Some Advanced Topics on Domain Decomposition Methds

Xing Cai

Department of Informatics
University of Oslo

Contents

Literature:

T. F. Chan, T. P. Mathew, *Domain decomposition algorithms*. Acta Numerica (1994), pp. 61–143.

B. F. Smith, P. E. Bjørstad, W. D. Gropp, *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.

J. Xu, *Iterative methods by space decomposition and subspace correction*. SIAM Review 34 (1992) pp. 581–613.

Schwarz framework

Let V be a finite dimensional Hilbert space with a bilinear form $a(\cdot,\cdot)$ and let V_0, V_1, ..., V_p denote the subspaces $V_i \subset V$. Let A be the linear operator such that

$$(Au, v) = a(u, v) \quad \forall v \in V.$$

We search for $u^* \in V$ such that

$$Au^* = f.$$

Assume that A is symmetric and positive definite (SPD), i.e.,

$$\begin{aligned} (Au, v) &= (u, Av) \quad \forall u, v \in V, \\ (Av, v) &> 0 \quad \forall v \in V \backslash \{0\}. \end{aligned}$$

The norm induced by $(A\cdot,\cdot) \equiv (\cdot,\cdot)_A$ is denoted by $\|\cdot\|_A$.

Linear iteration

$$u^{k+1} = u^k + B(f - Au^k),$$

where B is a preconditioner.

By defining $e^k = u^* - u^k$, we have

$$e^{k+1} = (I - BA)e^k.$$

Therefore

$$\left\|e^{k+1}\right\| \leq \|I - BA\| \left\|e^k\right\|.$$

In other words,

$$\|I - BA\| < 1$$

is a sufficient condition for achieving convergence.

Richardson's method

$$u^{k+1} = u^k + \tau B(f - Au^k).$$

Error propagation is as follows:

$$e^{k+1} = (I - \tau BA)e^k.$$

The best convergence result is obtained when

$$\tau = \frac{2}{\lambda_{\text{max}}(BA) + \lambda_{\text{min}}(BA)},$$

and therefore

$$\left\|e^k\right\| \leq \left(\frac{\kappa(BA) - 1}{\kappa(BA) + 1}\right)^k \left\|e^0\right\|,$$

where

$$\kappa(BA) = \frac{\lambda_{\text{max}}(BA)}{\lambda_{\text{min}}(BA)}$$

denotes the condition number of BA.

Preconditioned CG

Determination of τ requires knowledge of the extreme eigenvalues of BA, which is normally not available.

Suppose B is also SPD, we can use preconditioned CG method (no need to choose τ) to achieve faster convergence, where

$$\frac{\sqrt{\kappa(BA)} - 1}{\sqrt{\kappa(BA)} + 1}$$

is the improved convergence rate.

The objective is to have $\kappa(BA)$ as small as possible, preferably independent of the discretization.

Orthogonal projection

Schwarz preconditioners can be expressed as polynomials of orthogonal projection operators P_i: $V \to V_i$, where

$$a(P_i u, v) = a(u, v) \quad \forall v \in V_i,$$

and in its matrix form

$$P_i \equiv R_i^T A_i^{-1} R_i A.$$

Note that

1. $P_i P_i = R_i^T \left(A_i^{-1} R_i A R_i^T\right) A_i^{-1} R_i A = P_i$

2. $AP_i = AR_i^T A_i^{-1} R_i A = P_i^T A$

Two assumptions

Assumption 1 (partition property): Let $C_0 \geq 1$ be the minimal constant such that for any $u \in V$, where $u = \sum_{i=0}^{p} u_i,\ u_i \in V_i$,

$$\sum_i a(u_i, u_i) \leq C_0 a(u, u).$$

Assumption 2 (measure of orthogonality): Define $0 \leq \mathcal{E}_{ij} \leq 1$ as a matrix containing the minimal values that satisfy

$$a(u_i, u_j) \leq \mathcal{E}_{ij} a(u_i, u_i)^{1/2} a(u_j, u_j)^{1/2}, \quad u_i \in V_i,\ u_j \in V_j,\ i, j = 1, \ldots, p.$$

Additive Schwarz

The additive Schwarz preconditioner B_{AS} can be expressed by

$$B_{\mathrm{AS}} A = \sum_{i=0}^{p} R_i^T A_i^{-1} R_i A = \sum_{i=0}^{p} P_i$$

The convergence rate depends on

$$\kappa(B_{\mathrm{AS}} A) \equiv \frac{\lambda_{\max}(B_{\mathrm{AS}} A)}{\lambda_{\min}(B_{\mathrm{AS}} A)}$$

Study of $\kappa(B_{\mathrm{AS}} A) \Rightarrow \kappa(\sum P_i)$.

The bound on $\lambda_{\min}(\sum_{i=0}^{p} P_i)$

$$\begin{aligned} a(u,u) &= \sum_{i=0}^{p} a(u_i, u) = \sum_{i=0}^{p} a(u_i, P_i u) \\ &\leq \left(\sum_{i=0}^{p} a(u_i, u_i) \right)^{1/2} \left(\sum_{i=0}^{p} a(P_i u, P_i u) \right)^{1/2}. \end{aligned}$$

By Assumption 1 we have

$$a(u,u) \leq C_0^{1/2} a(u,u)^{1/2} \left(\sum_{i=0}^{p} a(P_i u, P_i u) \right)^{1/2}.$$

Thus

$$\begin{aligned} a(u,u)^{1/2} &\leq C_0^{1/2} \left(\sum_{i=0}^{p} a(P_i u, P_i u) \right)^{1/2} \\ &= C_0^{1/2} \left(\sum_{i=0}^{p} a(P_i u, u) \right)^{1/2}. \end{aligned}$$

Therefore

$$a(u,u) \leq C_0 a\left(\sum_i P_i u, u \right) \Rightarrow \frac{a\left(\sum_i P_i u, u\right)}{a(u,u)} \geq \frac{1}{C_0}.$$

$$\lambda_{\min}\left(\sum P_i\right) \geq 1/C_0.$$

The bound on $\lambda_{\max}(\sum_{i=0}^{p} P_i)$

$$\begin{aligned} a\left(\sum_{i=1}^{p} P_i u, \sum_{i=1}^{p} P_i u \right) &= \sum_{i,j=1}^{p} a(P_i u, P_j u) \\ \text{By Assumption 2} \quad &\leq \sum_{i,j=1}^{p} \mathcal{E}_{ij} a(P_i u, P_i u)^{1/2} a(P_j u, P_j u)^{1/2} \\ &\leq \rho(\mathcal{E}) \sum_{i=1}^{p} a(P_i u, P_i u) \\ &= \rho(\mathcal{E}) \sum_{i=1}^{p} a(P_i u, u) \\ &= \rho(\mathcal{E}) a\left(\sum_{i=1}^{p} P_i u, u \right) \\ &\leq \rho(\mathcal{E}) a(u,u)^{1/2} a\left(\sum_{i=1}^{p} P_i u, \sum_{i=1}^{p} P_i u \right)^{1/2} \end{aligned}$$

Note

$$\rho(\mathcal{E}) \equiv \sup_{x \neq 0} \frac{x^T \mathcal{E} x}{x^T x} \Rightarrow \sum_{ij} \mathcal{E}_{ij} x_i x_j \leq \rho(\mathcal{E}) \sum_i x_i^2.$$

Therefore

$$a\left(\sum_{i=1}^{p} P_i u, \sum_{i=1}^{p} P_i u\right) = a\left(\left[\sum_{i=1}^{p} P_i\right]^2 u, u\right) \le \rho^2(\mathcal{E}) a(u,u)$$

$$\Rightarrow \quad a\left(\sum_{i=1}^{p} P_i u, u\right) \le \rho(\mathcal{E}) a(u,u).$$

The largest eigenvalue of P_0 is 1 because

$$\begin{aligned} a(P_0 u, P_0 u) &= a(P_0 u, u) \\ &\le a(u,u)^{1/2} a(P_0 u, P_0 u)^{1/2}. \end{aligned}$$

In other words,

$$a(P_0 u, P_0 u) = a(P_0 u, u) \le a(u,u).$$

So

$$a\left(\sum_{i=0}^{p} P_i u, u\right) \le (1+\rho(\mathcal{E})) a(u,u)$$

$$\Rightarrow \quad \lambda_{\max}\left(\sum_{i=0}^{p} P_i\right) \le 1 + \rho(\mathcal{E}).$$

The condition number of $B_{AS}A = \sum_{i=0}^{p} P_i$

$$\kappa(B_{AS}A) \le C_0(1+\rho(\mathcal{E})).$$

Remarks:

1. $\rho(\mathcal{E})$ equals the minimum number of distinct colours N_c, such that those subspaces of $V_1, \dots, V_p$ that have a same colour are mutually orthogonal. (Bounded overlap means N_c independent of p.)
2. C_0 is independent of h and also independent of H if the overlap is uniformly of width $\mathcal{O}(H)$. (sufficient overlap)
3. Given $u_h \in V$ the partition $u_h = u_0 + u_1 + \dots + u_p$ can be done by

$$\begin{aligned} u_0 &= Q_0 u_h, \quad Q_0 \text{ is the } L^2 \text{ projection} \\ u_i &= I_h(\chi_i(u_h - u_0)), \quad \text{for } i = 1, \dots, p. \end{aligned}$$

where I_h is the finite element interpolation into V, and $\chi_1, \dots, \chi_p$ is a partition of unity subordinate to the overlapping subdomains $\widehat{\Omega}_1, \dots, \widehat{\Omega}_p$, satisfying:

$$0 \le \chi_i \le 1, \ \chi_i \in C_0^\infty\left(\widehat{\Omega}_i\right), \ \sum_{i=1}^{p} \chi_i = 1, \ |\nabla \chi_i| \le C\beta^{-1}H^{-1}.$$

Multiplicative Schwarz

$$u^{k+i/p} = u^{k+(i-1)/p} + R_i^T A_i^{-1} R_i \left(f - A u^{k+(i-1)/p}\right)$$

for $i = 1, \dots, p$.

Error propagation:

$$\Rightarrow \quad e^{k+1} = E_p e^k,$$

where $E_p \equiv (I - P_p)(I - P_{p-1}) \dots (I - P_1)$.

Want to show that there exists $0 < \delta < 1$ such that

$$\|E_p\|_A \le \delta \quad \text{with } \|E_p\|_A \equiv \sup_{v \in V} \frac{\|E_p v\|_A}{\|v\|_A}.$$

We note that

$$E_i = (I - P_i)E_{i-1}.$$

Therefore

$$\begin{aligned} a(E_i v, E_i v) &= a((I-P_i)E_{i-1}v, (I-P_i)E_{i-1}v) \\ &= a((I-P_i)E_{i-1}v, E_{i-1}v) \\ &= a(E_{i-1}v, E_{i-1}v) - a(P_i E_{i-1}v, E_{i-1}v) \end{aligned}$$

$$\Rightarrow \quad \|E_{i-1}v\|_A^2 - \|E_i v\|_A^2 = (P_i E_{i-1} v, E_{i-1} v)_A.$$

By noticing $E_0 \equiv I$ and summing up for $i = 1, \dots, p$, we get

$$\|v\|_A^2 - \|E_p v\|_A^2 = \sum_{i=1}^{p} (P_i E_{i-1} v, E_{i-1} v)_A.$$

We observe that

$$I - E_i = \sum_{j=1}^{i} P_j E_{j-1}.$$

So

$$\begin{aligned}(P_i v, v)_A &= (P_i v, E_{i-1} v)_A + (P_i v, (I - E_{i-1}) v)_A \\ &= (P_i v, E_{i-1} v)_A + \sum_{j=1}^{i-1} (P_i v, P_j E_{j-1} v)_A\end{aligned}$$

By Cauchy-Schwarz we have

$$\sum_{i=1}^{p} (P_i v, E_{i-1} v)_A \le \left(\sum_{i=1}^{p} (P_i v, v)_A \right)^{\frac{1}{2}} \left(\sum_{i=1}^{p} (P_i E_{i-1} v, E_{i-1} v)_A \right)^{\frac{1}{2}}.$$

By Assumption 2 we have

$$\sum_{i=1}^{p} \sum_{j=1}^{i-1} (P_i v, P_j E_{j-1} v)_A \le \rho(\mathcal{E}) \left(\sum_{i=1}^{p} (P_i v, v)_A \right)^{\frac{1}{2}} \left(\sum_{i=1}^{p} (P_i E_{i-1} v, E_{i-1} v)_A \right)^{\frac{1}{2}}.$$

Putting things together, we arrive at

$$\sum_{i=1}^{p} (P_i v, v)_A \le (1 + \rho(\mathcal{E}))^2 \sum_{i=1}^{p} (P_i E_{i-1} v, E_{i-1} v)_A.$$

Recall $\sum (P_i v, v)_A \ge \frac{1}{C_0} \|v\|_A^2$, we finally get

$$\|v\|_A^2 - \|E_p v\|_A^2 \ge \frac{1}{C_0 (1 + \rho(\mathcal{E}))^2} \|v\|_A^2 .$$

Therefore

$$\left(1 - \frac{1}{C_0 (1 + \rho(\mathcal{E}))^2} \right) \|v\|_A^2 \ge \|E_p v\|_A^2 .$$

$$\Rightarrow \quad \|E_p\|_A^2 = 1 - \frac{1}{C_0 (1 + \rho(\mathcal{E}))^2}.$$

Inexact subdomain solvers

Exact solution of a subproblem (A_i^{-1}) can still be expensive.

It is fully possible to use an inexact solver for the subproblems. That is, we replace A_i^{-1} with another $\tilde{A}_i^{-1}$, which is an approximate inverse.

For example

$$\tilde{B}_{\mathrm{AS}} A \equiv \sum_{i=0}^{p} R_i^T \tilde{A}_i^{-1} R_i A.$$

For multiplicative Schwarz iteration, we require

$$\rho(\tilde{A}_i^{-1} A_i) < 2.$$

Solving the coarse problem in parallel

Several alternatives:

1. Distributed solution of the coarse grid problem

2. Gather all the data in one processor, solve there and broadcast the result

3. Gather all the data to all processors, each processor solves it by itself

Data structure for parallel Schwarz methods

1. Construct only sub-matrices & sub-vectors for $\widehat{\Omega}_i$.

2. Avoid physical storage of global matrices & vectors. (The global linear algebra operations can be realized by invoking subdomain operations together with data exchange between neighboring subdomains.)

3. For each subdomain $\widehat{\Omega}_i$, $i = 1, \dots, p$, we denote by $\mathcal{N}_o^i$ the set of grid points shared by at least one neighboring subdomain. A subset of $\mathcal{N}_o^i$ is $\mathcal{N}_\Gamma^i$ that contains all the grid points on the internal (artificial) boundary.

4. For every grid point j, define
K_j (overlap count) – total number of $\widehat{\Omega}_i$, $1 \le i \le p$, where $j \in \mathcal{N}_o^i$,
$\tilde{K}_j$ (real overlap count) – total number of $\widehat{\Omega}_i$, $1 \le i \le p$, where $j \in \mathcal{N}_o^i \backslash \mathcal{N}_\Gamma^i$. (Note that $\tilde{K}_j \le K_j$.)

Parallel linear algebra operations

1. Parallel vector addition $w = u + v$.
On each subdomain $w_i = u_i + v_i$, no need for data exchange.

2. Parallel computation of $c = u \cdot v$.
On each subdomain $c_i = u_i \cdot v_i$. Then

$$\tilde{c}_i = c_i - \sum_{j \in \mathcal{N}_o^i} \frac{K_j - 1}{K_j} u_{i,j} v_{i,j}.$$

Finally $c = \sum_{i=1}^{p} \tilde{c}_i$. (collective communication)

3. Parallel computation of $v = Au$.
On each subdomain $v_i = A_i u_i$. Then for each $j \in \mathcal{N}_\Gamma^i$, receive the correct value on j from a neighboring subdomain k where $j \in \mathcal{N}_o^k \backslash \mathcal{N}_\Gamma^k$.

4. In each iteration of additive Schwarz, after the subdomain solve, we need to update the value on all nodes $j \in \mathcal{N}_o^i$.
First collect values from neighboring subdomains $k \ne i$ where $j \in \mathcal{N}_o^k \backslash \mathcal{N}_\Gamma^k$. Then take an average of those $\tilde{K}_j$ values.

Data structure for message passing

1. For each subdomain $\widehat{\Omega}_i$ first find all its neighboring subdomain(s) (there are shared grid points in between).
2. Find out all the grid points belonging to $\mathcal{N}_o^i$, and their associated K_j and $\tilde{K}_j$ values.
3. For each neighboring subdomain $\widehat{\Omega}_k$ allocate two arrays for updating values on Γ after e.g. parallel matrix-vector product. One array is to contain the values of points $j \in \mathcal{N}_o^i \backslash \mathcal{N}_\Gamma^i$ and $j \in \mathcal{N}_\Gamma^k$. The other array is for receiving incoming values from $\widehat{\Omega}_k$.
4. For each neighboring subdomain $\widehat{\Omega}_k$ allocate two arrays for updating values on all the overlapping grid points. One array is to contain the values of points $j \in \mathcal{N}_o^i \backslash \mathcal{N}_\Gamma^i$ and $j \in \mathcal{N}_o^k$. The other array is for receiving incoming values from $\widehat{\Omega}_k$.

Message passing in parallel Schwarz methods

Two types of data exchange:

1. Long arrays exchanged between two and two neighboring subdomains;
2. Short arrays exchanged among all subdomains by collective message passing routines.

Explicit synchronization is rarely necessary. Implicit synchronization is enforced by e.g. parallel computation of inner-product.

Schur complement methods

Iterative substructuring.

Interface solve.

For the two-subdomain case, let $B = \partial\Omega_1 \cap \partial\Omega_2$ and organize the unknowns as $u = (u_1, u_2, u_B)$.

$$\begin{pmatrix} A_{II}^{(1)} & 0 & A_{IB}^{(1)} \\ 0 & A_{II}^{(2)} & A_{IB}^{(2)} \\ A_{BI}^{(1)} & A_{BI}^{(2)} & A_{BB}^{(1)} + A_{BB}^{(2)} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_B \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_B \end{pmatrix}.$$

A reduced system

$$u_1 = A_{II}^{(1)^{-1}} \left(f_1 - A_{IB}^{(1)} u_B \right).$$

$$u_2 = A_{II}^{(2)^{-1}} \left(f_2 - A_{IB}^{(2)} u_B \right).$$

$$\Rightarrow \quad S u_B = \tilde{f}_B.$$

The Schur complement $S = S^{(1)} + S^{(2)}$ and

$$\begin{aligned} S^{(1)} &= A_{BB}^{(1)} - A_{BI}^{(1)} A_{II}^{(1)^{-1}} A_{IB}^{(1)}, \\ S^{(2)} &= A_{BB}^{(2)} - A_{BI}^{(2)} A_{II}^{(2)^{-1}} A_{IB}^{(2)}, \\ \tilde{f}_B &= f_B - A_{BI}^{(1)} A_{II}^{(1)^{-1}} f_1 - A_{BI}^{(2)} A_{II}^{(2)^{-1}} f_2. \end{aligned}$$

S is dense but has better condition number than that of A. ($\mathcal{O}(1/h) < \mathcal{O}(1/h^2)$)

Often S is not explicitly formed (too expensive). The action of $S^{(i)} = A_{BB}^{(i)} - A_{BI}^{(i)} A_{II}^{(i)^{-1}} A_{IB}^{(i)}$ on a vector can be realized by three sparse matrix-vector products and one matrix solve.

It is common practice to solve $S u_B = \tilde{f}_B$ iteratively by preconditioned Krylov subspace methods. Note that each matrix-vector product involves two subdomain solves ($A_{II}^{(1)^{-1}}$ and $A_{II}^{(2)^{-1}}$), which can be performed in parallel.

The main topic of Schur complement methods is the design of the preconditioner for S. No completely satisfactory choice.

Neumann-Dirichlet

Preconditioner is of form $S^{(1)^{-1}}$. That is, we solve

$$\left(S^{(1)} + S^{(2)} \right) u_B = \left(S^{(1)} + S^{(2)} \right) S^{(1)^{-1}} w_B = \tilde{f}_B.$$

The action of $S^{(1)^{-1}}$ can be calculated without explicitly forming $S^{(1)}$ by the following observation:

$$A^{(1)} = \begin{pmatrix} I & 0 \\ A_{BI}^{(1)} A_{II}^{(1)^{-1}} & I \end{pmatrix} \begin{pmatrix} A_{II}^{(1)} & 0 \\ 0 & S^{(1)} \end{pmatrix} \begin{pmatrix} I & A_{II}^{(1)^{-1}} A_{IB}^{(1)} \\ 0 & I \end{pmatrix}.$$

$$A^{(1)^{-1}} = \begin{pmatrix} I & -A_{II}^{(1)^{-1}} A_{IB}^{(1)} \\ 0 & I \end{pmatrix} \begin{pmatrix} A_{II}^{(1)^{-1}} & 0 \\ 0 & S^{(1)^{-1}} \end{pmatrix} \begin{pmatrix} I & 0 \\ -A_{BI}^{(1)} A_{II}^{(1)^{-1}} & I \end{pmatrix}.$$

$$\Rightarrow \quad S^{(1)^{-1}} v = (0 \;\; I) A^{(1)^{-1}} \begin{pmatrix} 0 \\ I \end{pmatrix} v.$$

The action of $S^{(1)^{-1}}$ is achieved by solving a discrete problem on Ω_1 with a Neumann boundary condition on the interface. The action of $S^{(2)}$ is achieved by solving a discrete problem on Ω_2 with a Dirichlet boundary condition. No inherent parallelism.

Neumann-Neumann

The (symmetric) preconditioner is of form

$$S^{(1)^{-1}} + S^{(2)^{-1}}.$$

The J operator

Explicit eigen-decomposition of S.

Probing

A purely algebraic technique. Construct a tridiagonal approximation to S by probing the action of S on a few carefully selected vectors.

Many non-overlapping subdomains

Preconditioners for Schur complement S:

2D: Block Jacobi, BPS, Vertex space preconditioner, Neumann-Neumann.

3D: Vertex space preconditioner, wirebasket, Neumann-Neumann.

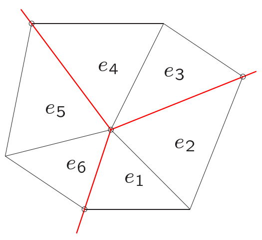
Need a “coarse grid” to enhance convergence rate greatly.

To overlap or not to overlap?

No definitive answer.

- Overlapping DD methods are easier to form and implement
- Overlapping DD methods more robust and easier to achieve an optimal convergence
- Overlapping DD methods can not avoid overhead work in overlapping regions
- Overlapping DD methods not very suitable for discontinuous coefficients

Partition of unstructured FE mesh



Reformulation as a graph partitioning problem.

Create $G = (\mathcal{V}, \mathcal{E})$, where

- one vertex corresponds to one mesh element, and
- a weighted edge between 2 vertices represents the number of shared mesh nodes between 2 neighboring mesh elements.

Graph partitioning

Given G, we want to partition the set of vertices $\mathcal{V}$ into

1. p equal-sized subsets $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_p$ (to achieve good load-balance), and that

2. *edge-cut* is minimized (to keep down the inter-processor communication volume).

The graph partitioning problem is NP-complete. But there are several algorithms that are able to find reasonably good partition:

Spectral partitioning algorithms;

Geometric partitioning algorithms;

Multilevel algorithms.

Multilevel graph partitioning algorithms

Efficient and flexible algorithms which consist of three phases:

1. Coarsening phase: a recursive process that generates a sequence of subsequently coarser graphs $G^0, G^1, \ldots G^m$. The stopping criterion is that the size of G^m is sufficiently small;

2. Initial partition phase: partitions the coarsest graph G^m into p subsets;

3. Refinement phase: a recursive process that projects the partition of G^m backward to G^0, while refining the partition along the way.

Coarsen a graph

Go through all the vertices in a random order.

For each vertex, we use a heuristic to match it with one of its so far unchosen neighboring vertices.

The two vertices then form a new super-vertex, and the edge connecting them is collapsed. Other edges that coming out from them are merged if necessary.

Some vertices may not find a neighbor to match.

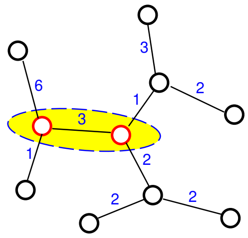
A new graph that has fewer vertices and edges arises from the old graph. So a good matching heuristic should rapidly reduce both the number of vertices and number of edges.

Matching heuristics used in coarsening

For every $v \in \mathcal{V}$,

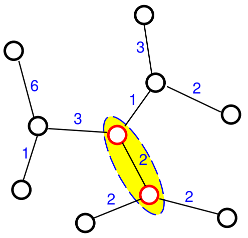
1. Random matching (RDM) - a random unchosen neighbor u is chosen.

2. Heady edge matching (HEM) - an unchosen neighboring vertex u is chosen, if the edge (v, u) has the heaviest weight among all the unchosen neighbors.



Matching heuristics used in coarsening

3. Gain vertex matching (GVM) - an unchosen neighboring vertex u is chosen such that the added weight of the edges that come out from the new super vertex is minimized.

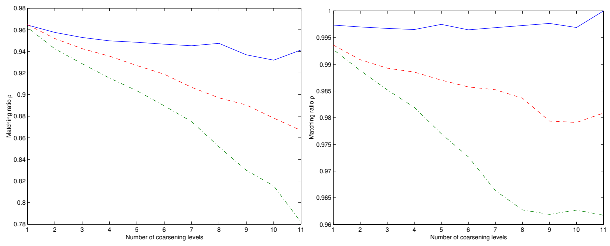


GVM tends to reduce the number of vertices faster.

Comparison between matching heuristics

Matching ratio

$$\varrho = 0.5 \cdot \frac{\#\text{ nodes in } G^k}{\#\text{ nodes in } G^{k+1}}$$



GVM vs. HEM vs. RDM

Initial partition

The coarsest graph G^m is very small.

We can therefore afford using a complex partitioning algorithm, e.g. recursive bisection with Kernighan & Lin heuristic for refinement.

See e.g.

G. Karypis, V. Kumar, *Multilevel k-way partitioning scheme for irregular graphs*. J. Parallel Distrb. Comput. 48 (1998), pp. 96–129.

B. W. Kernighan, S. Lin, *An efficient heuristic procedure for partitioning graphs*. The Bell System Technical Journal 49 (1970), pp. 291–307.

Refinement of the partition

The partition of G^{k+1} is projected backward to G^k. Every super vertex in G^{k+1} is split back into two vertices in G^k and the collapsed edges are recovered.

We also refine the partition of G^k to reduce edge-cut, while maintaining the load balance quality. That is, vertices are moved between subsets. There are many different refinement algorithms. One particular simple algorithm is as follows.

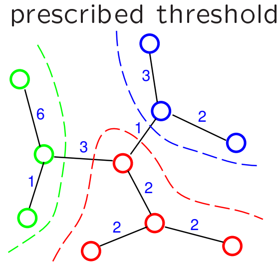
A simple refinement algorithm

Run a few refinement iterations, where in each iteration go through all the vertices randomly.

If a vertex is lying on the boundary, i.e., at least one of its neighboring vertices belongs to another subset, we will move the vertex into one of its neighboring subsets if

1. a largest possible non-negative decrease of edge-cut is achieved, and

2. load-imbalance is improved or remains within a prescribed threshold.



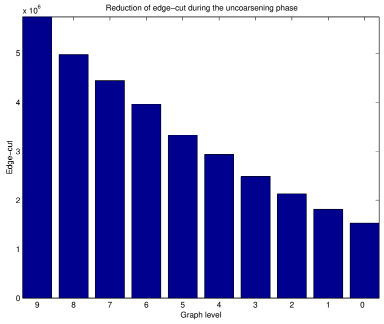
Effect of the refinement phase

Example: 48-way partition of a graph consisting of 320,048 vertices and 11,540,788 edges.

9 coarsening steps reduce the number of vertices to 646 and the number of edges to 22,094.

Initial partition of G^9 has edge-cut 5,741,502.

Refinements on all the graph levels during the uncoarsening process reduces EC to 1,529,995. Average number of neighbors drops from 26.33 to 10.13.



Partition of graph ⇒ partition of FE mesh

When a partition of a graph is ready, in form of vertex subsets $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_p$, we can partition the global FE mesh $\mathcal{M}$ into p sub-meshes $\mathcal{M}_i$.

The quality of load balance is indicated by:

$$\gamma_1 = \frac{\max_i |\mathcal{V}_i|}{|\mathcal{V}|/p},$$
$$\gamma_2 = \frac{\max_i |\mathcal{V}_i| - \min_i |\mathcal{V}_i|}{|\mathcal{V}|/p}.$$

Partition of graph ⇒ partition of FE mesh

Communication overhead can be indicated by the following two quantities:

1. Average number of neighbors: $D = \sum_i D_i/p$, where D_i is the number of neighboring sub-meshes for $\mathcal{M}_i$. (Number of messages $\mathcal{M}_i$ needs to exchange.)

2. Total volume of communication:

$$N = \sum_i \sum_{j \neq i} N_{i,j},$$

where $N_{i,j}$ is the number of mesh nodes shared between $\mathcal{M}_i$ and $\mathcal{M}_j$.