# CS 598: Communication Cost Analysis of Algorithms

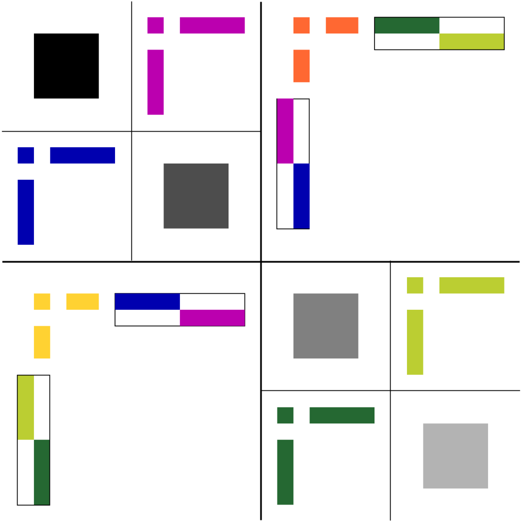
## Lecture 26: Hierarchically semi-separable (HSS) matrices

Edgar Solomonik

University of Illinois at Urbana-Champaign

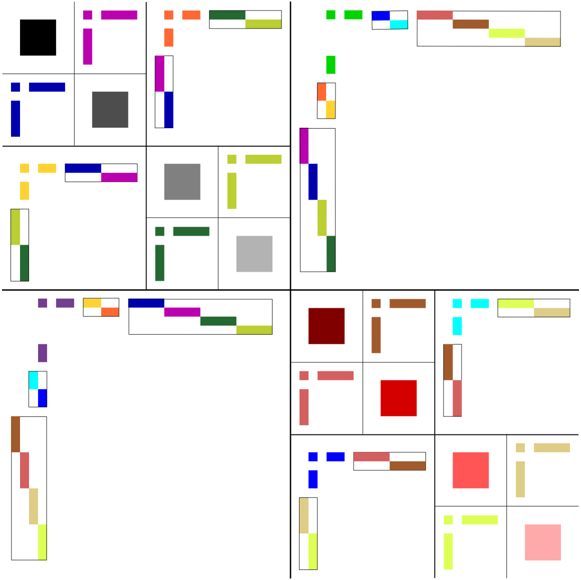
November 28, 2016

# HSS matrix, two levels



Hierarchically semi-separable (HSS) matrix, space padded around each matrix block, which are uniquely identified by dimensions and color

# HSS matrix, three levels



# HSS matrix formal definition

Consider matrix $\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$

- the $l$-level HSS factorization is

$$\mathcal{H}_l(\mathbf{A}) = \begin{cases} \{\mathbf{U}, \mathbf{V}, \mathbf{T}_{12}, \mathbf{T}_{21}, \mathbf{A}_{11}, \mathbf{A}_{22}\} & : l = 1 \\ \{\mathbf{U}, \mathbf{V}, \mathbf{T}_{12}, \mathbf{T}_{21}, \mathcal{H}_{l-1}(\mathbf{A}_{11}), \mathcal{H}_{l-1}(\mathbf{A}_{22})\} & : l > 1 \end{cases}$$

- the low-rank representation of the diagonal blocks is given by $\mathbf{A}_{21} = \bar{\mathbf{U}}_2 \mathbf{T}_{21} \bar{\mathbf{V}}_1^T$, $\mathbf{A}_{12} = \bar{\mathbf{U}}_1 \mathbf{T}_{12} \bar{\mathbf{V}}_2^T$ where for $a \in \{1, 2\}$,

$$\bar{\mathbf{U}}_a = \mathcal{U}_a(\mathcal{H}_l(\mathbf{A})) = \begin{cases} \mathbf{U}_a & : l = 1 \\ \begin{bmatrix} \mathcal{U}_1(\mathcal{H}_{l-1}(\mathbf{A}_{aa})) & \mathbf{0} \\ \mathbf{0} & \mathcal{U}_2(\mathcal{H}_{l-1}(\mathbf{A}_{aa})) \end{bmatrix} \mathbf{U}_a & : l > 1 \end{cases}$$

$$\bar{\mathbf{V}}_a = \mathcal{V}_a(\mathcal{H}_l(\mathbf{A})) = \begin{cases} \mathbf{V}_a & : l = 1 \\ \begin{bmatrix} \mathcal{V}_1(\mathcal{H}_{l-1}(\mathbf{A}_{aa})) & \mathbf{0} \\ \mathbf{0} & \mathcal{V}_2(\mathcal{H}_{l-1}(\mathbf{A}_{aa})) \end{bmatrix} \mathbf{V}_a & : l > 1 \end{cases}$$

# HSS matrix–vector multiplication

We now consider computing $\mathbf{y} = \mathbf{A}\mathbf{x}$

- with $\mathcal{H}_1(\mathbf{A})$ we would just compute $\mathbf{y}_1 = \mathbf{A}_{11}\mathbf{x}_1 + \mathbf{U}_1(\mathbf{T}_{12}(\mathbf{V}_2^T\mathbf{x}_2))$ and $\mathbf{y}_2 = \mathbf{A}_{22}\mathbf{x}_2 + \mathbf{U}_2(\mathbf{T}_{21}(\mathbf{V}_1^T\mathbf{x}_1))$
- for general $\mathcal{H}_l(\mathbf{A})$ we will perform an up-sweep and a down-sweep
  - up-sweep computes $\mathbf{w} = \begin{bmatrix} \bar{\mathbf{V}}_1^T\mathbf{x}_1 \\ \bar{\mathbf{V}}_2^T\mathbf{x}_2 \end{bmatrix}$ *at every tree node*
  - down-sweep computes a tree sum of $\begin{bmatrix} \bar{\mathbf{U}}_1\mathbf{T}_{12}\mathbf{w}_2 \\ \bar{\mathbf{U}}_2\mathbf{T}_{21}\mathbf{w}_1 \end{bmatrix}$
- the up-sweep is performed by using the nested structure of $\bar{\mathbf{V}}$

$$\mathbf{w} = \mathcal{W}(\mathcal{H}_l(\mathbf{A}), \mathbf{x}) = \begin{cases} \begin{bmatrix} \mathbf{V}_1^T & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_2^T \end{bmatrix} \mathbf{x} & : l = 1 \\ \begin{bmatrix} \mathbf{V}_1^T & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_2^T \end{bmatrix} \begin{bmatrix} \mathcal{W}(\mathcal{H}_{l-1}(\mathbf{A}_{11}), \mathbf{x}_1) \\ \mathcal{W}(\mathcal{H}_{l-1}(\mathbf{A}_{22}), \mathbf{x}_2) \end{bmatrix} & : l > 1 \end{cases}$$

# HSS matrix–vector multiplication, down-sweep

We now employ each $\mathbf{w} = \mathcal{W}(\mathcal{H}_l(\mathbf{A}), \mathbf{x})$ from the root to the leaves to get

$$\mathbf{y} = \mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{U}_1 \mathbf{T}_{12} \mathbf{w}_2 \\ \mathbf{U}_2 \mathbf{T}_{21} \mathbf{w}_1 \end{bmatrix} + \begin{bmatrix} \mathbf{A}_{11} \mathbf{x}_1 \\ \mathbf{A}_{22} \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{U}}_1 & \mathbf{0} \\ \mathbf{0} & \bar{\mathbf{U}}_2 \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{0} \end{bmatrix} \mathbf{w} + \begin{bmatrix} \mathbf{A}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{22} \end{bmatrix} \mathbf{x}$$

- using the nested structure of $\bar{\mathbf{U}}_a$ and $\mathbf{v} = \begin{bmatrix} \mathbf{U}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{U}_2 \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{0} \end{bmatrix} \mathbf{w}$,

$$\mathbf{y}_a = \begin{bmatrix} \mathcal{U}_1(\mathcal{H}_{l-1}(\mathbf{A}_{aa})) & \mathbf{0} \\ \mathbf{0} & \mathcal{U}_2(\mathcal{H}_{l-1}(\mathbf{A}_{aa})) \end{bmatrix} \mathbf{v}_a + \mathbf{A}_{aa} \mathbf{x_a} \quad \text{for} \quad a \in \{1, 2\}$$

- which gives the down-sweep recurrence

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{z} = \mathcal{Y}(\mathcal{H}_l(\mathbf{A}), \mathbf{x}, \mathbf{z}) = \begin{cases} \begin{bmatrix} \mathbf{U}_1 \mathbf{q}_1 \\ \mathbf{U}_2 \mathbf{q}_2 \end{bmatrix} + \begin{bmatrix} \mathbf{A}_{11} \mathbf{x}_1 \\ \mathbf{A}_{22} \mathbf{x}_2 \end{bmatrix} & : l = 1 \\ \begin{bmatrix} \mathcal{Y}(\mathcal{H}_{l-1}(\mathbf{A}_{11}), \mathbf{x}_1, \mathbf{U}_1 \mathbf{q}_1) \\ \mathcal{Y}(\mathcal{H}_{l-1}(\mathbf{A}_{22}), \mathbf{x}_2, \mathbf{U}_2 \mathbf{q}_2) \end{bmatrix} & : l > 1 \end{cases}$$

where $\mathbf{q} = \begin{bmatrix} \mathbf{0} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{0} \end{bmatrix} \mathcal{W}(\mathcal{H}_l(\mathbf{A}), \mathbf{x}) + \mathbf{z}$

# Prefix sum as HSS matrix–vector multiplication

We can express the $n$-element prefix sum $y(i) = \sum_{j=1}^{i-1} x(j)$ as

$$\mathbf{y} = \mathbf{L}\mathbf{x} \quad \text{where} \quad \mathbf{L} = \begin{bmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 1 & \cdots & 1 & 0 \end{bmatrix}$$

- $\mathbf{L}$ is an $\mathcal{H}$-matrix since $\mathbf{L}_{21} = \mathbf{1_n}\mathbf{1_n}^{\mathsf{T}} = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}$
- $\mathbf{L}$ also has rank-1 HSS structure, in particular

$$\mathcal{H}_l(\mathbf{L}) = \begin{cases} \left\{ \mathbf{1_2}, \mathbf{1_2}, \begin{bmatrix} 0 \end{bmatrix}, \begin{bmatrix} 1 \end{bmatrix}, \begin{bmatrix} 0 \end{bmatrix}, \begin{bmatrix} 0 \end{bmatrix} \right\} & : l = 1 \\ \left\{ \mathbf{1_4}, \mathbf{1_4}, \begin{bmatrix} 0 \end{bmatrix}, \begin{bmatrix} 1 \end{bmatrix}, \mathcal{H}_{l-1}(\mathbf{L}_{11}), \mathcal{H}_{l-1}(\mathbf{L}_{22}) \right\} & : l > 1 \end{cases}$$

so each $\mathbf{U}, \mathbf{V}, \bar{\mathbf{U}}, \bar{\mathbf{V}}$ is a vector of 1s, $\mathbf{T}_{12} = \begin{bmatrix} 0 \end{bmatrix}$ and $\mathbf{T}_{21} = \begin{bmatrix} 1 \end{bmatrix}$

# Prefix sum HSS up-sweep

We can use the HSS structure of $\mathbf{L}$ to compute the prefix sum of $\mathbf{x}$

- recall that the up-sweep recurrence has the general form

$$\mathbf{w} = \mathcal{W}(\mathcal{H}_l(\mathbf{A}), \mathbf{x}) = \begin{cases} \begin{bmatrix} \mathbf{V}_1^T & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_2^T \end{bmatrix} \mathbf{x} & : l = 1 \\ \begin{bmatrix} \mathbf{V}_1^T & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_2^T \end{bmatrix} \begin{bmatrix} \mathcal{W}(\mathcal{H}_{l-1}(\mathbf{A}_{11}), \mathbf{x}_1) \\ \mathcal{W}(\mathcal{H}_{l-1}(\mathbf{A}_{22}), \mathbf{x}_2) \end{bmatrix} & : l > 1 \end{cases}$$

- for the prefix sum this becomes

$$\mathbf{w} = \mathcal{W}(\mathcal{H}_l(\mathbf{L}), \mathbf{x}) = \begin{cases} \mathbf{x} & : l = 1 \\ \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mathcal{W}(\mathcal{H}_{l-1}(\mathbf{L}_{11}), \mathbf{x}_1) \\ \mathcal{W}(\mathcal{H}_{l-1}(\mathbf{L}_{22}), \mathbf{x}_2) \end{bmatrix} & : l > 1 \end{cases}$$

- so the up-sweep computes $\mathbf{w} = \begin{bmatrix} \mathcal{S}(\mathbf{x_1}) \\ \mathcal{S}(\mathbf{x_2}) \end{bmatrix}$ where $\mathcal{S}(\mathbf{a}) = \sum_i a(i)$

# Prefix sum HSS down-sweep

The down-sweep has the general structure

$$\mathbf{y} = \mathcal{Y}(\mathcal{H}_l(\mathbf{A}), \mathbf{x}, \mathbf{z}) = \begin{cases} \begin{bmatrix} \mathbf{U}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{U}_2 \end{bmatrix} \mathbf{q} + \begin{bmatrix} \mathbf{A}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{22} \end{bmatrix} \mathbf{x} & : l = 1 \\ \begin{bmatrix} \mathcal{Y}(\mathcal{H}_{l-1}(\mathbf{A}_{11}), \mathbf{x}_1, \mathbf{U}_1\mathbf{q}_1) \\ \mathcal{Y}(\mathcal{H}_{l-1}(\mathbf{A}_{22}), \mathbf{x}_2, \mathbf{U}_2\mathbf{q}_2) \end{bmatrix} & : l > 1 \end{cases}$$

where $\mathbf{q} = \begin{bmatrix} \mathbf{0} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{0} \end{bmatrix} \mathcal{W}(\mathcal{H}_l(\mathbf{A}), \mathbf{x}) + \mathbf{z}$, for the prefix sum:

- $\begin{bmatrix} \mathbf{0} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{0} \end{bmatrix} \mathcal{W}(\mathcal{H}_l(\mathbf{L}), \mathbf{x}) = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \mathcal{S}(\mathbf{x_1}) \\ \mathcal{S}(\mathbf{x_2}) \end{bmatrix} = \begin{bmatrix} 0 \\ \mathcal{S}(\mathbf{x_1}) \end{bmatrix} = \mathbf{q} - \mathbf{z}$

$$\mathbf{y} = \mathcal{Y}(\mathcal{H}_l(\mathbf{L}), \mathbf{x}, \mathbf{z}) = \begin{cases} \begin{bmatrix} z(1) \\ x(1) + z(2) \end{bmatrix} & : l = 1 \\ \begin{bmatrix} \mathcal{Y}(\mathcal{H}_{l-1}(\mathbf{L}_{11}), \mathbf{x}_1, \mathbf{1_2} z(1)) \\ \mathcal{Y}(\mathcal{H}_{l-1}(\mathbf{L}_{22}), \mathbf{x}_2, \mathbf{1_2}(\mathcal{S}(\mathbf{x}_1) + z(2)) \end{bmatrix} & : l > 1 \end{cases}$$

- initially the prefix $\mathbf{z} = \mathbf{0}$ and it will always be the case that $z(1) = z(2)$

# Short pause

# Cost of HSS matrix–vector multiplication

The down-sweep and the up-sweep perform small dense matrix–vector multiplications at each recursive step

- lets assume $k$ is the dimension of the leaf blocks and the rank at each level (number of columns in each $\mathbf{U}_a$, $\mathbf{V}_a$)
- the computation cost for both the down-sweep and up-sweep is

$$T(n,k) = 2T(n/2,k) + O(k^2 \cdot \gamma), \quad T(k,k) = O(k^2 \cdot \gamma)$$
$$T(n,k) = O(nk \cdot \gamma)$$

- Q: what parallelization approach would be sensible for small $k$?
- if we assign each tree node to a single processor for the first $\log_2(P)$ levels, and execute a different leaf subtree with a different processor

$$\begin{aligned} T(n,k,P) &= 2T(n,k,P/2) + O(k^2 \cdot \gamma + k \cdot \beta + \alpha) \\ &= O((nk/P + k^2 \log(P)) \cdot \gamma + k \log(P) \cdot \beta + \log(P) \cdot \alpha) \end{aligned}$$

# Synchronization-efficient HSS multiplication

Our first algorithm would require $O(\log(P))$ BSP supersteps

- Q: what would we need to do to reduce this to $O(1)$?
- we can observe that the leaf subtrees can be computed independently

$$T_{\text{leaf-subtrees}}(n,k,P) = O(nk/P \cdot \gamma + k \cdot \beta + \alpha)$$

- thus we can focus on doing the up-sweep and down-sweep on a binary tree with $\log_2(P)$ levels
  - executing the root subtree sequentially would yield a cost of

$$T_{\text{root-subtree}}(n,k,P) = O(Pk^2 \cdot \gamma + Pk \cdot \beta + \alpha)$$

  - this could be prohibitive on a large number of processors
  - instead have $P^r$ $(r < 1)$ processors compute subtrees with $P^{1-r}$ leaves, then recurse on the $P^r$ roots

$$T_{\text{rec-tree}}(k,P) = T_{\text{rec-tree}}(k,P^r) + O(P^{1-r}k^2 \cdot \gamma + P^{1-r}k \cdot \beta + \alpha)$$

  - the algorithm has BSP complexity

$$T_{\text{rec-tree}}(k,P) = O(P^{1-r}k^2 \cdot \gamma + P^{1-r}k \cdot \beta + \log_{1/r}(\log(P)) \cdot \alpha)$$

# Synchronization-efficient HSS multiplication

We can do better by exploiting what the HSS tree nodes are doing

- again lets focus on the top tree with $P$ leaves (leaf subtrees)
- lets try to assign each processor a unique path from a leaf to the root
- given $\mathbf{w} = \mathcal{W}(\mathcal{H}_l(\mathbf{A}), \mathbf{x})$ at every node its clear each processor can compute a down-sweep path in the subtree independently
- for the up-sweep, we can realize that the tree applies a linear transformation, so we can sum the results computed in each path
- for each tree node, there is a contribution from every processor assigned a leaf of the subtree of the node
- therefore, there are $P-1$ sums of a total of $O(P\log(P))$ contributions, for a total of $O(kP\log(P))$ elements
- we can do these with $\min(P, k\log_2(P))$ processors, each obtaining $\max(P, k\log_2(P))$ contributions, so

$$T_{\text{root-paths}}(k, P) = O(k^2\log(P)\cdot\gamma + (k\log(P) + P)\cdot\beta + \alpha)$$

- we have not improved the asymptotic number of messages, but only the number of synchronizations, and can leverage efficient reductions

# HSS multiplication by multiple vectors

Consider multiplication $\mathbf{C} = \mathbf{AB}$ where $\mathbf{A} \in \mathbb{R}^{n \times n}$ is HSS and $\mathbf{B} \in \mathbb{R}^{n \times b}$

- lets consider the case that $P \leq b \ll n$
- if we assign each processor all of $\mathbf{A}$, each can compute a column of $\mathbf{C}$ simultaneously
- this requires a prohibitive amount of memory usage
- Q: could you propose a good BSP algorithm for this problem?
- A: use transpose like in the $b$ scans problem
  - perform leaf-level multiplications, processing $n/P$ rows of $\mathbf{B}$ with each processor (call intermediate $\bar{\mathbf{C}}$)
  - transpose $\bar{\mathbf{C}}$ and apply $\log_2(P)$ root levels of HSS tree to columns of $\bar{\mathbf{C}}$ independently
- this algorithm requires replication only of the root $O(\log(P))$ levels of the HSS tree, $O(Pb)$ data
- for large $k$ or larger $P$ different algorithms may be desirable