RT 08

Symposium on Interactive Ray Tracing 2008
Los Angeles, California

Efficient Clustered BVH Update Algorithm for Highly-Dynamic Models

Kirill Garanzha

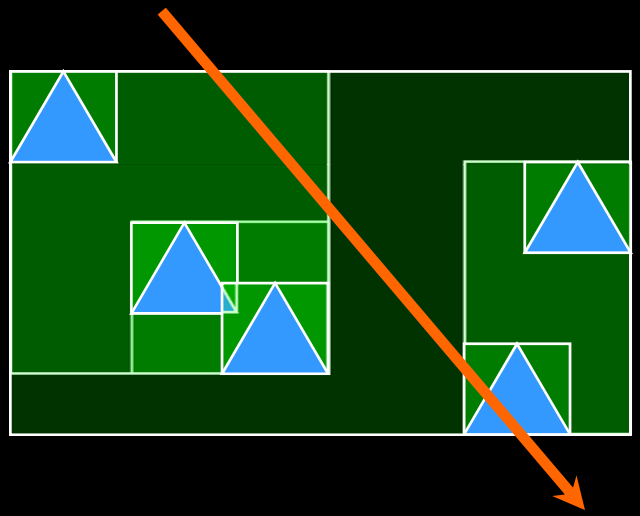
Department of Software for Computers
Bauman Moscow State Technical University, Russia

BVH summary

RT 08

BVH is the tree that encloses scene objects

- **Advantages:**
  - Fast refitting
  - Predictable memory consumption
  - Fastest empty space passing for SAH trees



- **Disadvantages:**
  - Not ordered traversal of the ray in the space
  - Tree-quality degradation after refitting
  - Not very fast SAH-based build

Dynamic BVH related *RT 08*

- **Fast BVH build assume some tree-quality degradation for ray tracing**
  - BIH [Wächter EGSR06]
  - Build from hierarchy [Hunt RT07]
  - SAH binning [Wald RT07]
  - Selective restructuring [Yoon EGSR07]

Dynamic BVH related *RT 08*

- Fast BVH build assume some tree-quality degradation for ray tracing

- In general these approaches apart from “Selective restructuring” produce splits for every BVH-node in every frame – brute force

Dynamic BVH related

RT 08

How to get rid of brute force?

- Lazy build
- Selective restructuring
- Avoid full reconstruction of nodes without strong dynamism under them

Idea

RT 08

- To measure cheaply the dynamism under node while refitting the BVH
- To apply the cheapest update technique for the node that saves good SAH cost:
  1. Leave it refitted
  2. Relocate BVH-cluster in proper position of the tree
  3. Rebuild the structure under node by exploiting SAH-binning, existing hierarchy, degree of dynamism

Dynamism detection

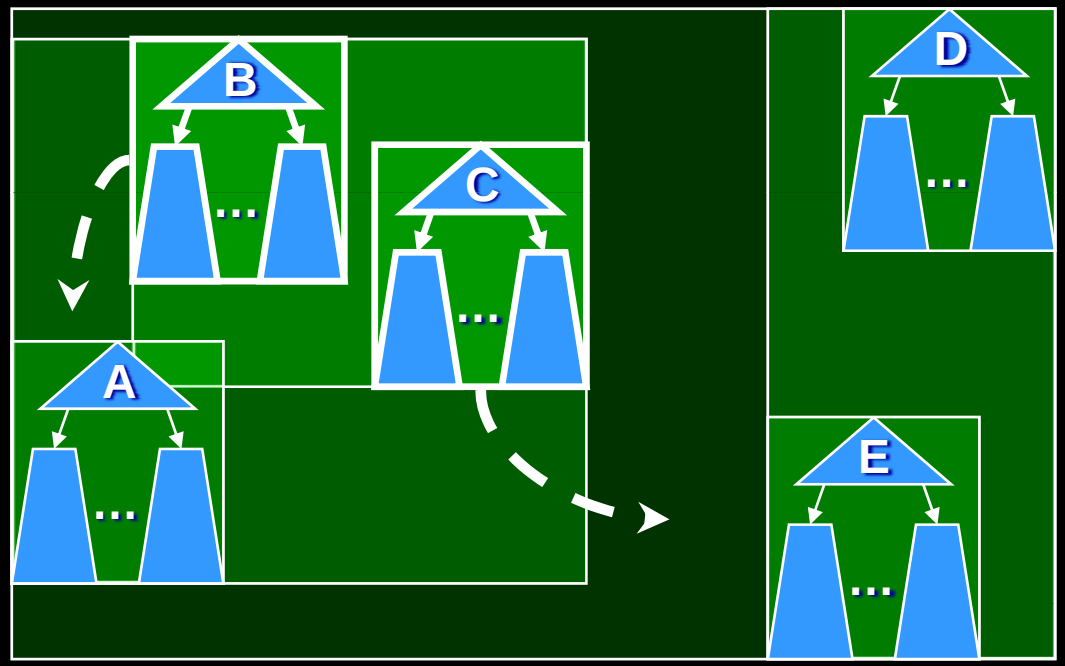
RT 08

The type of dynamism is detected in the bottom-up refitting process for every affected node:

- Migrating node represents a BVH-cluster of coherently moving triangles
- Exploded node represents a cluster of strong triangle explosions and growing overlaps

Detect migrating node *RT 08*

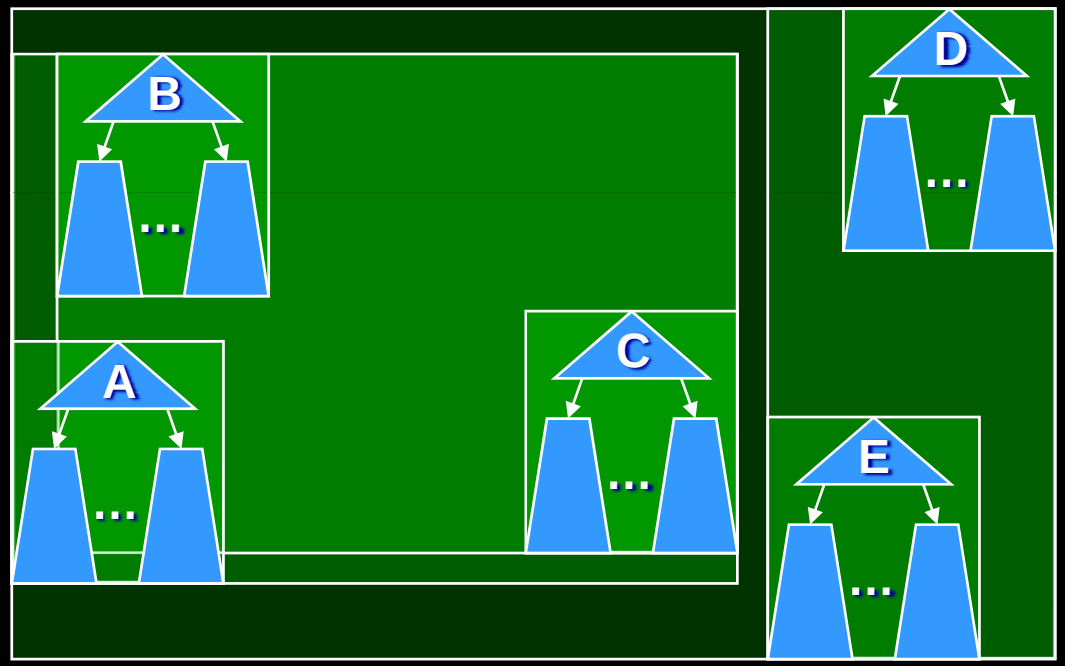
Example:



Nodes B and C are moving. They represent clusters of coherently moving triangles

Detect migrating node *RT 08*

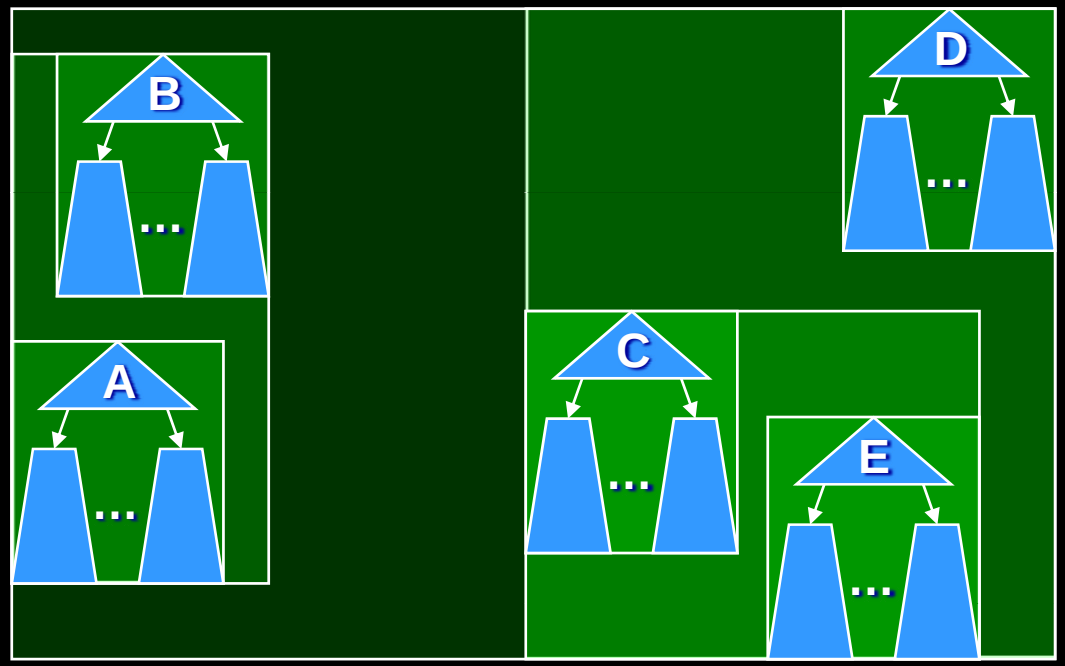
Example:



The SAH cost of the node (B+C) is improved.
But nodes B and C should be united with others

Detect migrating node *RT 08*

Example:



Like now…

Detect migrating node *RT 08*

Formula:

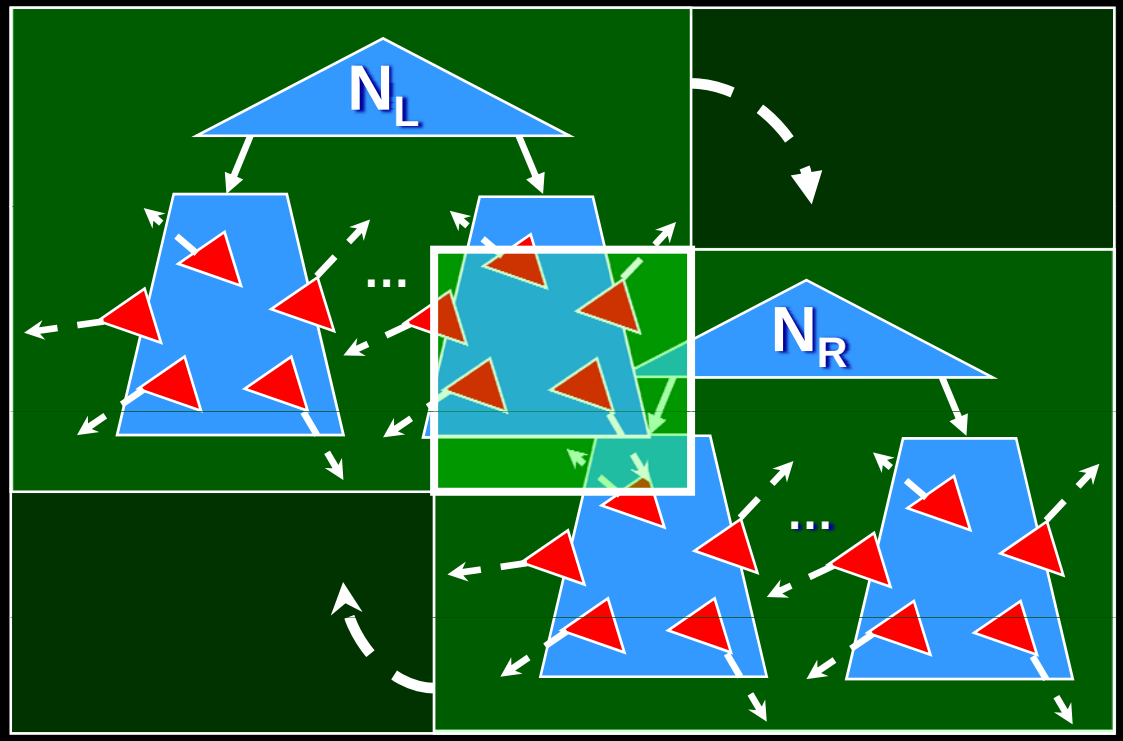
if $$\frac{SAMC_i}{Sav_{all}} < 1$$

then N_L **and** N_R **represent migrating clusters and should be relocated**

- N_L **and** N_R **are children of** N
- dM **is the predefined threshold**

Detect exploded node *RT 08*

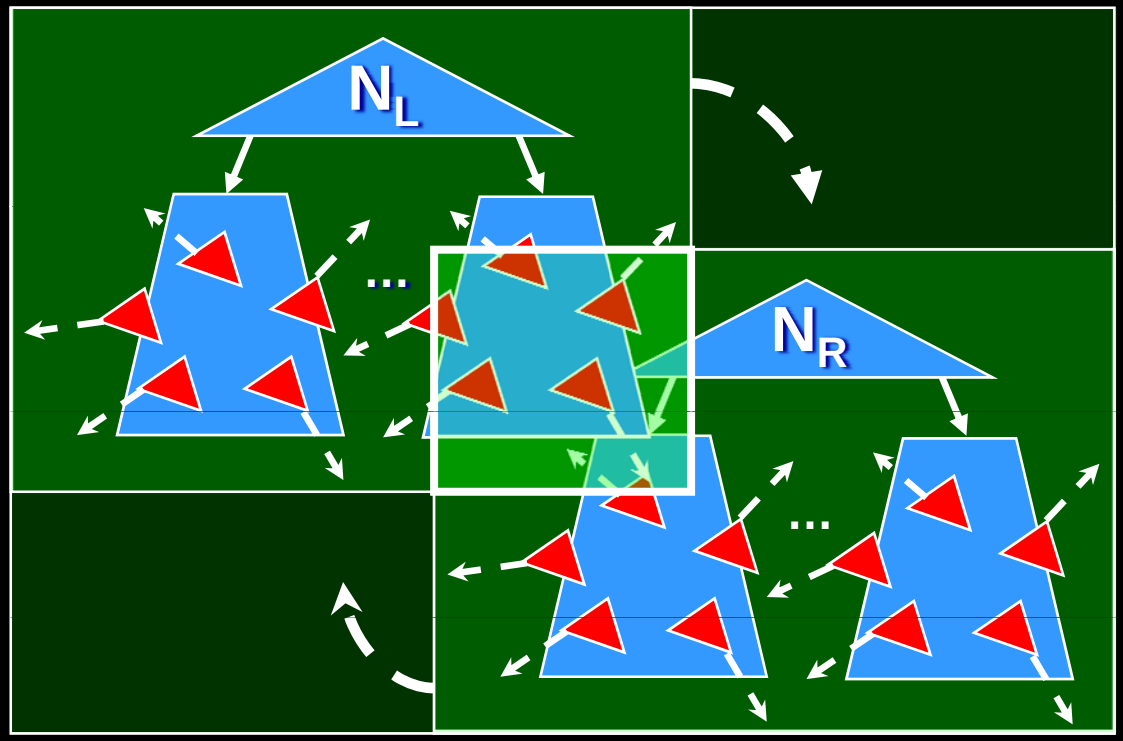
Example:



The overlap between N_L and N_R is likely to grow if SAH cost of N is increasing

Detect exploded node *RT 08*

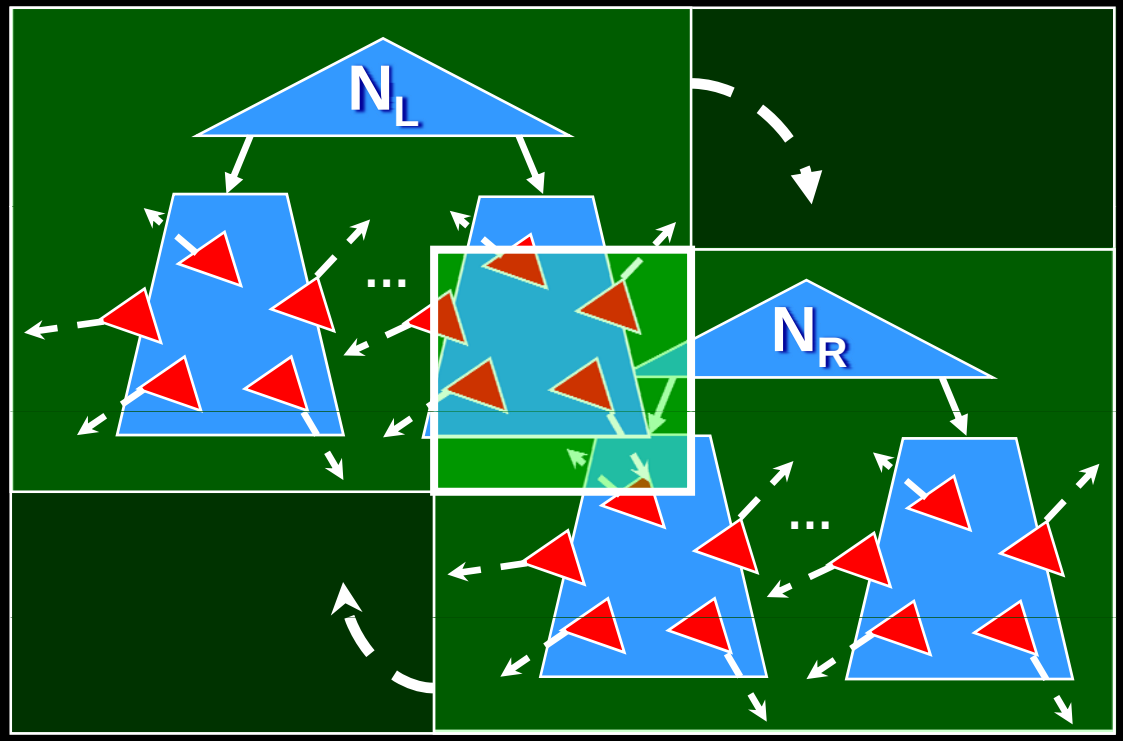
Example:



N is worth to rebuild if there is a big % of <u>broken nodes</u> under it

Detect exploded node *RT 08*

Example:



<u>Broken node</u> – migrating or exploded. **Broken count** measures the dynamism under node

Detect exploded node *RT 08*

Formula:

if

- There is big overlap between N_L and N_R
- There is big % of broken nodes under N

then The cluster with root N should be restructured

- N_L and N_R are children of N
- dE is the predefined threshold

Refitting phase

RT 08

The bottom-up refitting phase:

1. Updates BVs
2. Detects the dynamism
3. Accumulates broken counts

At the output it produces 2 arrays with migrating and exploded nodes

Independent clusters

RT 08

How to obtain a lot of smaller exploded clusters to perform independent rebuilds on them?

Independent clusters

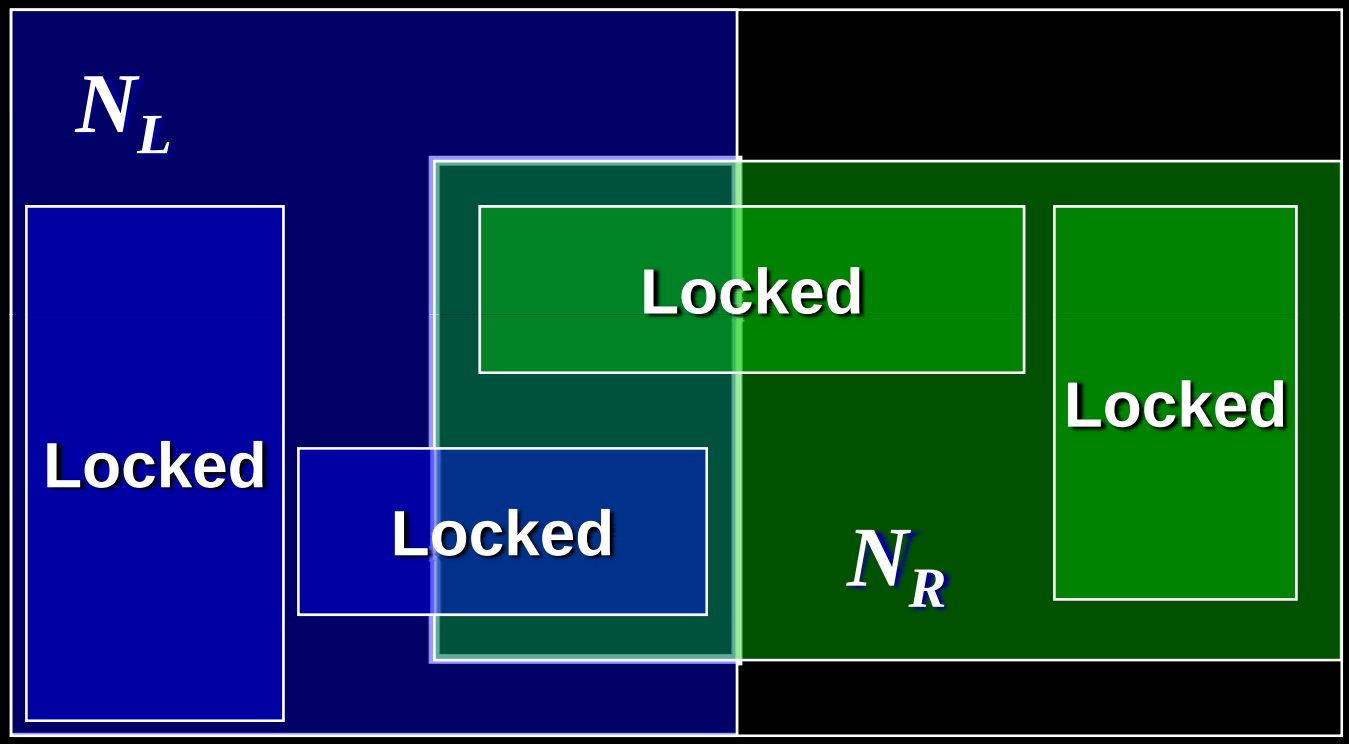
RT 08

When exploded node N is detected:

- Lock it. N will be considered as atomic for rebuild process on a cluster above it.
- Unlock all exploded descendants of N that intersect the BV of the overlap between N_L and N_R

Independent clusters

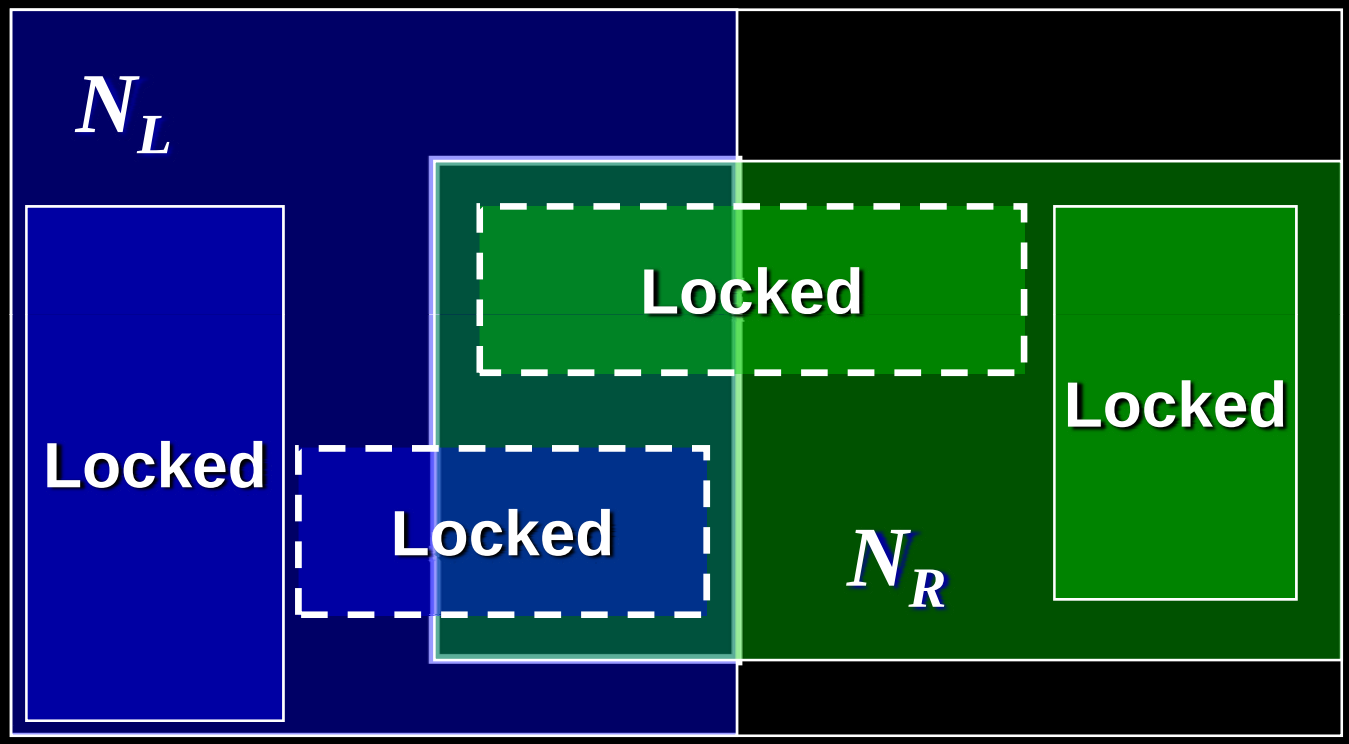
RT 08



Locked exploded descendants will be rebuilt independently and will be considered as atomic for some rebuild process above them

Independent clusters

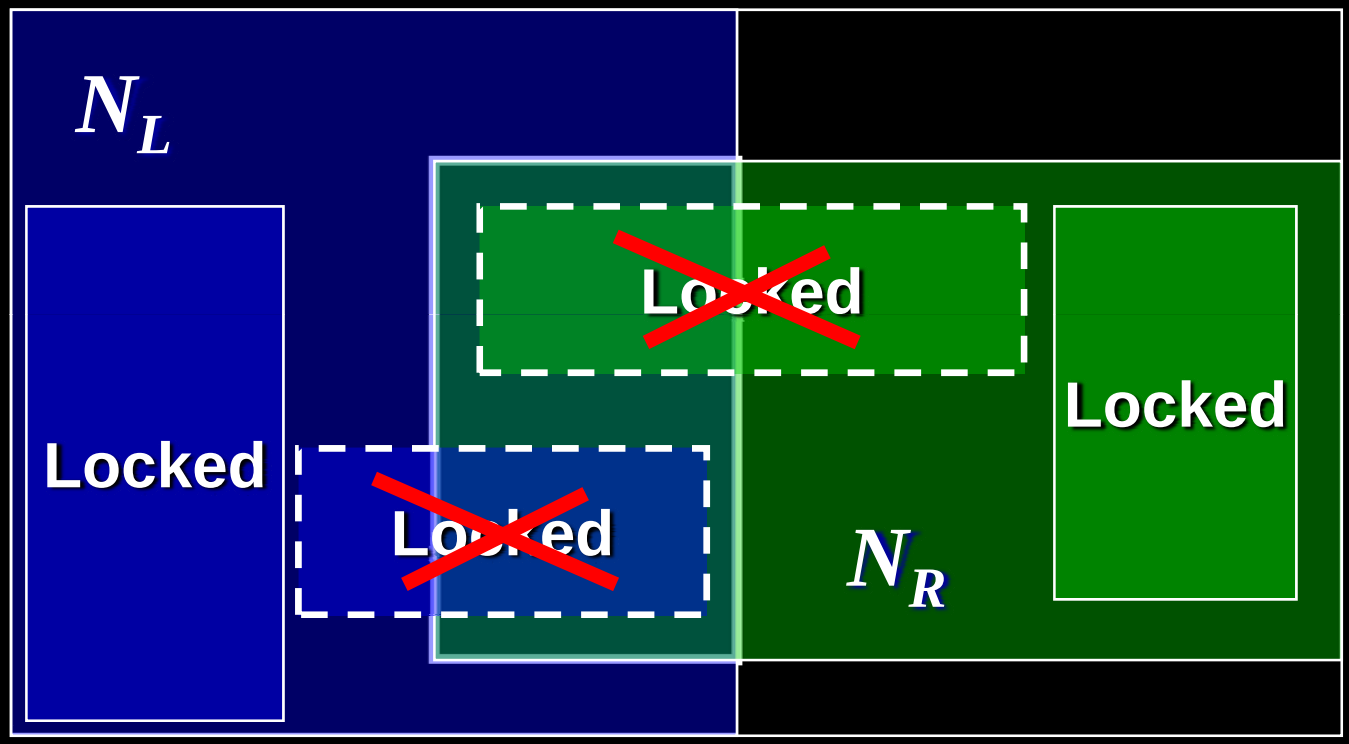
RT 08



But if some of them intersect the BV of the overlap within some ancestor N then they create obstacles in the process eliminating the overlap

Independent clusters

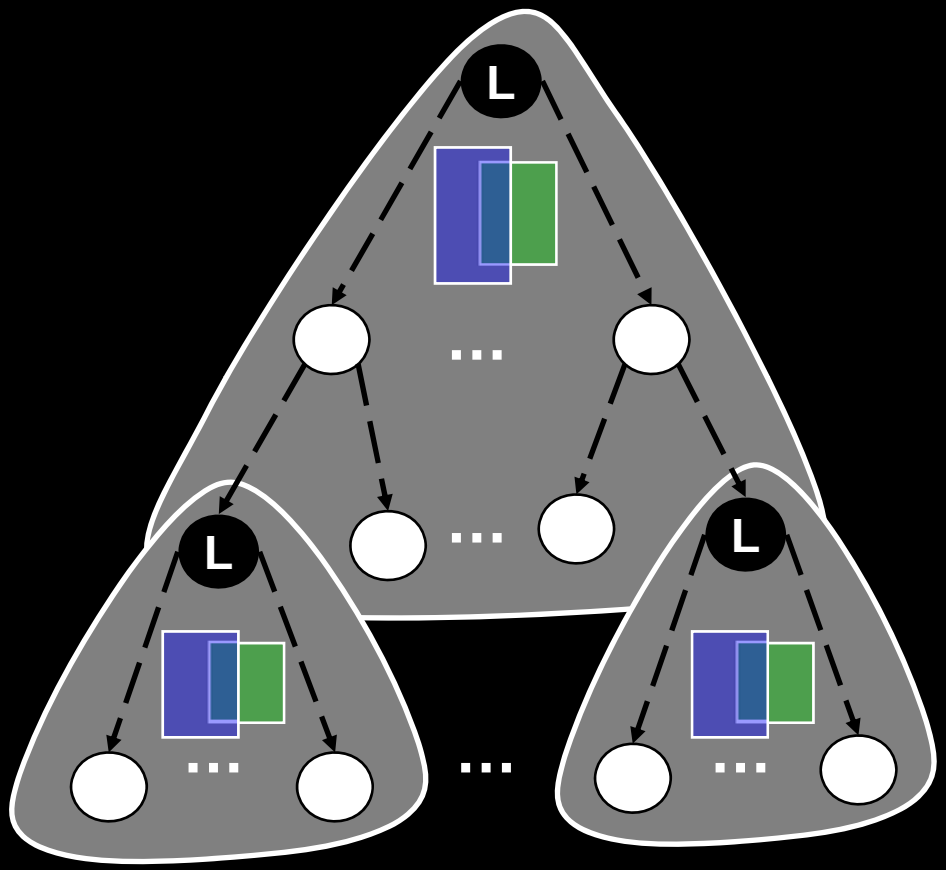
RT 08



So they should be unlocked

Independent clusters

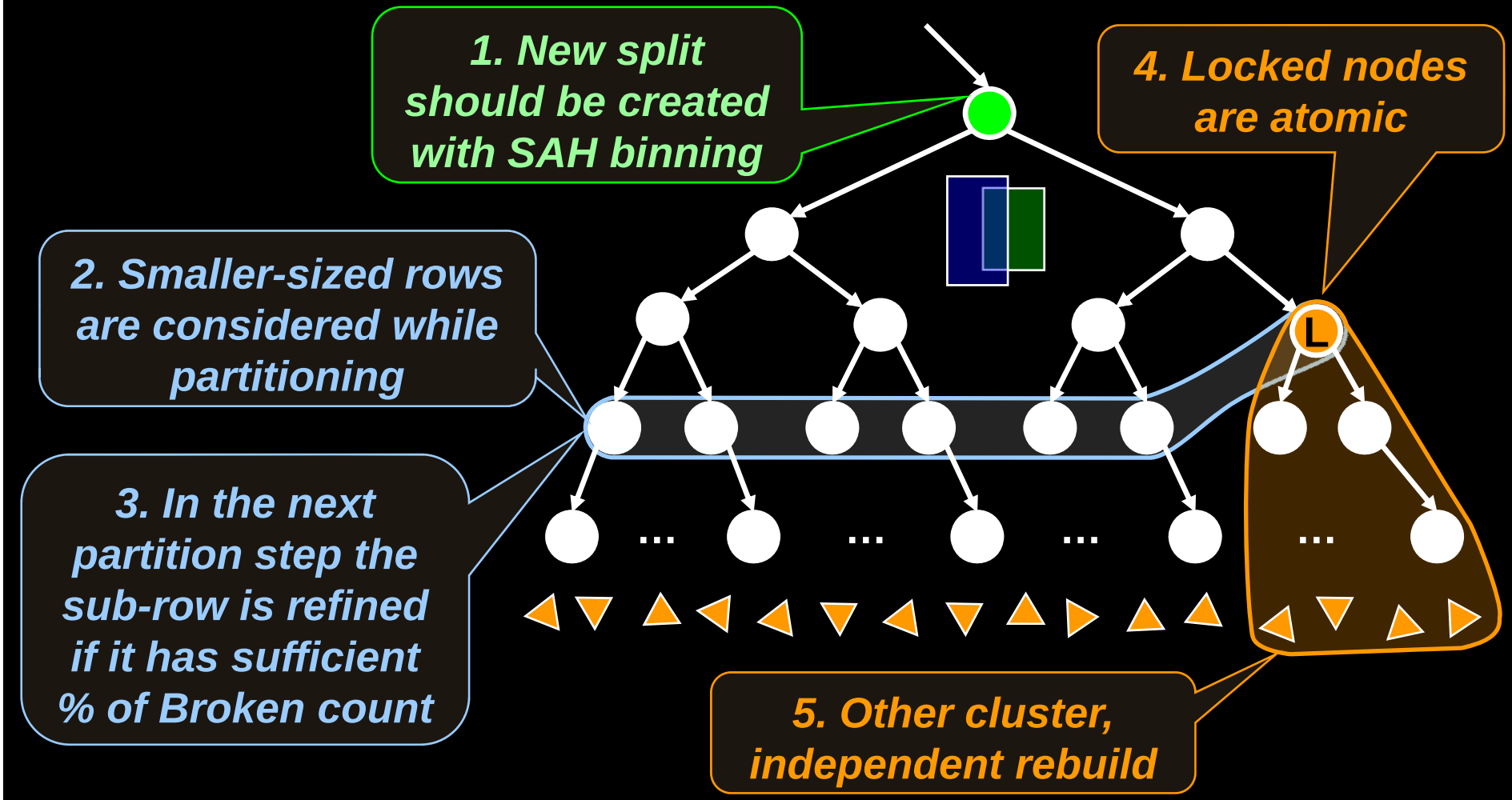
RT 08



The set of independent clusters for rebuilding may form the hierarchy. Their roots are locked

Hierarchical rebuild

RT 08



This way saves some rebuild operations

Cluster migration

RT 08

Migrating nodes are processed in 3 loops:

1. Unlink each node from the tree
2. Adjust the remaining tree and search reinsertion nodes
3. Insert each node back into the tree

At every loop all migrating nodes are passed in the “end-begin” order – i.e. a loop starts from nodes of higher levels

Cluster migration

RT 08

Migrating nodes are processed in 3 loops:

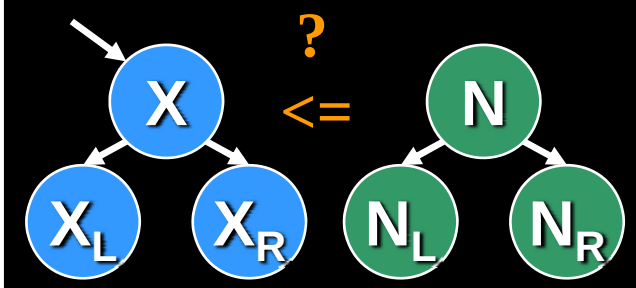
1. Unlink each node from the tree
2. Adjust the remaining tree and search reinsertion nodes
3. Insert each node back into the tree

The formula that detects migrating nodes and these 3 loops automatically resolve the problem of the tree-thinning effect

Cluster migration

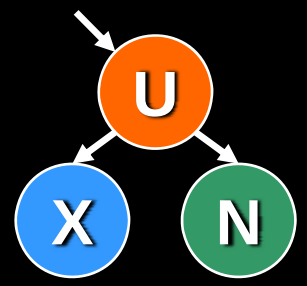
RT 08

Where insert N when start from X?

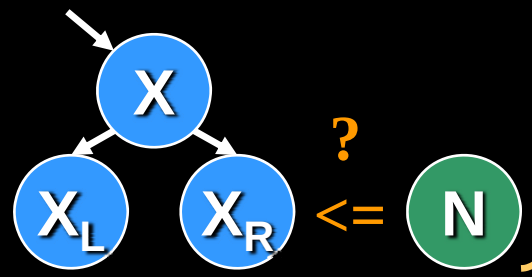


The insertion is a recursive process of taking decisions

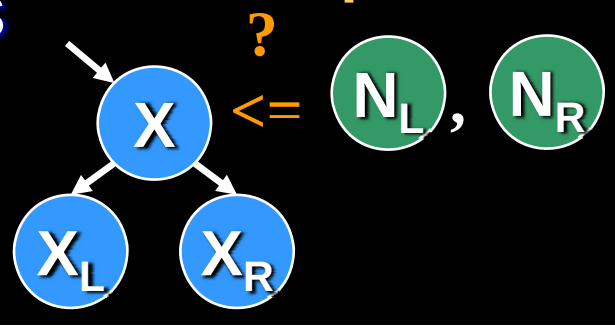
1. Unite X and N



2. Descend into one of the children



3. Decompose N



The variant of decision is taken according to its SAH evaluation

N is decomposed when its insertion produces severe overlap in X.

Memory manager

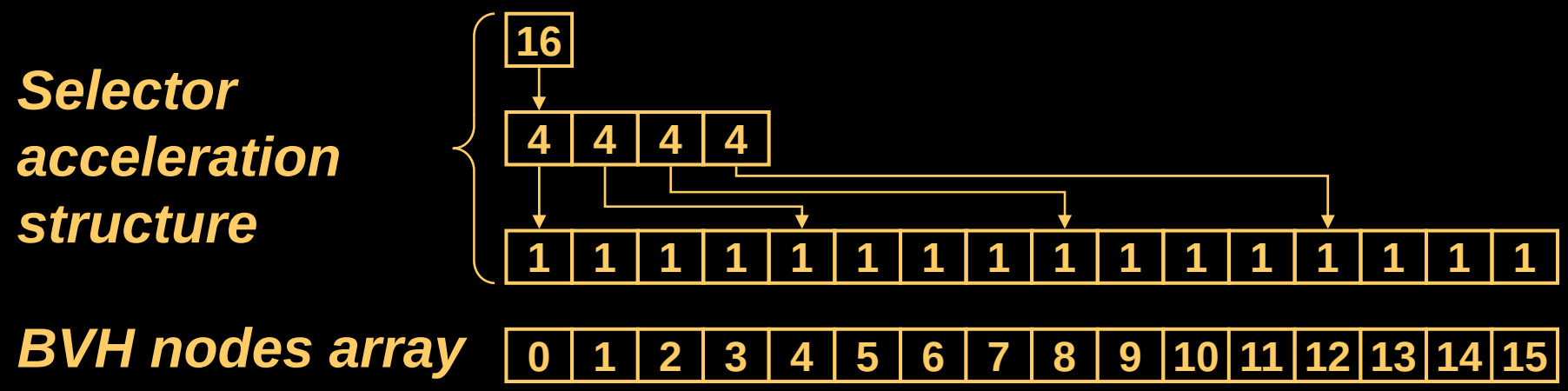
RT 08

- Problem: frequent updates of pointers in the proposed highly-dynamic structure result in cache-efficiency degradation.
- Property of restructurings: in either rebuild or insert process a new BVH-node is allocated under some parent node.

Memory manager

RT 08

Acceleration structure for pre-allocated array of BVH-nodes improves the situation:

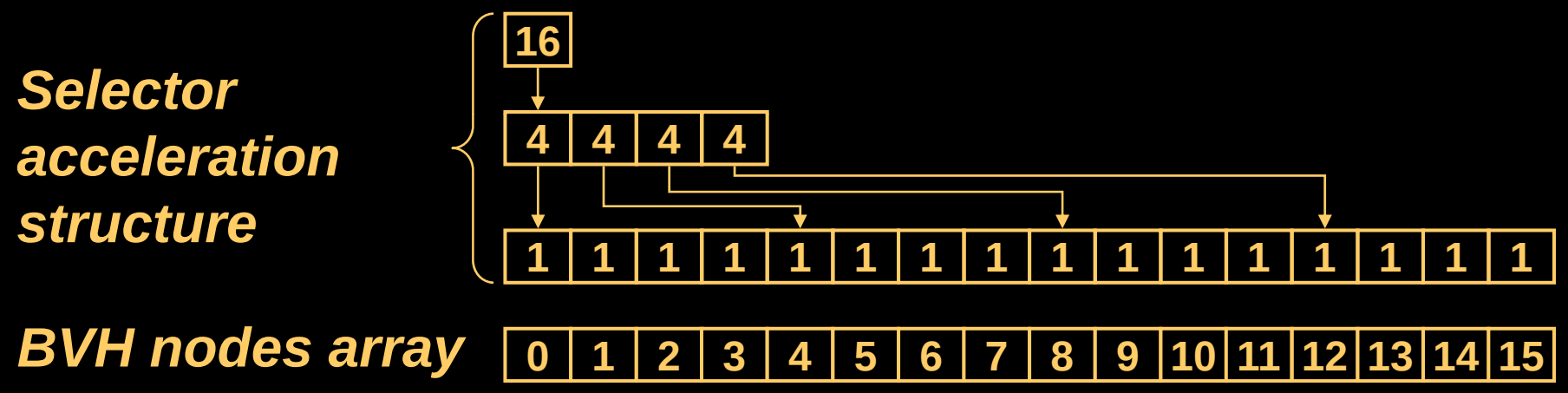


It accelerates the allocation of a free BVH-node that is the nearest to the given one in the memory space (e.g. nearest to parent)

Memory manager

RT 08

Acceleration structure for pre-allocated array of BVH-nodes improves the situation:



This strategy tries to keep the BVH-layout of reasonable cache-efficiency

Benchmarks

RT 08

BVH Updater – Core 2 Quad 2.4 GHz (1 core utilization)

Ray Tracer – GeForce 8800 GTX and CUDA, mono-ray tracer

UNC Dynamic models:

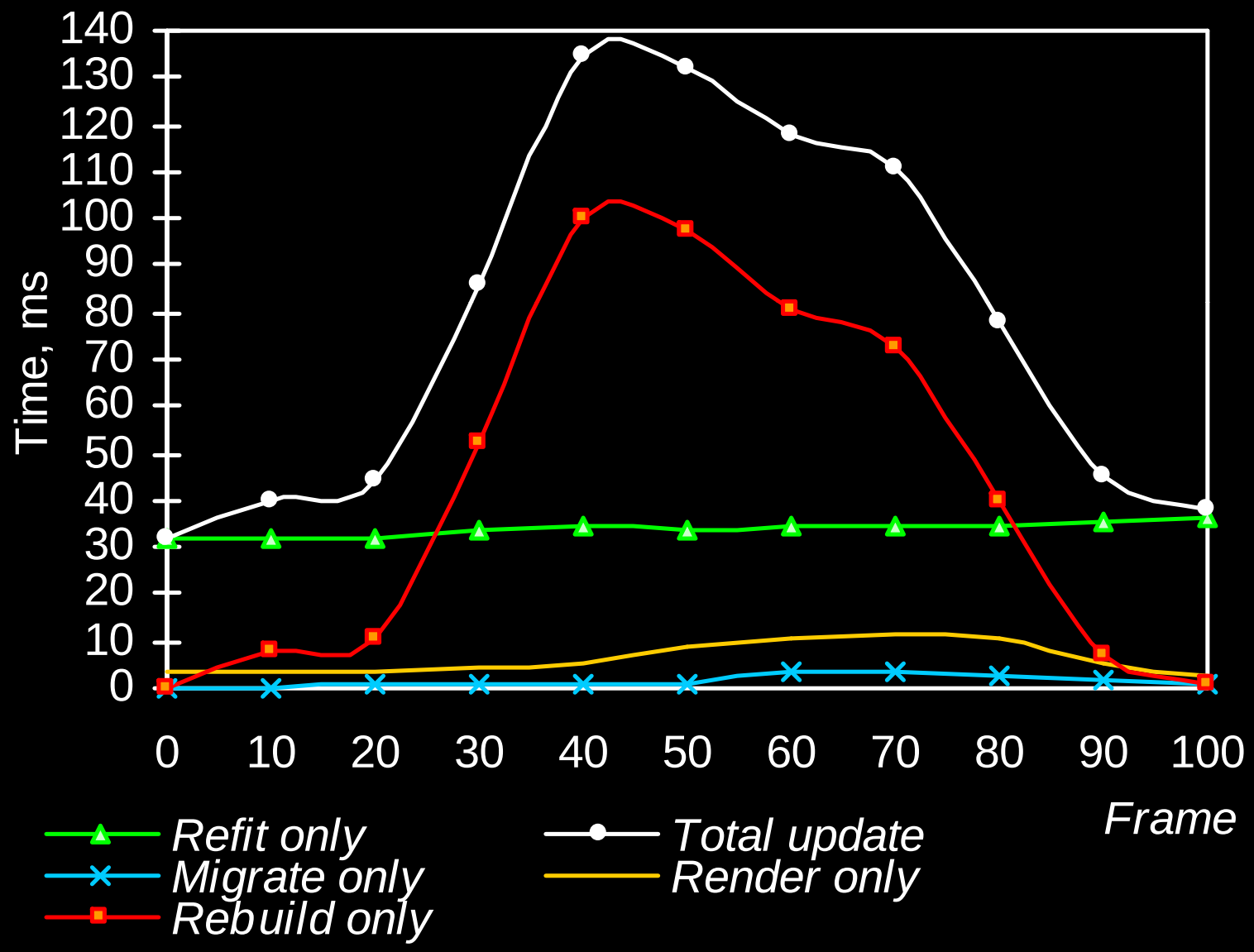
Exploding dragon (252K triangles)

Cloth simulation (92K triangles)

Colliding balls (146K triangles)

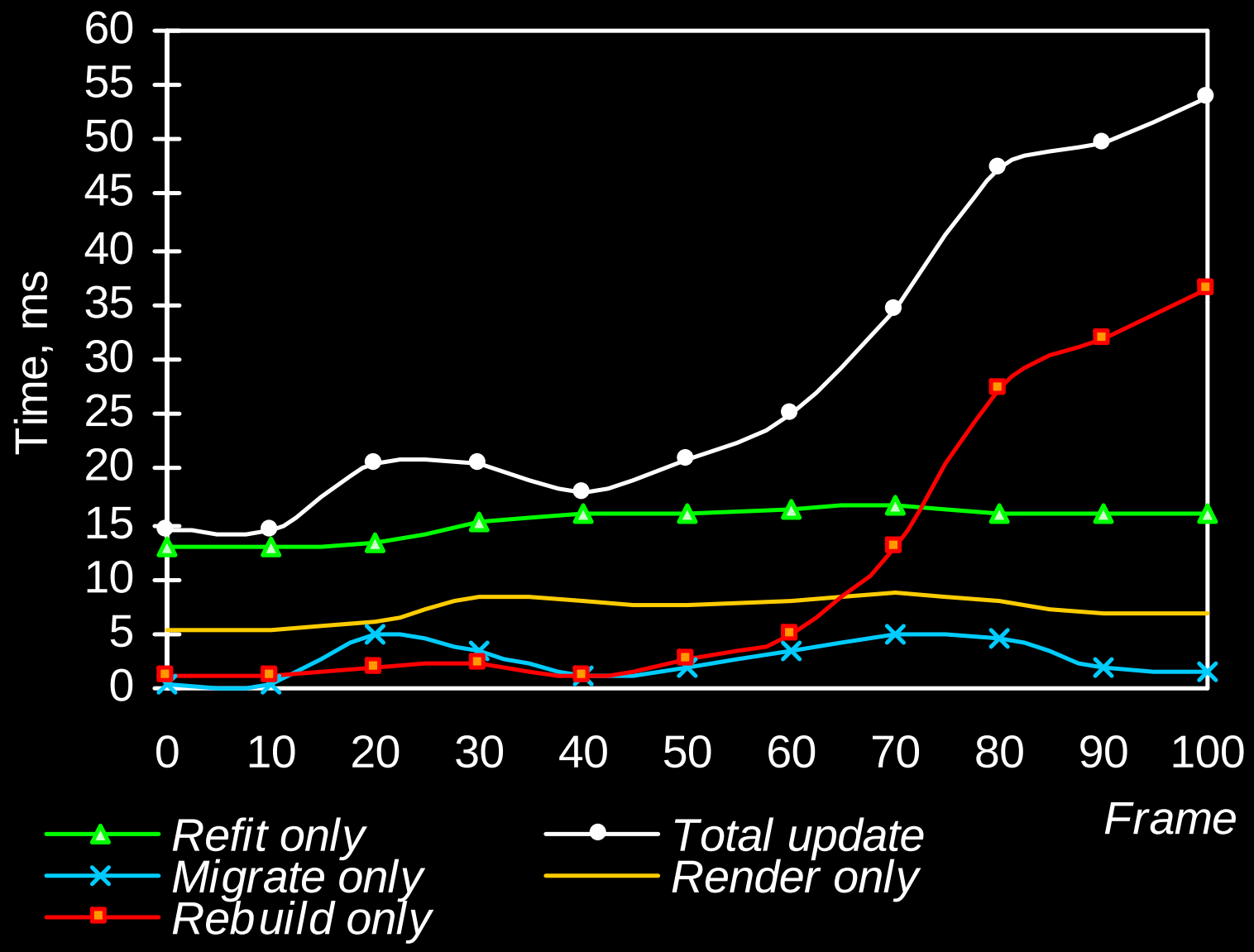
Dragon timings

RT 08



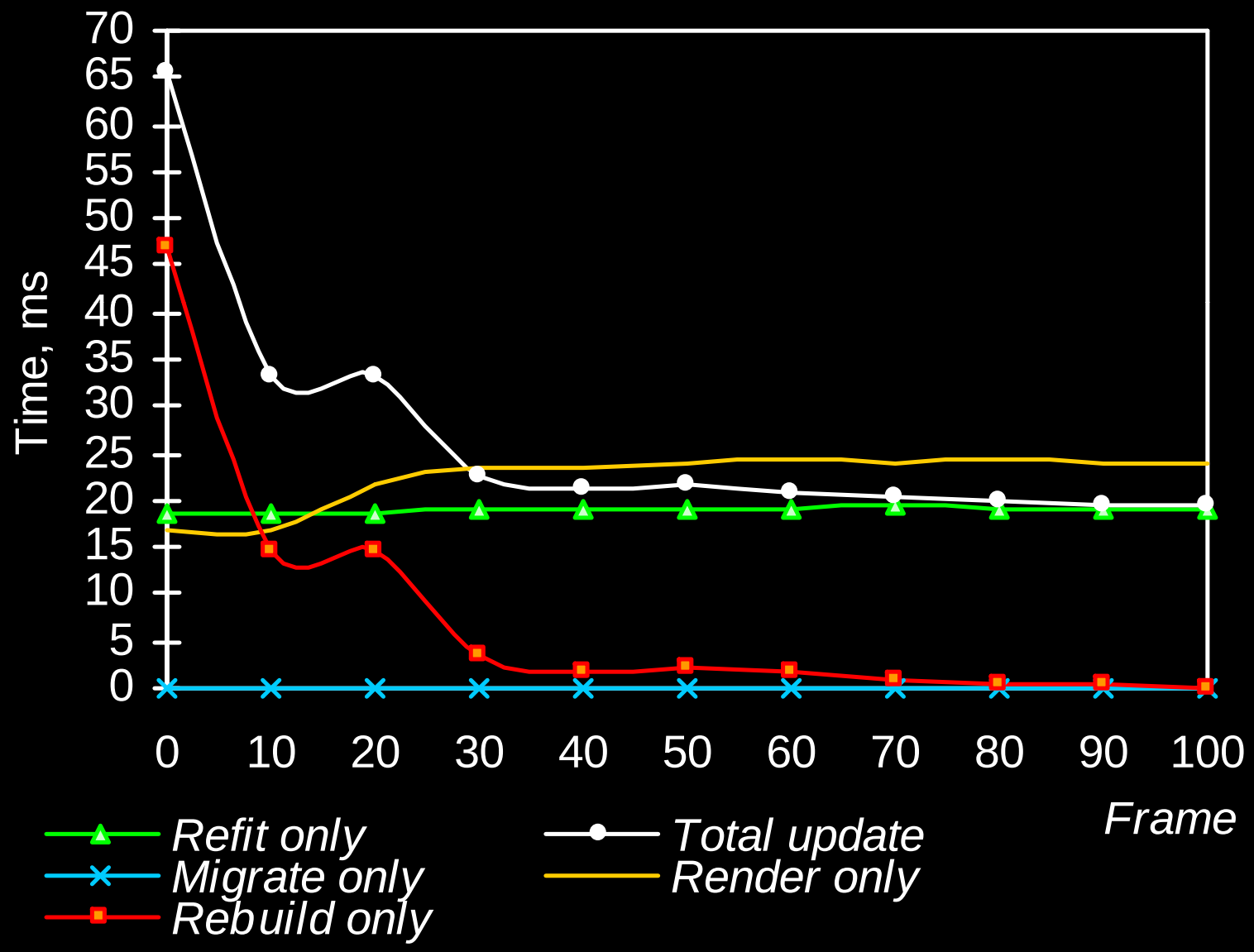
Cloth timings

RT 08



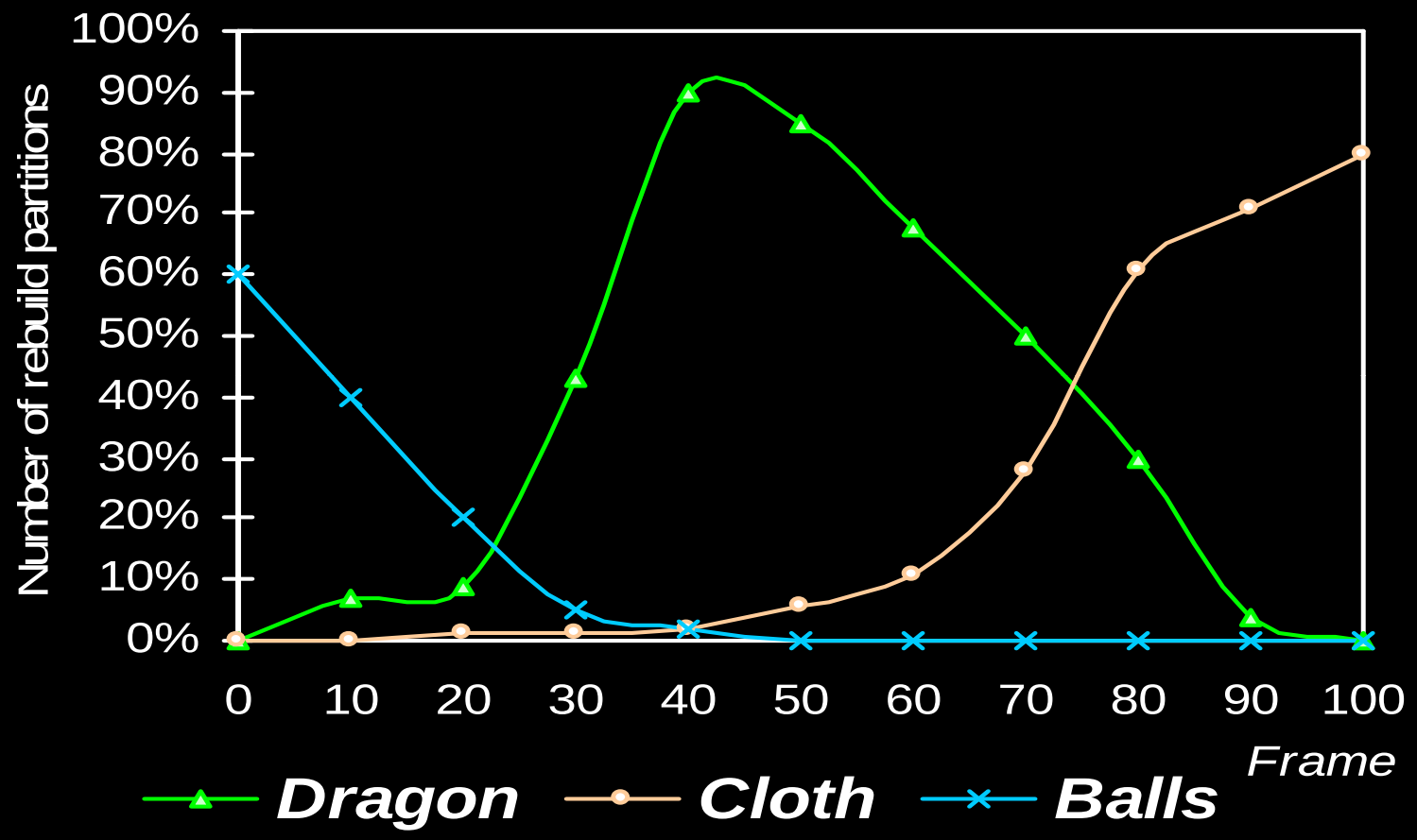
Balls timings

RT 08



Rebuild partitions

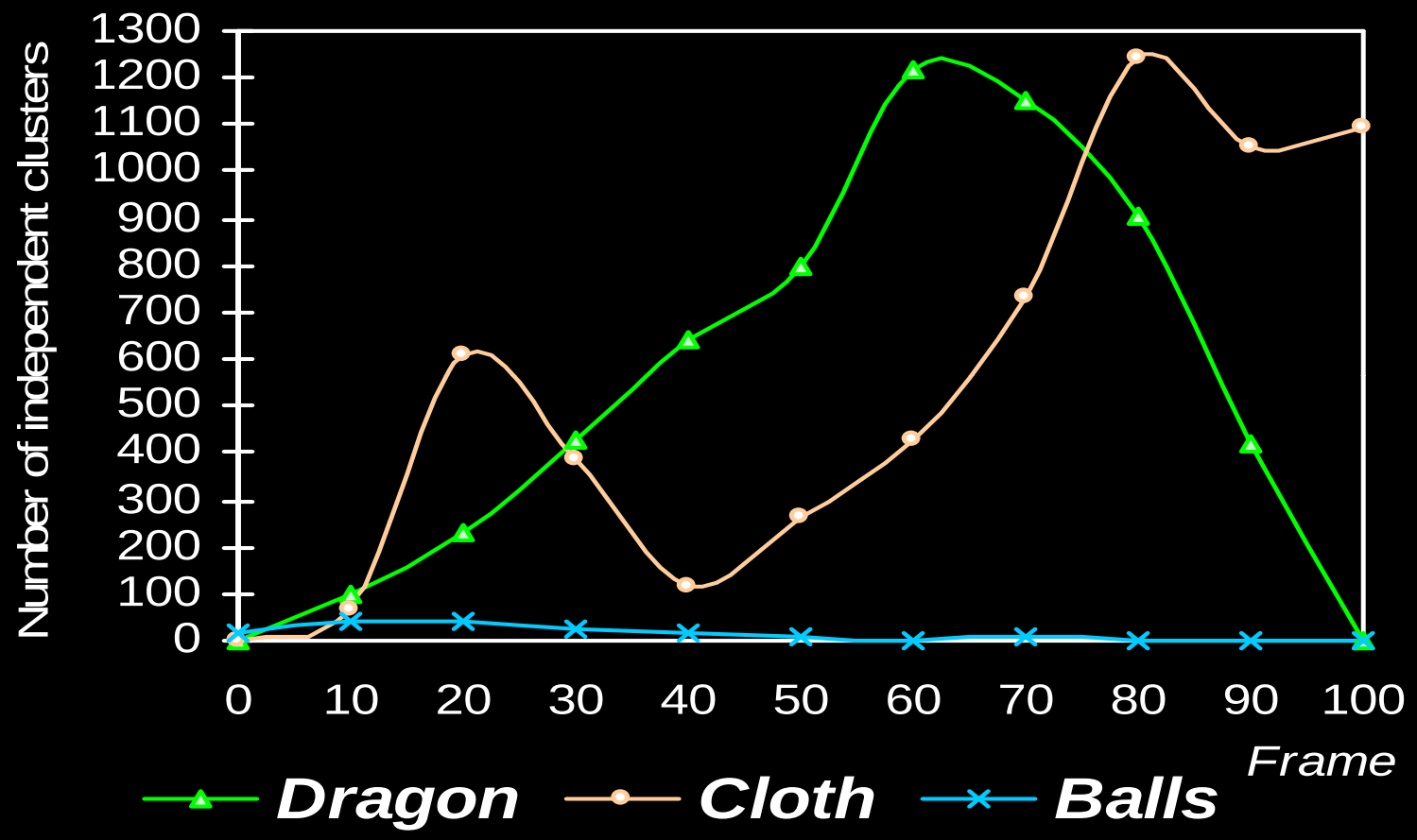
RT 08



Relative number of rebuild partitions. Model specific 100% = number of partitions for the full-rebuild (i.e. number of inner nodes)

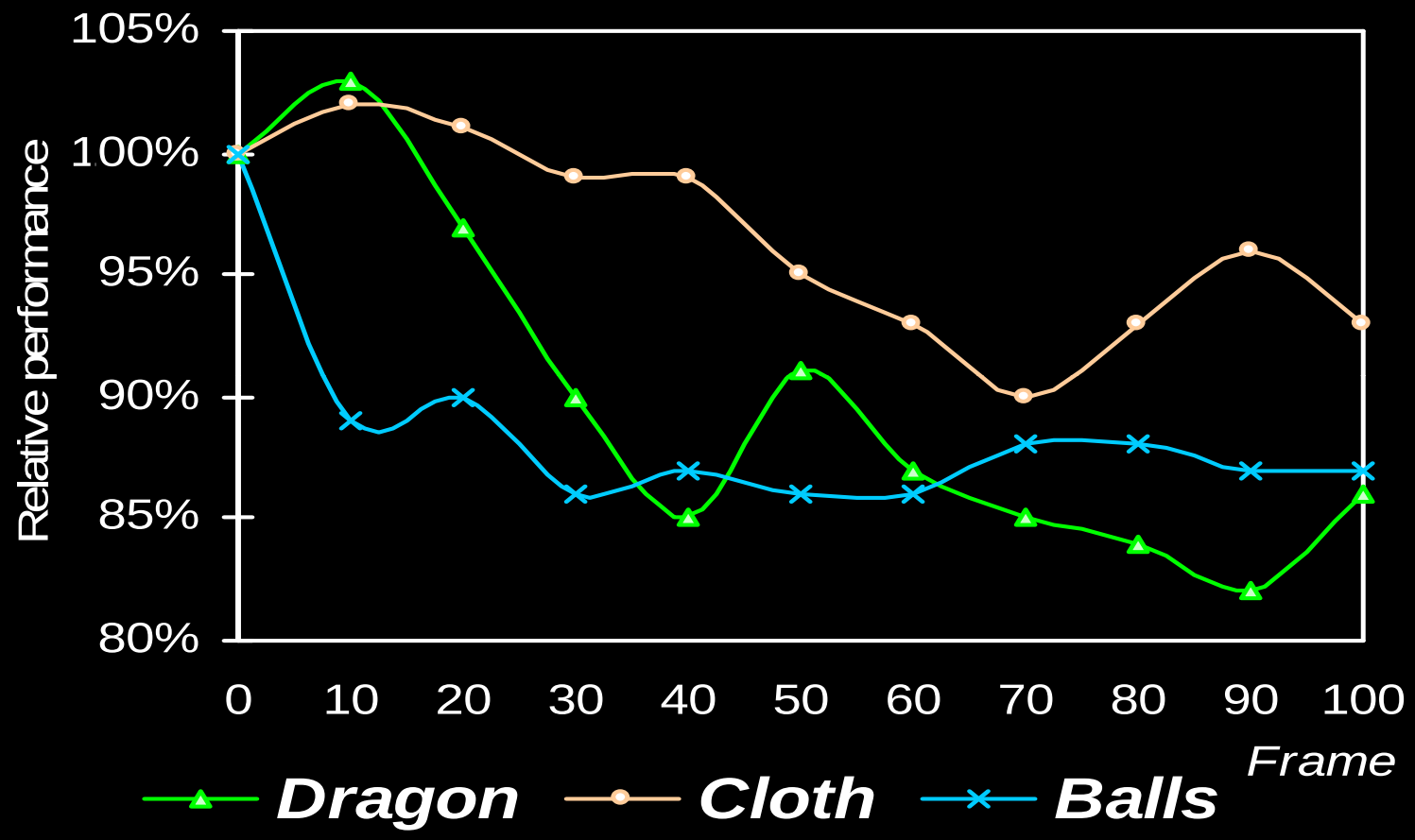
Independent clusters

RT 08



The number of detected independent exploded clusters

Rendering performance RT 08



Relative rendering performance of the BVH in comparison to the one produced by full SAH binned-rebuild

Dragon video

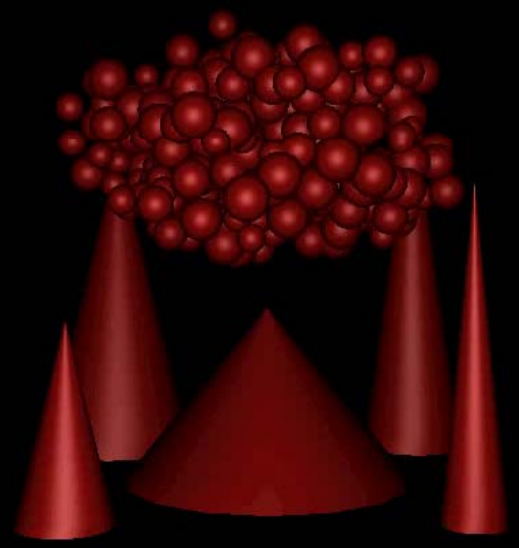
RT 08

Cloth video

RT 08

Balls video

RT 08



The algorithm adapts to the rigid motion associating every ball with a separate subtree and then exploits only migrating updates

Conclusions

RT 08

- The algorithm unites advantages of several updating techniques for the BVH.
- Cheapest update techniques are utilized when they can keep reasonable BVH quality.
- The algorithm is applicable to various types of dynamic models.
- Lots of produced independent clusters for rebuild will be useful in a future parallel version.