The Multilevel Fast Multipole Method

Ramani Duraiswami

Nail Gumerov

Review

- FMM aims at accelerating the matrix vector product
- Matrix entries determined by a set of source points and evaluation points (possibly the same)
- Function Φ has following point-centered representations about a given point x_*
  - ❑ Local (valid in a neighborhood of a given point)
  - ❑ Far-field or multipole (valid outside a neighborhood of a given point)
  - ❑ In many applications Φ is singular
- Representations are usually series
  - ❑ Could be integral transform representations
- Representations are usually approximate
  - ❑ Error bound guarantees the error is below a specified tolerance

$$\mathbf{v} = \mathbf{\Phi}\mathbf{u},$$

$$\mathbf{\Phi} = \begin{pmatrix} \Phi(\mathbf{y}_1,\mathbf{x}_1) & \Phi(\mathbf{y}_1,\mathbf{x}_2) & \dots & \Phi(\mathbf{y}_1,\mathbf{x}_N) \\ \Phi(\mathbf{y}_2,\mathbf{x}_1) & \Phi(\mathbf{y}_2,\mathbf{x}_2) & \dots & \Phi(\mathbf{y}_2,\mathbf{x}_N) \\ \dots & \dots & \dots & \dots \\ \Phi(\mathbf{y}_M,\mathbf{x}_1) & \Phi(\mathbf{y}_M,\mathbf{x}_2) & \dots & \Phi(\mathbf{y}_M,\mathbf{x}_N) \end{pmatrix}.$$

$$\mathsf{X} = \{\mathbf{x}_1,\mathbf{x}_2,\dots,\mathbf{x}_N\}, \quad \mathbf{x}_i \in \mathsf{R}^d, \quad i = 1,\dots,N,$$
$$\mathsf{Y} = \{\mathbf{y}_1,\mathbf{y}_2,\dots,\mathbf{y}_M\}, \quad \mathbf{y}_j \in \mathsf{R}^d, \quad j = 1,\dots,M.$$

$$v_j = \sum_{i=1}^{N} u_i \Phi(\mathbf{y}_j,\mathbf{x}_i), \quad j = 1,\dots,M.$$

Review

$$\Phi(\mathbf{y}_j,\mathbf{x}_i) = \sum_{m=0}^{p-1} A_m(\mathbf{x}_i) F_m(\mathbf{y}_j) + Error(p,\mathbf{x}_i,\mathbf{y}_j).$$

$$v_j = \sum_{i=1}^{N} u_i \Phi(\mathbf{y}_j,\mathbf{x}_i) = \sum_{i=1}^{N} u_i \sum_{m=0}^{p-1} A_m(\mathbf{x}_i) F_m(\mathbf{y}_j) + \sum_{i=1}^{N} u_i Error(p,\mathbf{x}_i,\mathbf{y}_j)$$
$$= \sum_{m=0}^{p-1} B_m F_m(\mathbf{y}_j) + Error_j(p,N), \quad j = 1,\ldots,M.$$

- One representation, valid in a given domain, can be converted to another valid in a subdomain contained in the original domain
- Factorization trick is at core of the FMM speed up
- Representations we use are factored … separate points x_i and y_j
- Data is partitioned to organize the source points and evaluation points so that for each point we can separate the points over which we can use the factorization trick, and those we cannot.
- Hierarchical partitioning allows use of different factorizations for different groups of points
- Accomplished via MLFMM

Prepare Data Structures

- Convert data set into integers given some maximum number of bits allowed/dimensionality of space
- Interleave
- Sort
- Go through the list and check at what bit position two strings differ
  - For a given *s* determine the number of levels of subdivision needed
  - *s* is the maximum number of points in a box at the finest level

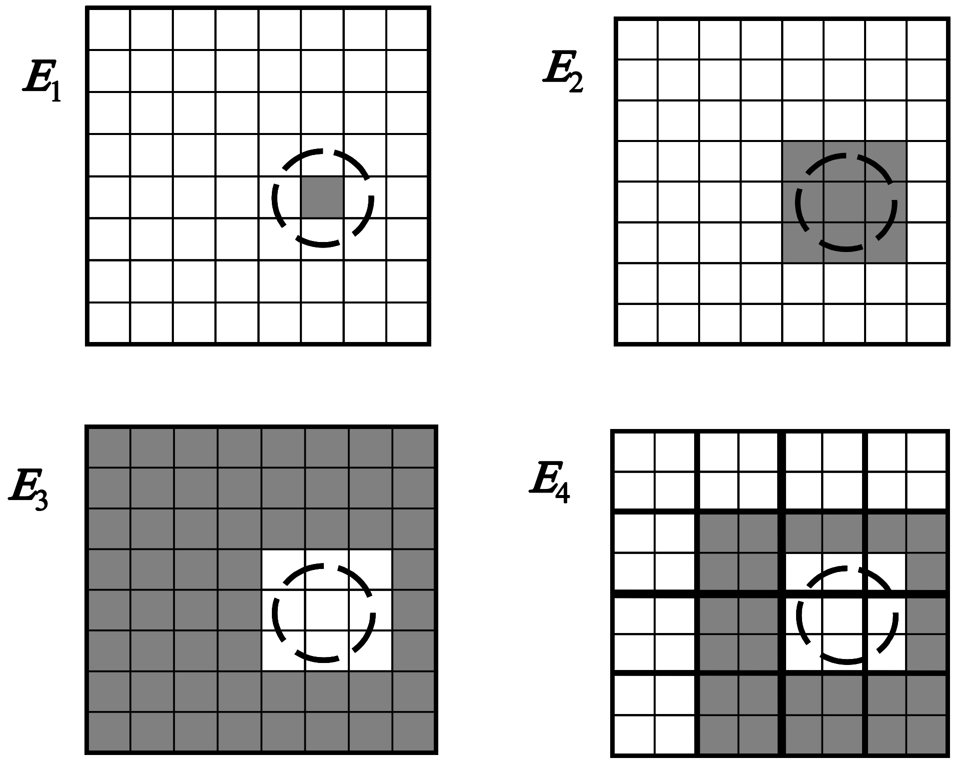
Hierarchical Spatial Domains

E_1: box

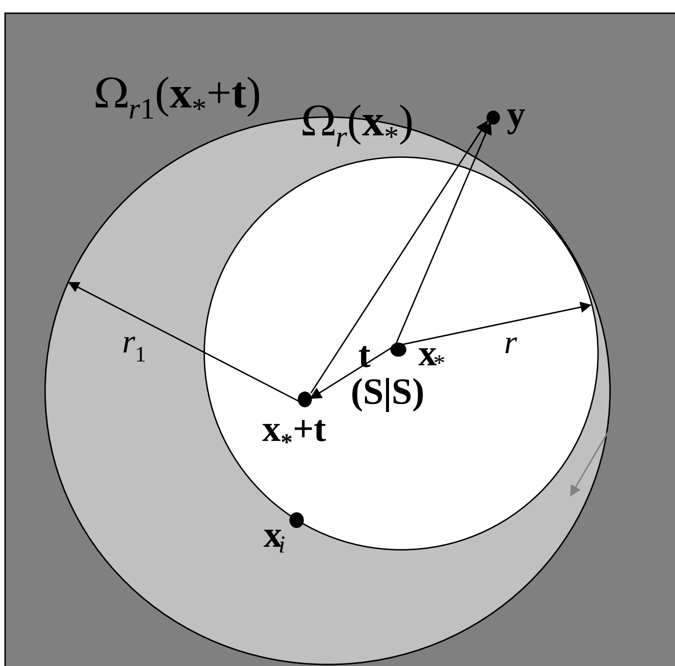
E_2: points in the box and in neighboring boxes

E_3: points in boxes outside neighborhood

E_4: points belonging to neighbors of parent box, but which do not belong to E_2



S|S-reexpansion (Far to Far, or Multipole to Multipole, or M2M)



Original expansion
Is valid only here!

$$|\mathbf{y} - \mathbf{x}_* - \mathbf{t}| > r_1 = r + |\mathbf{t}|$$

Since
$\Omega_{r1}(\mathbf{x}_*+\mathbf{t}) \subset \Omega_r(\mathbf{t})$!

Also

$$|\mathbf{x}_i - \mathbf{x}_*| < r$$

singular point !

UPWARD PASS

- Partition sources into a source hierarchy.
- Stop hierarchy so that boxes at the finest level contain at most *s* sources
- Let the number of levels be *L*
- Consider the finest level
- For non-empty boxes we create *S* expansion about center of the box $\Phi(x_i,y)=\sum^P u_i B(x_*,x_i)\, S(x_*,y)$

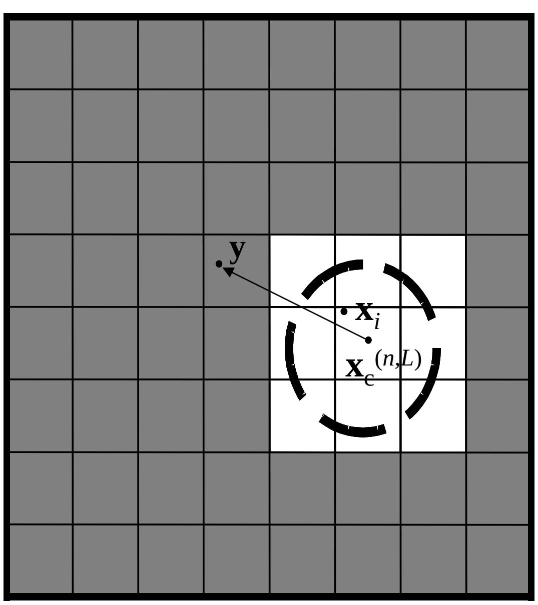
$$\Phi_1^{(n,L)}(\mathbf{y}) = \mathbf{C}^{(n,L)} \circ \mathbf{S}(\mathbf{y} - \mathbf{x}_c^{(n,L)}),$$

$$\mathbf{C}^{(n,L)} = \sum_{\mathbf{x}_i \in E_1(n,L)} u_i \mathbf{B}\left(\mathbf{x}_i, \mathbf{x}_c^{(n,L)}\right).$$

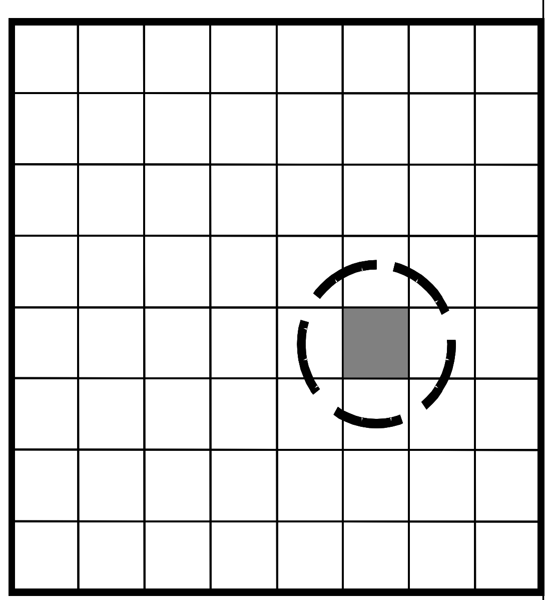
- We need to keep these coefficients. $\boldsymbol{C}^{(n,l)}$ for each level as we will need it in the downward pass
- Then use S/S translations to go up level by level up to level 2.
- Cannot go to level 1 (Why?)

- *S* expansion is valid in the domain *E_3* outside domain *E_1* (provided *d<9*)

E_3



E_1



UPWARD PASS

- At the end of the upward pass we have a set of *S* expansions (i.e. we have coefficients for them)
- we have a set of coefficients $\boldsymbol{C}^{(n,l)}$ for $n=1,\ldots,2^{ld}$ $\quad l=L,\ldots,2$
- Each of these expansions is about a center, and is valid in some domain
- We would like to use the coarsest expansions in the downward pass (have to deal with fewest numbers of coefficients)
- But may not be able to --- because of domain of validity
- Upward pass works on source points and builds representations to be used in the downward pass, where the actual product will be evaluated

DOWNWARD PASS

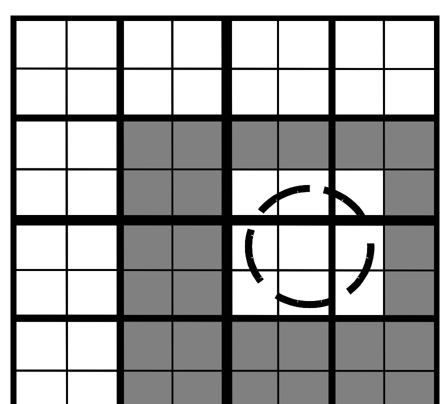
- Starting from level 2, build an *R* expansion in boxes where *R* expansion is valid

$$\Phi_4^{(n,l)}(\mathbf{y}) = \widetilde{\mathbf{D}}^{(n,l)} \circ \mathbf{R}(\mathbf{y}-\mathbf{x}_c^{(n,l)}),$$

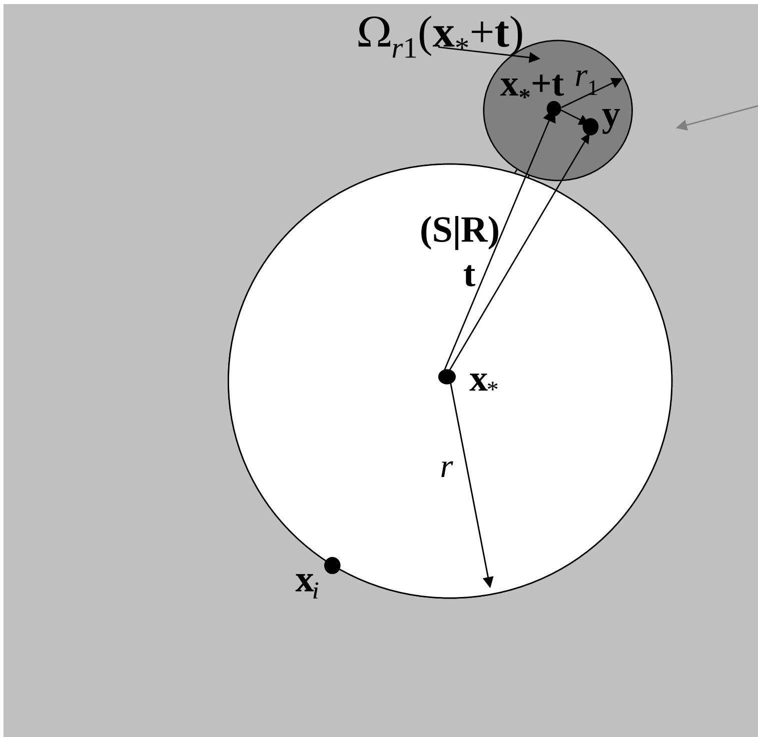
$$\widetilde{\mathbf{D}}^{(n,l)} = \sum_{m\in I_4(n,l)} (\mathbf{S}|\mathbf{R})\left(\mathbf{x}_c^{(n,l)}-\mathbf{x}_c^{(m,l)}\right)\mathbf{C}^{(m,l)}.$$

- Must to do $S|R$ translation
- The *S* expansion is not valid in boxes immediately surrounding the current box
- So we must exclude boxes in the E_4 neighborhood

$\boldsymbol{E}_4$



S|R-reexpansion (Far to Local, or Multipole to Local, or M2L)



Original expansion
Is valid only here!

$|\mathbf{y} - \mathbf{x}_* - \mathbf{t}| < r_1 = |\mathbf{t}| - r$

Since
$\Omega_{r1}(\mathbf{x}_*+\mathbf{t}) \subset \Omega_r(\mathbf{t})$!

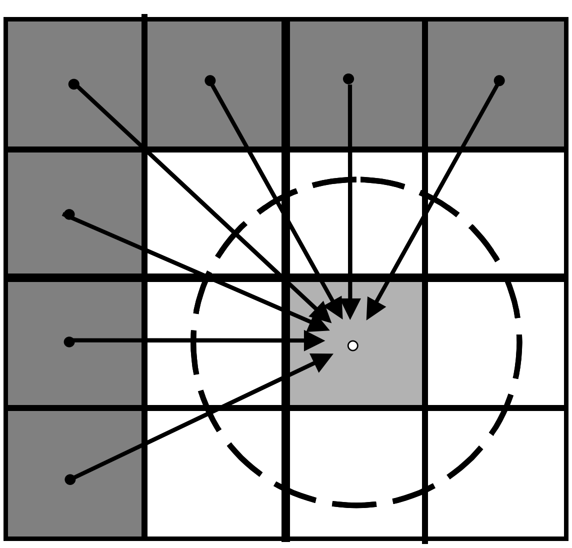
Also

$|\mathbf{x}_i - \mathbf{x}_*| < r$

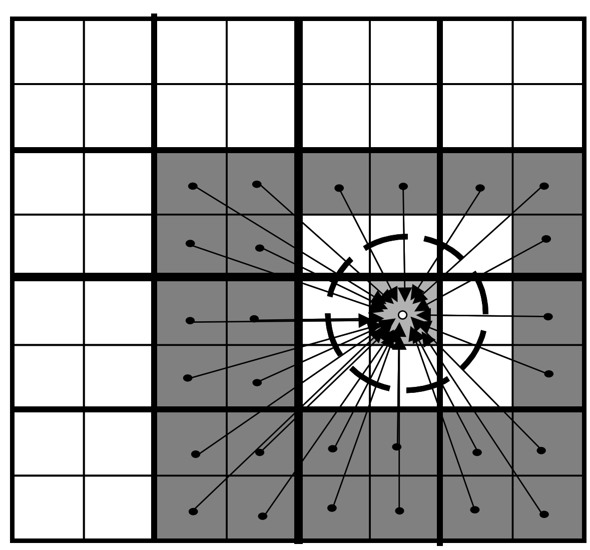
singular point !

Downward Pass. Step 1.

Level 2:



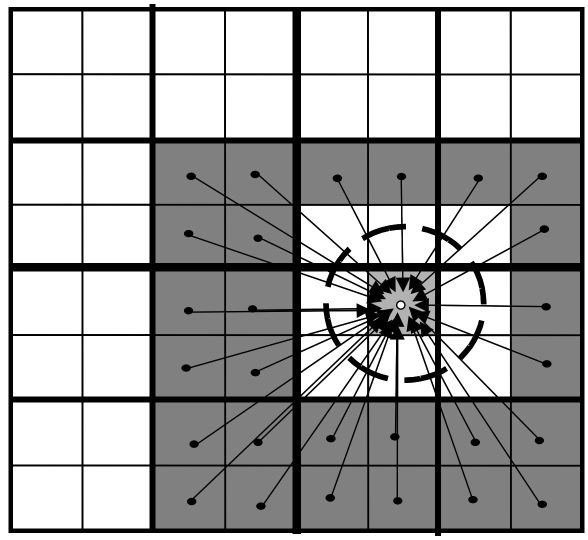
Level 3:



CSCAMM FAM04: 04/19/2004
© Duraiswami & Gumerov, 2003-2004

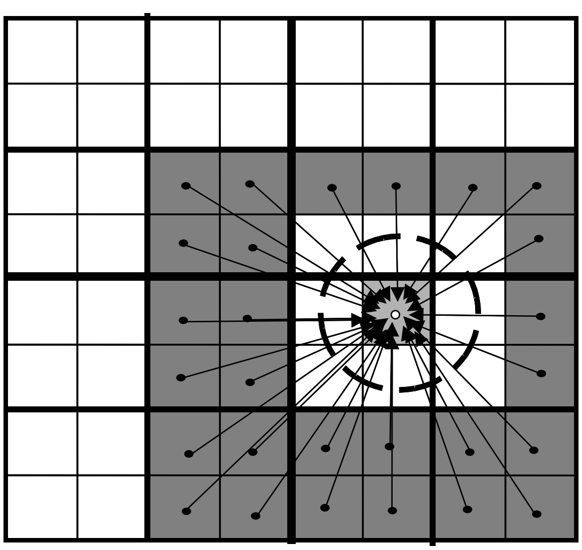
Downward Pass. Step 1.

THIS MIGHT BE
THE MOST EXPENSIVE
STEP OF THE ALGORITHM



Downward Pass. Step 1.

$$P_4 = PowerOfE_4Neighborhood = 3^d 2^d - 3^d = 3^d\left(2^d - 1\right)$$



$$\begin{aligned} d &= 1: & P_4 &= 3, \\ d &= 2: & P_4 &= 27, \\ d &= 3: & P_4 &= 189, \\ d &= 4: & P_4 &= 1215, \\ &\ldots \end{aligned}$$

Exponential Growth

Total number of S|R-translations per 1 box in *d*-dimensional space

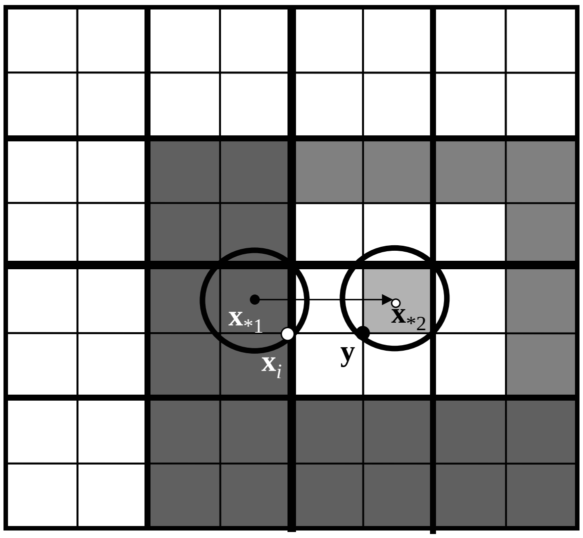
(far from the domain boundaries)

Domains of Expansion Validity (6). S|R-translation.

■ S-expansion coefficients can be $S|R$-translated (converted to R-expansion coefficients)

$$|\mathbf{y}-\mathbf{x}_{*2}| < |\mathbf{x}_{*1}-\mathbf{x}_{*2}| - |\mathbf{x}_i-\mathbf{x}_{*1}|,$$

$$\mathbf{A}(\mathbf{x}_i,\mathbf{x}_{*2}) = (\mathbf{S}|\mathbf{R})(\mathbf{x}_{*2}-\mathbf{x}_{*1})\mathbf{B}(\mathbf{x}_i,\mathbf{x}_{*1})$$



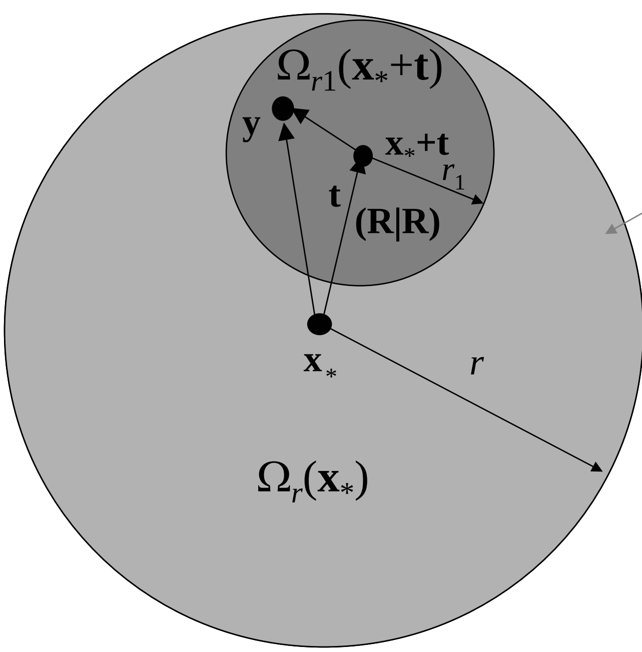
$$|\mathbf{y}-\mathbf{x}_*| < r_{\min}(l), \quad |\mathbf{x}_i-\mathbf{x}_*| < r_{\min}(l),$$

$$\min|\mathbf{x}_{*1}-\mathbf{x}_{*2}| = 2size(l).$$

$$2^{-l-1}\sqrt{d} + 2^{-l-1}\sqrt{d} < 2^{-l+1}, \quad d < 4.$$

$d < 4$

R|R-reexpansion (Local to Local, or L2L)



Original expansion
Is valid only here!

$$|\mathbf{y} - \mathbf{x}_* - \mathbf{t}| < r_1 = r - |\mathbf{t}|$$

Since $\Omega_{r1}(\mathbf{x}_*+\mathbf{t}) \subset \Omega_r(\mathbf{t})$!

Downward Pass Step 2

- Now consider we already have done the S|R translation at some level at the center of a box.
- So we have a R expansion that includes contribution of most of the points, but not of points in the E_4 neighborhood
- We can go to a finer level to include these missed points
- But we will now have to translate the already built R expansion to a box center of a child
  - ❑ (Makes no sense to do S|R again, since many S|R are consolidated in this R expansion)
- Add to this translated one, the S|R of the E_4 of the finer level

- Formally

Step 2. At $l = 2$ we have

$$\Phi_3^{(n,2)}(\mathbf{y}) = \Phi_4^{(n,2)}(\mathbf{y}), \quad \mathbf{D}^{(n,2)} = \widetilde{\mathbf{D}}^{(n,2)},$$

Form $\Phi_3^{(n,l)}(\mathbf{y})$ (or expansion coefficients of this function) by adding $\Phi_4^{(Parent(n),l-1)}(\mathbf{y})$ to $(\mathbf{R}|\mathbf{R})$- translated coefficients of the parent box to the child center:

$$\Phi_3^{(n,l)}(\mathbf{y}) = \mathbf{D}^{(n,l)} \circ \mathbf{R}(\mathbf{y} - \mathbf{x}_c^{(n,l)}),$$

$$\mathbf{D}^{(n,l)} = \widetilde{\mathbf{D}}^{(n,l)} + (\mathbf{R}|\mathbf{R})\left(\mathbf{x}_c^{(n,l)} - \mathbf{x}_c^{(m,l-1)}\right)\mathbf{D}^{(m,l-1)}, \quad m = Parent(n).$$

$$\Phi_4^{(n,l)}(\mathbf{y}) = \widetilde{\mathbf{D}}^{(n,l)} \circ \mathbf{R}(\mathbf{y} - \mathbf{x}_c^{(n,l)}),$$

$$\widetilde{\mathbf{D}}^{(n,l)} = \sum_{m \in I_4(n,l)} (\mathbf{S}|\mathbf{R})\left(\mathbf{x}_c^{(n,l)} - \mathbf{x}_c^{(m,l)}\right)\mathbf{C}^{(m,l)}.$$

Downward Pass. Step 2.

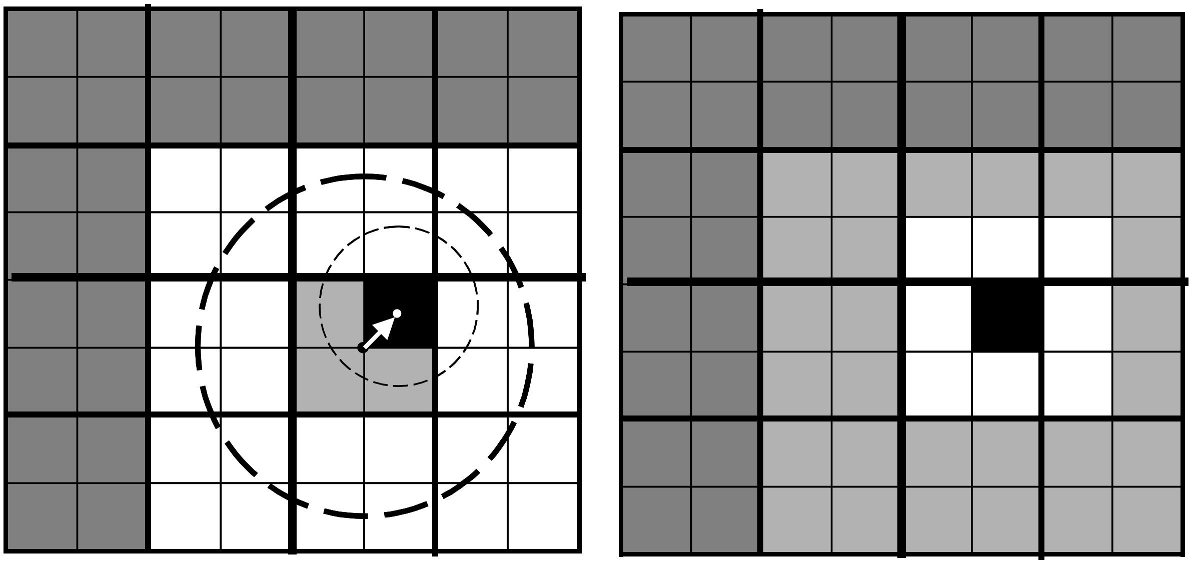


Figure shows that local-to-local translation is applicable in this case (smaller sphere is located completely inside the larger sphere), and junction of structures $E_3(n,l)$ and $E_4(n,l+1)$ produces $E_3(n,l+1)$:

$$E_3(n,l+1) = E_3(n,l) \cup E_4(n,l+1).$$

Domains of Expansion Validity (5). R|R and S|S-translations.

- R-expansion coefficients can be $R|R$-translated:

$$|\mathbf{y}-\mathbf{x}_{*2}| < |\mathbf{x}_i-\mathbf{x}_{*1}| - |\mathbf{x}_{*1}-\mathbf{x}_{*2}| :$$

$$\mathbf{A}(\mathbf{x}_i,\mathbf{x}_{*2}) = (\mathbf{R}|\mathbf{R})(\mathbf{x}_{*2}-\mathbf{x}_{*1})\mathbf{A}(\mathbf{x}_i,\mathbf{x}_{*1})$$

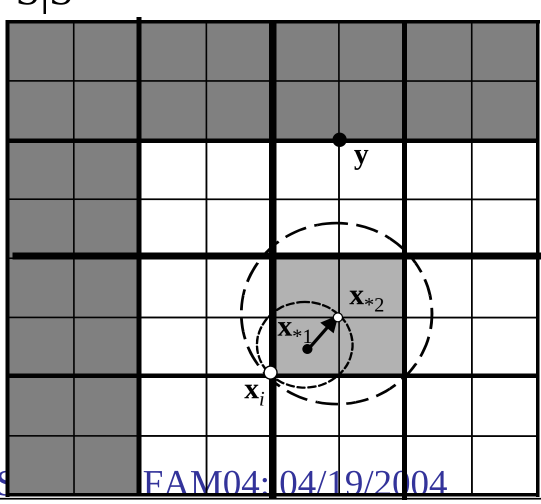
- S-expansion coefficients can be $S|S$-translated:

$$|\mathbf{y}-\mathbf{x}_{*2}| > |\mathbf{x}_{*1}-\mathbf{x}_{*2}| + |\mathbf{x}_i-\mathbf{x}_{*1}|,$$

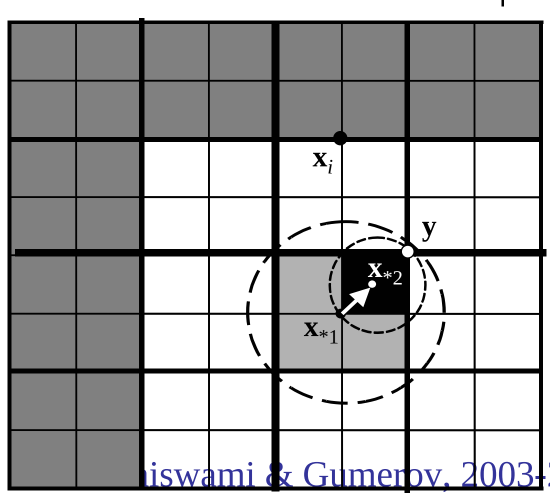
$$\mathbf{B}(\mathbf{x}_i,\mathbf{x}_{*2}) = (\mathbf{S}|\mathbf{S})(\mathbf{x}_{*2}-\mathbf{x}_{*1})\mathbf{B}(\mathbf{x}_i,\mathbf{x}_{*1})$$

Not as restrictive as S|R

S|S



R|R



Final Summation

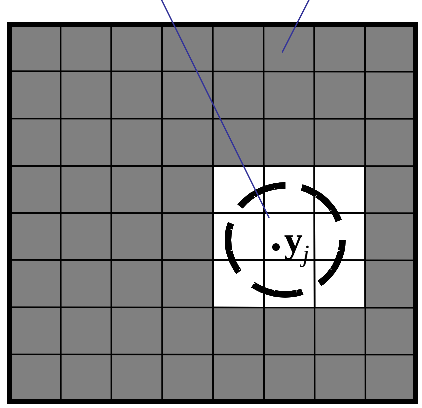
- At this point we are at the finest level.
- We cannot do any S|R translation for x_i ‘s that are in the E_3 neighborhood of our y_j’s
- Must evaluate these directly

Final Summation

As soon as coefficients $\mathbf{D}^{(n,L)}$ are determined total potential can be computed for any point $\mathbf{y}_j \in E_1(0,0)$, where $\Phi_2^{(n,l)}(\mathbf{y})$ can be computed straightforward. So:

$$v_j = \Phi(\mathbf{y}_j) = \sum_{\mathbf{x}_i \in E_2(n,L)} u_i \Phi(\mathbf{y}_j, \mathbf{x}_i) + \mathbf{D}^{(n,L)} \circ \mathbf{R}(\mathbf{y}_j - \mathbf{x}_c^{(n,L)}), \quad \mathbf{y}_j \in E_1(n,L).$$

Contribution of E_2 Contribution of E_3



Cost of FMM --- Upward Pass

- Upward Step1. Cost of creating an S expansion for each source point. *O(NP)*
- Upward Step2. Cost of performing an S|S translation
  - If we use expensive (matrix vector) method cost is $O(P^2)$ for one translation.
- Step 2 is repeated from level *L-1* to level *2*

$$CostUpward_2 = 2^d\left(2^{(L-1)d} + 2^{(L-2)d} + \ldots + 2^{2d}\right)CostSS(P)$$
$$< \frac{2^d}{2^d-1}\left(2^{Ld}-1\right)CostSS(P) \sim \frac{N}{s}CostSS(P)$$

- Total Cost of Upward Pass $\sim NP + (N/s)\,(P^2)$

COST of MLFMM

- Cost of downward pass, step 1 is the cost of performing S|R translations at each level

$$CostDownward_1 \lesssim P_4(d)\left(2^{2d} + \ldots + 2^{Ld}\right)CostSR(P) \sim P_4(d)\frac{N}{s}CostSR(P),$$

- At the downward pass, 2nd step we have the cost of the R|R translation, and S|R translation from the E_4 neighbourhood (already accounted for above)

$$CostDownward_2 = 2^d\left(2^{2d} + \ldots + 2^{(L-1)d}\right)CostRR(P) \sim \frac{N}{s}CostRR(P),$$

- Final summation cost is $CostEvaluation = M(P_2(d)sCostFunc + P).$

- Total

$$CostMLFMM = (M+N)P + (P_4(d)+2)\frac{N}{s}CostTranslation(P) + P_2(d)sMCostFunc$$

Itemized Cost of MLFMM

Regular mesh:

$$N = 2^{L_* d}, \quad s = 2^{L_s d}, \quad L = L_{\max} = L_* - L_s.$$

Assume that all translation costs are the same, *CostTranslation*(*P*)

$$CostUpward_1 = NCostExpansion(P) = O(NP).$$

$$CostUpward_2 = 2^d\left(2^{(L-1)d} + 2^{(L-2)d} + \ldots + 2^{2d}\right)CostSS(P) < \frac{2^d}{2^d - 1}\left(2^{Ld} - 1\right)CostSS(P) \sim \frac{N}{s}CostSS(P)$$

$$CostDownward_1 \lesssim P_4(d)\left(2^{2d} + \ldots + 2^{Ld}\right)CostSR(P) \sim P_4(d)\frac{N}{s}CostSR(P),$$

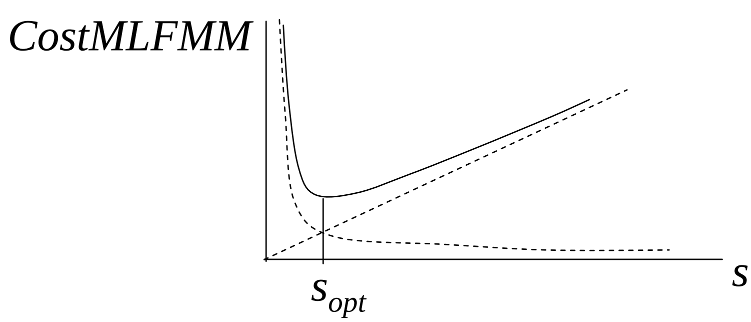
$$CostDownward_2 = 2^d\left(2^{2d} + \ldots + 2^{(L-1)d}\right)CostRR(P) \sim \frac{N}{s}CostRR(P),$$

$$CostEvaluation = M(P_2(d)sCostFunc + P).$$

Powers of E_4 and E_2 neighborhoods

$$CostMLFMM = (M+N)P + (P_4(d)+2)\frac{N}{s}CostTranslation(P) + P_2(d)sMCostFunc$$

Optimization of the Grouping Parameter



$$CostMLFMM = (M+N)P + (P_4(d)+2)\frac{N}{s}CostTranslation(P) + P_2(d)sMCostFunc$$

$$\frac{\partial CostMLFMM}{\partial s} = -(P_4(d)+2)\frac{N}{s^2}CostTranslation(P) + P_2(d)MCostFunc = 0$$

$$s_{opt} = \left[\frac{N(P_4(d)+2)CostTranslation(P)}{MP_2(d)CostFunc}\right]^{1/2}.$$

$$CostMLFMM_{opt} = (M+N)P + 2[MN(P_4(d)+2)P_2(d)CostTranslation(P)CostFunc]^{1/2}.$$

Optimization of the Grouping Parameter (Example)

$$s_{opt} = \left[\frac{N(P_4(d)+2)CostTranslation(P)}{MP_2(d)CostFunc}\right]^{1/2}.$$

$$CostMLFMM_{opt} = (M+N)P + 2[MN(P_4(d)+2)P_2(d)CostTranslation(P)CostFunc]^{1/2}.$$

Example:

$$N = M, \quad P_4(d) = 3^d\left(2^d - 1\right), \quad P_2(d) = 3^d,$$

$$CostTranslation(P) = P^2, \quad CostFunc = 1$$

$$s_{opt} \sim 2^{d/2}P; \quad CostMLFMM_{opt} \sim 2NP\left(1 + 3^d 2^{d/2}\right)$$

$$For\ d = 2, \quad P = 10, \quad s_{opt} \sim 38, \quad CostMLFMM_{opt} \sim 38NP = 380N.$$

If non-optimized,

$$s = 1; \quad CostMLFMM_{opt} \sim NP\left(2 + 3^d 2^d P\right)$$

$$For\ d = 2, \quad P = 10, \quad s = 1, \quad CostMLFMM_{opt} \sim 360NP = 3600N.$$

In this example optimization results in about 10 times savings!

DEMO

- Yang Wang (wpwy@umiacs.umd.edu), "Java Implementation and Simulation of the Fast Multipole Method for 2-D Coulombic Potential Problems," AMSC 698R course project report, 2003.
- http://brigade.umiacs.umd.edu/~wpwy/applet/FmmApplet.html
- Seems to work with Mozilla and Netscape …IE has problems

Some Numerical Experiments with MLFMM

N.A. Gumerov, R. Duraiswami & E.A. Borovikov

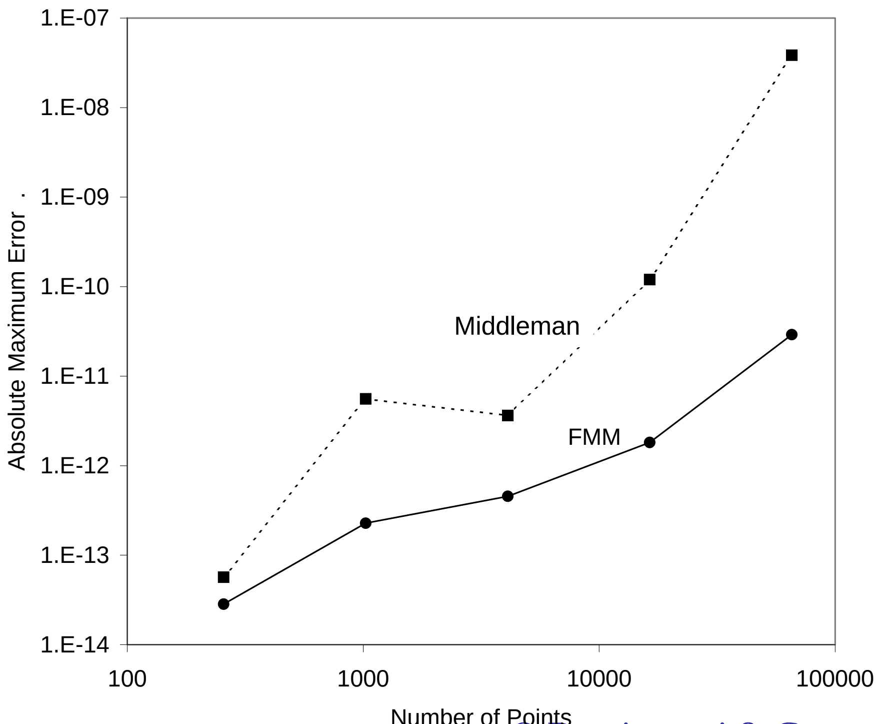
Data Structures, Optimal Choice of Parameters, and Complexity Results for Generalized Multilevel Fast Multipole Methods in *d* Dimensions.

UMIACS TR 2003-28,
Also issued as *Computer Science Technical Report CS-TR-# 4458*.
University of Maryland, College Park, 2003.

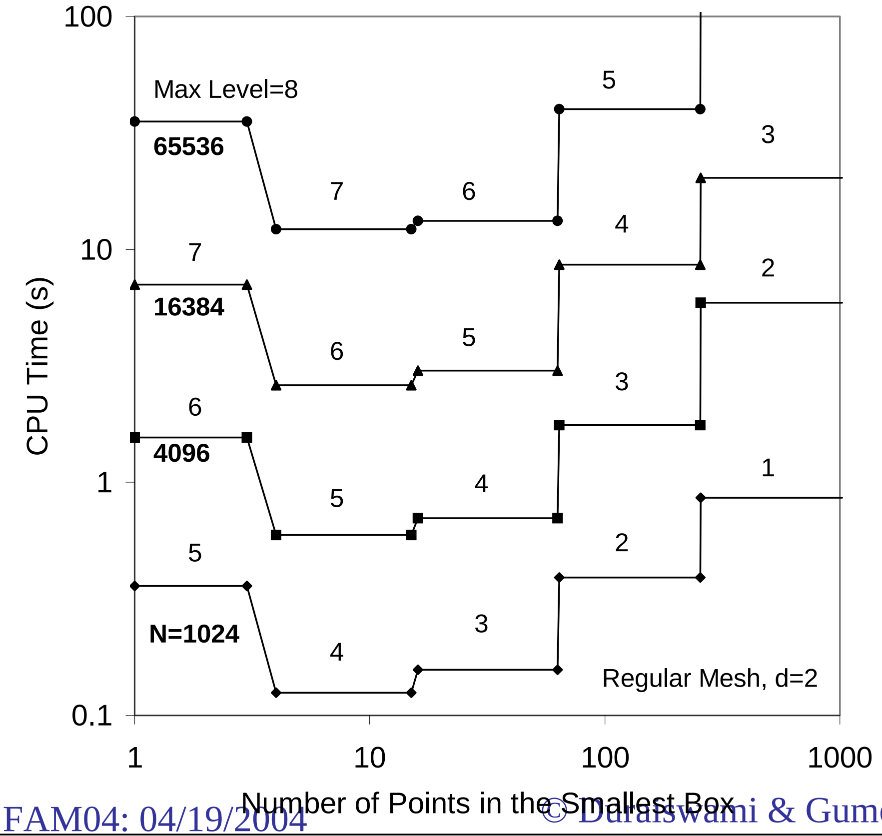
Available online via
http://www.umiacs.umd.edu/~ramani/pubs/umiacs-tr-2003-28.pdf

Error Test. FMM vs Middleman.

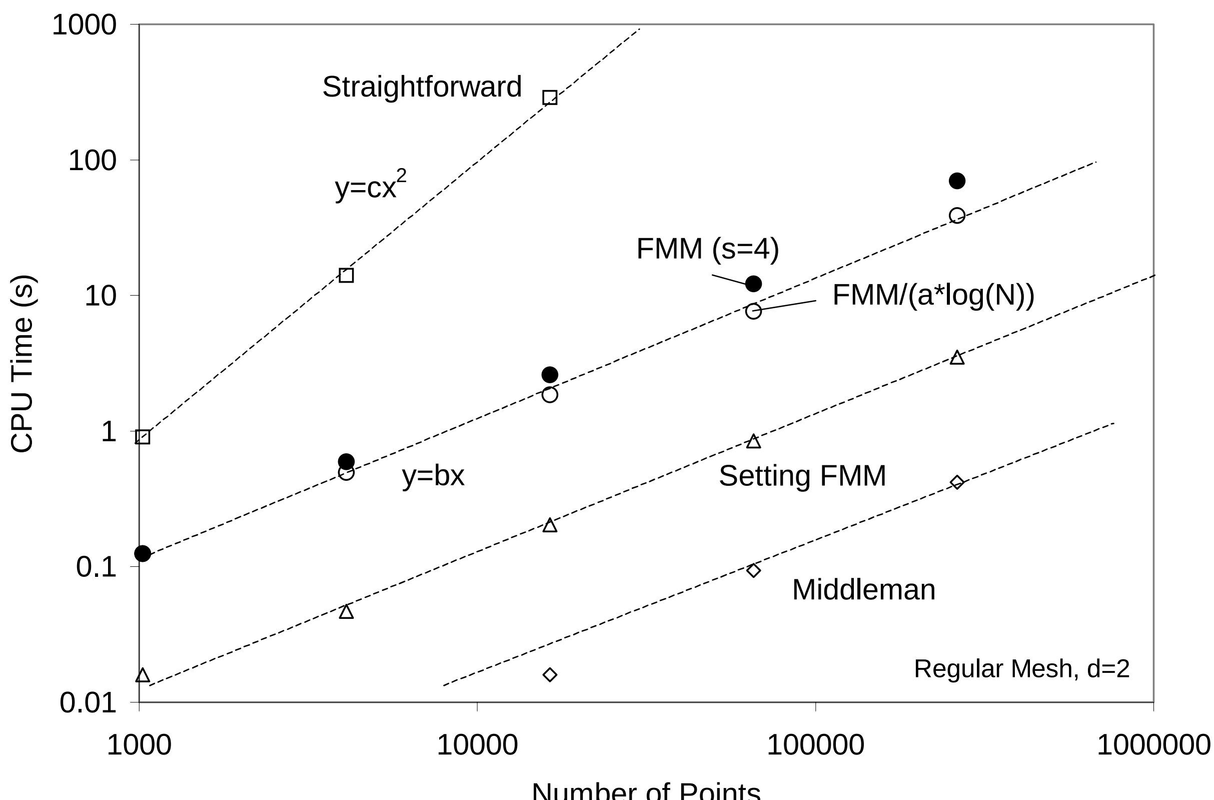
Regular Mesh, $N = M$.



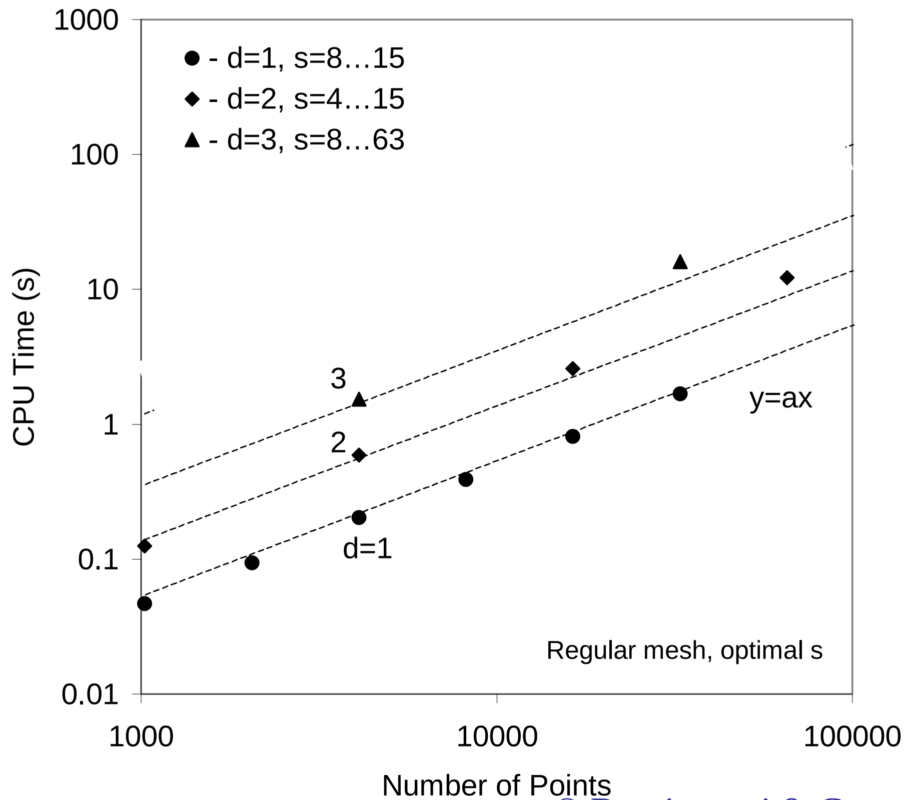
Test with Varying Grouping Parameter.



Test with Varying N.

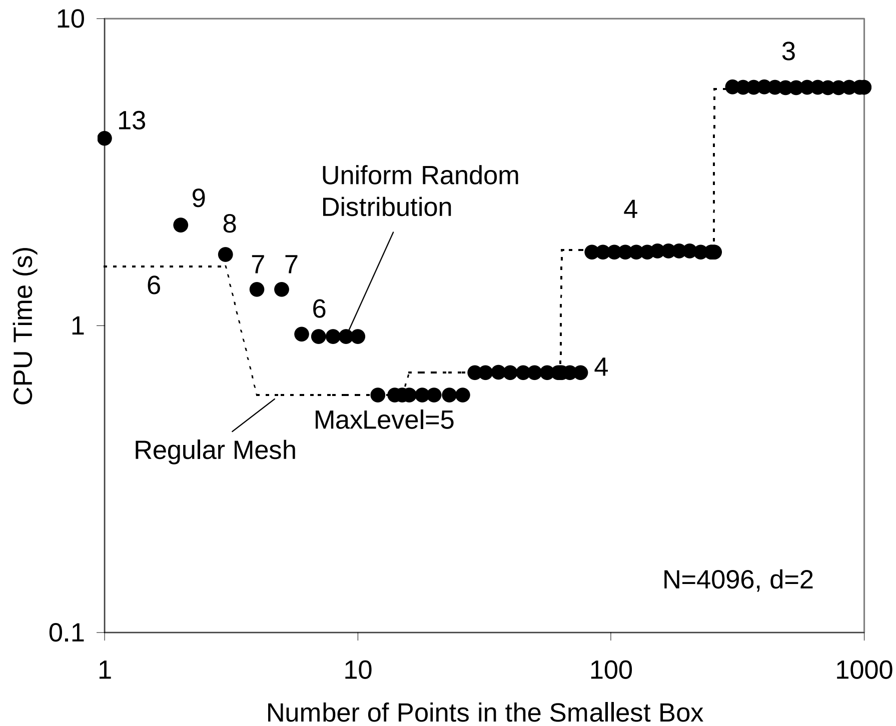


Comparisons for different dimensionalities

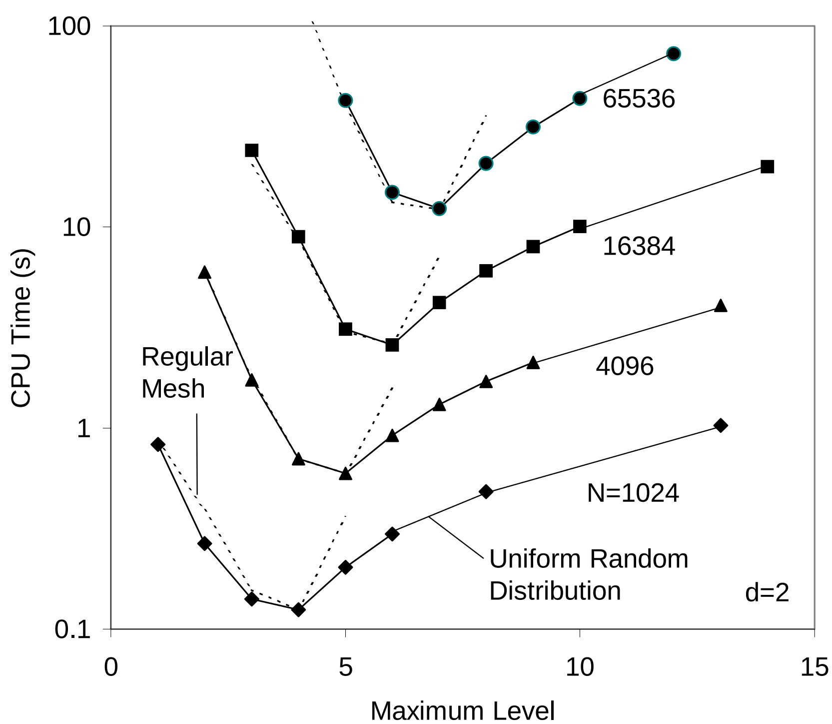


Random Distributions

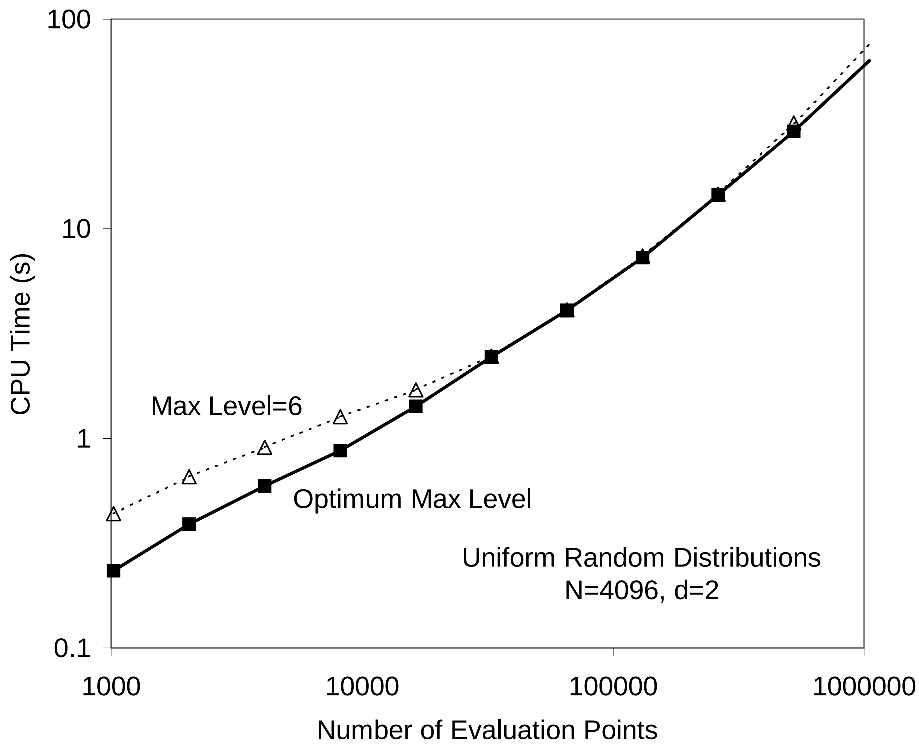
Dependence of CPU Time on the Grouping Parameter, s



Dependence of CPU Time on the Maximum Space Subdivision Level



Dependence of CPU Time on *M*



Adaptive FMM

- H. Cheng, L. Greengard, and V. Rokhlin, "A Fast Adaptive Multipole Algorithms in Three Dimensions," Journal of Computational Physics, 155:468-498, 1999 .
- N.A. Gumerov, R. Duraiswami, and Y.A. Borovikov, "Data structures and algorithms for adaptive multilevel fast multipole methods," in preparation.