_e & \left[\det(J)\left(\frac{\partial s}{\partial x}\frac{\partial s}{\partial x}+\frac{\partial s}{\partial y}\frac{\partial s}{\partial y}\right)\right]_e \end{bmatrix}$$

$$K_{i,j} = \begin{bmatrix} D_{rr}(i,j) & D_{rs}(i,j) \\ D_{sr}(i,j) & D_{ss}(i,j) \end{bmatrix} = \begin{bmatrix} \left(\frac{\partial \phi_i}{\partial r},\frac{\partial \phi_j}{\partial r}\right)_{\hat{e}} & \left(\frac{\partial \phi_i}{\partial r},\frac{\partial \phi_j}{\partial s}\right)_{\hat{e}} \\ \left(\frac{\partial \phi_i}{\partial s},\frac{\partial \phi_j}{\partial r}\right)_{\hat{e}} & \left(\frac{\partial \phi_i}{\partial s},\frac{\partial \phi_j}{\partial s}\right)_{\hat{e}} \end{bmatrix}$$

“Tensor” K

$$K_{i,j} = \begin{bmatrix} \left(\frac{\partial \phi_i}{\partial r}, \frac{\partial \phi_j}{\partial r}\right)_{\hat{e}} & \left(\frac{\partial \phi_i}{\partial r}, \frac{\partial \phi_j}{\partial s}\right)_{\hat{e}} \\ \left(\frac{\partial \phi_i}{\partial s}, \frac{\partial \phi_j}{\partial r}\right)_{\hat{e}} & \left(\frac{\partial \phi_i}{\partial s}, \frac{\partial \phi_j}{\partial s}\right)_{\hat{e}} \end{bmatrix}$$

	$\phi_{0,0}$		$\phi_{1,0}$		$\phi_{0,1}$	
$\phi_{0,0}$	1	1	0	-1	-1	0
	1	1	0	-1	-1	0
$\phi_{1,0}$	0	0	0	0	0	0
	-1	-1	0	1	1	0
$\phi_{0,1}$	-1	-1	0	1	1	0
	0	0	0	0	0	0

- Example of “tensor” K
- p=1
- S(1,2)=-G(2)-G(4)

$$S^e{}_{i,j} = K_{i,j} : G^e$$

Optimization problem

$$S^e{}_{i,j} = K_{i,j} : G^e$$

- Objective: To generate code to minimize the number of multiply add pairs (MAPs) when building the local stiffness matrix

Possible Optimizations

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- Example of "tensor" K
- p=2
- Many possible optimizations for $K(i,j):G^e$ dot product

Possible Optimizations – 0 Blocks

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- 0 blocks
- S(1,5)=0
- Dot product unnecessary
- 0 MAPs

Possible Optimizations - Same Blocks

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- Same blocks
- S(3,2)=S(1,2)
- 0 MAPs

Possible Optimizations - 1 NZ Blocks

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- 1 NZ block
- S(6,6)=3G(1)
- 1 MAP

Possible Optimizations - 2 NZ Blocks

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- 2 NZ block
- S(1,2) = -4G(2)-4G(4)
- 2 MAPs

Possible Optimizations – 3 NZ Blocks

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- 3 NZ block
- S(2,5)=-8G(1) -4G(2)-4G(3)
- 3 MAPs

Possible Optimizations – Scalar Multiple Blocks

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- Scalar multiple blocks
- S(1,3)=-4S(1,2)
- 1 MAP

Possible Optimizations – More Complex

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- Partial scalar multiple blocks
- $S(2,2) = -S(2,5) + 8G(4)$
- 2 MAPs
- Many other optimizations possible…

Implementation

- Modified 2D Helmholtz code
- "Tensor" representation of local stiffness matrix
- Implemented several optimizations
  - 0 block
  - Same block
  - 1 NZ, 2NZ, 3NZ
  - Scalar multiple block
  - Partial scalar multiple block
- Graph model for more complex block relationships
- Implementation reports an optimal (minimal MAPs) set of operations to build stiffness matrix
  - Optimal for block relationships used

Implementation

S =

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Implementation – 0 Blocks

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- No change to S

Implementation

S =

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Implementation – Same Blocks

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- 0 MAPs
- S(i,j)=S(k,l)

Implementation

S =

0	0	0	0	0	0
0	0	$S(1,2)$	0	0	0
0	$S(2,1)$	0	0	0	0
0	$S(2,4)$	0	$S(2,2)$	0	$S(1,4)$
0	$S(2,5)$	0	$S(4,5)$	$S(2,2)$	$S(3,5)$
0	0	0	$S(4,1)$	$S(5,3)$	0

Implementation

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

- Build graph from remaining blocks
- One vertex for each block
- One dot product vertex
- Weighted Edges
- Weights – work to determine a block given another block

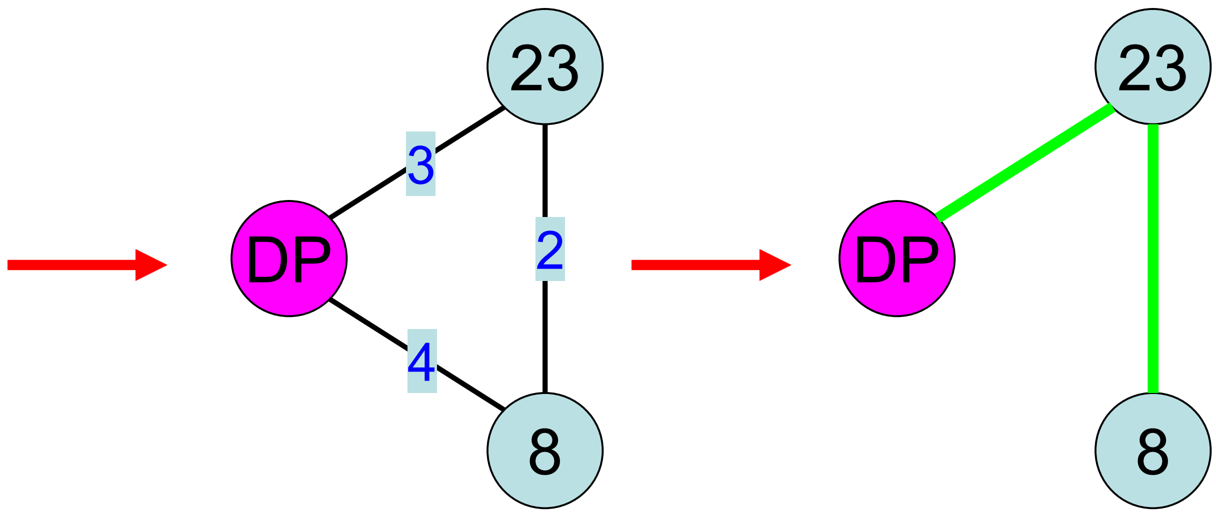
Graph Problem

B8

8	4
4	8

B23

0	-4
-4	-8



Graph

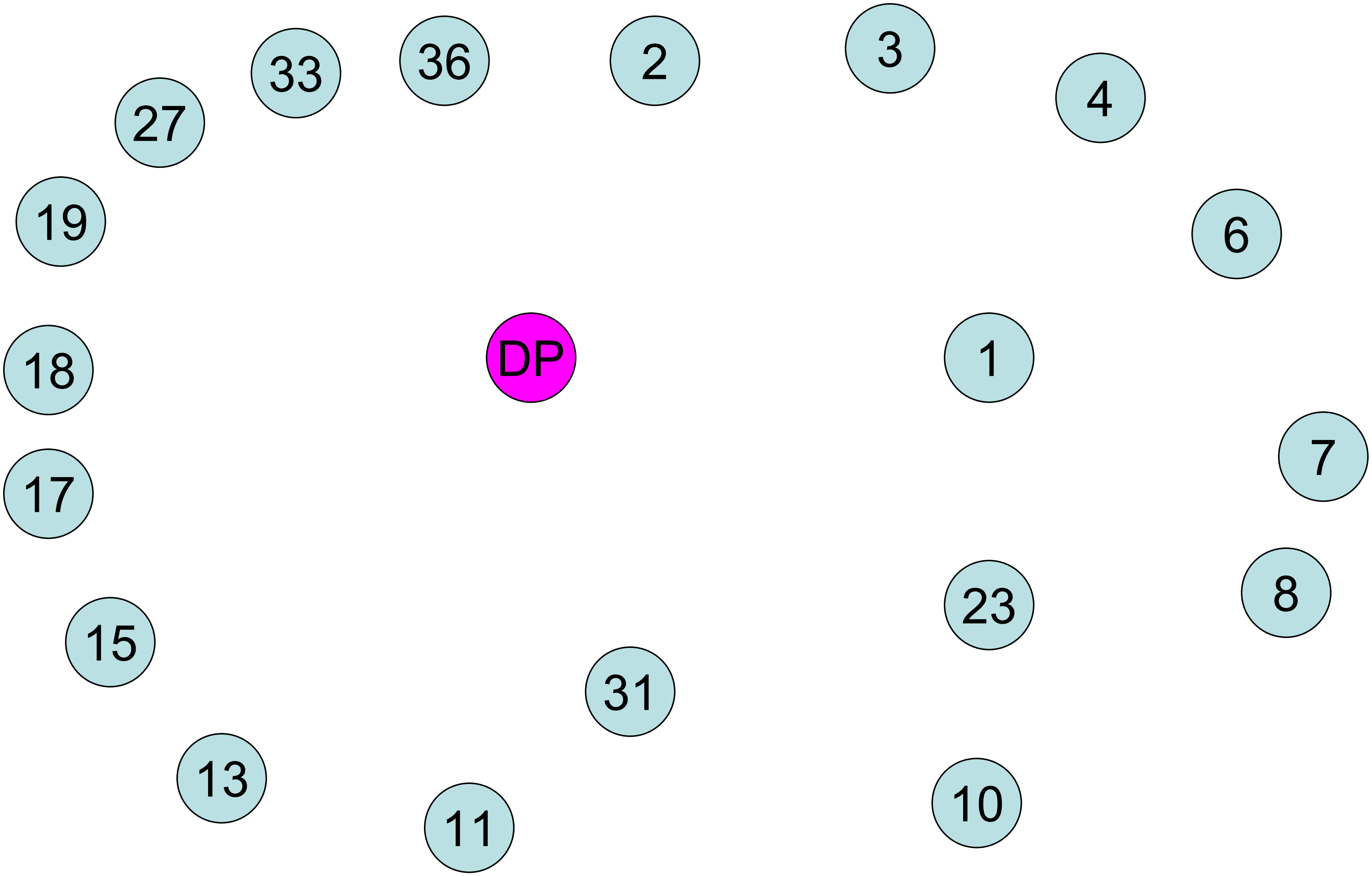
MST(5)

S8=-S23+8G1

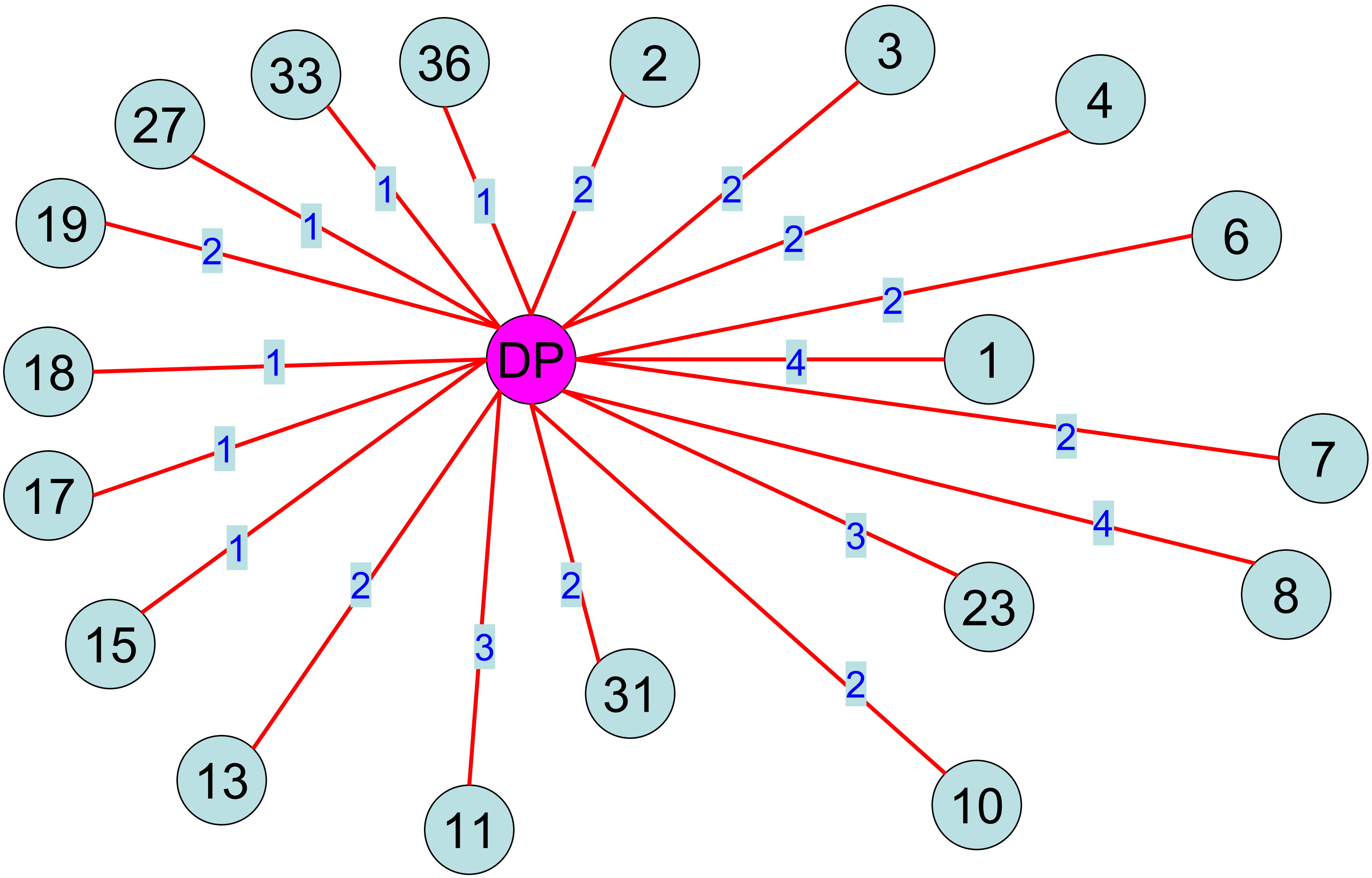
Implementation

3	3	0	-4	0	1	-4	0	0	0	1	0
3	3	0	-4	0	1	-4	0	0	0	1	0
0	0	8	4	0	-4	0	4	-8	-4	0	0
-4	-4	4	8	0	-4	4	0	-4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	-4	-4	0	3	0	0	4	0	-1	0
-4	-4	0	4	0	0	8	4	0	-4	-4	0
0	0	4	0	0	0	4	8	-4	-8	-4	0
0	0	-8	-4	0	4	0	-4	8	4	0	0
0	0	-4	0	0	0	-4	-8	4	8	4	0
1	1	0	0	0	-1	-4	-4	0	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0

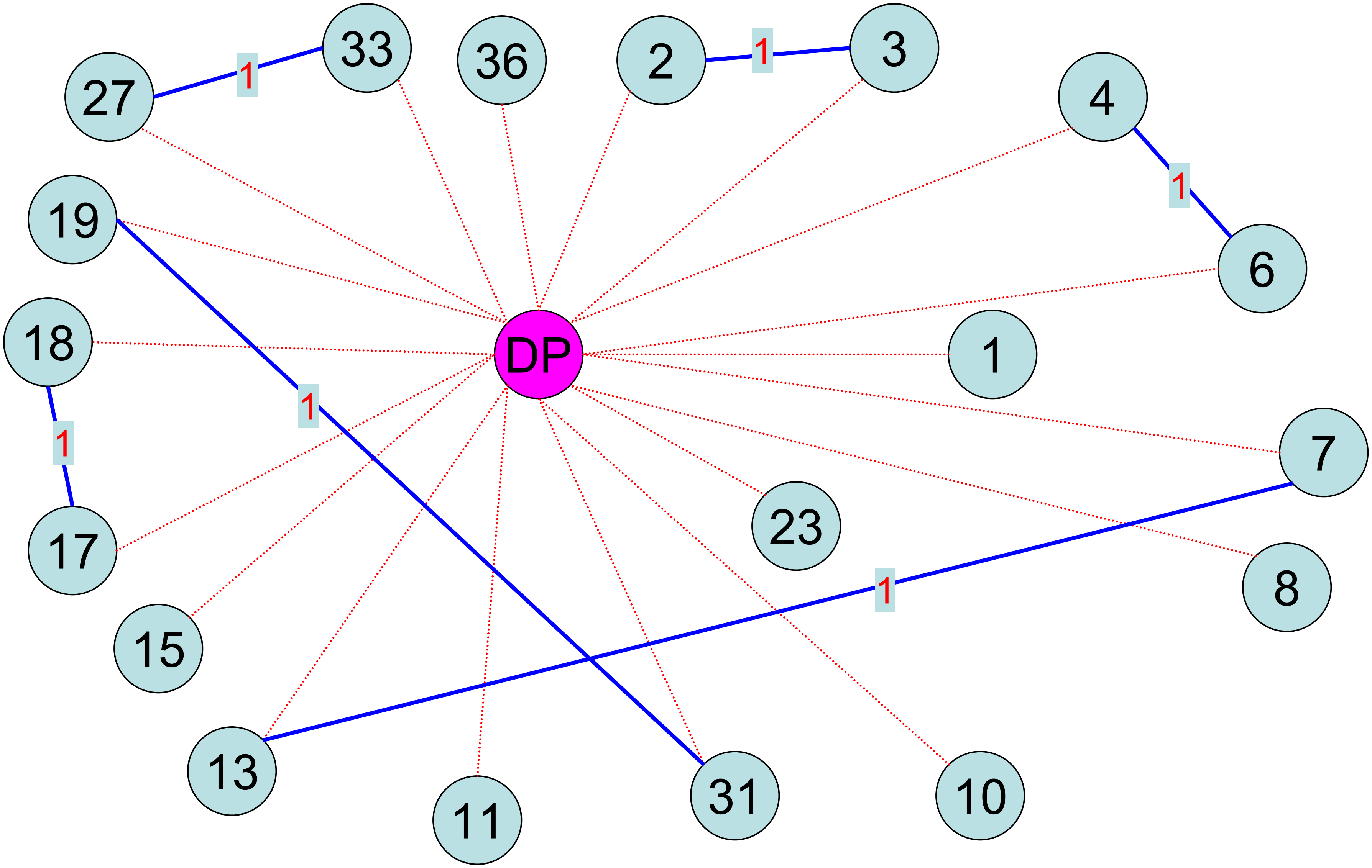
Graph Vertices



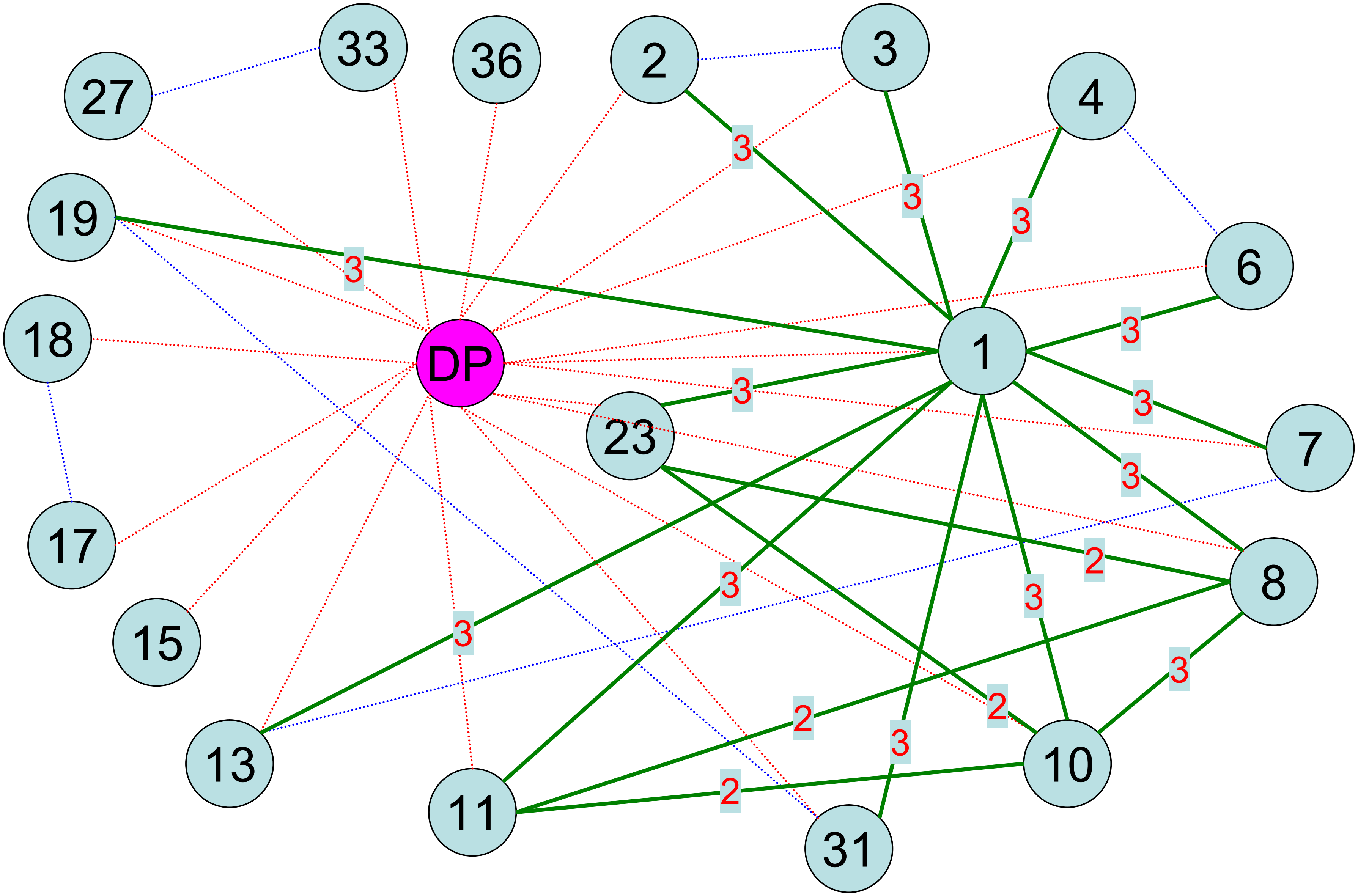
Graph - Dot Product Edges



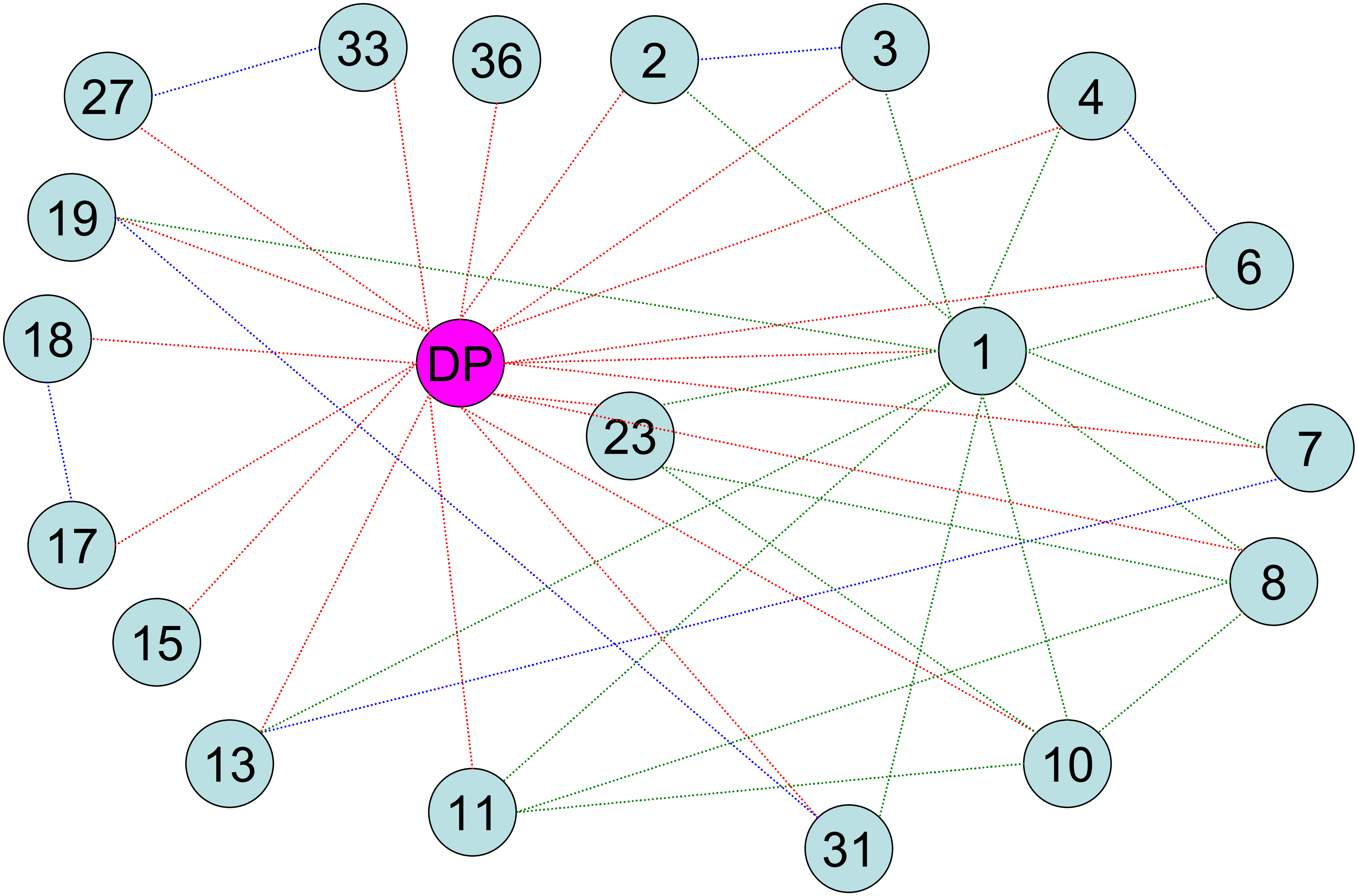
Graph - Scalar Multiple Edges



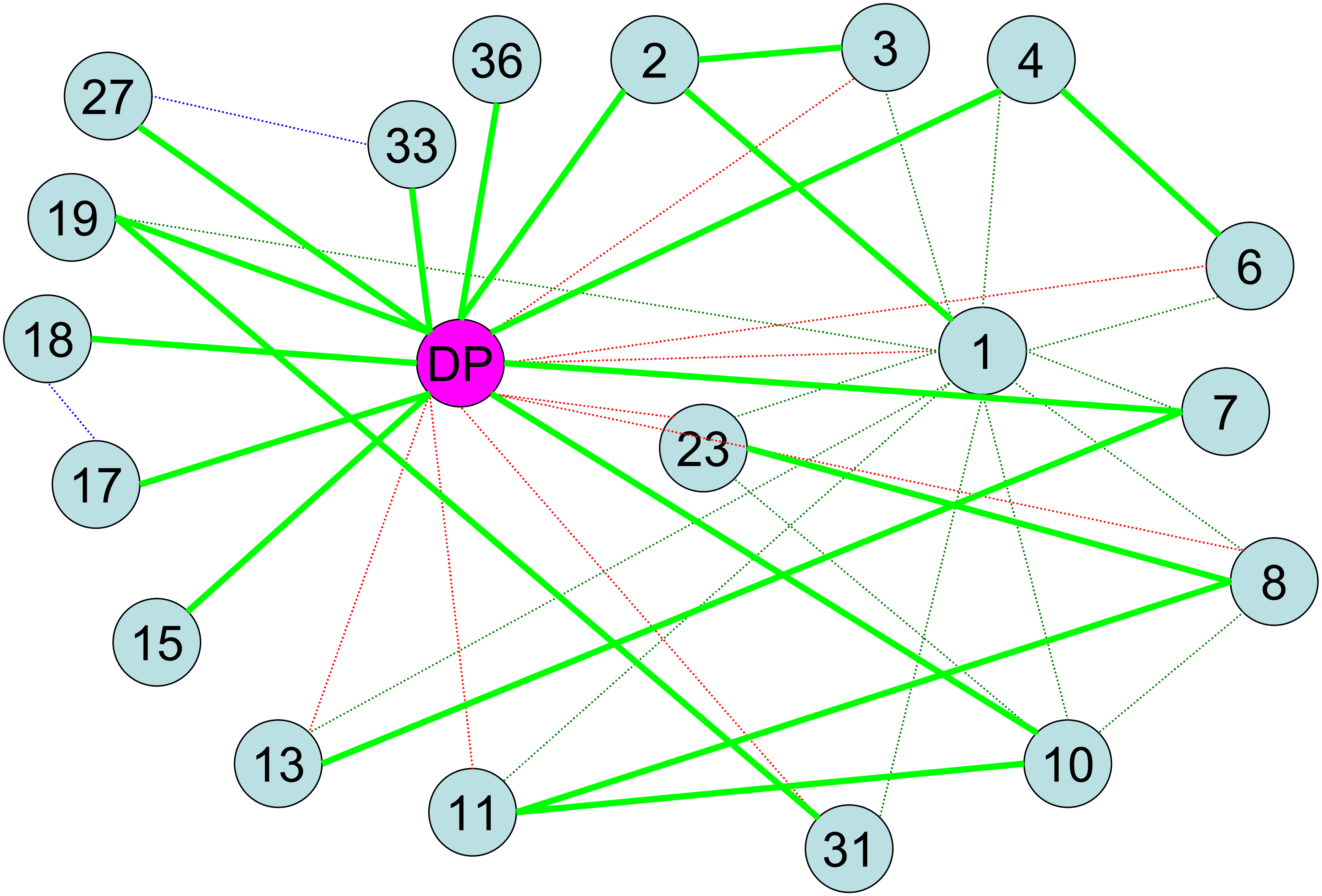
Graph – Partial Scalar Multiple Edges



Graph



Kruskal's Algorithm -> MST



Implementation

$$
S = \begin{bmatrix}
-\frac{3}{4}S(1,2) + 3G(1) + 3G(3) & -4G(2) - 4G(4) & -\frac{1}{4}S(1,2) & -4G(1) - 4G(3) & 0 & -\frac{1}{4}S(1,4) \\
-4G(3) - 4G(4) & -S(2,5) + 8G(4) & S(1,2) & 4G(2) + 4G(3) & -S(2,4) - 8G(1) & 0 \\
-\frac{1}{4}S(2,1) & S(2,1) & 3G(4) & 0 & 4G(3) & -G(3) \\
-4G(1) - 4G(2) & S(2,4) & 0 & S(2,2) & -S(2,2) + 8G(1) & S(1,4) \\
0 & S(2,5) & 4G(2) & S(4,5) & S(2,2) & S(3,5) \\
-\frac{1}{4}S(4,1) & 0 & -G(2) & S(4,1) & S(5,3) & 3G(1)
\end{bmatrix}
$$

144 MAPs → 29 MAPs

Implementation

$$S(1,2) = -4*G(2) - 4*G(4);$$
$$S(1,4) = -4G(1) - 4G(3);$$
$$S(2,1) = -4*G(3);$$
$$S(2,4) = 4*G(2) + 4*G(3);$$
$$S(3,3) = 3*G(4);$$
$$S(3,5) = 3*G(3);$$
$$S(3,6) = -1*G(3);$$
$$S(4,1) = -4*G(1) - 4*G(2);$$
$$S(5,3) = 4*G(2);$$
$$S(6,3) = -1*G(2);$$
$$S(6,6) = 3*G(1);$$
$$S(1,1) = -0.75*S(1,2);$$
$$S(1,3) = -0.25*S(1,2);$$
$$S(1,6) = -0.25*S(1,4);$$
$$S(2,5) = -1*S(2,4) - 8*G(1);$$

$$S(2,2) = -1*S(2,5) + 8*G(4);$$
$$S(3,1) = -0.25*S(2,1);$$
$$S(4,5) = -1*S(2,2) + 8*G(1);$$
$$S(6,1) = -0.25*S(4,1);$$
$$S(2,3) = S(1,2);$$
$$S(3,2) = S(2,1);$$
$$S(4,2) = S(2,4);$$
$$S(4,4) = S(2,2);$$
$$S(4,6) = S(1,4);$$
$$S(5,2) = S(2,5);$$
$$S(5,4) = S(4,5);$$
$$S(5,5) = S(2,2);$$
$$S(5,6) = S(3,5);$$
$$S(6,4) = S(4,1);$$
$$S(6,5) = S(5,3);$$

FErari Results

Order	Entries	Base MAPs	FErari MAPs
1	6	24	7
2	21	84	15
3	55	220	45
4	120	480	176
5	231	924	443
6	406	1624	867

My Results

Order	Entries	Base MAPs	My Code MAPs
1	9	36	15(14)
2	36	144	29(28)
3	100	400	155
4	225	900	443
5	441	1764	814
6	784	3136	1387
7	1296	5184	2211

Conclusions

- Greatly reduced MAPs when building stiffness matrix
- Reduction for lower order FE simple
  - Many zeros in K
  - Graph algorithms unnecessary (p=1, 1 MAP)
- Reduction for higher order FE more interesting
  - Very few zeros in K
  - More complex algorithms necessary
- Graph model limited
  - Each S(i,j) can only exploit knowledge from one previously calculated S(k,l).
  - Can't optimize for a K block being a linear combination of 2 other K blocks

Acknowledgements

Boman, Erik. Unpublished notes and correspondence. SNL. 2005.

Kirby, R. C., Anders Logg, L. Ridgeway Scott, Andy R. Terrel (2005). "Topological optimization of the evaluation of finite element matrices." University of Chicago Technical Reports: 1-22.

Kirby, R. C., Matthew Knepley, Anders Logg, L. Ridgeway Scott (2005). "Optimizing the Evaluation of Finite Element Matrices." SIAM Journal on Scientific Computing 27(no 3): 741-758.

Professor Olson, et al. 2D Helmholtz code.