INF 2063 – Tópicos em CG III

Visualização de Modelos Massivos

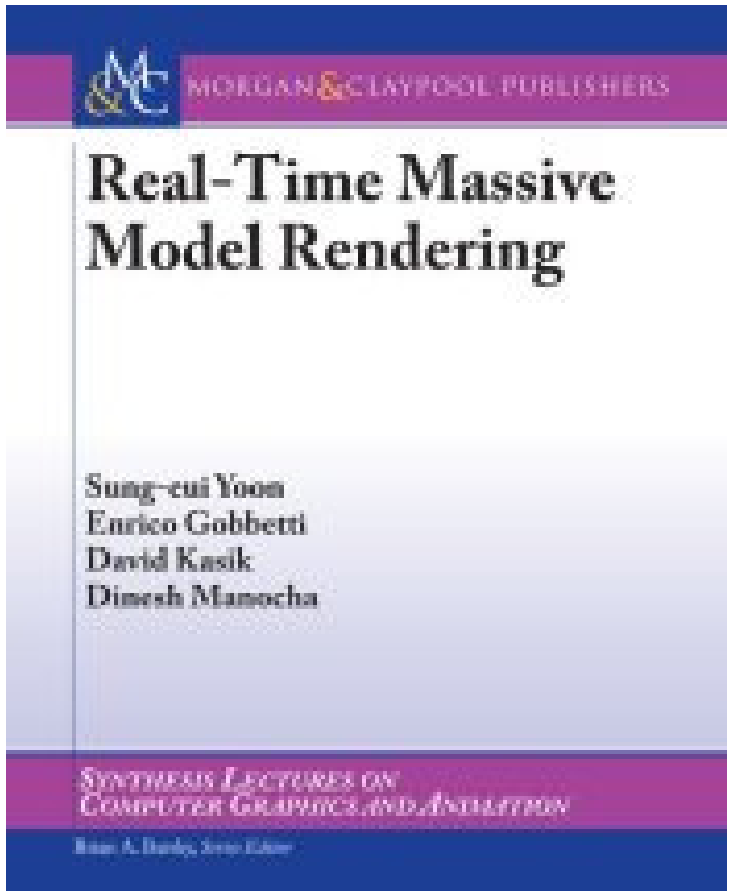
Prof. Alberto Raposo
Tecgraf / DI / PUC-Rio

Aula 06

Cache Coherent Data Management – Parte 2

Alberto Raposo - 2010

Livro



1. Introdução √
2. Visibilidade √
3. Simplificação e Níveis de Detalhe √
4. Representações Alternativas √
5. Cache-Coherent Data Management (hoje – parte 2)
6. Conclusões (hoje)

Tecgraf PUC-RIO VR GROUP

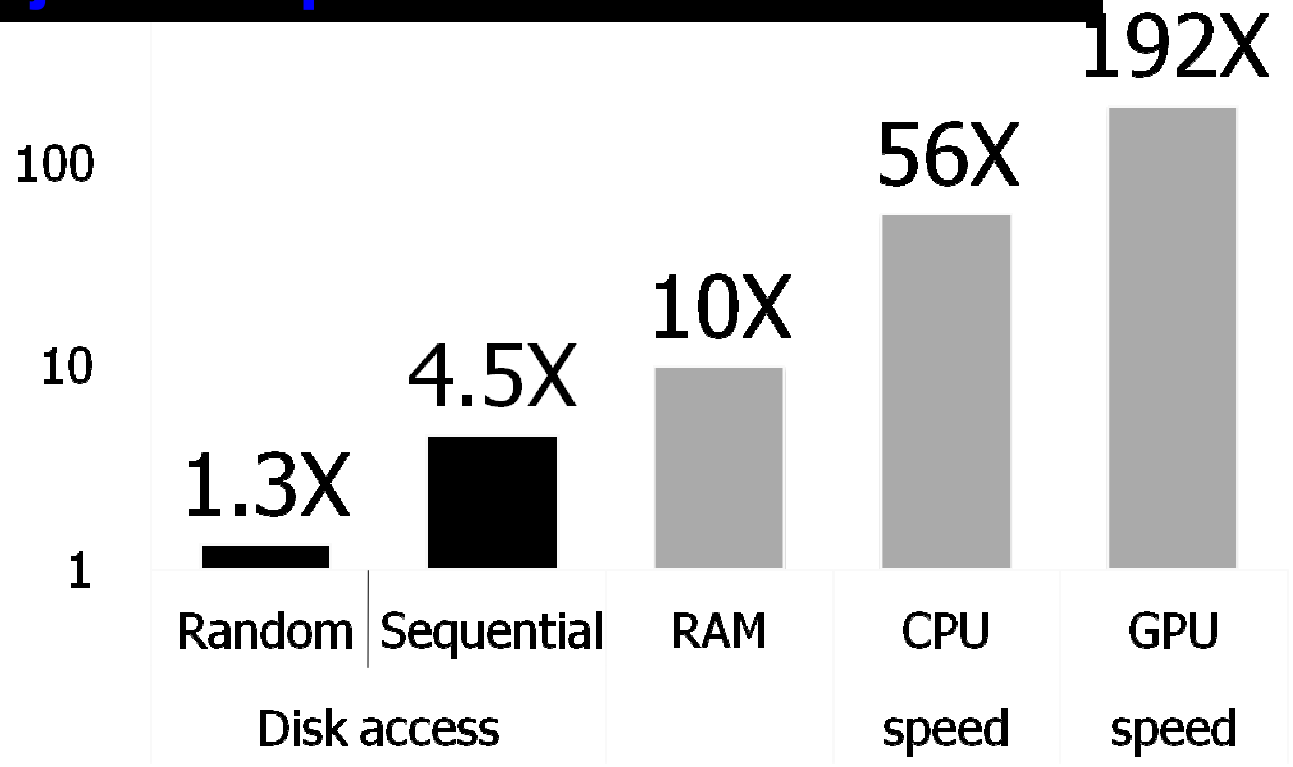
Alguns Slides de

Sung-Eui Yoon
KAIST

URL: http://jupiter.kaist.ac.kr/~sungeui/

Current Architecture Trends

Data access time becomes the major computational bottleneck!



Accumulated growth rate during 1999~2009 (log scale)

Tecgraf PUC-RIO VR GROUP

Para reduzir tempos de acesso a memória

- 2 técnicas-padrão
  - Computation Reordering
    - Reordena as operações dos algoritmos para reduzir cache misses
  - Data Layout Optimization
    - Reconfigura layout de dados para reduzir cache misses
    - Cache-aware vs cache-oblivious

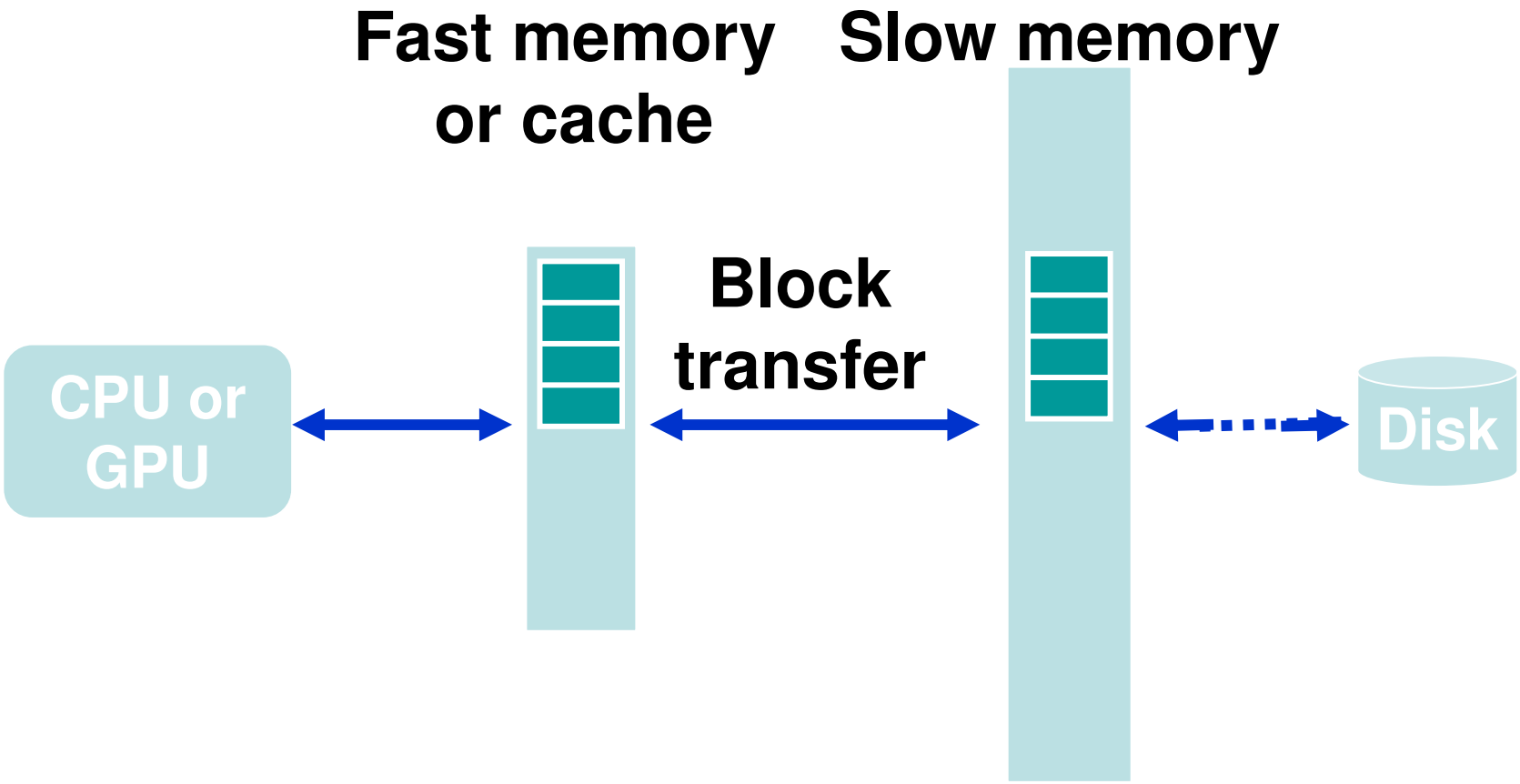
Cache-aware vs Cache-oblivious

- **Cache-aware layouts**
  - **Optimized for particular cache parameters (e.g., block size)**
- **Cache-oblivious layouts**
  - **Minimize data access time without any knowledge of cache parameters**
  - **Even work with various hardware and memory hierarchies**

Orthogonal Approaches

- Cache-coherent layouts
- Random-accessible compressed data
- Cache-oblivious ray reordering

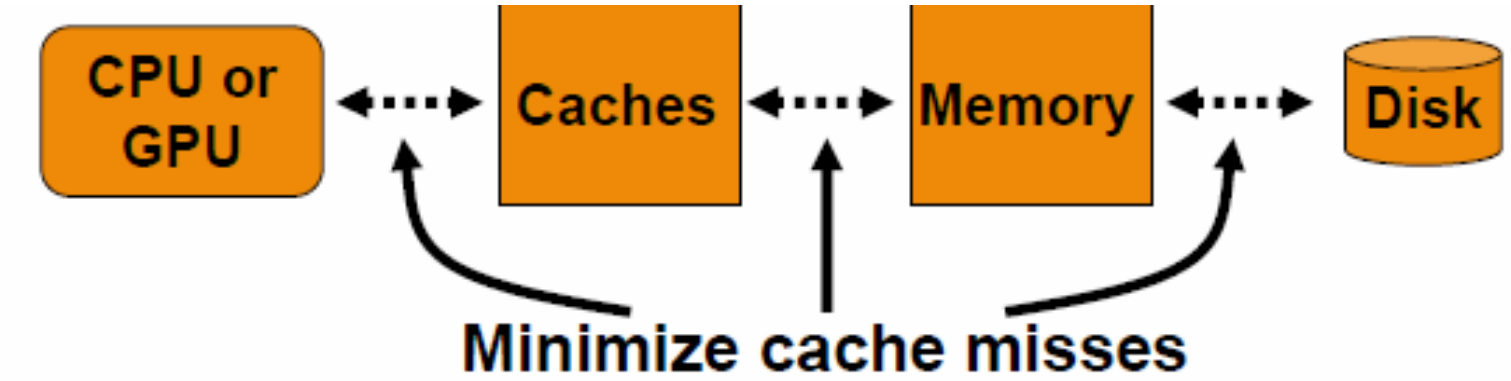
Block-based I/O Model



Access time: 10^{-6} **sec** 10^{-4} **sec** **1 sec**

Tecgraf PUC-RIO VR GROUP

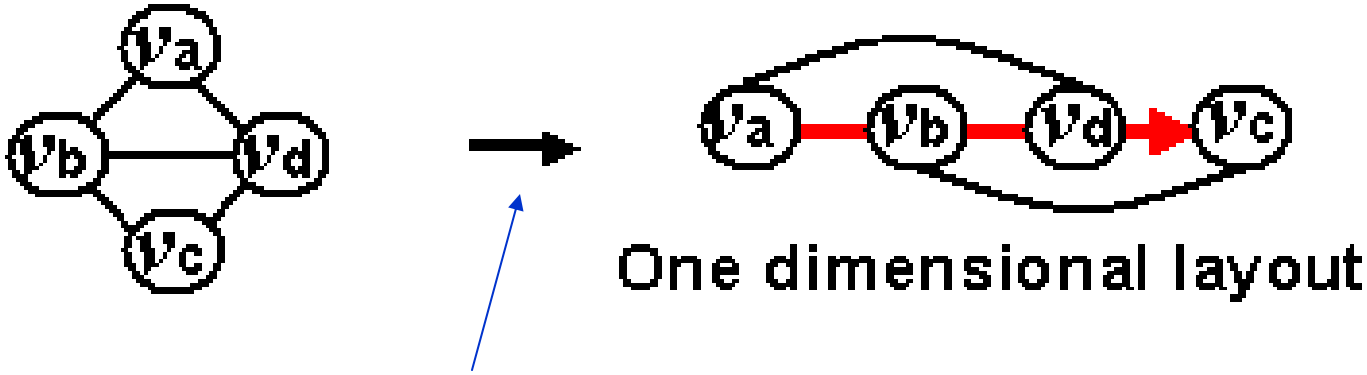
Cache Coherent Layouts



Reduce expensive I/O accesses!

Cache Coherent Data Layouts

- Ideia básica



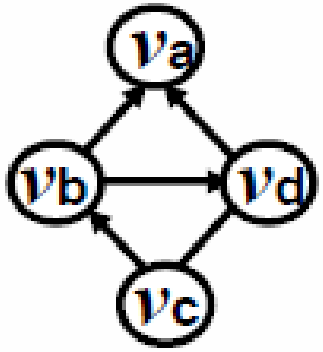
Otimizacao do layout dos dados

O grafo a esquerda representa o padrão de acesso aos dados. Cada arco do grafo assume uma probabilidade igual de acesso a partir de um nó (dado). Uma função de otimização vai gerar um layout linear, para armazenamento em cache que minimiza os cache misses.

Tecgraf PUC-RIO VR GROUP

Cache Coherent Data Layouts

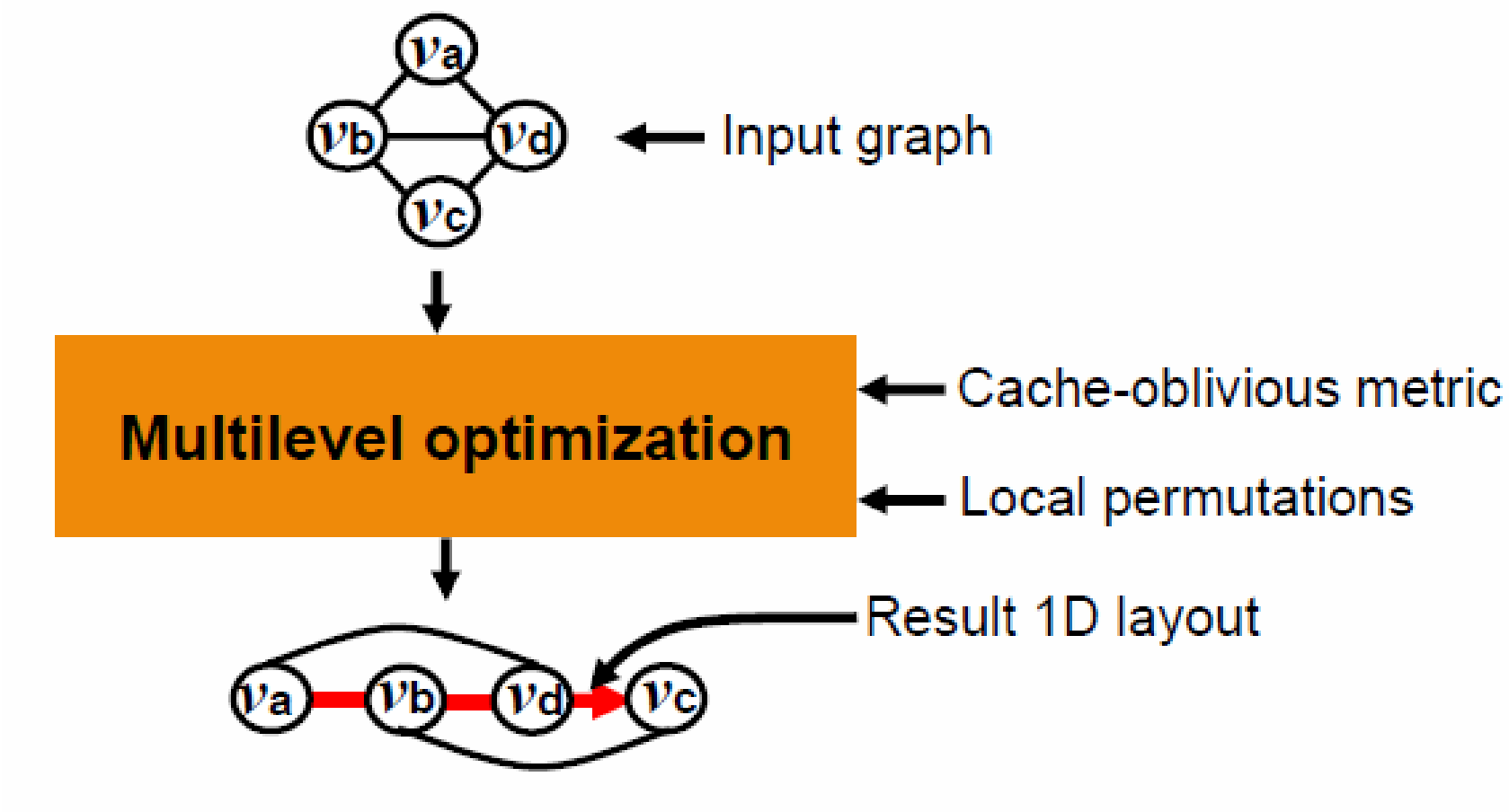
- **Directed graph, G = (V, E)**
  - **Represents access patterns of applications**
- **Vertex**
  - **Data element**
  - **(e.g., mesh vertex or mesh triangle)**
- **Edge**
  - **Connects two vertices if they are likely to be accessed sequentially**



Em Computação Gráfica

- Sendo os vértices de uma malha os dados (os nós do grafo de acesso)
  - Arcos do grafo seriam as arestas dos triângulos
- Sendo os triângulos os dados (os nós no grafo de acesso)
  - Arcos do grafo podem ser a adjacência dos triângulos
- Dados também podem ser nós de uma hierarquia de volumes envolventes

Cache Coherent Data Layouts

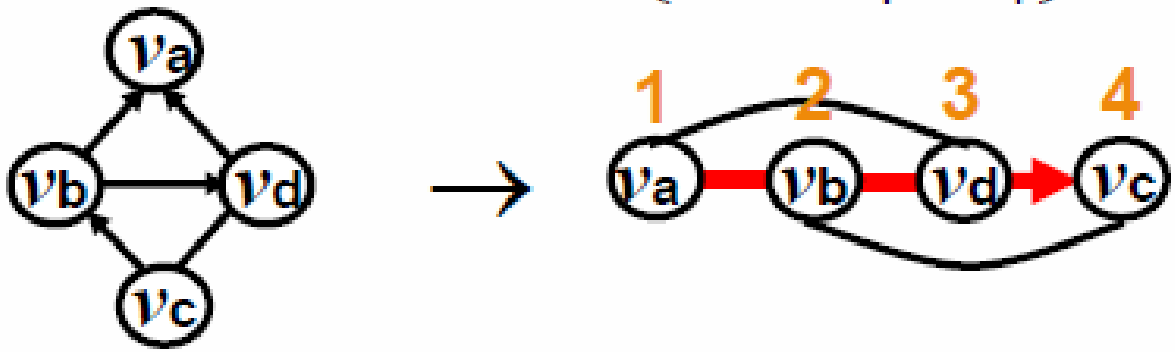


Tecgraf PUC-RIO VR GROUP

Função de Otimização

- **Vertex layout of G = (V, E)**
  - **One-to-one mapping of vertices to indices in the 1D layout**

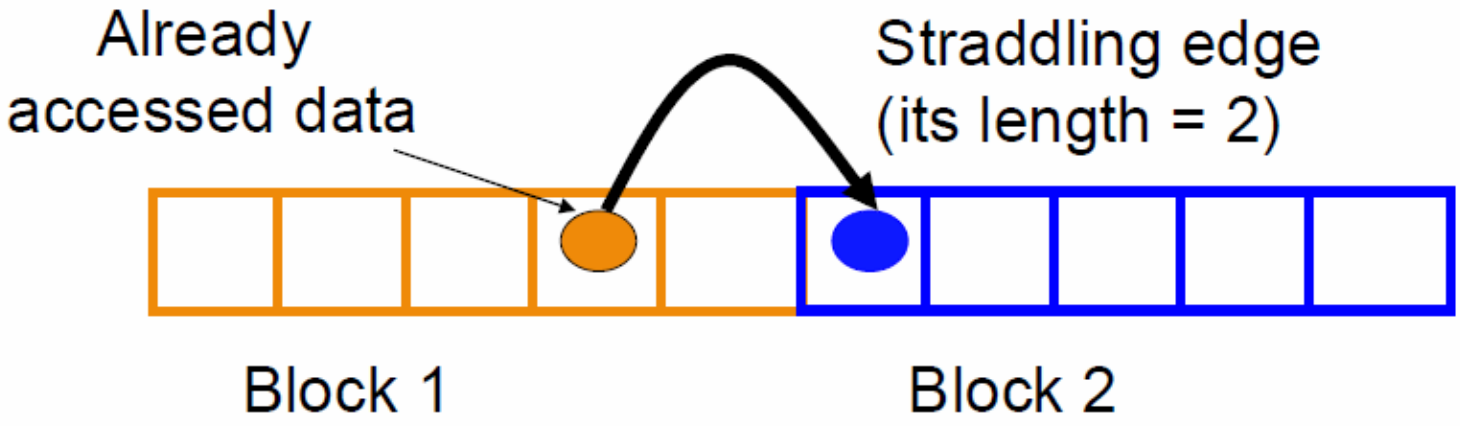
$$\varphi : V \rightarrow \{1, \ldots, |V|\}$$



- **Compute a φ that minimizes the expected number of cache misses**

Cache-aware, 1 block

- **Cache misses when a cache holds only one block**



- **Layout computation**
  - **Minimize the number of straddling edges**
  - **Graph partitioning**

Tecgraf PUC-RIO VR GROUP

Cache-coherent layouts of meshes

- Achar a função φ que tem o valor mínimo em cache-oblivious
- Solução “força-bruta”: checa-se todos os possíveis layouts de vértices, pela permutação dos vértices no grafo
  - Exponencial! → Inviável
- Solução possível: multi-level layout
  - Não garante layout ótimo
  - Resultados comprovam que produz layouts com muito boa coerência de cache.

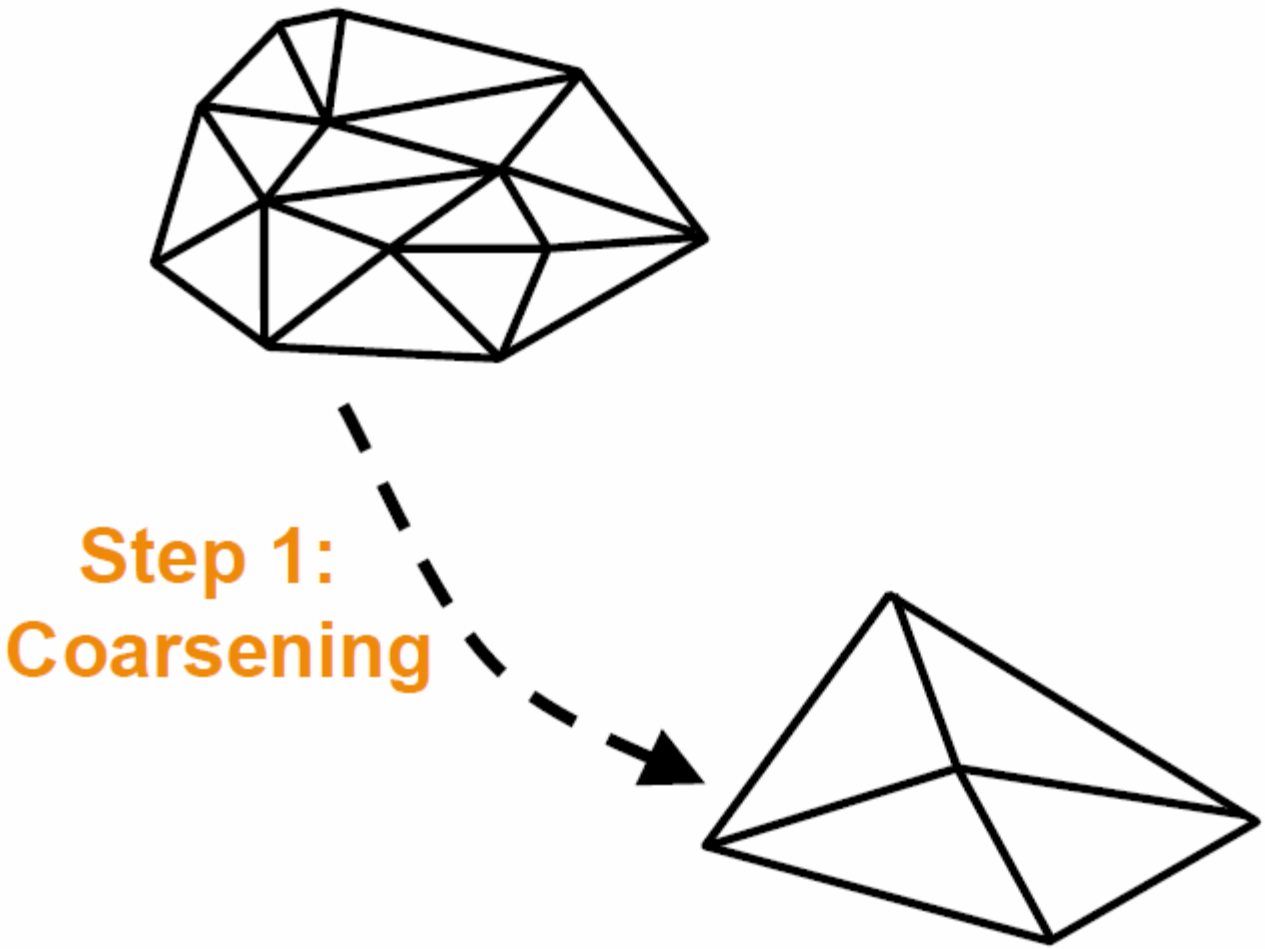
Tecgraf PUC-RIO VR GROUP

Multi-level optimization

- Três passos
  1. Simplifica (coarsen) o grafo
  2. Faz permutãções e otimiza grafo simplificado
  3. Refina o grafo com as operações inversas que foram feitas no passo 1
     - Refina ordenação do grafo com permutações locais.

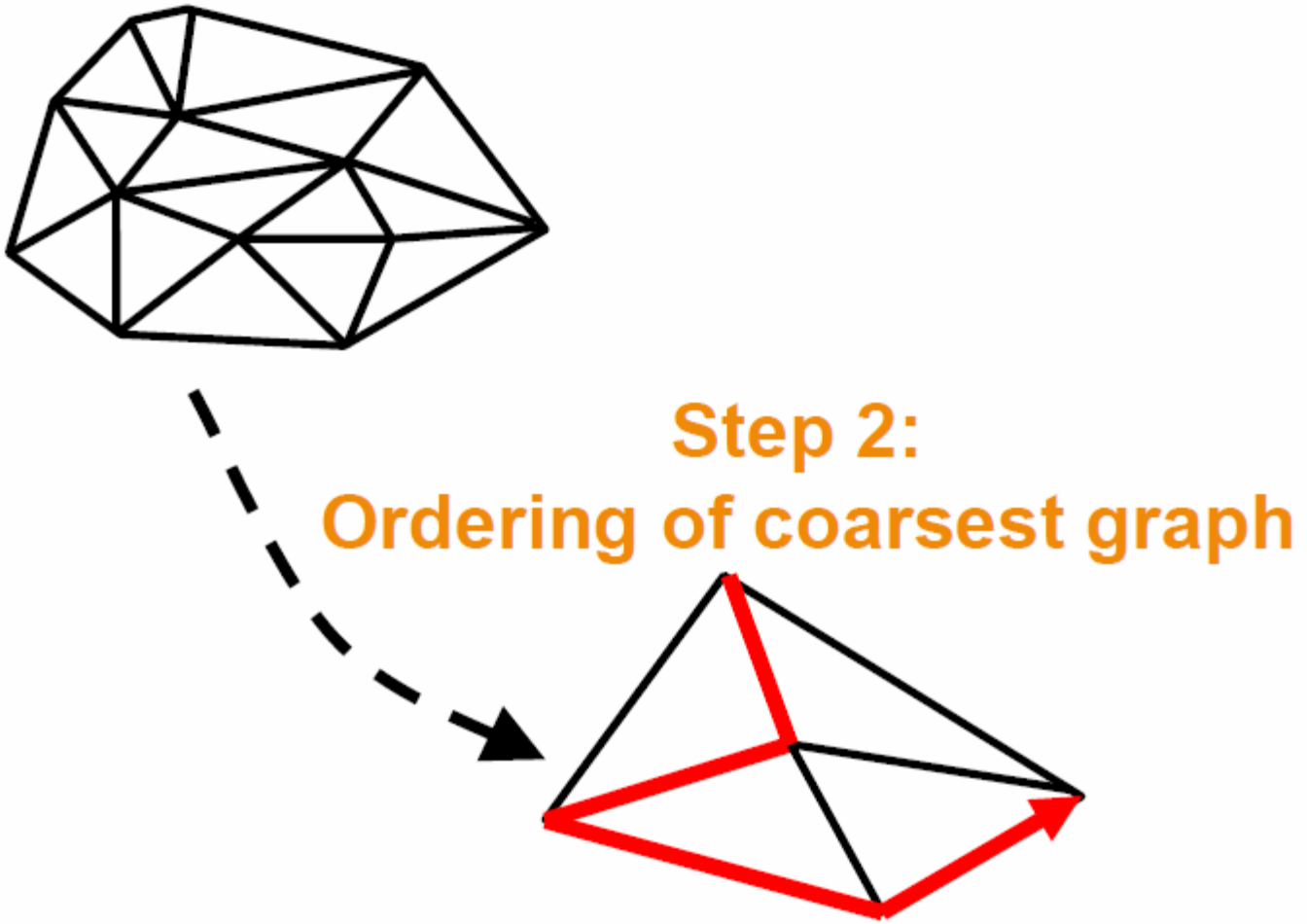
Tecgraf PUC-RIO VR GROUP

Multi-level optimization



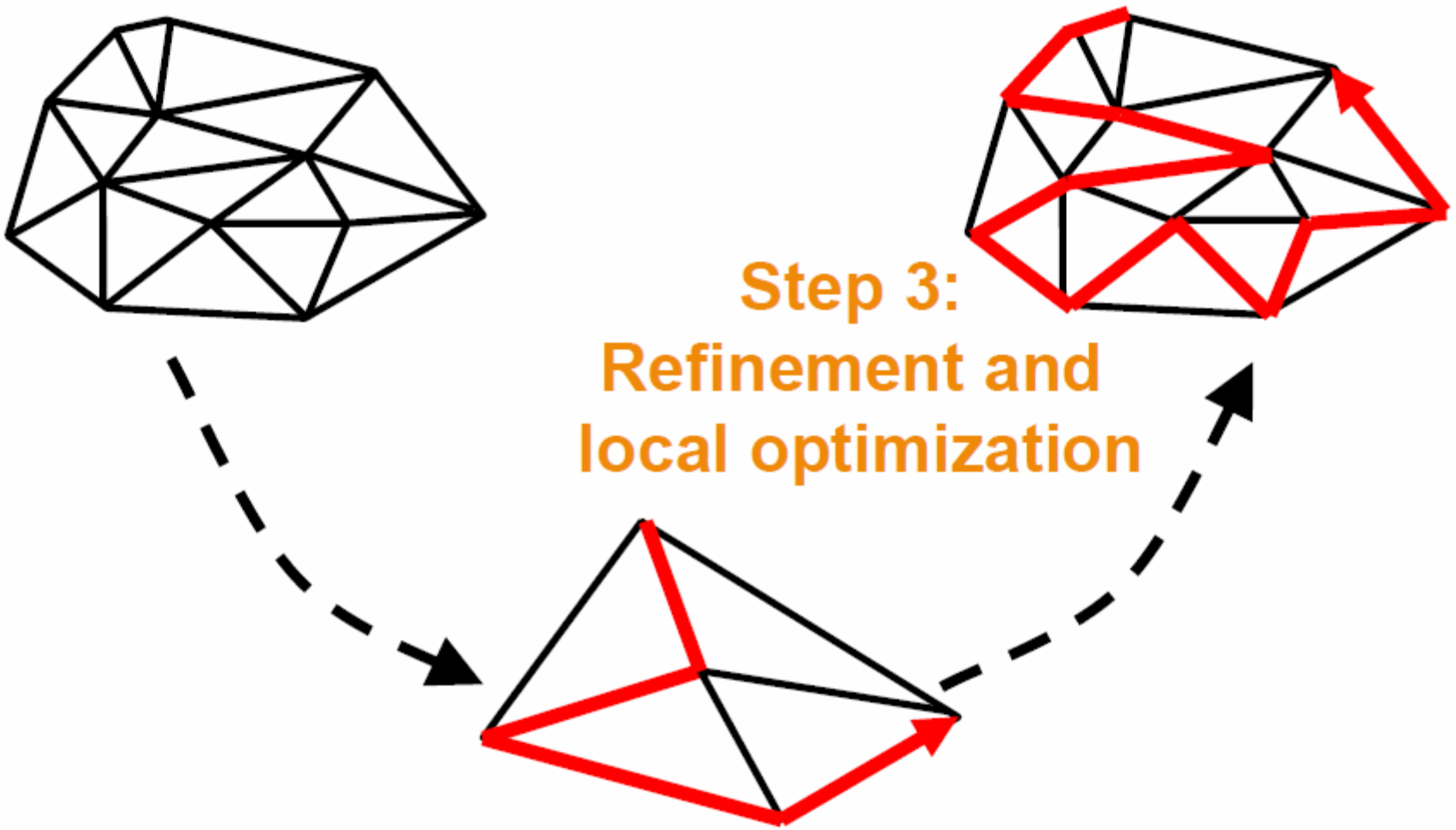
Tecgraf PUC-RIO VR GROUP

Multi-level optimization



Tecgraf PUC-RIO VR GROUP

Multi-level optimization



Tecgraf PUC-RIO VR GROUP

Etapa 1: Simplificação do Grafo

- A estrutura do grafo deve ser bem preservada a cada simplificação
  - Se não for assim, o grafo simplificado não é um bom candidato para a otimização da coerência de cache
- Autor diz que usa biblioteca de particionamento chamada Metis
  - Particiona-se o grafo em k chunks, minimizando os arcos entre essas chunks
    - Faz-se particionamento recursivo até que cada chunk tenha no máximo k nós.
    - K é tipicamente igual a 4.

Etapa 2: Ordenação do Grafo Simplificado

- Considerando-se k=4, há apenas 4! = 24 possíveis ordenações em cada chunk.
  - Calcula-se o custo de cada uma delas, considerando métricas de cache oblivious, e escolhe-se o layout de menor custo.

Etapa 3: Refinamento do Grafo

- Usa-se a operação inversa à da etapa 1 para refinar o grafo.
- Para cada nó, cria-se k novos nós
  - Permutações locais desses k novos nós definem sua ordem no local do nó anterior.

Tecgraf PUC-RIO VR GROUP

Out-of-core multi-level optmization

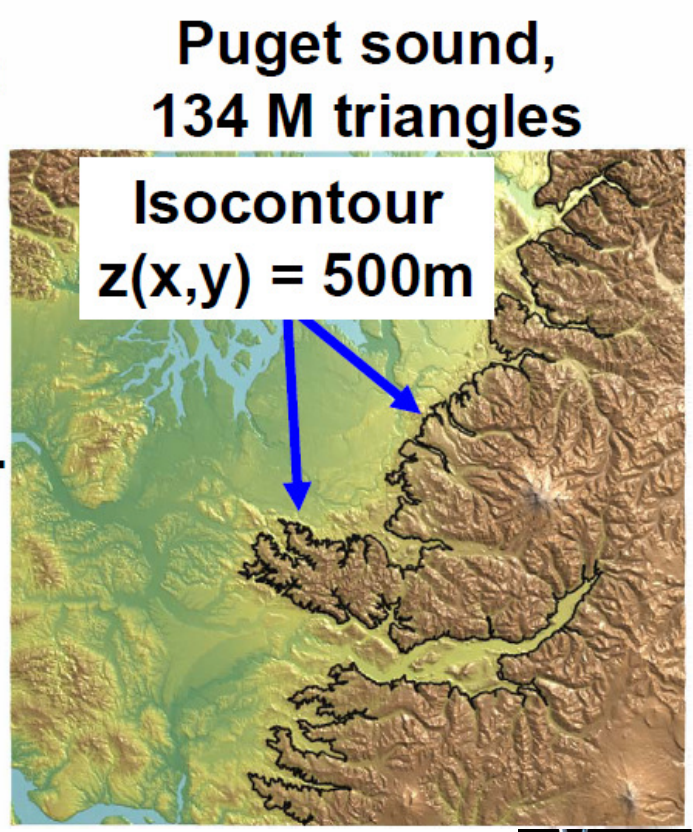
- Para modelos massivos
- Estátua de St. Matthew (372M triângulos): 2.6 horas de processamento

Aplicações

- Extração de iso-contornos (comuns em GIS)

  - **Uses contour tree [van Kreveld et al. 97]**
  - **Use mesh as the input graph**
  - **Extract an isocontour that is orthogonal to z-axis**

Ganho de ~20x !!!



Aplicações / Desafio

- View-dependent Meshes
  - É caro computar strips de triângulos cache-coherent, uma vez que esse tipo de malha muda sua conectividade a cada quadro.

Cache-coherent layouts of hierarchies

- Organiza as hierarquias que são usadas para acelerar a performance da renderização
  - Exemplo: BVH (Boundary Volume Hierarchy)
    - Uso: visibility culling, ray tracing, detecção de colisão, etc.

BVH

- Nós-folha representam os triângulos do modelo original.
- Nós intermediários representam os volumes envolventes (esferas, AABB, OBB), etc
- Grandes modelos podem requerer muita memória
  - Ex., custo de armazenagem de uma hierarquia de OBB é aproximadamente 64B por nó.
    - Dezenas de milhões de triângulos levam a GB de espaçc de memória

Cache-coherent layout de BVHs

- Colocação adequada dos nós de uma BVH em cache para reduzir número de cache misses em tempo de execução.
- S.-E. Yoon & D. Manocha. “Cache-efficient layouts of bounding volume hierarchies”. Eurographics 2006.

Fatores a considerar para a otimização

- Padrão de acesso aos dados da BVH
  - Root to leaf
  - Right first, left first...
- Construção especializada
  - No caso da malha de triângulos, supunha-se que cada arco do grafo tinha mesma possibilidade de ser acessado
  - Aqui isso não é verdade
    - Volumes maiores tendem a ser acessados mais frequentemente que volumes menores

Fatores a considerar para a otimização

- 2 localidades:
  - Localidade pai-filho
    - Uma vez que um nó da hierarquia está sendo acessado, seus filhos têm alta probabilidade de serem acessados
  - Localidade espacial
    - Uma vez que um nó esteja sendo acessado, os nós espacialmente próximos a ele têm alta probabilidade de serem acessados

2 localidades

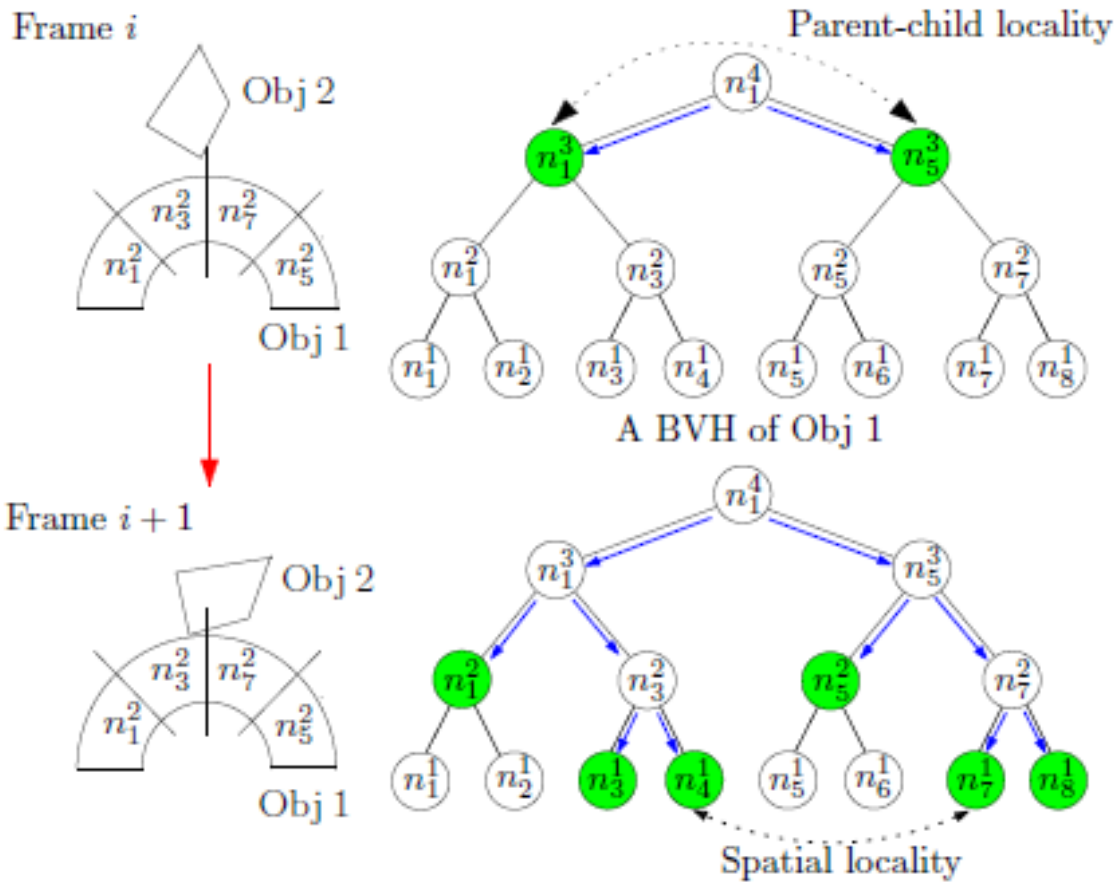


Figure 3: ***Two localities within BVHs:*** We show two successive frames from a dynamic simulation and the change in access patterns (shown with blue arrows) of a BVH. In this simulation, object 2 drops on object 1, as shown on the left. The access pattern of the BVH of object 1 during each frame is shown on the right. The BVs from the 2nd level in the BVH are shown within object 1 on the left. We also illustrate the front traversed within each BVH during each frame in green. The top BVH shows the parent-child locality, when the root node, n_1^4, of the BVH of object 1 collides with the BVs of objects 2. During frame $i+1$, object 2 is colliding with object 1. In this configuration, the BVs n_3^2 and n_7^2 (and their sub-nodes) are accessed due to their close spatial locality.

Tecgraf PUC-RIO VR GROUP

Otimização do layout da BVH

1. Divisão da BVH-tree em clusters de tamanho fixo
   - Por exemplo, tendo cada cluster o tamanho de um bloco de cache
   - Cada cluster vai conter nós de BV que tenham mais possibilidades de serem acessados juntos
     - Localidade pai-filho
2. Layout dos vários clusters
   - Localidade espacial

Otimização do layout da BVH

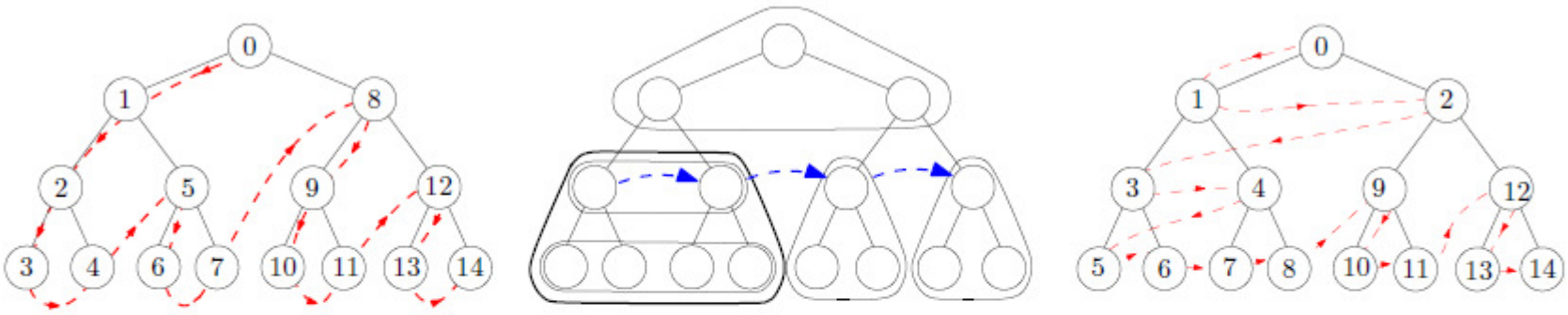


Figure 4: ***Layout computation of a BVH:*** A depth-first layout of a BVH is shown in the leftmost figure and a cache-oblivious layout of the same tree is shown in the rightmost figure. The number within each BV node in the leftmost and the rightmost figures is an index of the ordering of BVs in the layout. The middle figure shows the output of the clustering step. The topmost cluster is the root cluster and the rest are child clusters. Directed edges (shown in blue) indicate ordering between clusters. Also, the middle figure indicates that leftmost cluster is merged with its neighboring cluster.

Aplicações

- Detecção de colisão

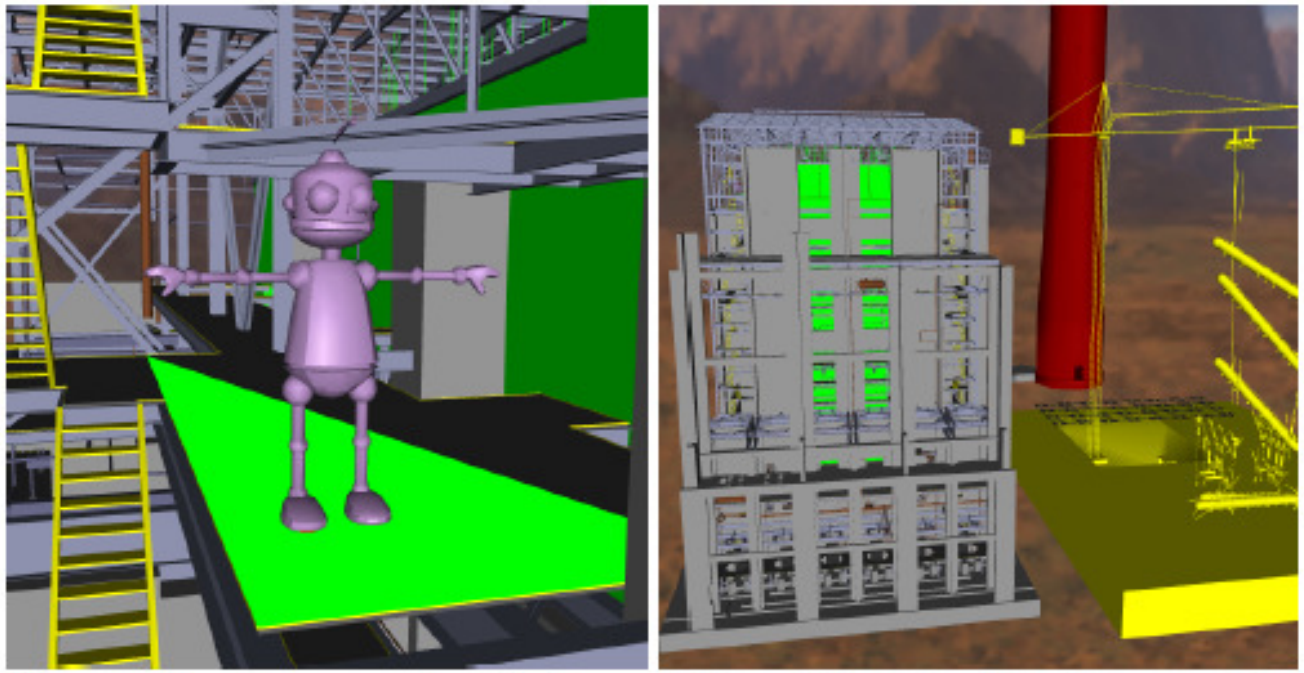
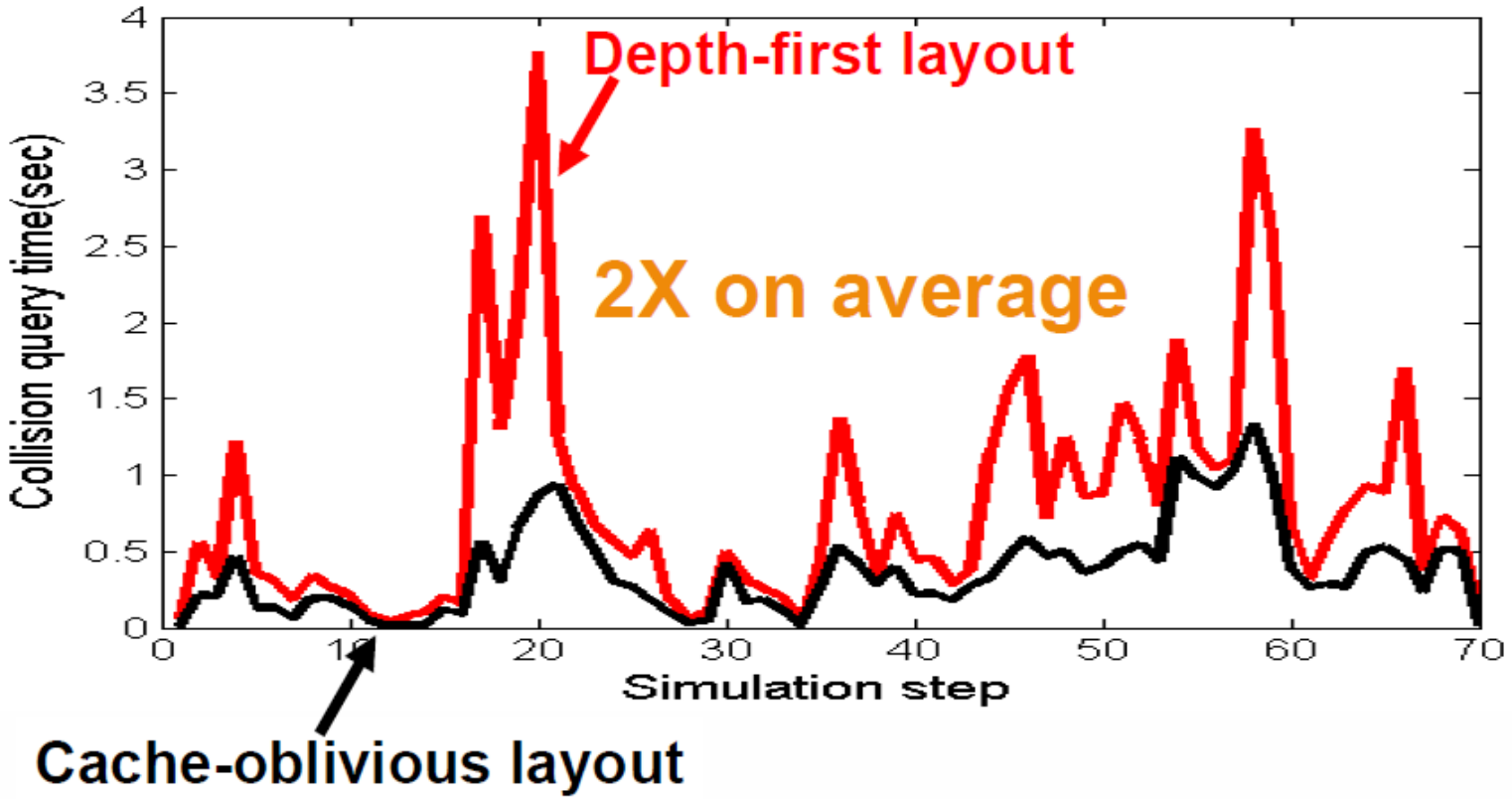


Figure 2: ***Hugo and 1M Power Plant Models:*** The Hugo robot model is placed in the top left of the power plant model, whose overall shape is shown on the right. We are able to achieve 35%–2600% performance improvement in collision detection by using our cache-efficient layouts of the OBB-tree over other tested layouts.

Tecgraf PUC-RIO VR GROUP

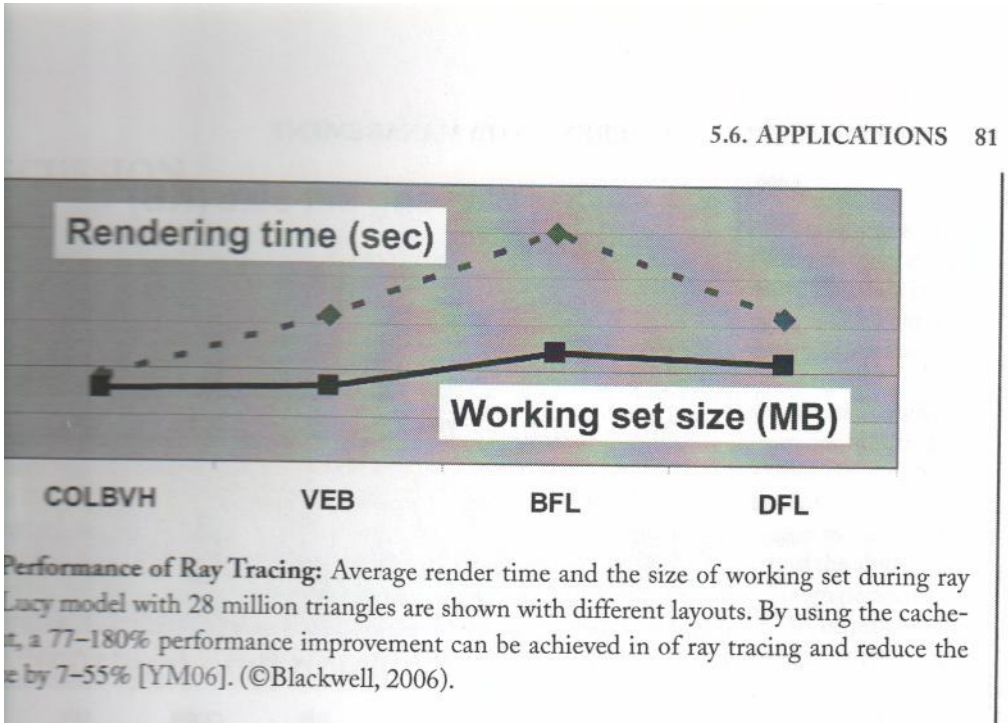
Aplicações

- Detecção de colisão



Aplicações

- Ray-tracing
  - 30% a 300% de melhoria de desempenho



Performance do Ray-tracing com Cache-oblivious layouts of BVH (COLBVH) contra VEB (van Emde Boas layout), Breadth-first (BFL), and Depth-first (DFL) layouts.

Tecgraf PUC-RIO VR GROUP

Aplicações

- Vídeo:
  - http://gamma.cs.unc.edu/COLBVH/Final.mp4

Orthogonal Approaches

- Cache-coherent layouts
- Random-accessible compressed data
- Cache-oblivious ray reordering

Random-accessible compressed data

- Sung-eui Yoon, Peter Lindstrom, "Random-Accessible Compressed Triangle Meshes“, IEEE Transactions on Visualization and Computer Graphics, pp. 1536-1543, November/December, 2007.

- Tae-Joon Kim, Bochang Moon, Duksu Kim, and Sung-Eui Yoon. “RACBVHs: Random-Accessible Compressed Bounding Volume Hierarchies”. IEEE Transactions on Visualization and Computer Graphics, 99, pp. 273-286, (Rapid post) 2009.

- Tae-Joon Kim, Yongyoung Byun, Yongjin Kim, Bochang Moon, Seungyong Lee, Sung-Eui Yoon. “HCCMeshes: Hierarchical-Culling oriented Compact Meshes”. Eurographics 2010.

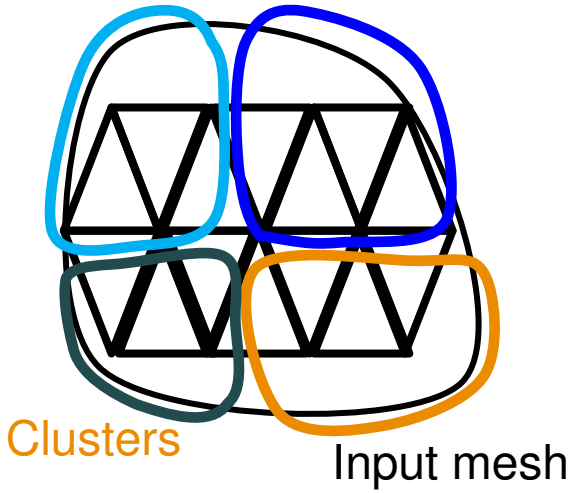
Tecgraf PUC-RIO VR GROUP

Random-accessible compressed data

- Compression methods of meshes and hierarchies
  - Reduce the memory requirements
  - Supports random accesses on meshes and hierarchies
  - Can be useful to many different applications

Random-access compressed meshes

- Decompose the mesh into spatially coherent clusters
  - Efficiently performed by decomposing an original layout
- Compress each cluster separately while preserving the mesh layout
  - Each cluster serves as an access point

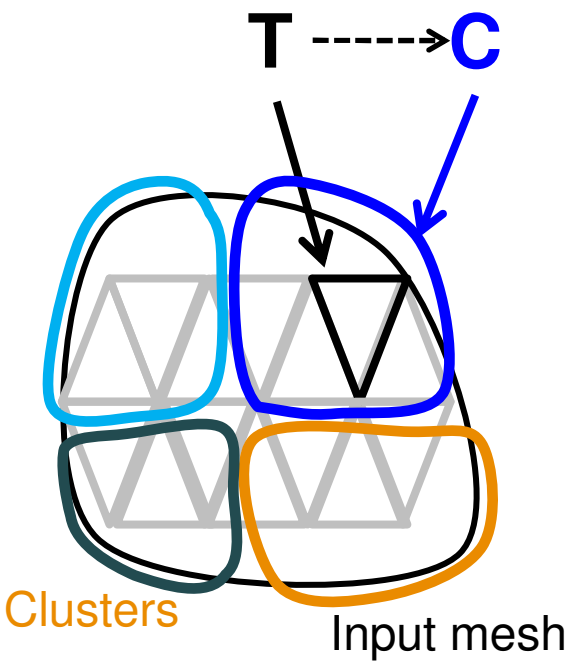


Tecgraf PUC-RIO VR GROUP

Random-access compressed meshes

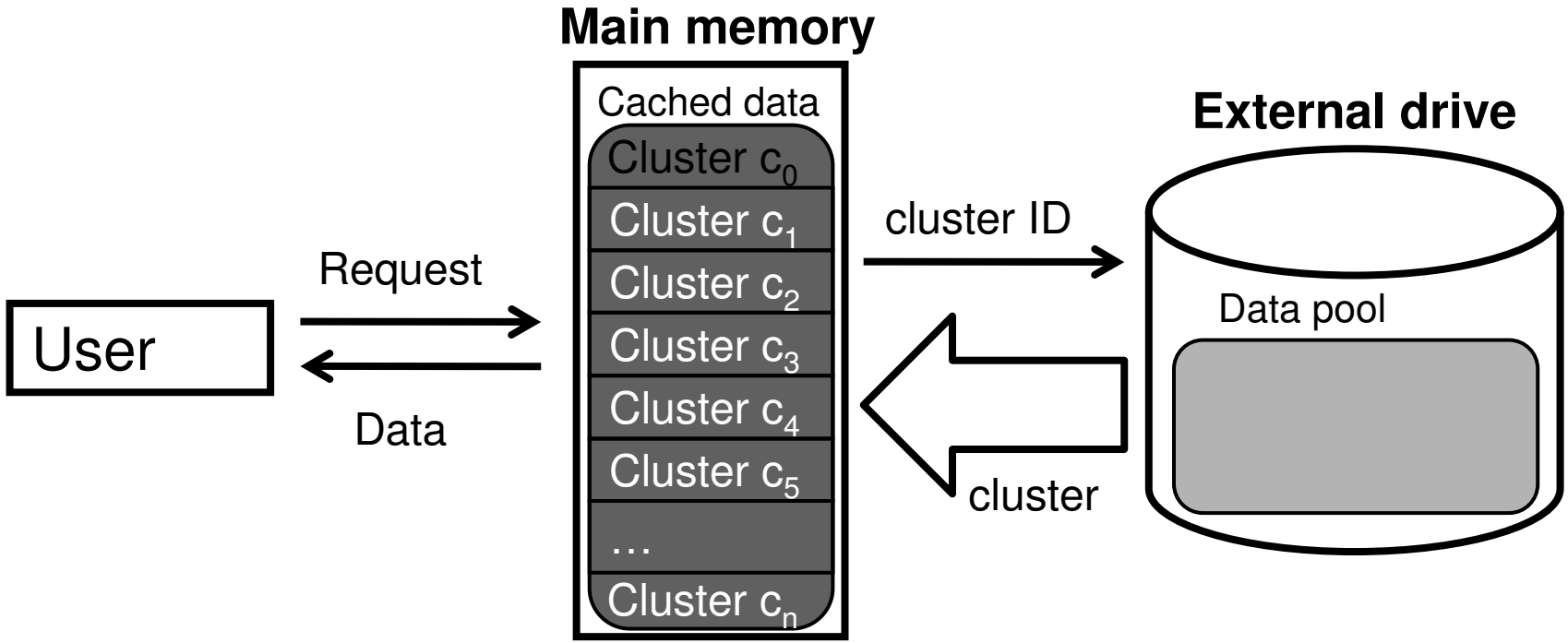
- Decompress the cluster containing a data required by an application

- Dynamically construct connectivity information
  - Provide correct connectivity information for the requested data even with partially loaded data



Tecgraf PUC-RIO VR GROUP

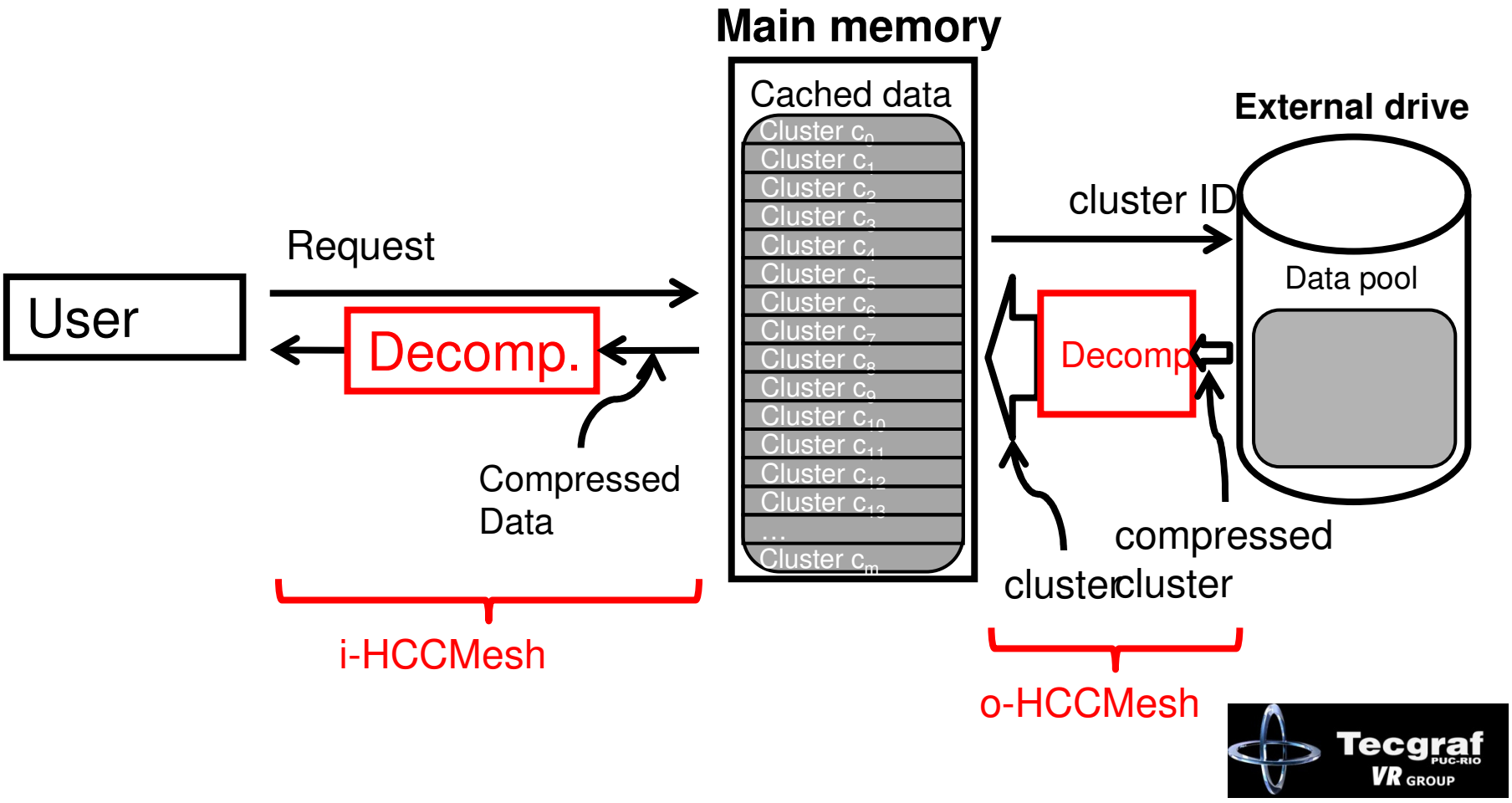
Data Access Framework - Out-of-Core Technique



Tecgraf PUC-RIO VR GROUP

HCCMeshes

Support hierarchical random access!



Main Benefits

- Use a lower memory space and working set size
  - o-HCCMeshes have 20:1 compression ratios
  - i-HCCMeshes have 6:1 compression ratios
- Improve runtime performance
- Vídeo
  - http://sglab.kaist.ac.kr/HCCMesh/HCCMesh_video.mov

Tecgraf PUC-RIO VR GROUP

Orthogonal Approaches

- Cache-coherent layouts
- Random-accessible compressed data
- Cache-oblivious ray reordering

Cache-oblivious ray reordering

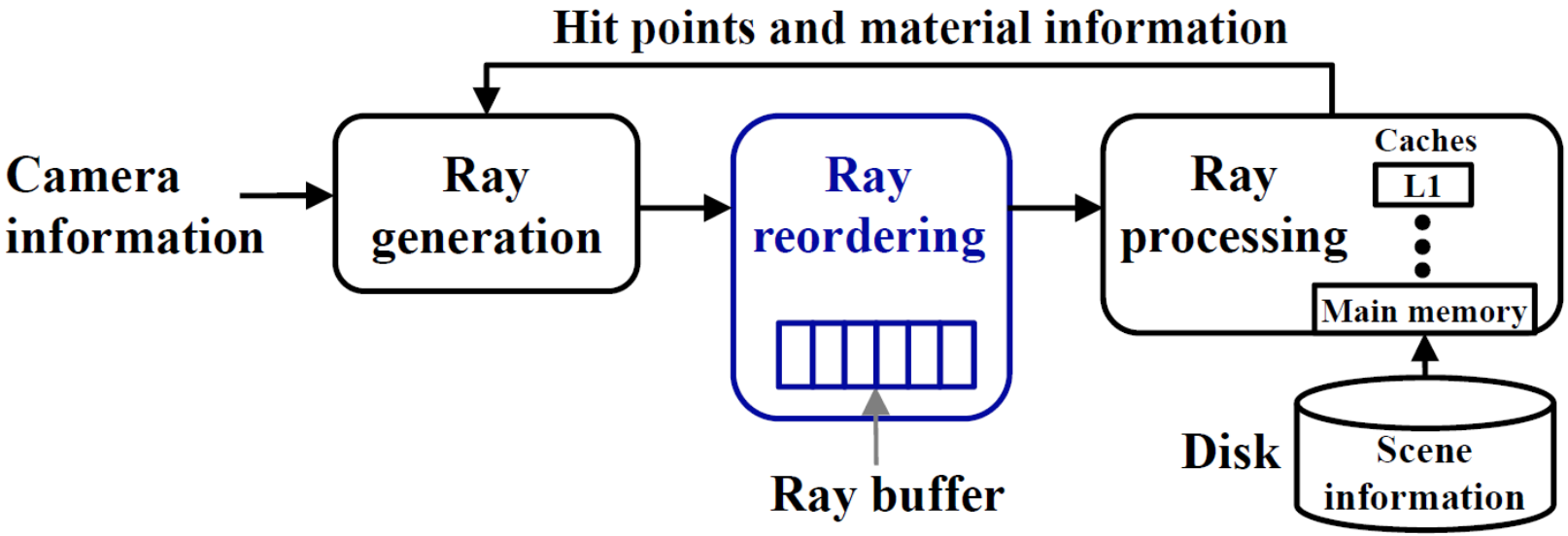
- Bochang Moon, Youngyong Byun, Tae-Joon Kim, Pio Claudio, Hye-sun Kim, Yun-ji Ban, Seung Woo Nam, and Sung-Eui Yoon. “Cache-Oblivious Ray Reordering”, (To appear at) ACM Transactions on Graphics, Vol. 29, No. 3, June 2010.

Tecgraf PUC-RIO VR GROUP

Cache-oblivious ray reordering

- Desafio
  - Secondary rays generated show low ray coherence
    - Result in low cache utilizations
  - In case of ray tracing massive models, expensive cache misses occur
- Objetivo
  - Design an efficient algorithm for converting incoherent secondary rays to coherent
  - Achieve a high cache coherence of these rays
  - The performance improvement of ray tracing

Cache-oblivious ray reordering



Tecgraf PUC-RIO VR GROUP

Exemplo

- This scene has 128 M triangles and takes 15.7 GB.

Vídeo: http://sglab.kaist.ac.kr/CORR/CORR.mov

Tecgraf PUC-RIO VR GROUP

Conclusões (do Cap. 5)

- Métodos de otimização de layout ainda são caros
  - Necessidade de métodos mais rápidos e que produzam otimizações de melhor qualidade.

Tecgraf PUC-RIO VR GROUP

Conclusões (Cap. 6 do livro)

- Visualização de Modelos Massivos é área ativa de pesquisa em CG, visualização científica, GIS, planejamento urbano, etc.
  - Avanços nos últimos 5 anos superam os dos 45 anos anteriores
- Aplicações como Google Earth, Microsoft Virtual Earth, Visible Human Project mostram “mercado” para essas ideias
- O livro focou em
  - Visibility culling
  - Redução de complexidade
  - Cache-friendly techniques

Conclusões (Cap. 6 do livro)

- Apesar dos tópicos terem sido focados em rasterização, muitas das técnicas apresentadas podem ser aplicadas em Ray-Tracing
  - Tema a ser abordado na sequência do nosso curso
- O livro não focou em
  - Modelos dinâmicos
    - Maior parte das técnicas apresentadas precisam ser recomputadas quando modelo se altera... Perde-se tempo-real!
  - Integração com outras técnicas interativas
    - Navegação multi-escala, detecção de colisão, etc.
  - Renderização foto-realista e iluminação global