Efficient GPU-only Tree Walks in ChaNGa

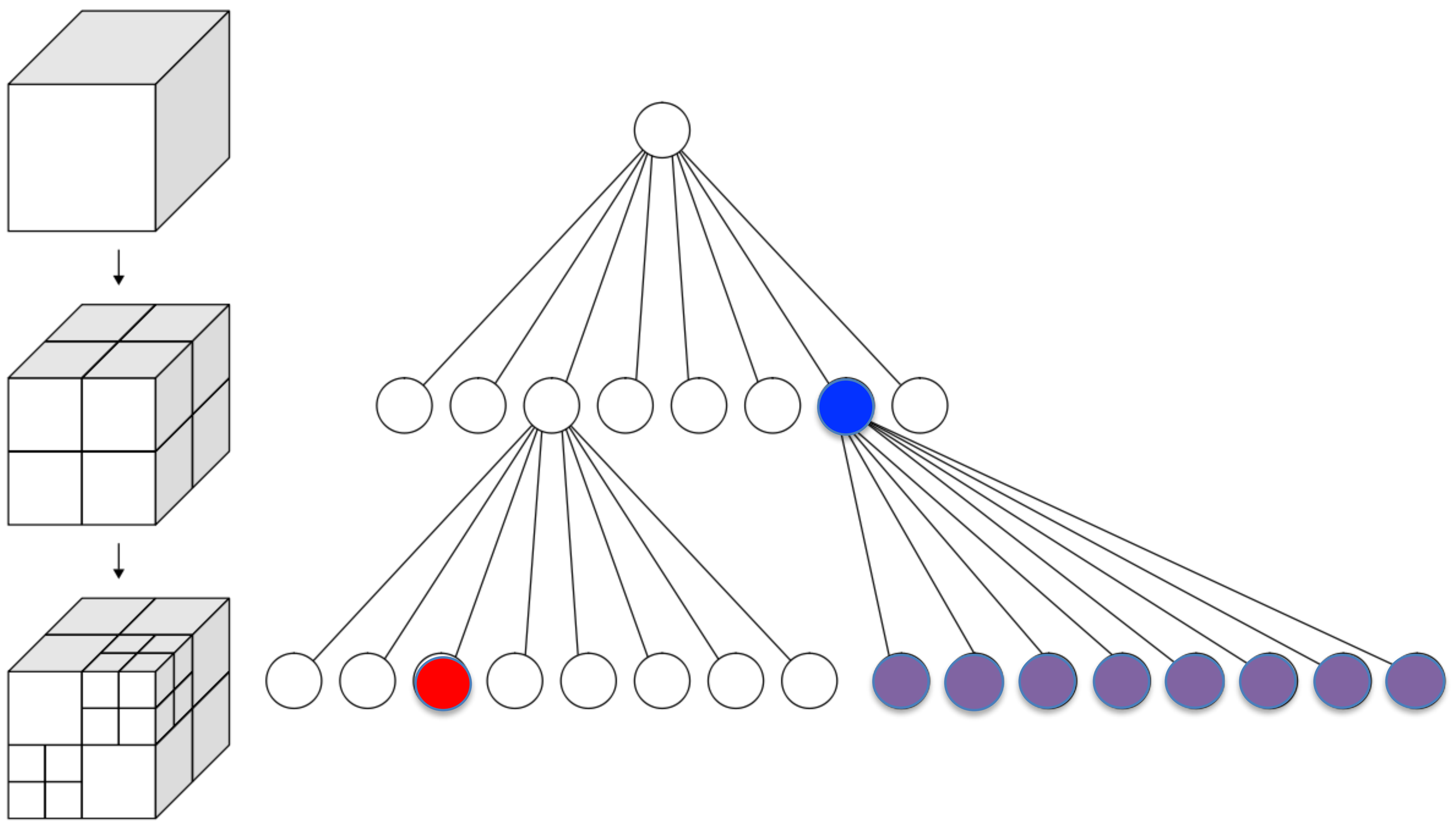
Jianqiao Liu, Milind Kulkarni
Purdue University

gpus!

- GPUs are an important component of modern supercomputers, and are becoming increasingly important to obtain peak performance
  - Blue Waters (2007) had 1 GPU (K20) for every 16 CPU cores
  - Summit (2018) has 1 GPU (Volta) for every 7 CPU cores
- ChaNGa, unsurprisingly, leverages GPUs for maximum performance
- But can we do better?

barnes-hut refresher

- Accelerate n-body codes by subdividing space into **octree**
- Compute forces on **red** bodies by traversing tree
- Approximate contribution from **purple** bodies by using summary information at **blue** node

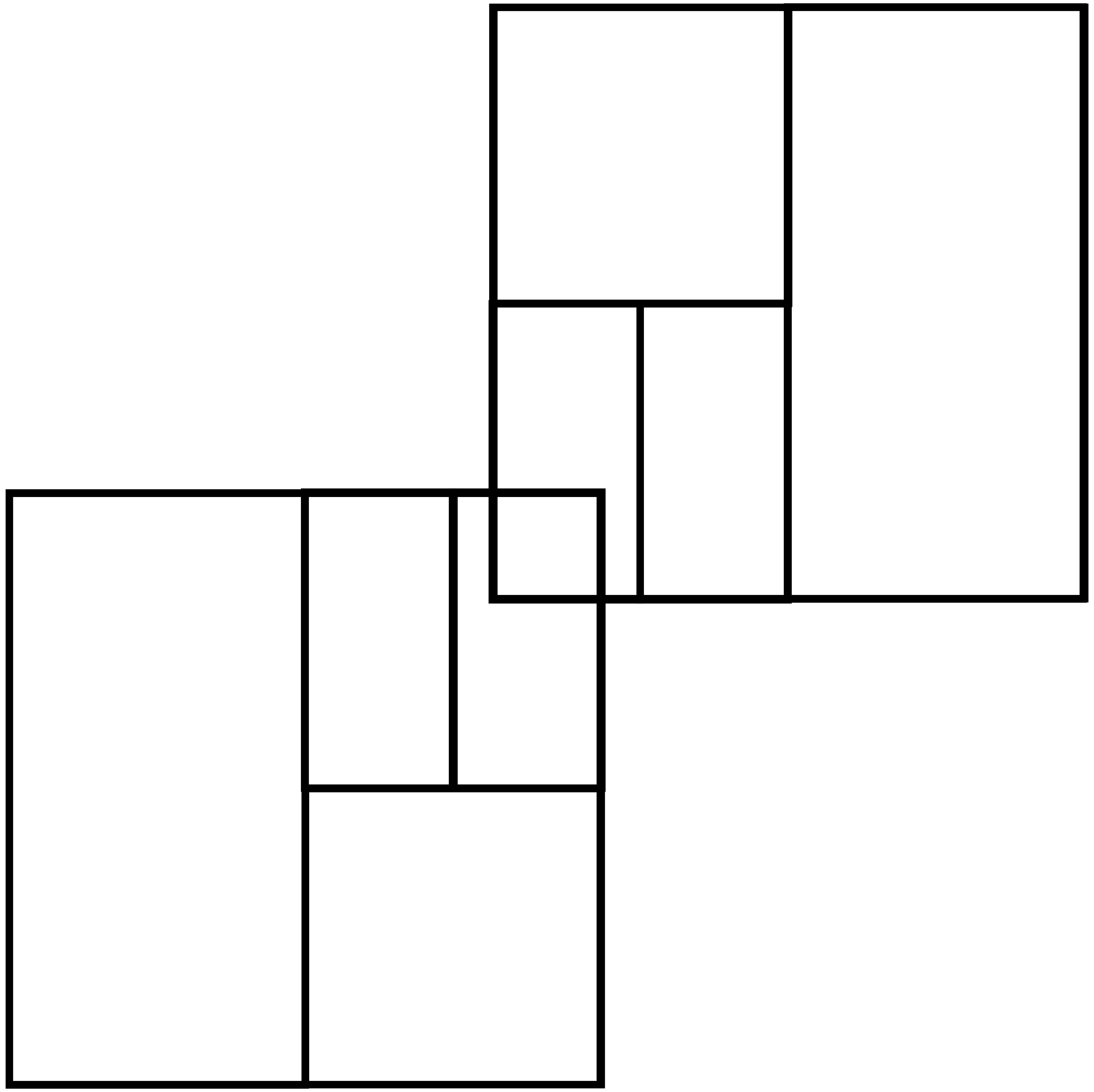


https://en.wikipedia.org/wiki/Octree

dual-tree approach

- Classical Barnes-Hut is a **single tree** approach: for each leaf node, traverse the tree
→ O(n log n) force computation, O(n log n) traversals

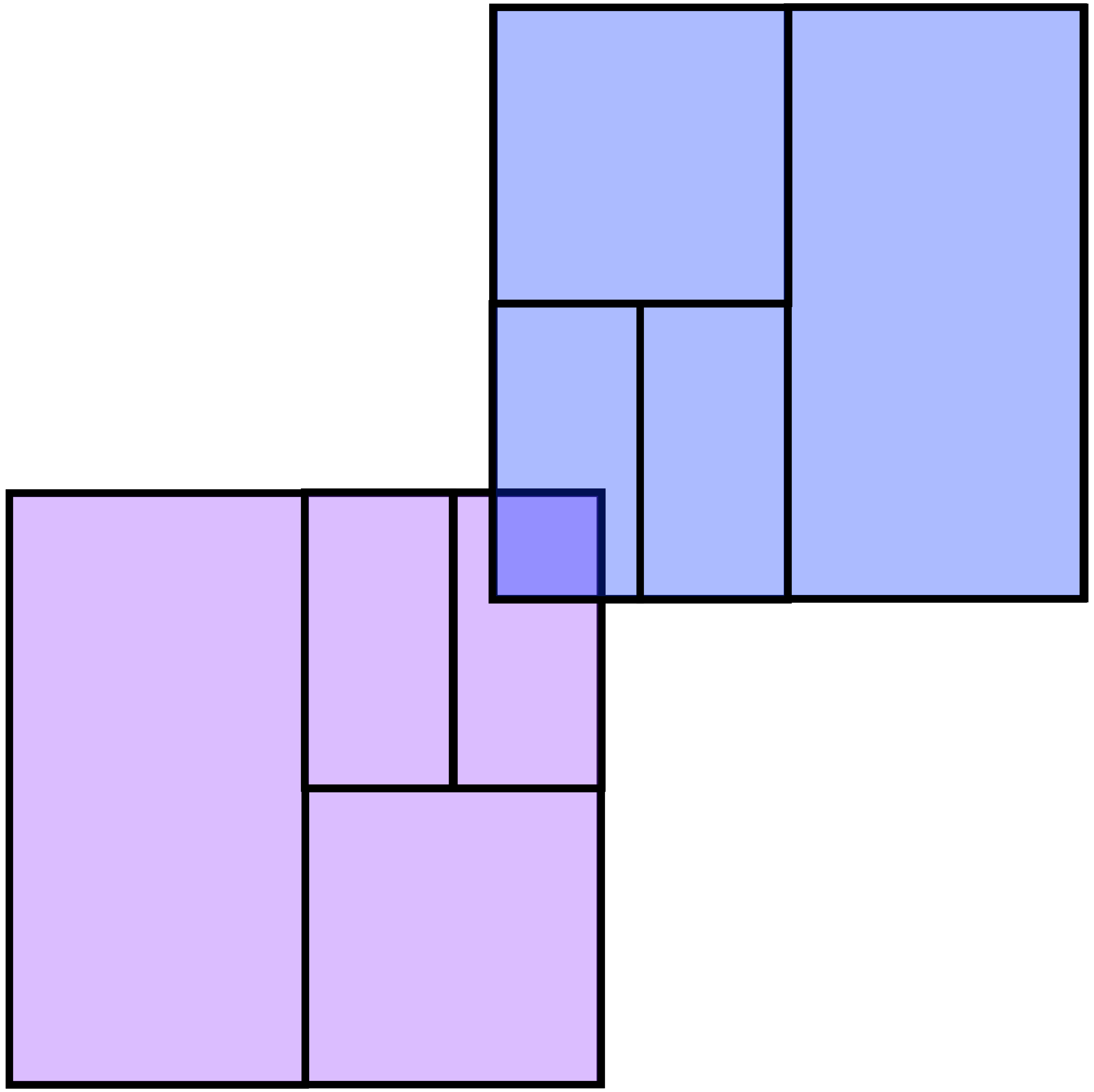
- Can also adopt a **dual tree** approach: for each *interior node* traverse the tree
→ O(n log n) force computation, **O(n) traversals**



dual-tree approach

- Classical Barnes-Hut is a **single tree** approach: for each leaf node, traverse the tree
  → O(n log n) force computation, O(n log n) traversals

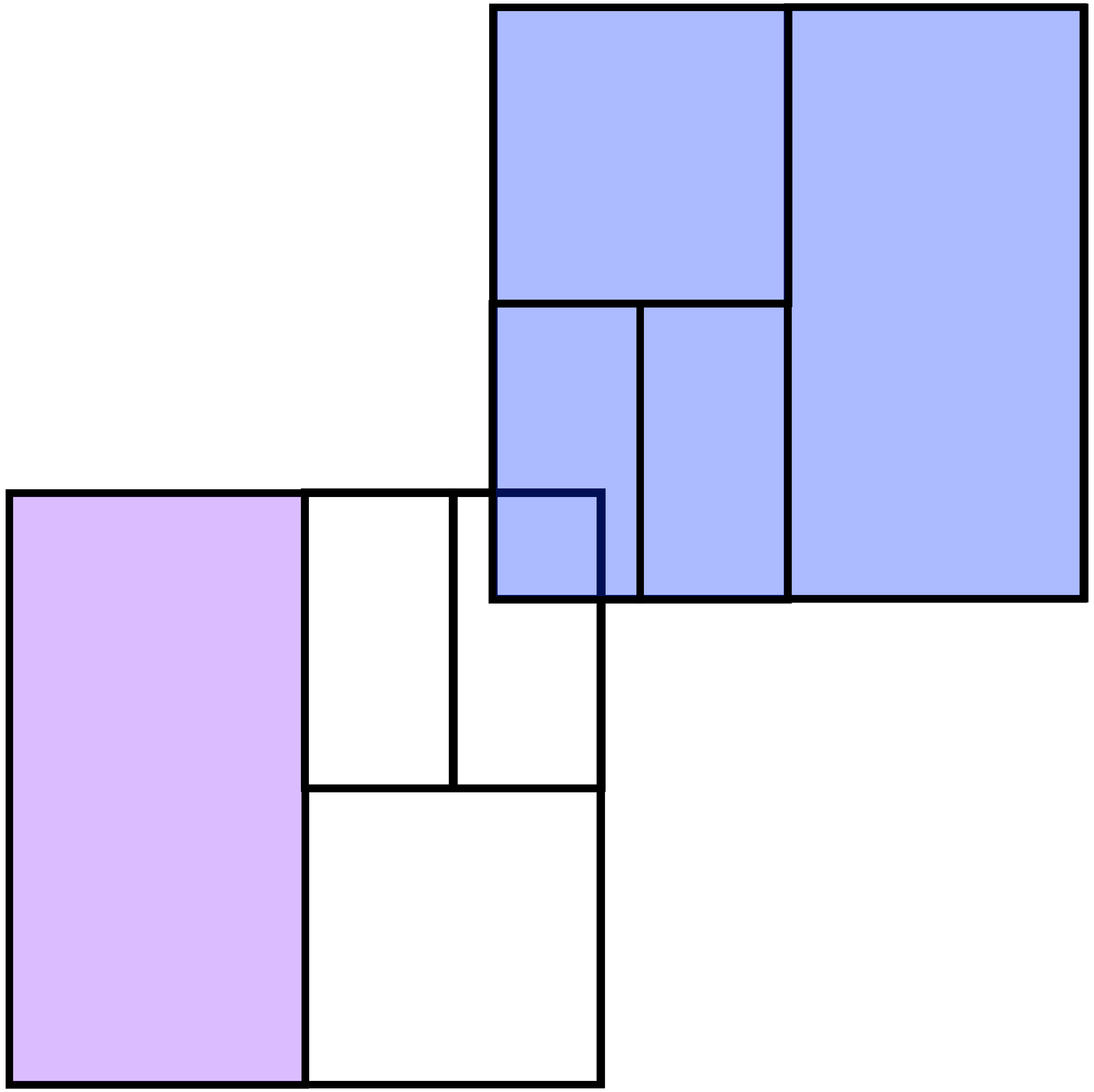
- Can also adopt a **dual tree** approach: for each *interior node* traverse the tree
  → O(n log n) force computation, **O(n) traversals**

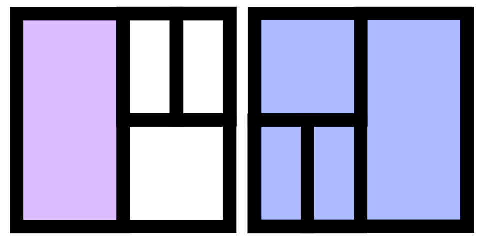


dual-tree approach

- Classical Barnes-Hut is a **single tree** approach: for each leaf node, traverse the tree → O(n log n) force computation, O(n log n) traversals

- Can also adopt a **dual tree** approach: for each *interior node* traverse the tree → O(n log n) force computation, **O(n) traversals**

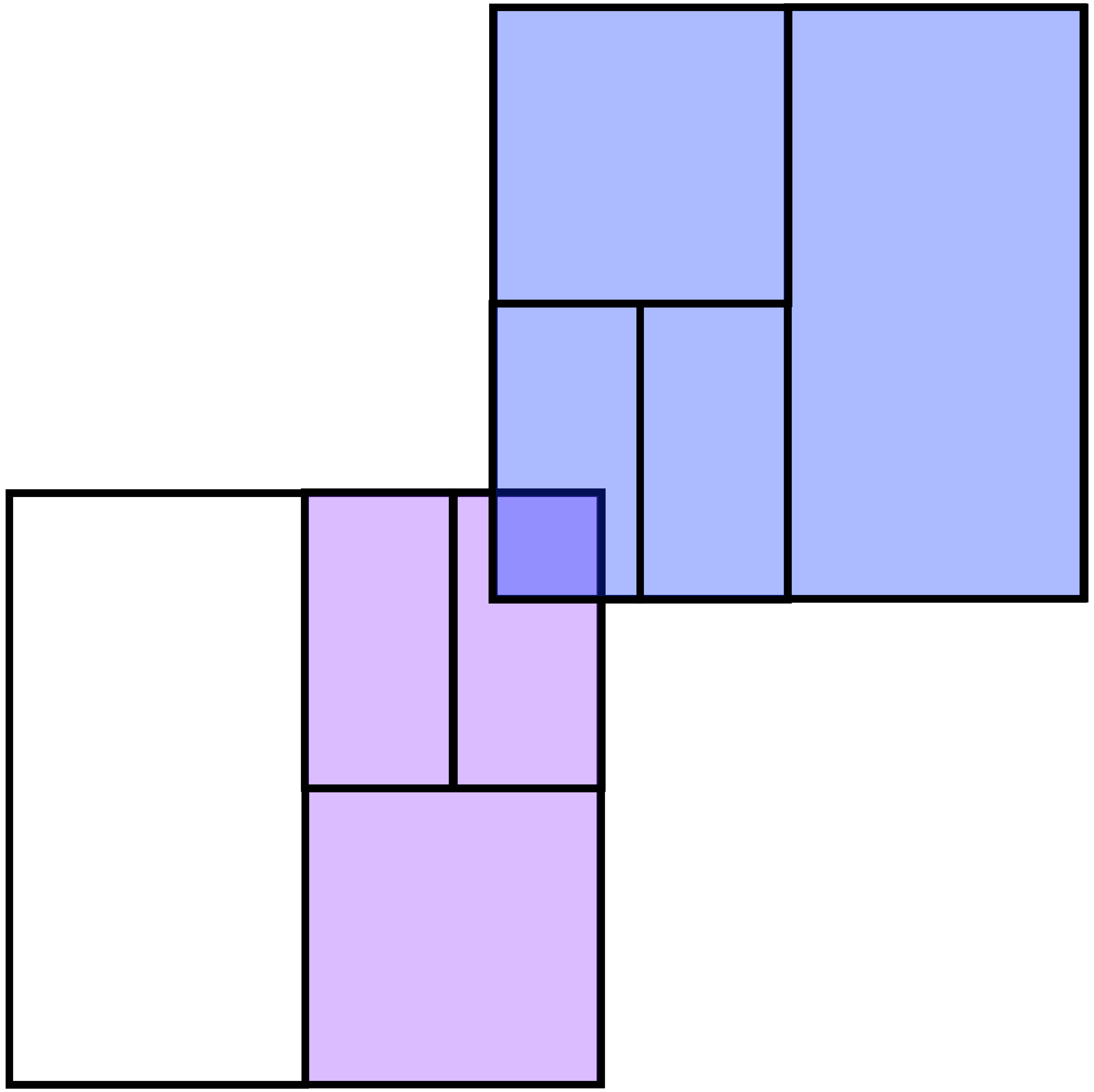


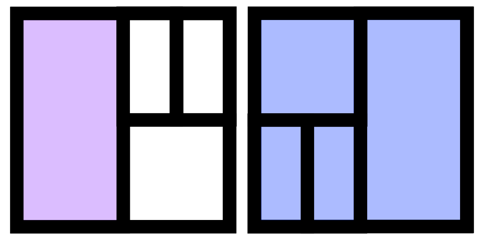


dual-tree approach

- Classical Barnes-Hut is a **single tree** approach: for each leaf node, traverse the tree
  → O(n log n) force computation, O(n log n) traversals

- Can also adopt a **dual tree** approach: for each *interior node* traverse the tree
  → O(n log n) force computation, **O(n) traversals**

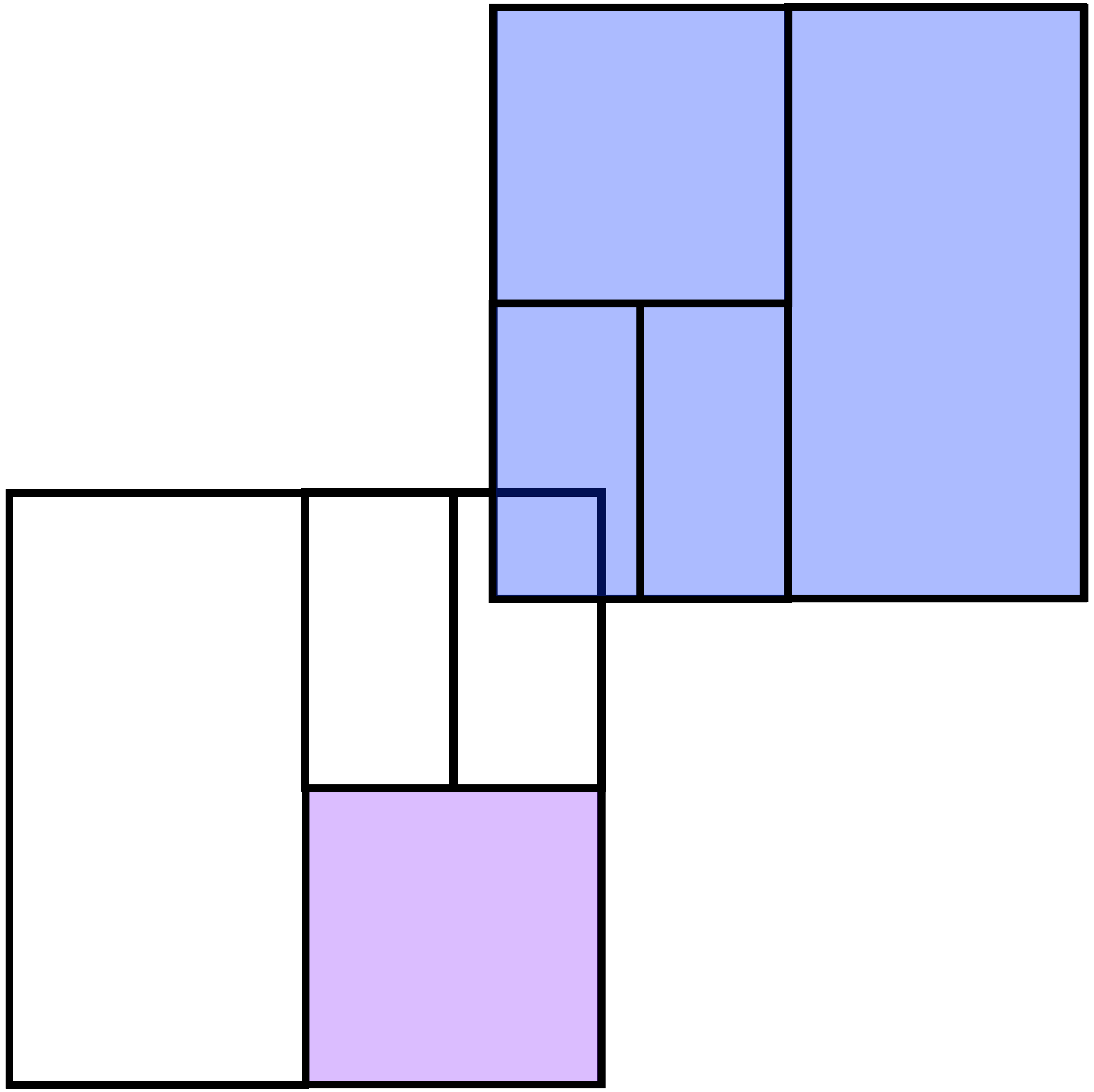


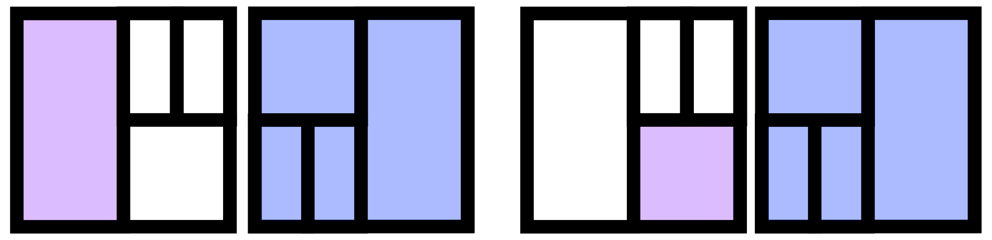


dual-tree approach

- Classical Barnes-Hut is a **single tree** approach: for each leaf node, traverse the tree
  → O(n log n) force computation, O(n log n) traversals

- Can also adopt a **dual tree** approach: for each *interior node* traverse the tree
  → O(n log n) force computation, **O(n) traversals**

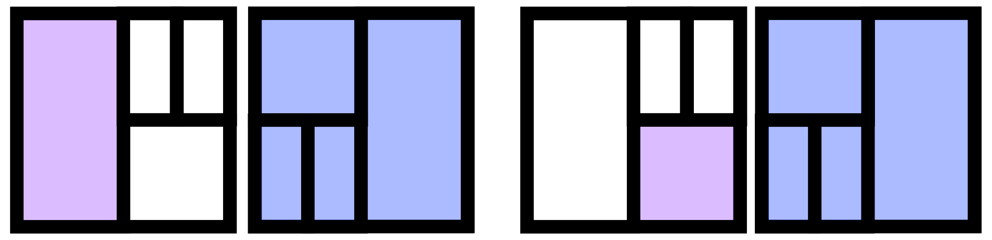


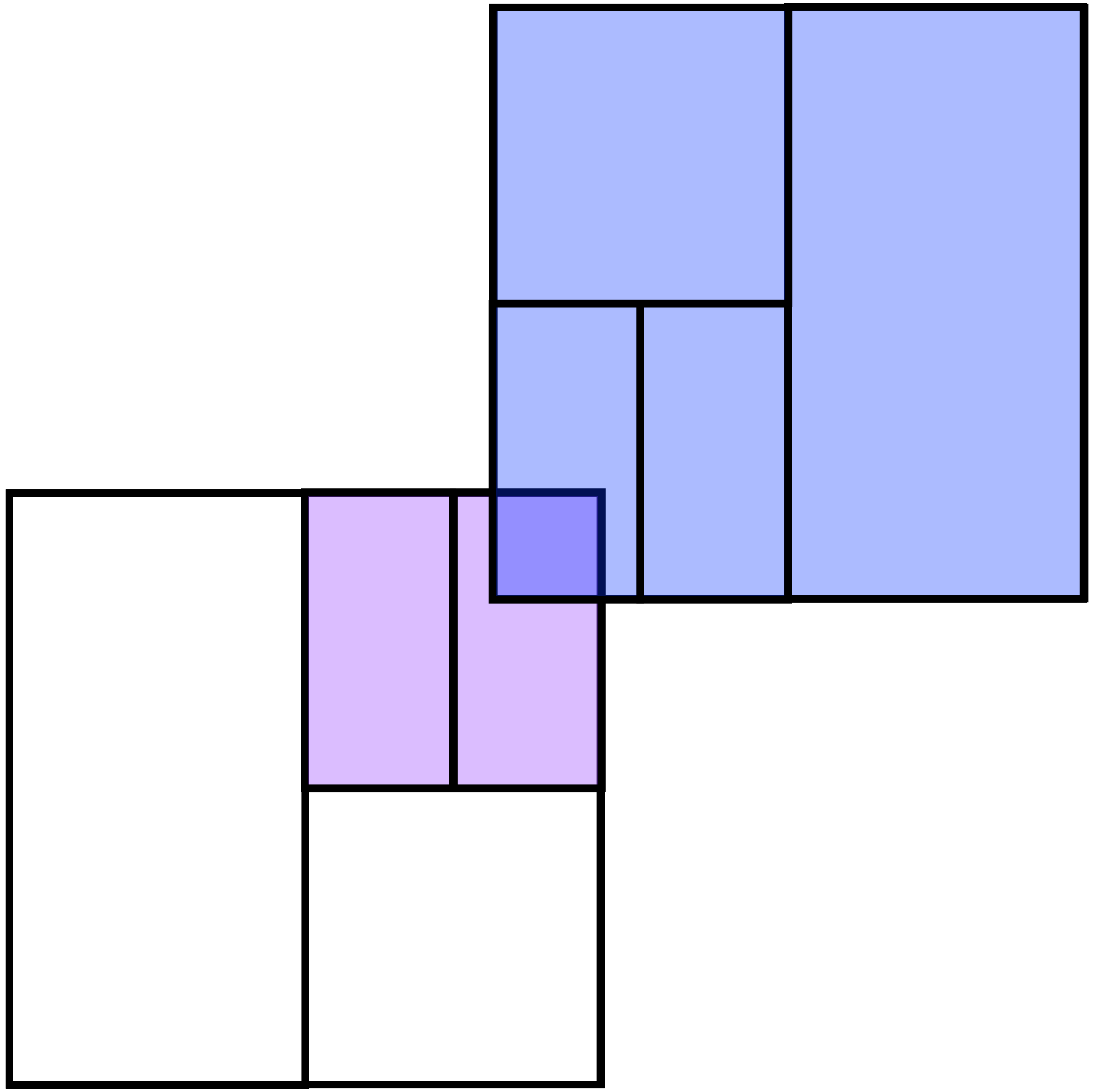


dual-tree approach

- Classical Barnes-Hut is a **single tree** approach: for each leaf node, traverse the tree
  → O(n log n) force computation, O(n log n) traversals

- Can also adopt a **dual tree** approach: for each *interior node* traverse the tree
  → O(n log n) force computation, **O(n) traversals**

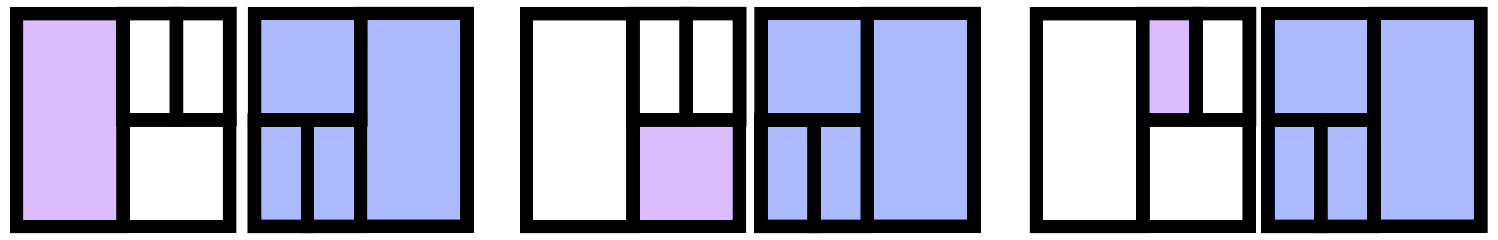


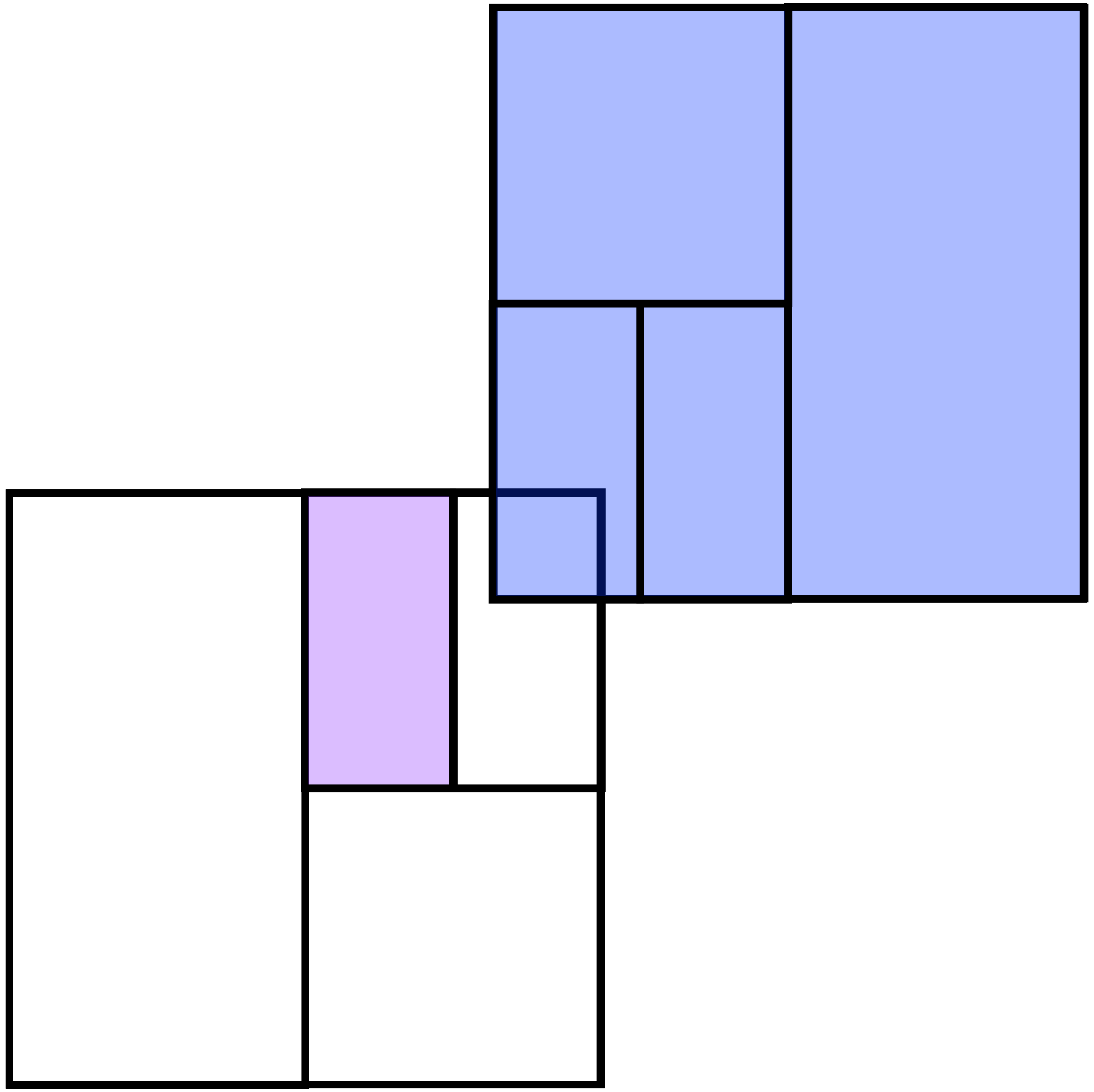


dual-tree approach

- Classical Barnes-Hut is a **single tree** approach: for each leaf node, traverse the tree
  → O(n log n) force computation, O(n log n) traversals

- Can also adopt a **dual tree** approach: for each *interior node* traverse the tree
  → O(n log n) force computation, **O(n) traversals**

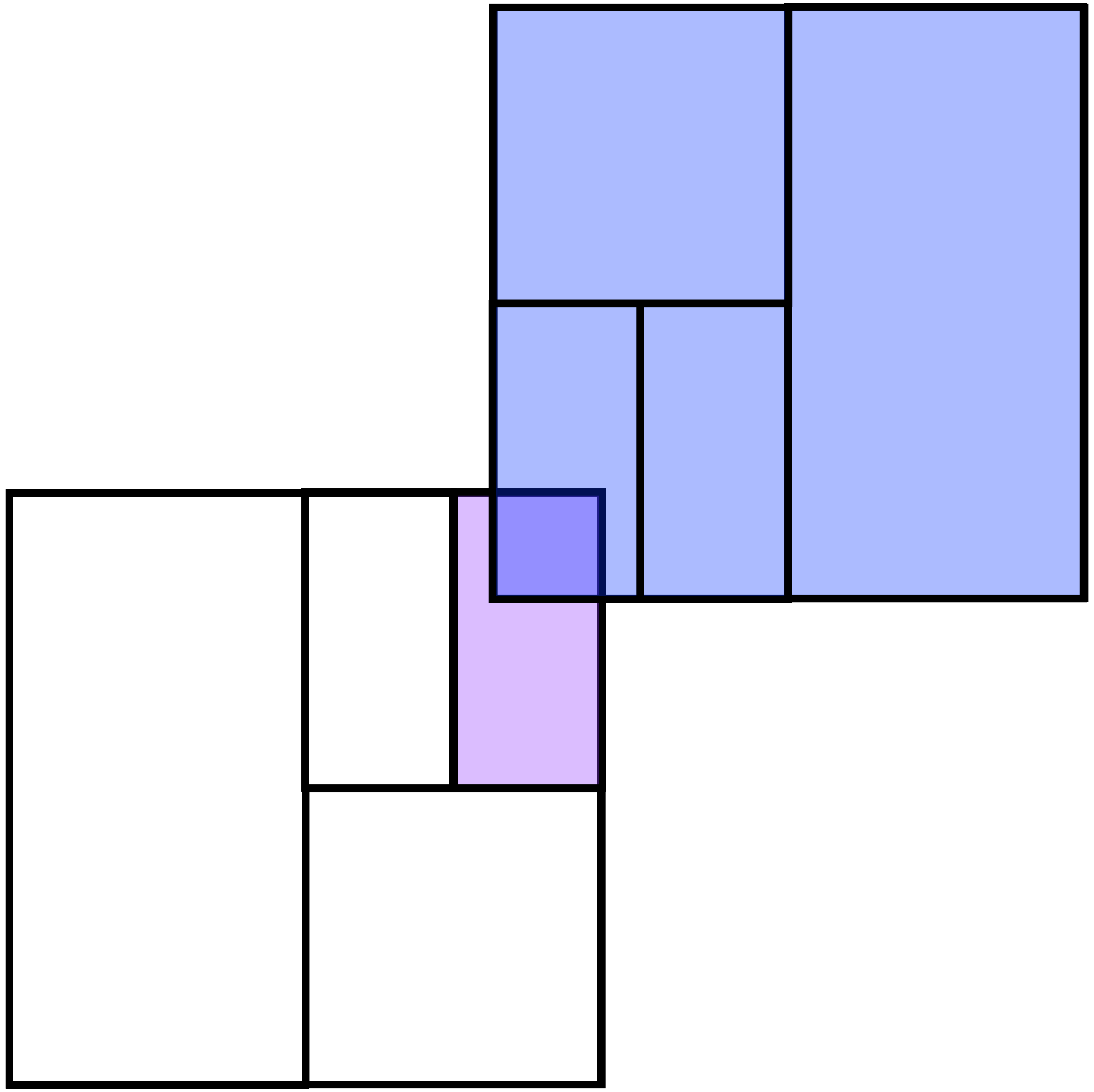


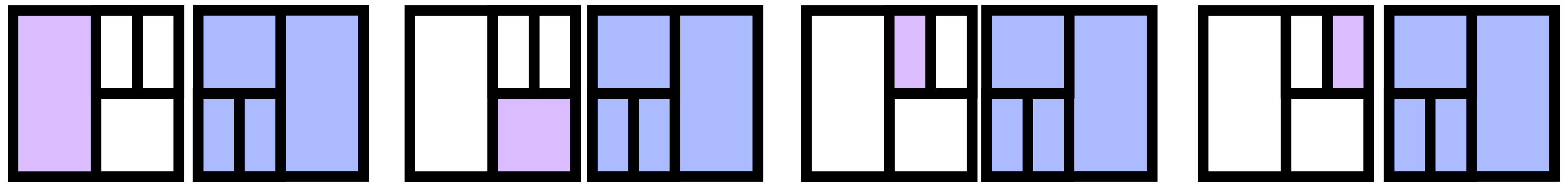


dual-tree approach

- Classical Barnes-Hut is a **single tree** approach: for each leaf node, traverse the tree
→ O(n log n) force computation, O(n log n) traversals

- Can also adopt a **dual tree** approach: for each *interior node* traverse the tree
→ O(n log n) force computation, **O(n) traversals**



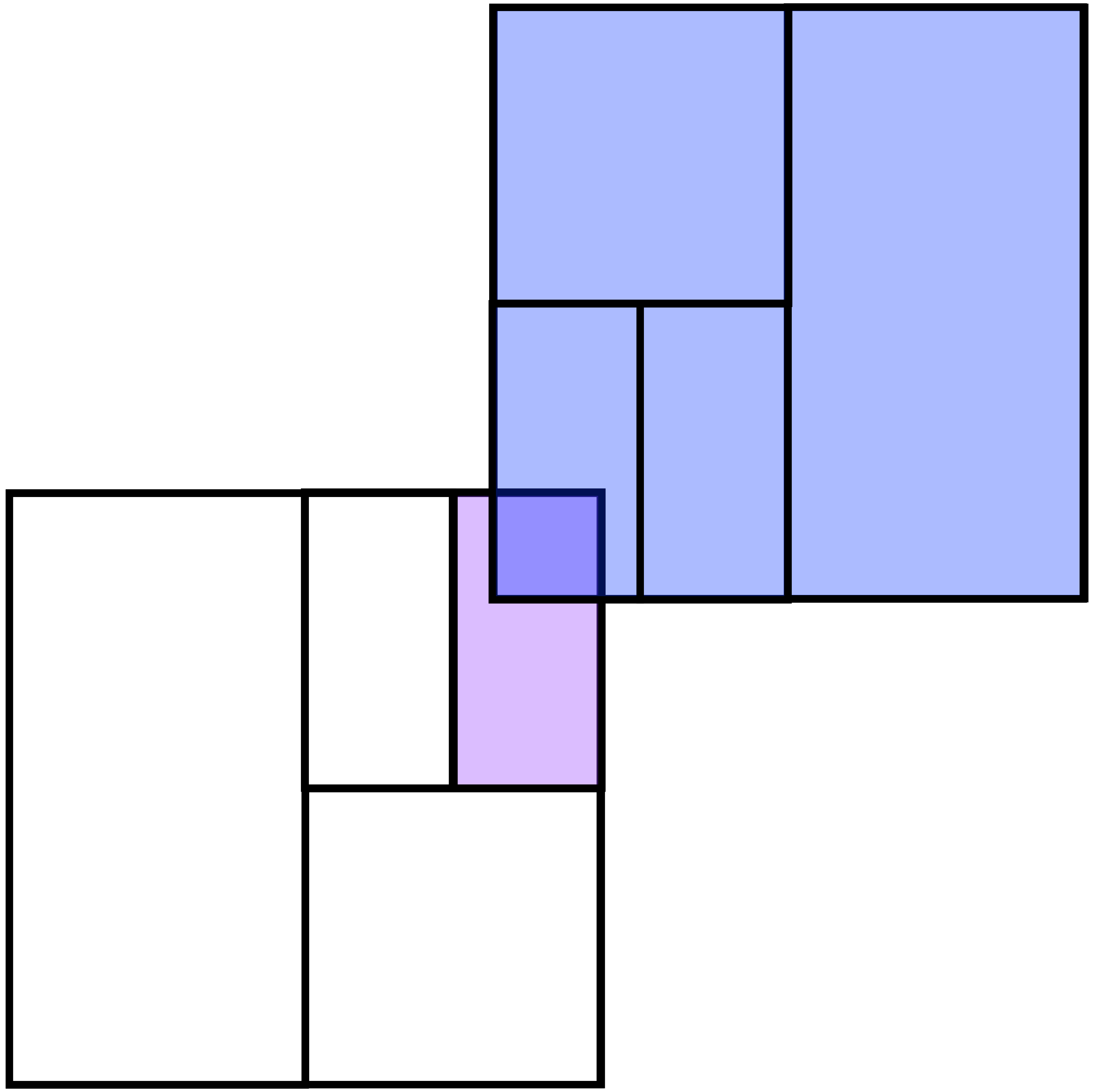


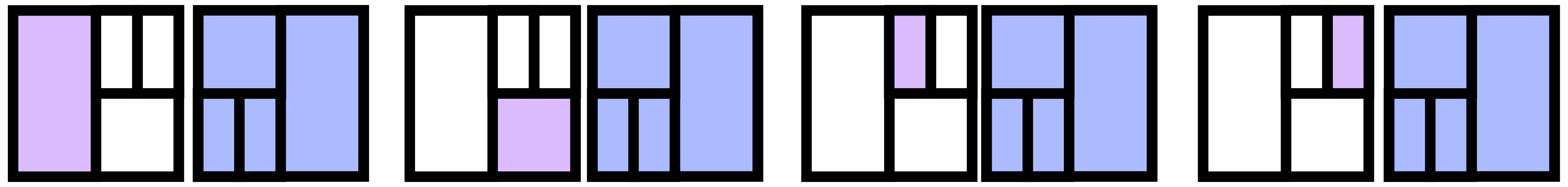
moving to gpus

- Key challenge for Barnes-Hut (and other tree traversals): *significant irregularity* so does not map well to GPUs

- Existing approach in ChaNGa: CPU computes **interaction lists** and sends to GPU for computation

- Goal: put whole computation on GPU

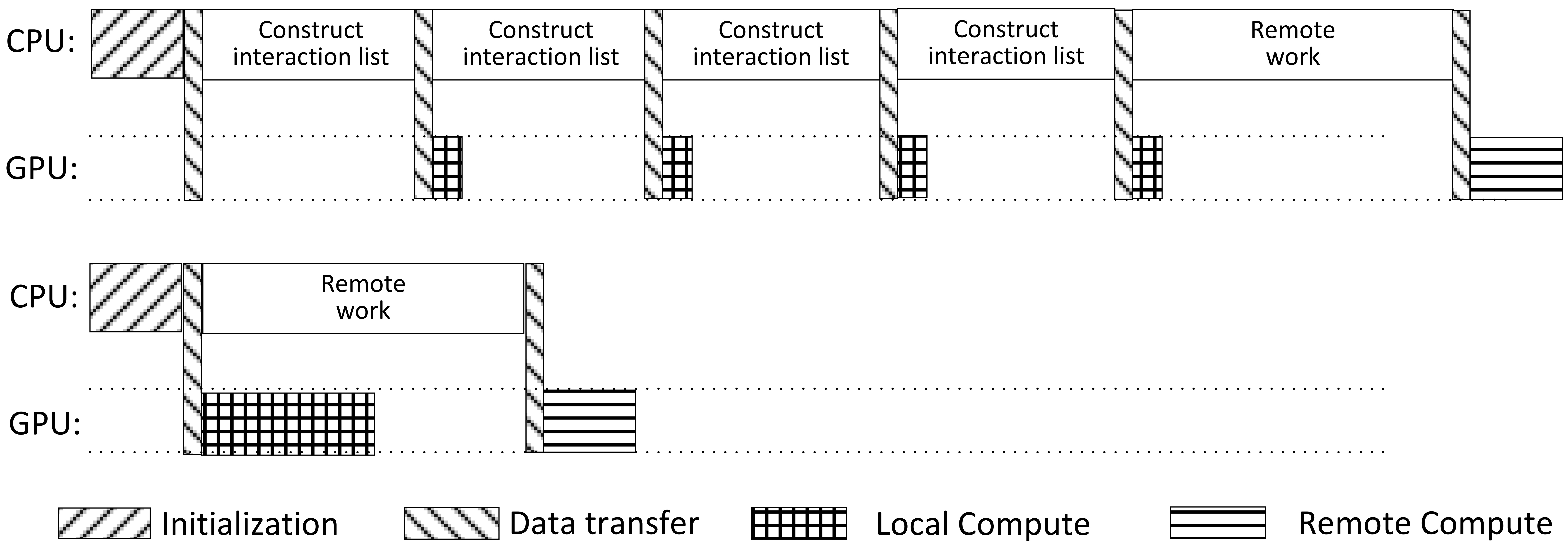




return to single tree

- Putting dual-tree computation on GPUs is challenging
  - Asymptotic complexity wins come from sacrificing parallelism during traversal to do cell-cell interactions, but GPUs need parallelism to keep them busy
- Instead, return to single-tree computation for local tree walks
  - Adopt many existing effective implementation tricks [Burtscher and Pingali; Goldfarb et al.; Liu et al.]
  - Tweak **open criterion** (traversal conditions) to work better for single-tree traversals

full single-tree walk on gpu



✓ Less CPU/GPU communication

✓ No latency while waiting for CPU to compute interaction lists

✓ Free up CPU to do other computations (e.g., remote tree walks)

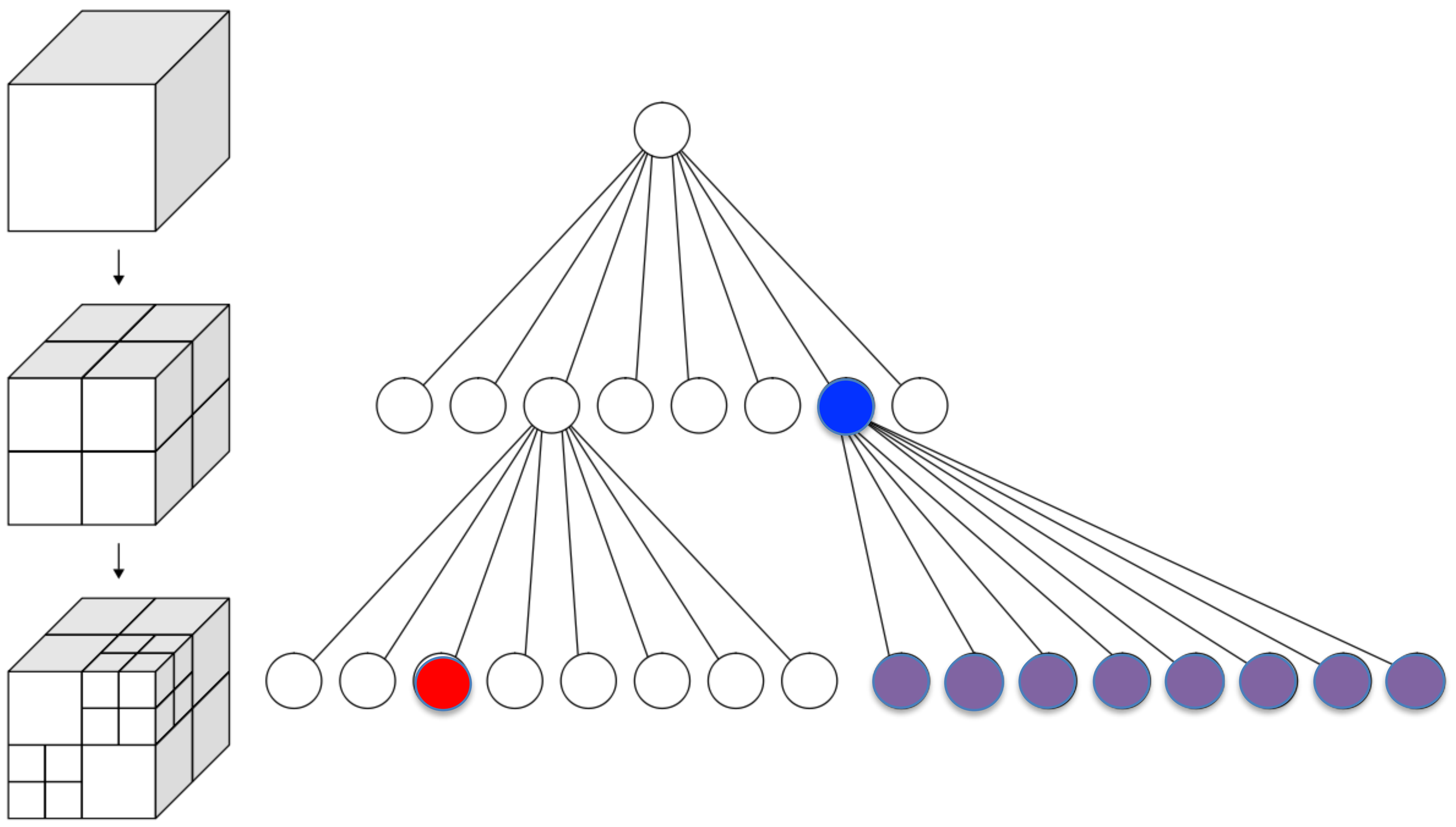
✗ Loses asymptotic complexity (back to O(n log n) traversals) but OK for local tree walks

results

P100 Speed test (in seconds)									
Configuration	bucket_size	Original ChaNGa		new ChaNGa					
		32	64	32		64		Average Speedup
		Runtime(s)	Runtime(s)	Runtime(s)	Speedup	Runtime(s)	Speedup	
1 node, 1 process per node	lambs, 3M, theta=0.6	9.58	5.10	1.06	9.01x	0.85	6.01x	**8.25x**
	lambb, 80M, theta=0.6	359.67	189.29	31.85	**11.29x**	26.01	7.28x	
	dwf1, 5M, theta=0.7	16.89	9.16	1.71	9.86x	1.40	6.54x	
	dwf1.6144, 50M, theta=0.7	194.84	103.93	19.69	9.90x	16.95	6.13x	
1 node, 4 processes per node	lambs, 3M, theta=0.6	3.08	1.66	1.22	2.53x	0.89	1.88x	2.13x
	lambb, 80M, theta=0.6	101.22	54.38	29.55	3.43x	23.18	2.35x	
	dwf1, 5M, theta=0.7	6.26	3.42	3.15	1.99x	1.95	1.76x	
	dwf1.6144, 50M, theta=0.7	67.52	37.07	40.73	1.66x	25.20	1.47x	
1 node, 8 processes per node	lambs, 3M, theta=0.6	1.89	1.07	1.05	1.80x	0.77	1.38x	1.55x
	lambb, 80M, theta=0.6	55.16	30.94	24.07	2.29x	19.83	1.56x	
	dwf1, 5M, theta=0.7	3.49	1.90	2.40	1.45x	1.55	1.22x	
	dwf1.6144, 50M, theta=0.7	38.40	20.71	26.75	1.44x	16.32	1.27x	
8 nodes, 1 process per node	lambs, 3M, theta=0.6	1.92	1.04	1.07	1.80x	0.78	1.33x	**1.80x**
	lambb, 80M, theta=0.6	49.49	27.47	15.41	3.21x	10.41	2.64x	
	dwf1, 5M, theta=0.7	3.51	1.90	2.37	1.48x	1.55	1.22x	
	dwf1.6144, 50M, theta=0.7	39.10	20.67	27.36	1.43x	16.56	1.25x	
8 nodes, 4 processes per node	lambs, 3M, theta=0.6	1.50	0.88	0.90	1.67x	0.67	1.31x	1.53x
	lambb, 80M, theta=0.6	41.11	22.13	16.94	2.43x	13.36	1.66x	
	dwf1, 5M, theta=0.7	2.27	1.37	1.68	1.35x	1.20	1.14x	
	dwf1.6144, 50M, theta=0.7	22.93	12.46	14.92	1.54x	10.49	1.19x	
8 nodes, 8 processes per node	lambs, 3M, theta=0.6	0.80	0.57	0.57	1.39x	0.45	1.27x	**1.40x**
	lambb, 80M, theta=0.6	21.55	11.70	10.15	2.12x	7.58	1.54x	
	dwf1, 5M, theta=0.7	1.28	0.82	1.05	1.22x	0.74	1.10x	
	dwf1.6144, 50M, theta=0.7	11.80	6.50	8.66	1.36x	5.43	1.20x	

summary

- GPUs are ill-suited for dual-tree walks, so ChaNGa didn't use the GPU for tree walks
- Switch local tree walk to classical single-tree walk and put it on GPU
- Lose in asymptotic complexity, but massive win in parallelism
- Work is in ChaNGa main branch as of August 2018



https://en.wikipedia.org/wiki/Octree