Multi-Resolution Techniques

Sung-Eui Yoon
(윤성의)

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

At the Previous Class

- **The overview of the course**
  - **Culling techniques**

Problems

- Even after visibility culling we can still have too many visible triangles

372 million triangles (10GB)

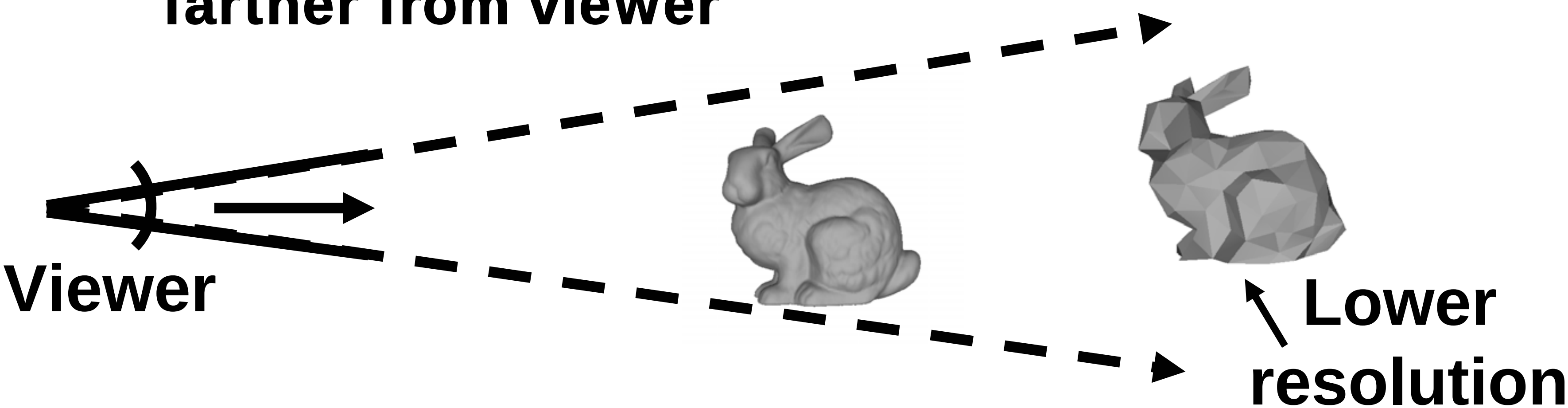
13M Triangles

Problems

- **Won't this problem go away with faster GPUs?**
  - **The real world has virtually infinite complexity**
  - **Our ability to model and capture this complexity can outpace rendering performance**

Multi-Resolution or Levels of Detail

- Basic idea
  - Render with fewer triangles when model is farther from viewer



- Methods
  - Polygonal simplification
  - Parametric and subdivision surfaces
  - Image impostors

Polygonal Simplification

- **Method for reducing the polygon count of mesh**

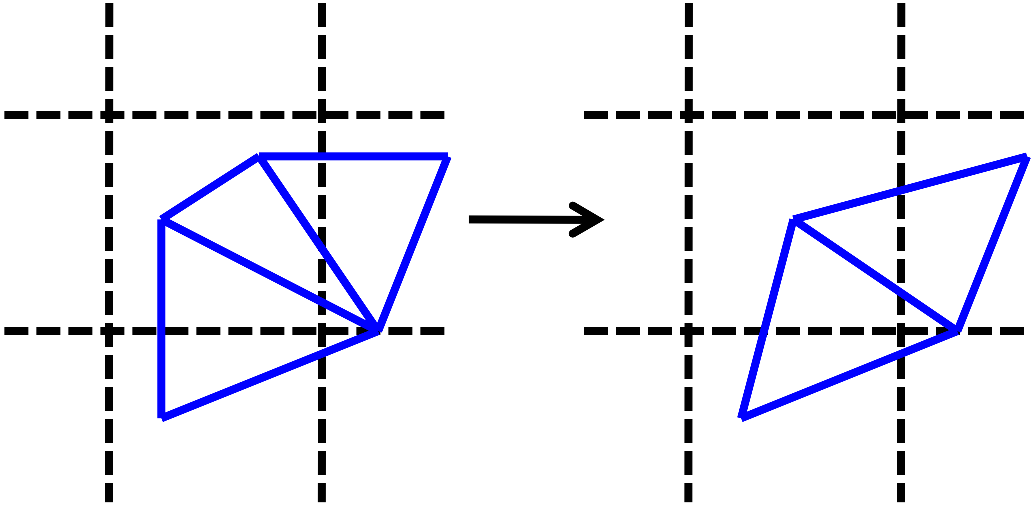
- **Two main components**
  - **Simplification operators**
  - **Simplification error metrics**

Simplification Operators

- Vertex clustering
- Edge collapse
- Vertex removal

Vertex Clustering [Rossignac & Borrel 93]

- Impose a grid on the model
- Compute weighted average vertex in each cell
- Triangles become:
  - Triangles
  - Lines
  - Points

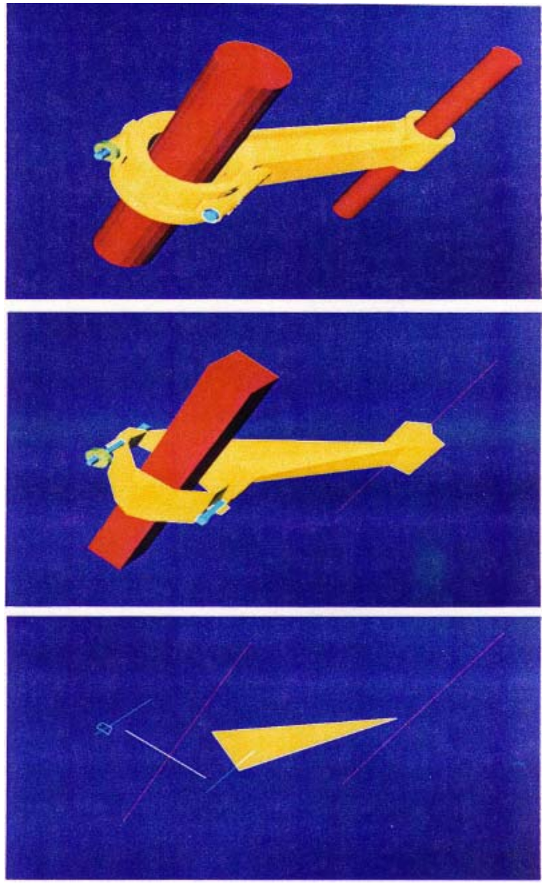


KAIST

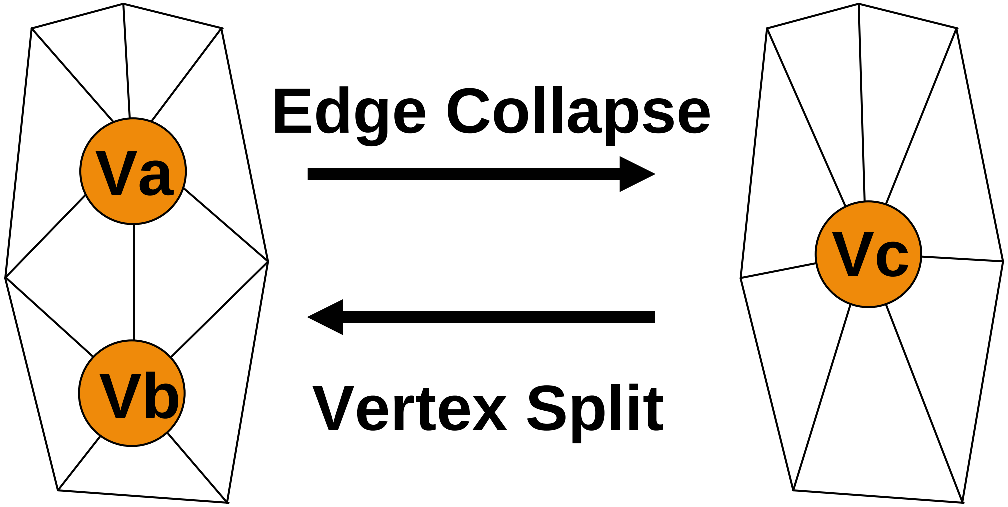
Vertex Clustering

- Main benefits
  - Simple and robust

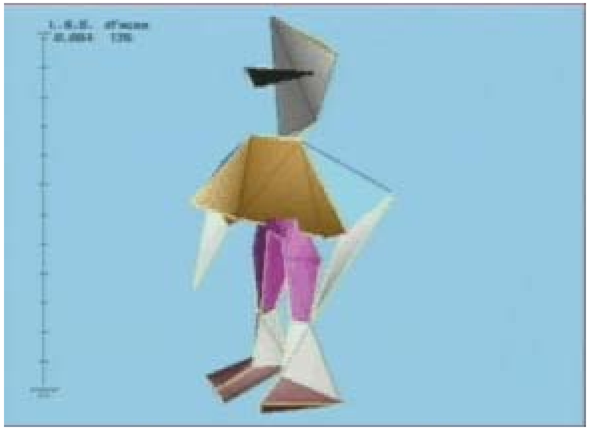
- Disadvantages
  - Hard to target a polygon count
  - Poor error control



Edge Collapse [Hoppe93]

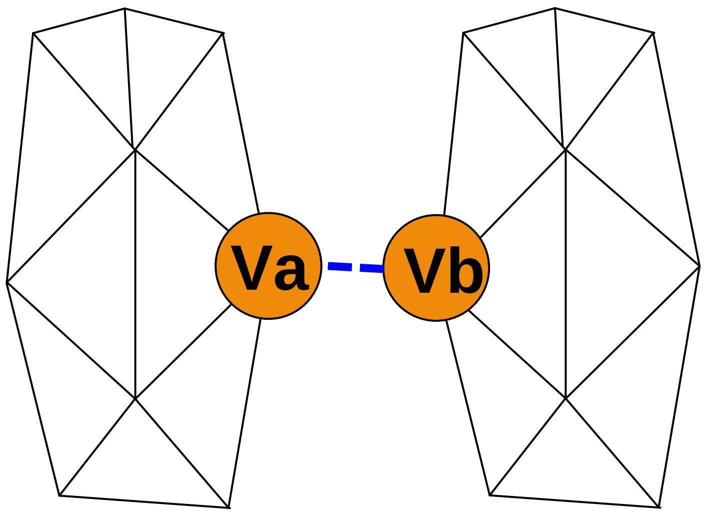


- **Fine grained**
  - **Reduce two triangles at most**
- **Allows simple geomorphs**
- **Topology preserving**



Virtual Edge Collapse

- Extension of edge collapse to two vertices not connected by an edge
  - Allows topological simplification
  - Usually limited to small distance to avoid $O(n^2)$ virtual edges



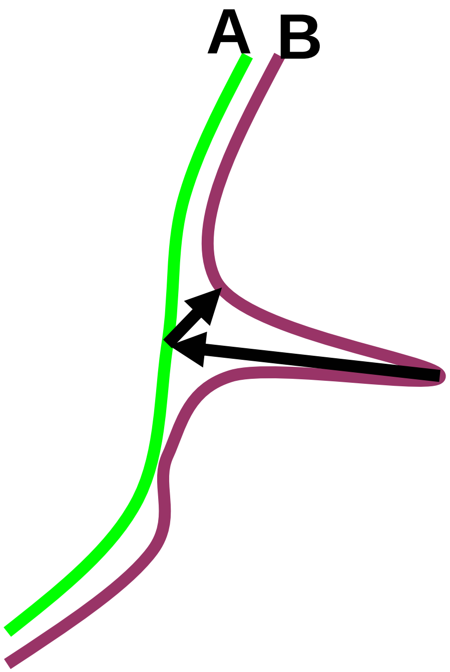
Simplification Error Metric

- Control and quantify the quality of the LOD
  - Ultimately, we want to measure appearance
  - Typically, we use a geometric measure as an approximation
- Error measures are used in three ways
  - To pick which parts are simplified
  - To determine the simplified mesh from the operation (e.g. position of the new vertex)
  - To choose an LOD at runtime
- Two common LOD selection criteria
  - Target frame rate vs. target quality

KAIST

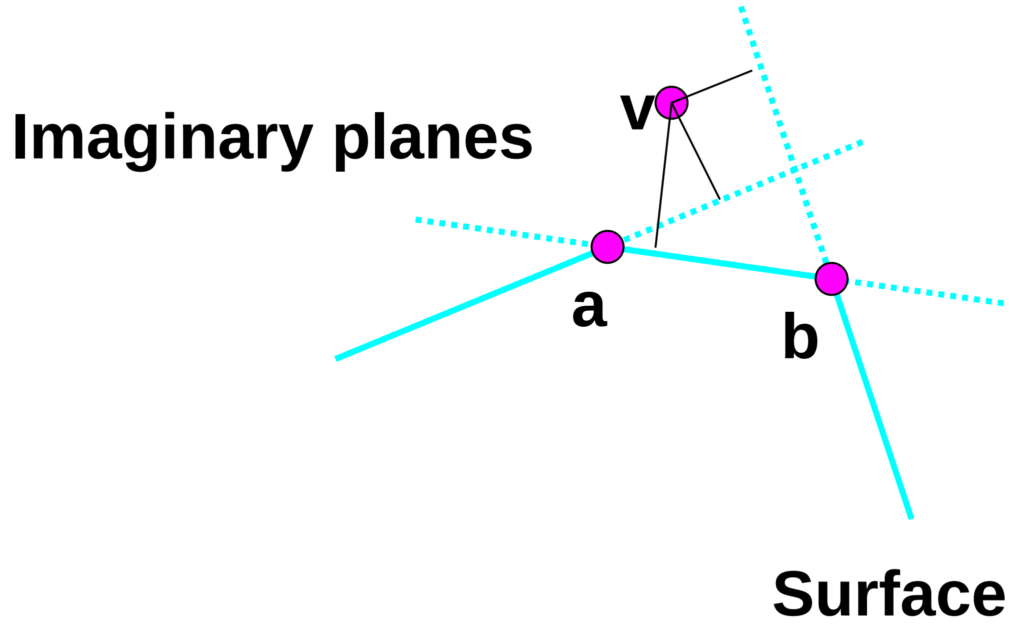
Hausdorff Distance

- **A measure of surface deviation**
  - **$H(A,B)=\max(h(a,b),h(b,a))$, where $h(A,B)=\max_a \min_b(|a-b|)$**
  - **h is called the one-sided Hausdorff distance**
- **Provides a bound on the maximum possible error**
  - **Project to screen space to get deviation in pixels**



Vertex Plane Distance [Ranford 96]

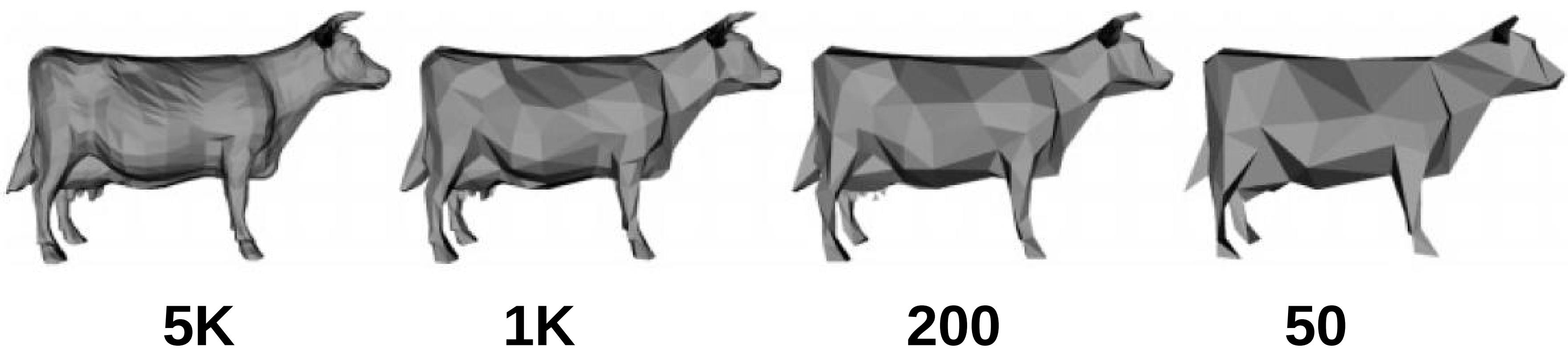
- One metric is the max distance between the vertex and the planes of the supported triangles



- $E = \max_p (p \bullet v)$

Quadric Error Metric [Garland & Heckbert 97]

- Use sum of squared distance rather than max distance
  - Can be efficiently computed and empirically shows very good results



Excerpted from [Garland & Heckert 97]

KAIST

Attributes

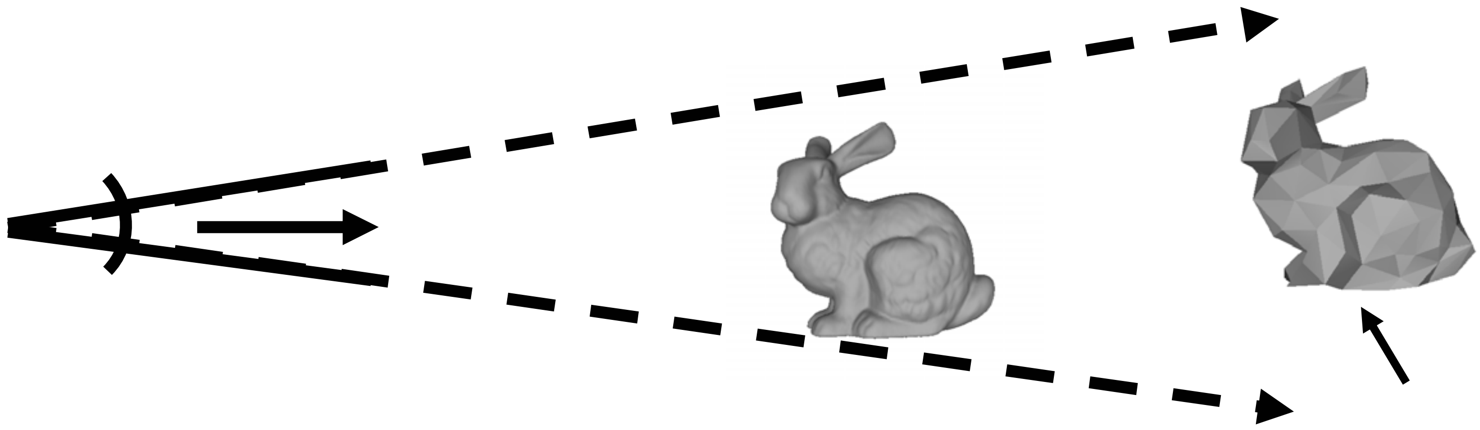
- Vertices have more than just position:
  - Colors
  - Normals
  - Texture coords
  - Shader programs

Overall Simplification Process

- Compute simplification candidates
- While (there is candidate)
  - Pick a candidate with the smallest error
  - Perform the simplification

View-Dependent Rendering

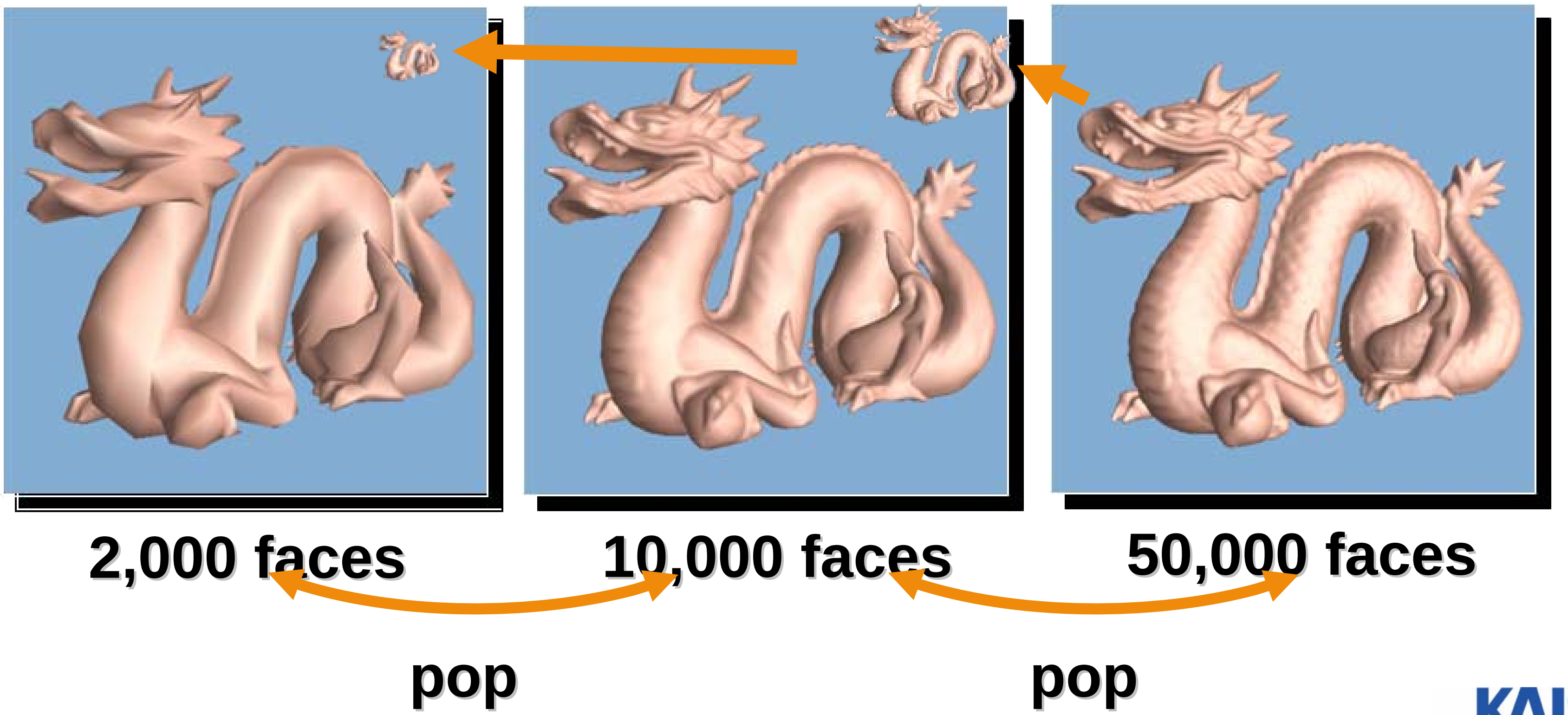
- Use different resolutions according to view points
  - [Clark 76, Funkhouser and Sequin 93]



- Static levels-of-detail (LODs)
- Dynamic (or view-dependent) simplification

Static LODs

- Pre-compute discrete simplified meshes
  - Switch between them at runtime
  - Has very low LOD selection overhead

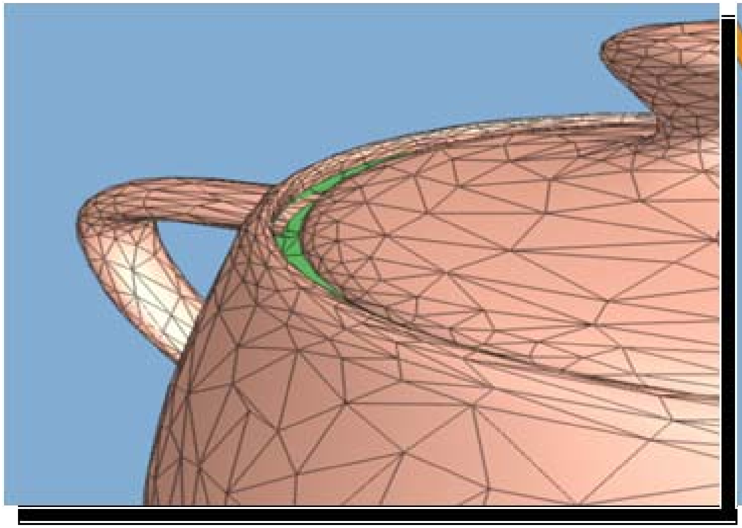


Excerpted from Hoppe's slides

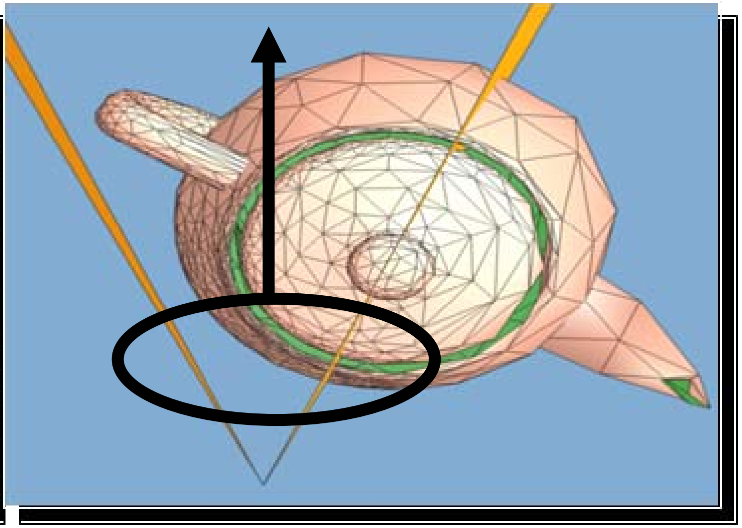
Dynamic Simplification

- **Provides smooth and varying LODs over the mesh** **[Hoppe 97]**

1st person's view

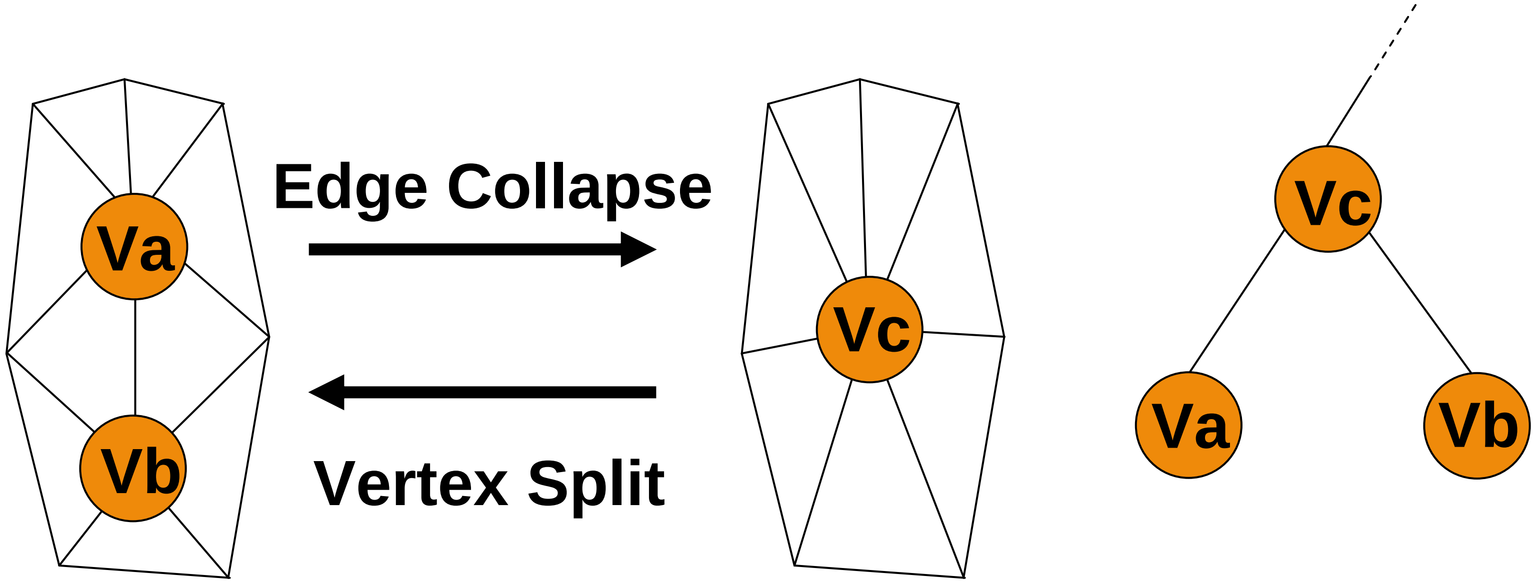


3rd person's view

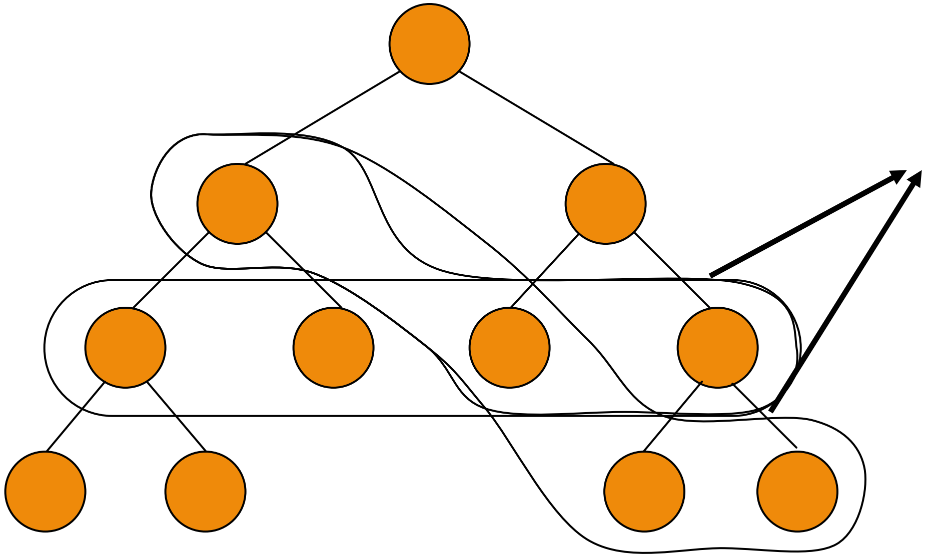


Play video

Vertex Hierarchy



View-Dependent Refinement



Front representing a LOD mesh

Dynamic Simplification: Issues

- Representation
- Construction
- Runtime computation
- Integration with other acceleration techniques

Dynamic Simplification: Issues

- Representation
- Construction
- Runtime computation
- Integration with other acceleration techniques

22+GB for 100M triangles [Hoppe 97]

Dynamic Simplification: Issues

- Representation
- Construction
- Runtime computation
- Integration with other acceleration techniques

Dynamic Simplification: Issues

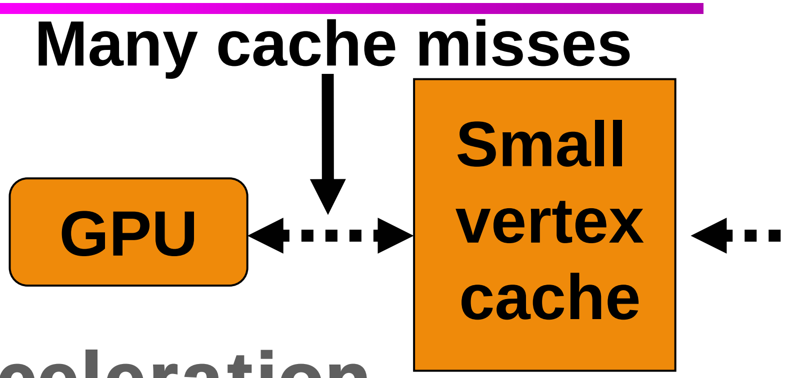
- Representation
- Construction
- Runtime computation
- Integration with other acceleration techniques

Rendering throughput of 3M triangles per sec

[Lindstrom 03]

Dynamic Simplification: Issues

- Representation
- Construction
- Runtime computation
- Integration with other acceleration techniques



Dynamic Simplification: Issues

- Representation
- Construction
- Runtime computation
- Integration with other acceleration techniques

Toward Scale-able Dynamic Simplification Method

- View-dependent rendering [Yoon et al. VIS 04]
  - New multi-resolution hierarchy (CHPM)
  - Out-of-core construction for general meshes
  - Applied to collision detection [Yoon et al. SGP 04] and shadow computation [Lloyd et al. EGSR 06]
- Cache-oblivious layouts [Yoon et al. SIG 05]
  - High GPU utilization during rendering

KAIST

Live Demo – View-Dependent Rendering [Yoon et al., SIG 05]

30 Pixels of error

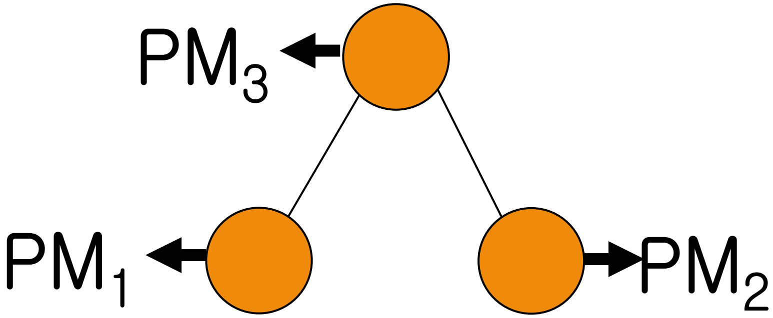
Pentium 4

GeForce Go 6800 Ultra

1GB RAM

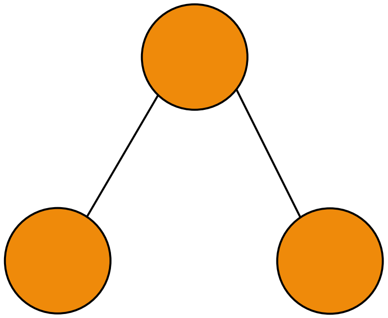
Clustered Hierarchy of Progressive Meshes (CHPM)

- **Novel dynamic simplification representation**
  - **Cluster hierarchy**
  - **Progressive meshes**



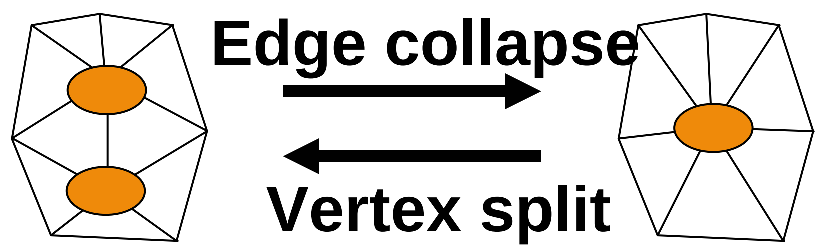
Clustered Hierarchy of Progressive Meshes (CHPM)

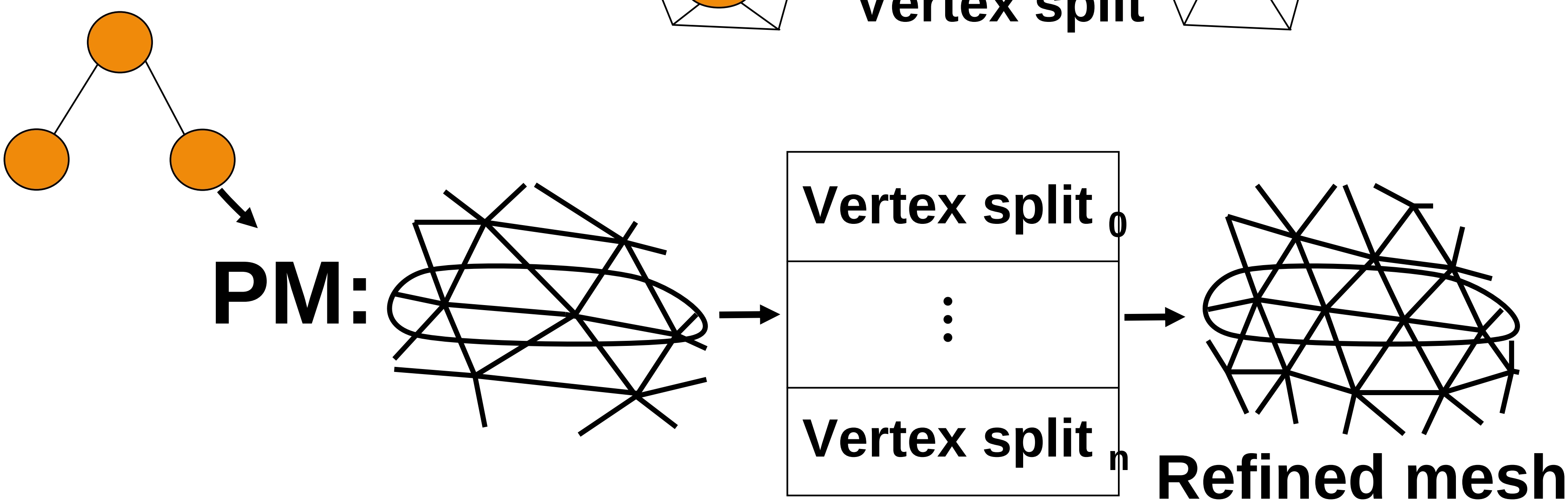
- **Clusters**
  - **Spatially localized regions of the mesh**
  - **Main processing unit for view-dependent computation**
- **Cluster hierarchy**
  - **Used for visibility computations and out-of-core rendering**



Clustered Hierarchy of Progressive Meshes (CHPM)

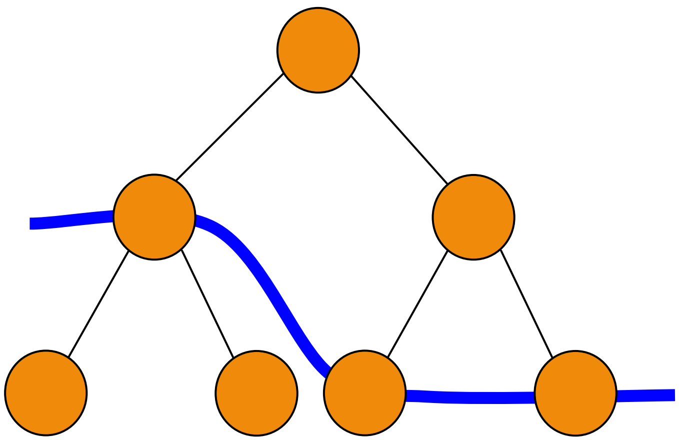
- **Progressive mesh (PM) [Hoppe 96]**
  - **Each cluster contains a PM as an LOD representation**





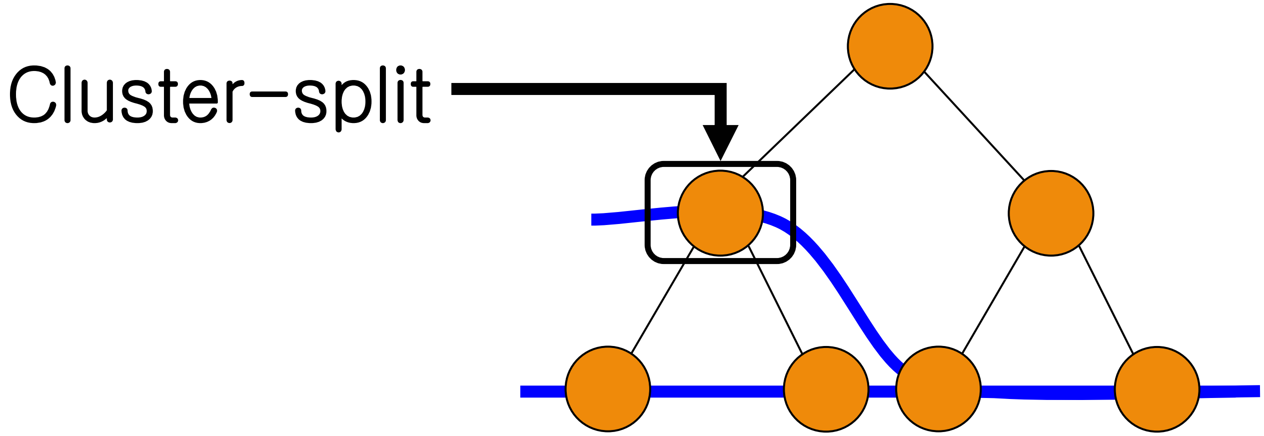
Two-Levels of Refinement at Runtime

- **Coarse-grained view-dependent refinement**
  - **Provided by selecting a front in the cluster hierarchy**
  - **Inter-cluster level refinements**



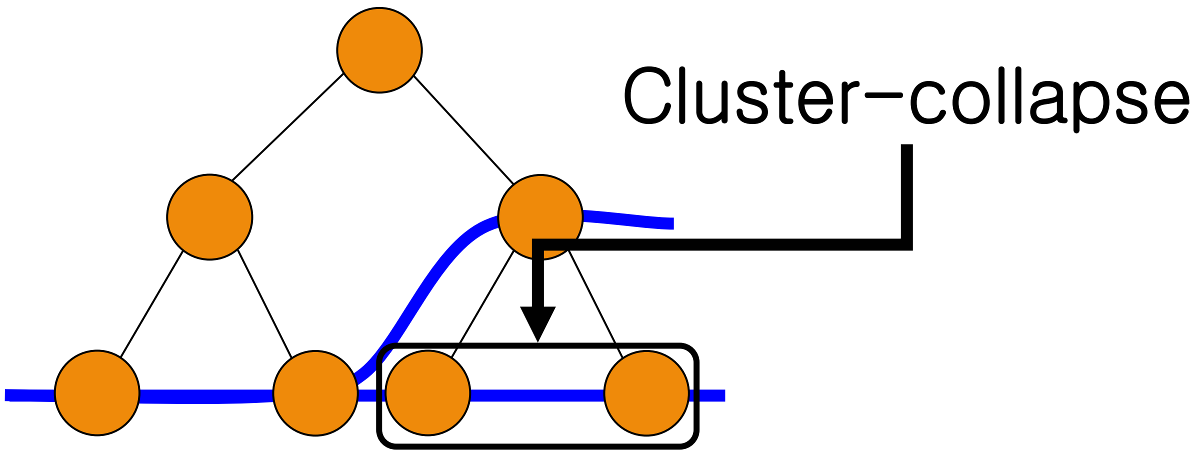
Two-Levels of Refinement at Runtime

- **Coarse-grained view-dependent refinement**
  - **Provided by selecting a front in the cluster hierarchy**
  - **Inter-cluster level refinements**



Two-Levels of Refinement at Runtime

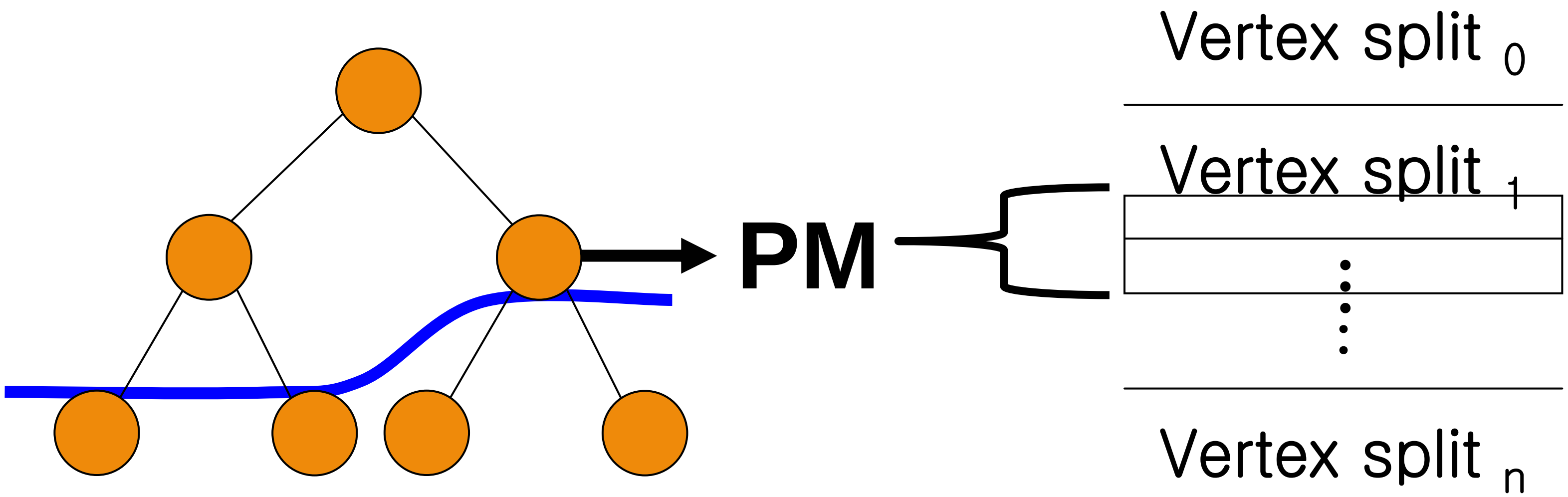
- **Coarse-grained view-dependent refinement**
  - **Provided by selecting a front in the cluster hierarchy**
  - **Inter-cluster level refinements**



Two-Levels of Refinement at Runtime

- **Fine-grained local refinement**
  - **Supported by performing vertex splits in PMs**
  - **Intra-cluster refinements**

Efficient and effective representation for massive models!



Main Properties of CHPM

- Low refinement cost
  - 1 or 2 order of magnitude lower than previous representations
- Alleviates visual popping artifacts
  - Provides smooth transition between different LODs

Overview of Building a CHPM

Input model

Cluster decomposition

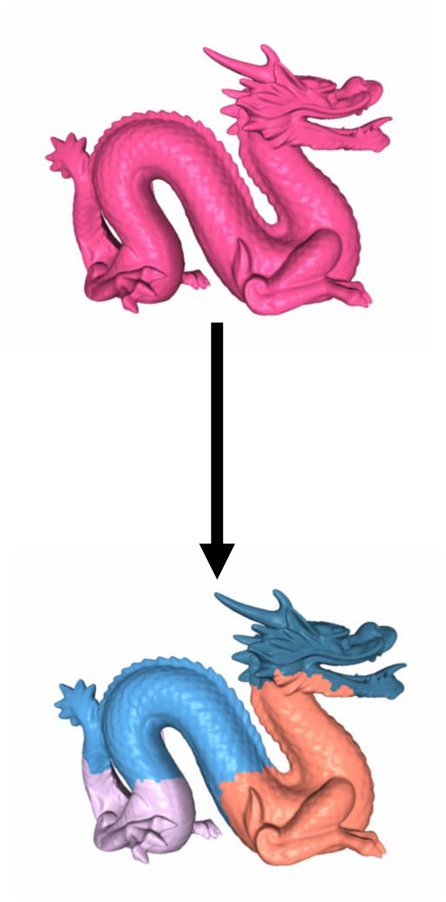
Cluster hierarchy generation

Performed out-of-core

Hierarchical simplification

CHPM

Overview of Building a CHPM



Input model

↓

Cluster decomposition

↓

Cluster hierarchy generation

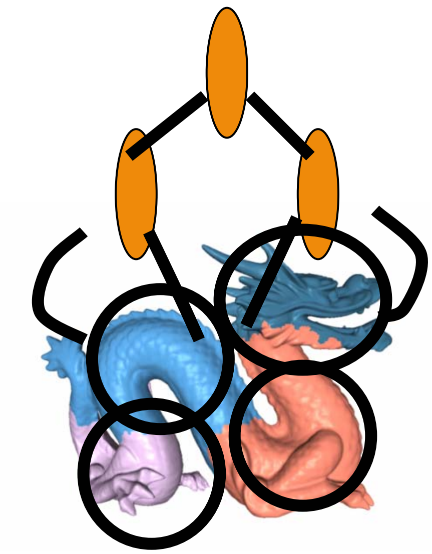
↓

Hierarchical simplification

↓

CHPM

Overview of Building a CHPM



Input model

↓

Cluster decomposition

↓

Cluster hierarchy generation

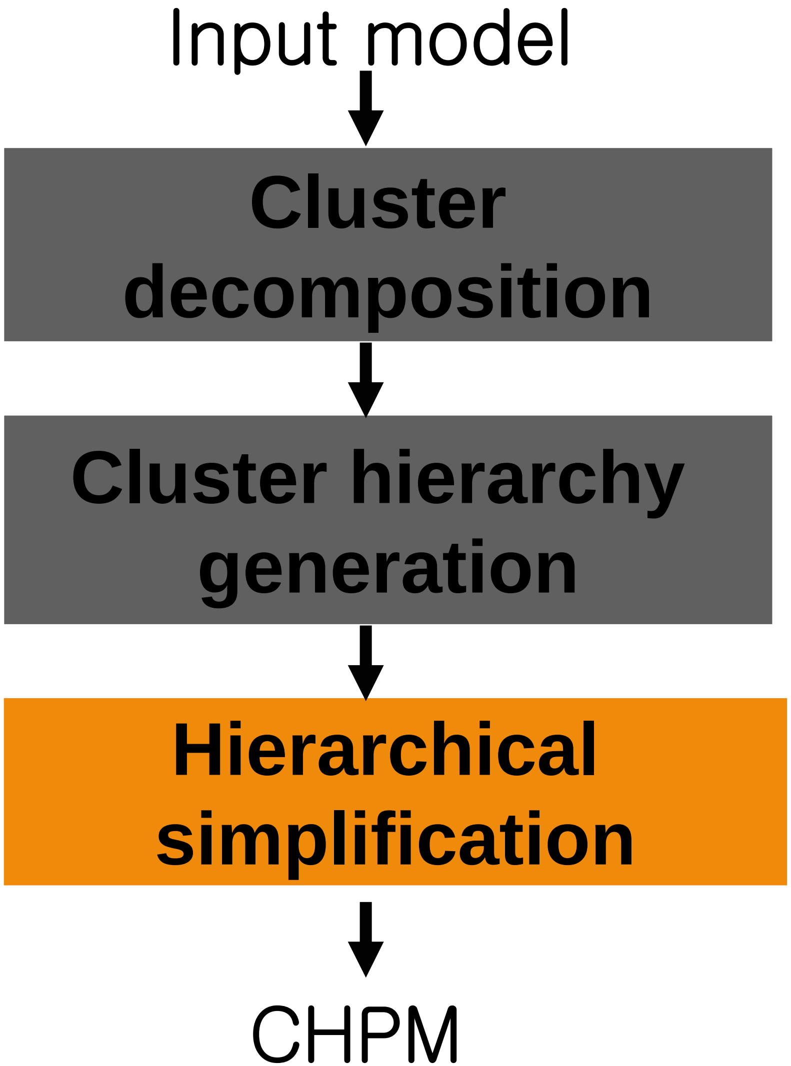
↓

Hierarchical simplification

↓

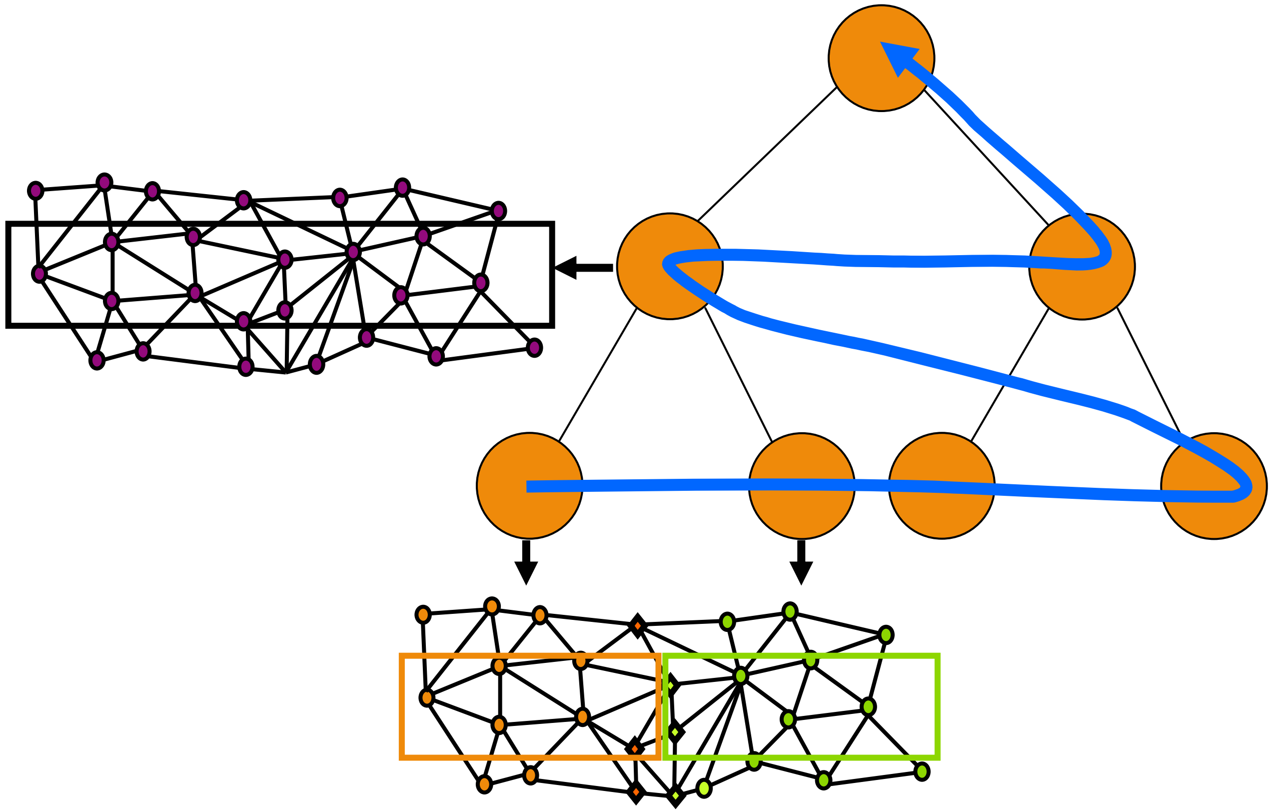
CHPM

Overview of Building a CHPM



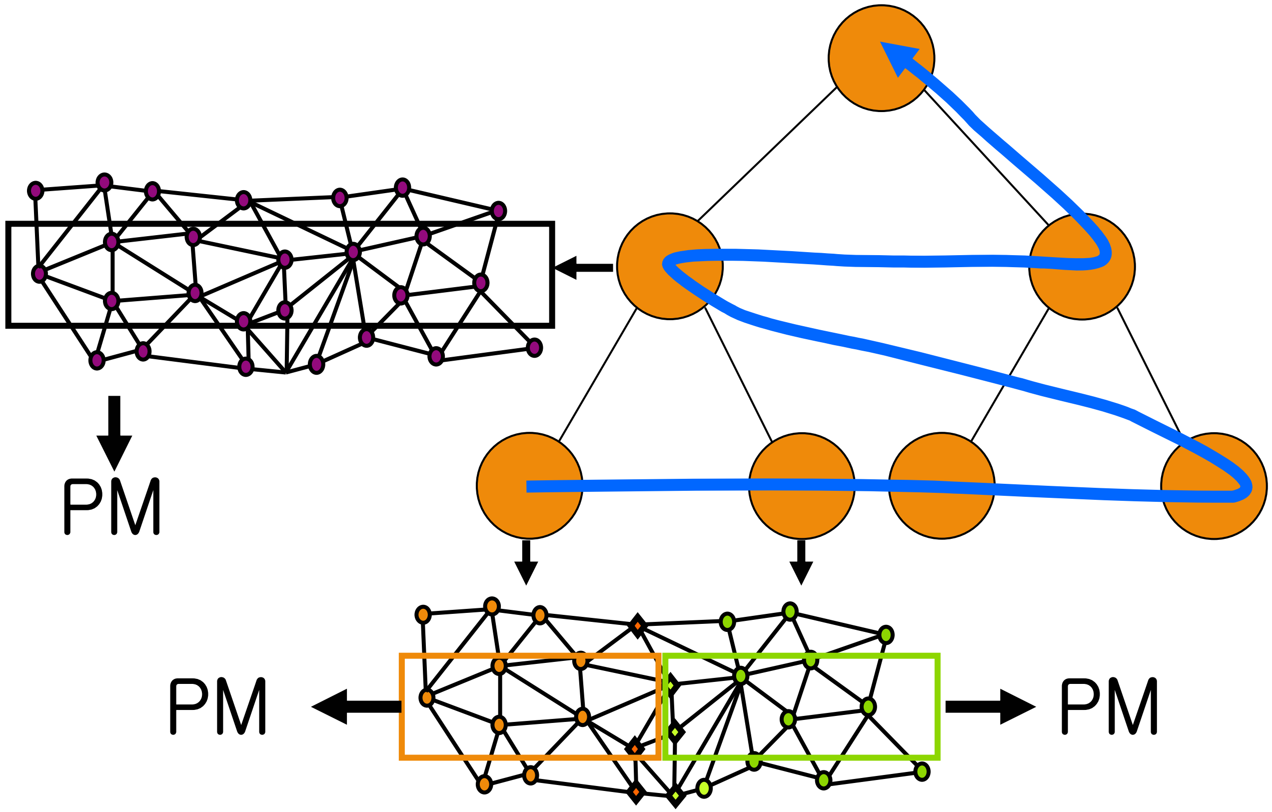
Out-of-Core Hierarchical Simplification

- **Simplifies clusters bottom-up**

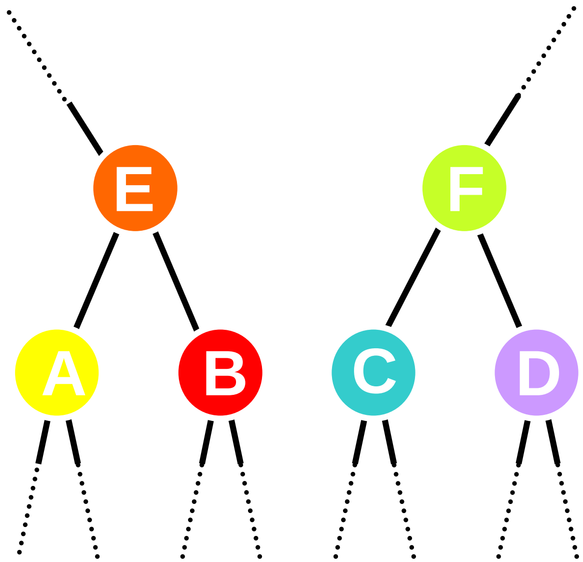


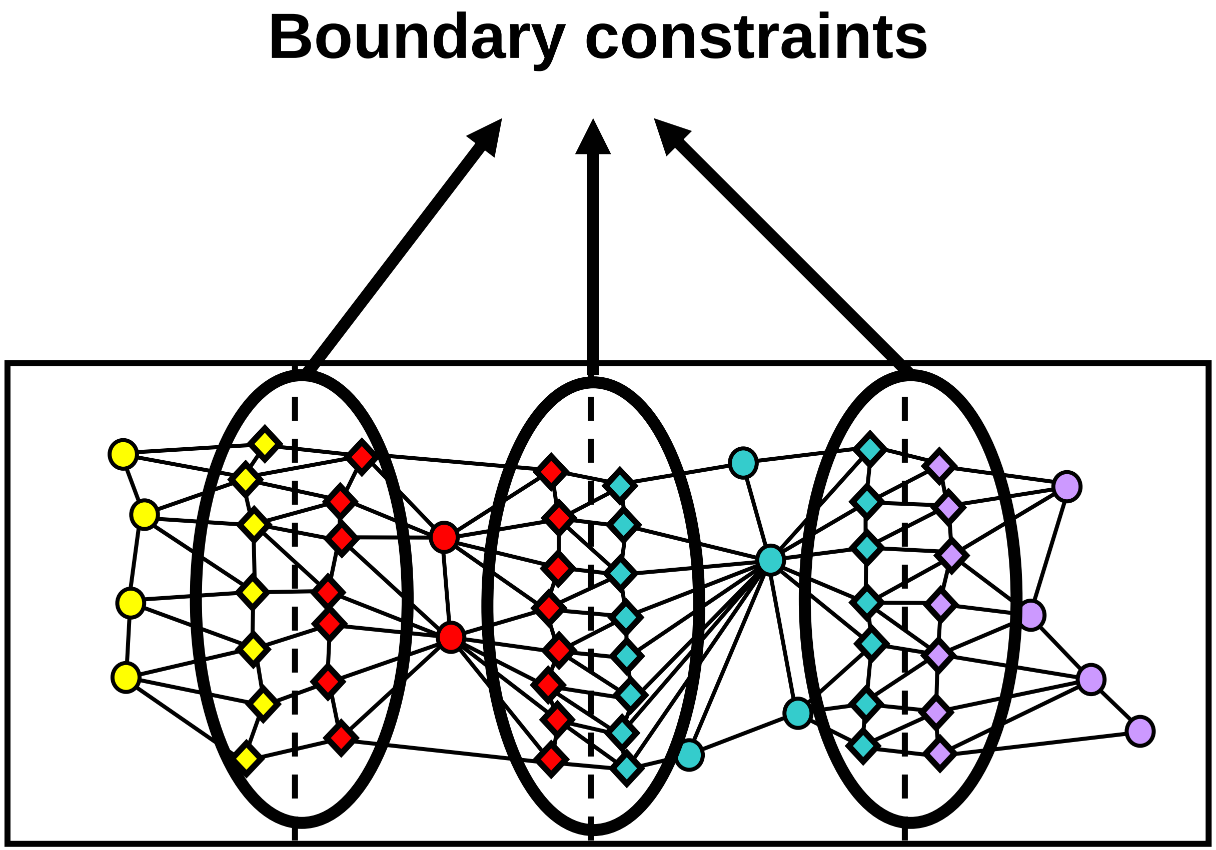
Out-of-Core Hierarchical Simplification

- **Simplifies clusters bottom-up**

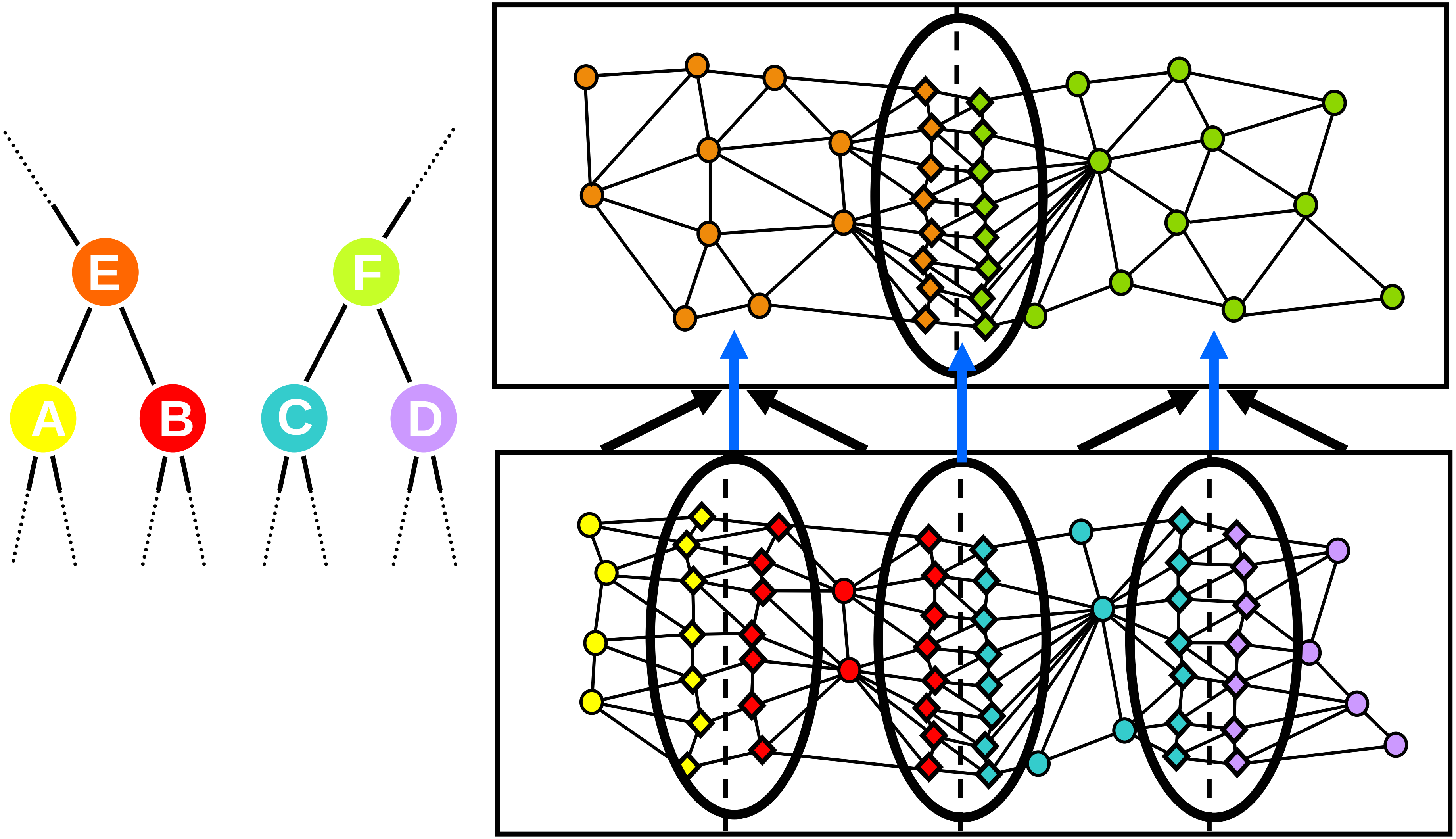


Boundary Simplification





Boundary Simplification

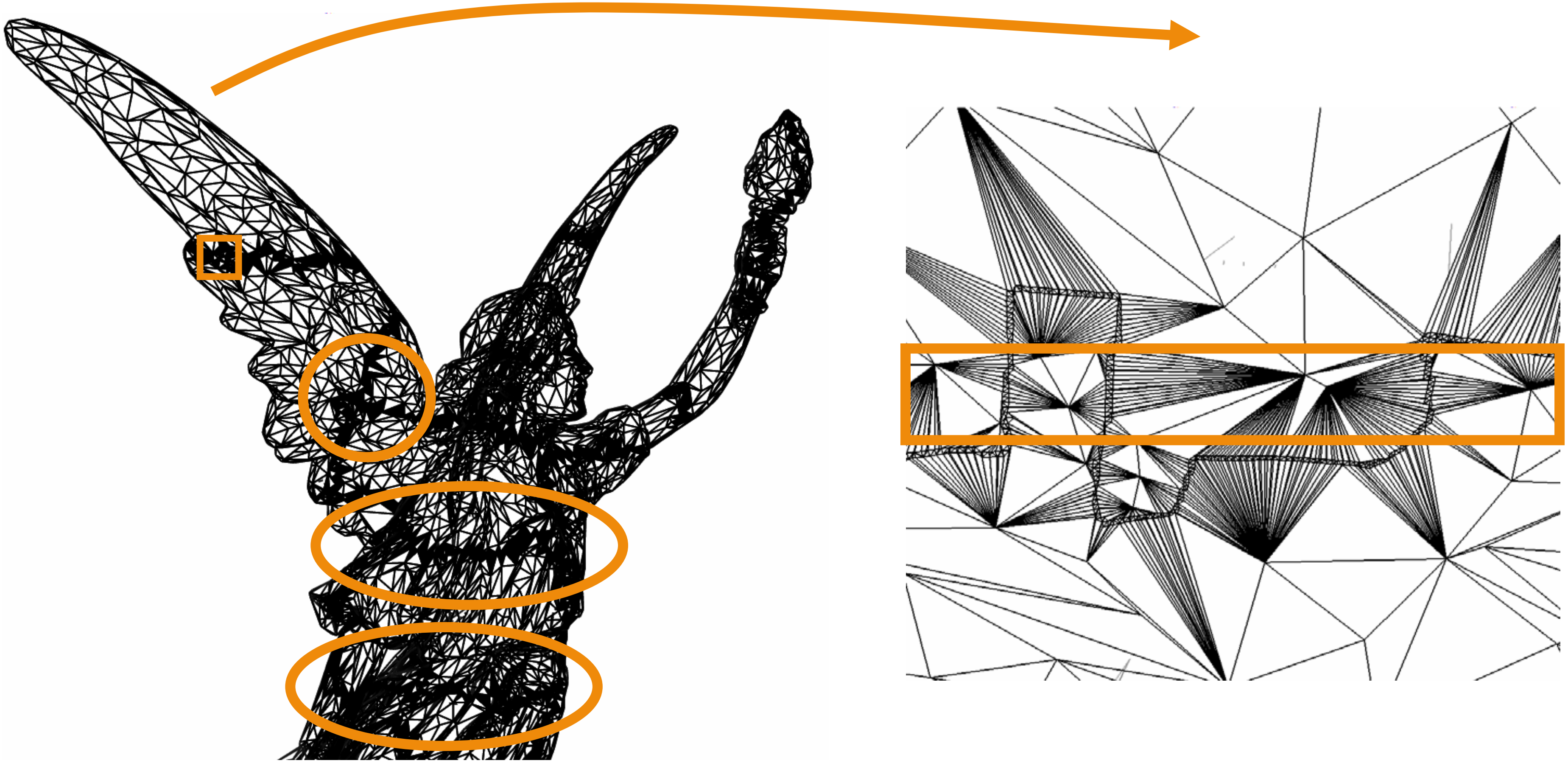


Boundary Constraints

- Common problem in many hierarchical simplification algorithms
  - [Hoppe 98; Prince 00; Govindaraju et al. 03]

Boundary Constraints

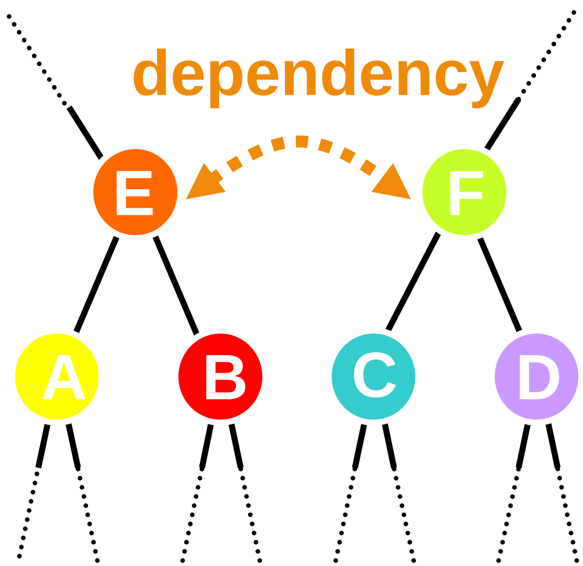
Boundary Constraints

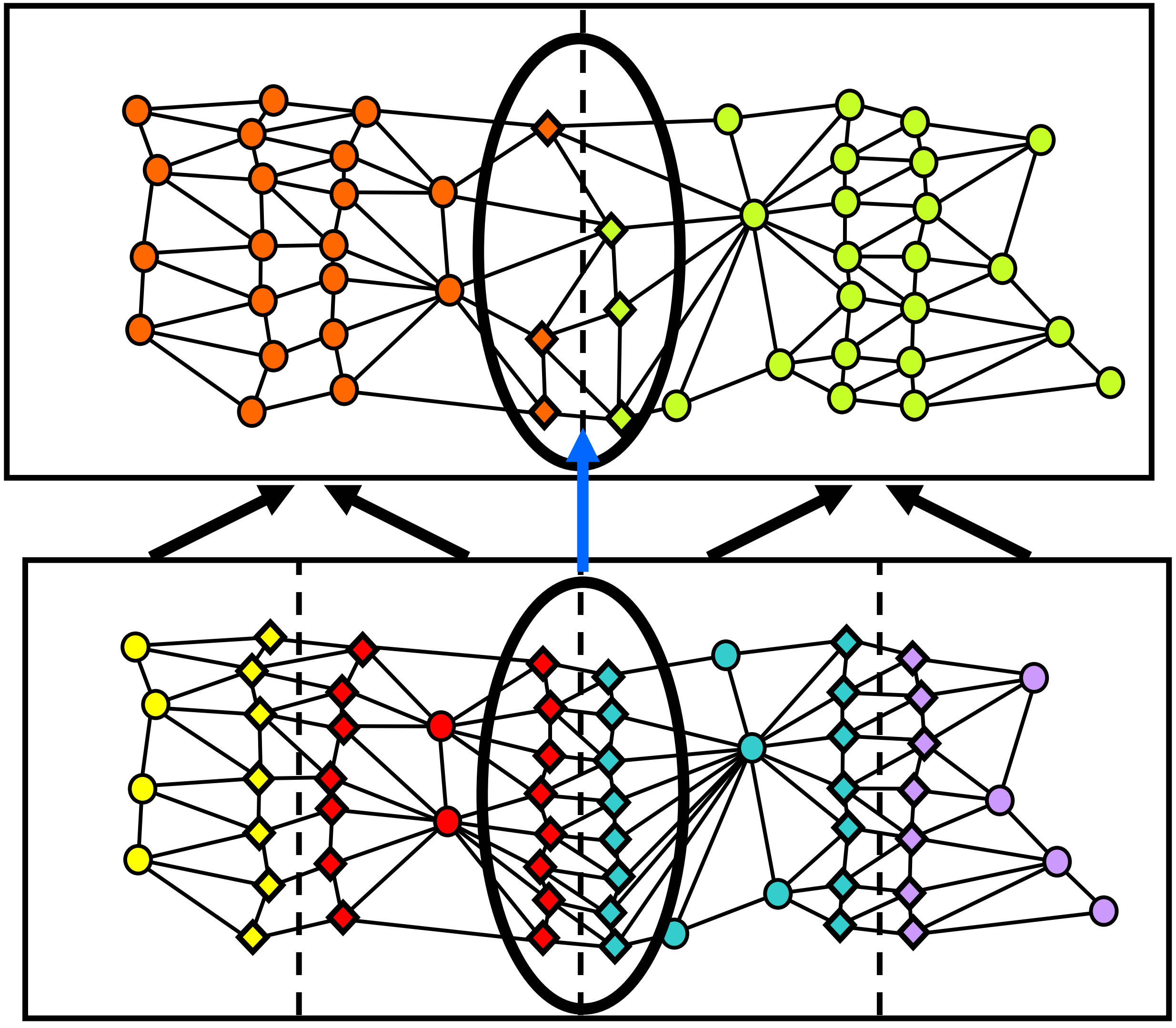


Cluster Dependencies

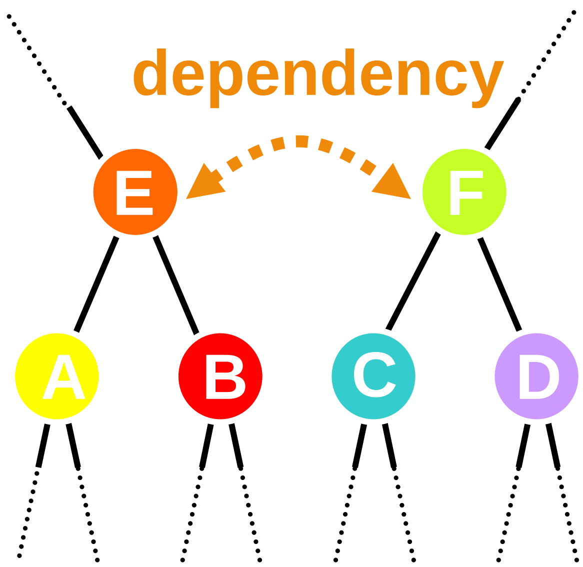
- Replaces preprocessing constraints with runtime dependencies

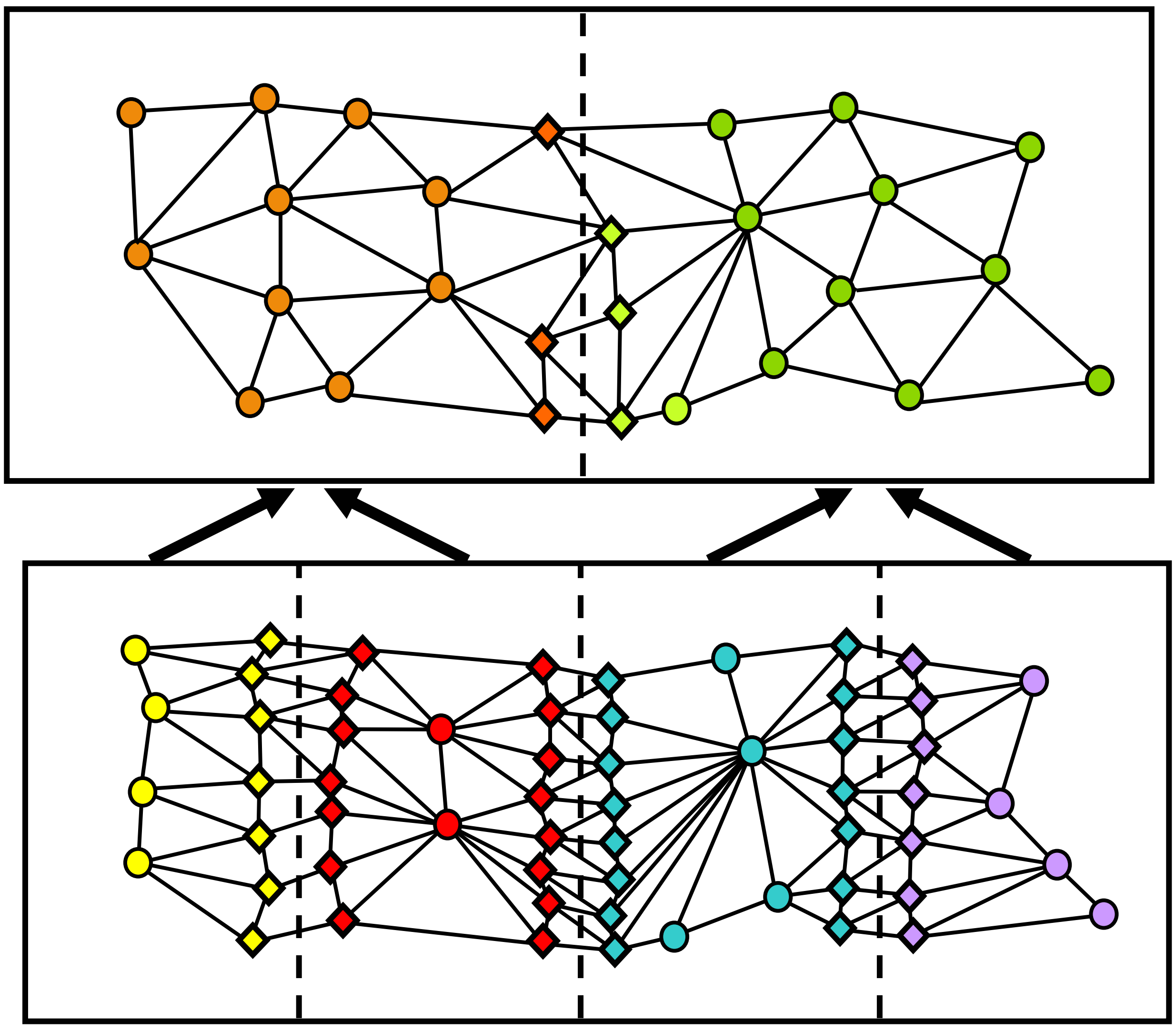
Cluster Dependencies



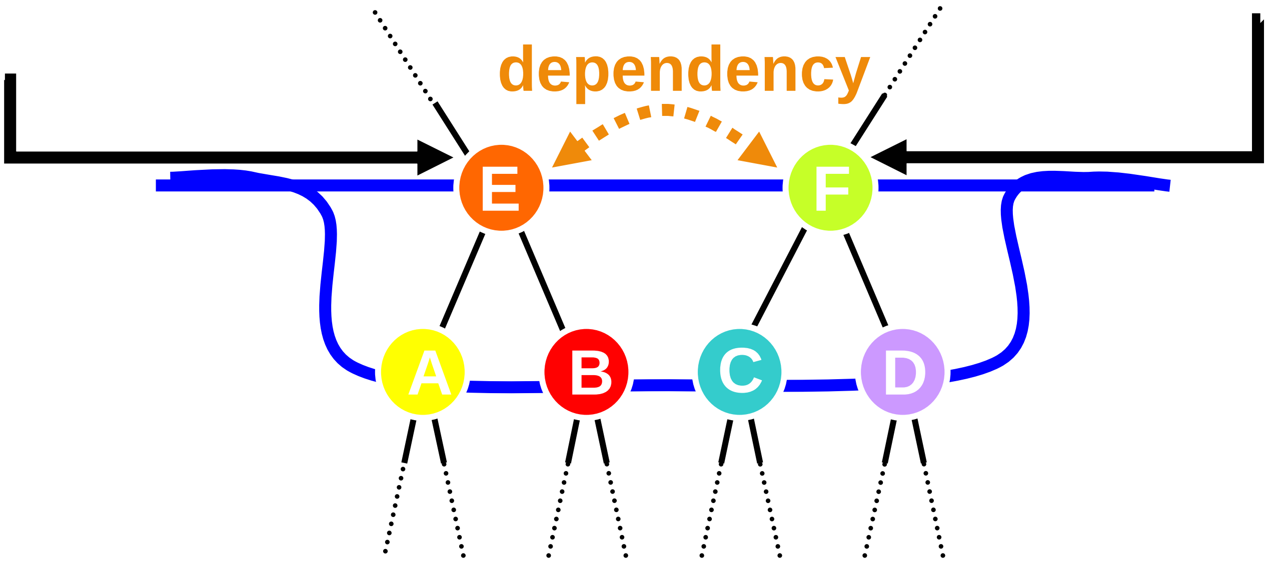


Cluster Dependencies



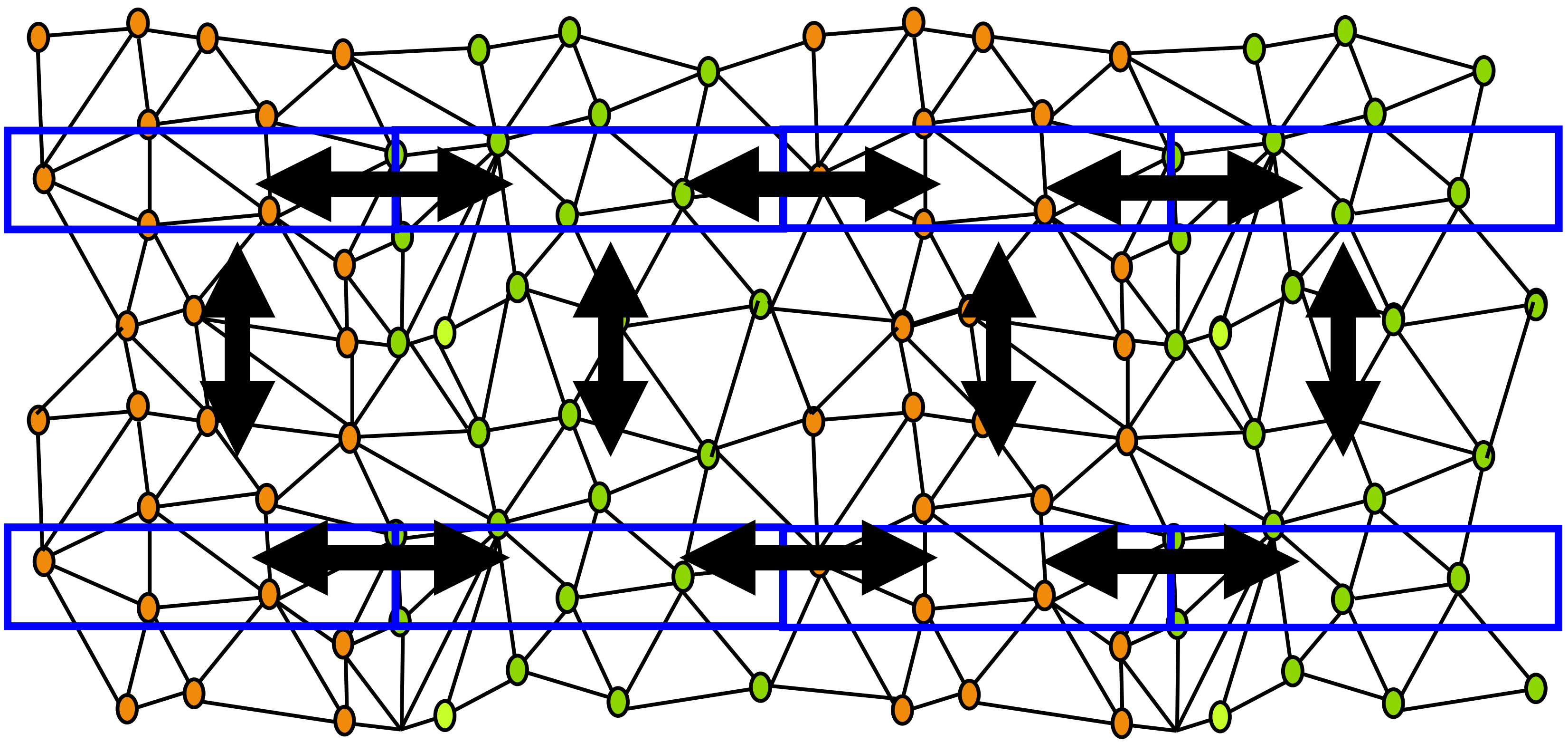


Cluster Dependencies at Runtime

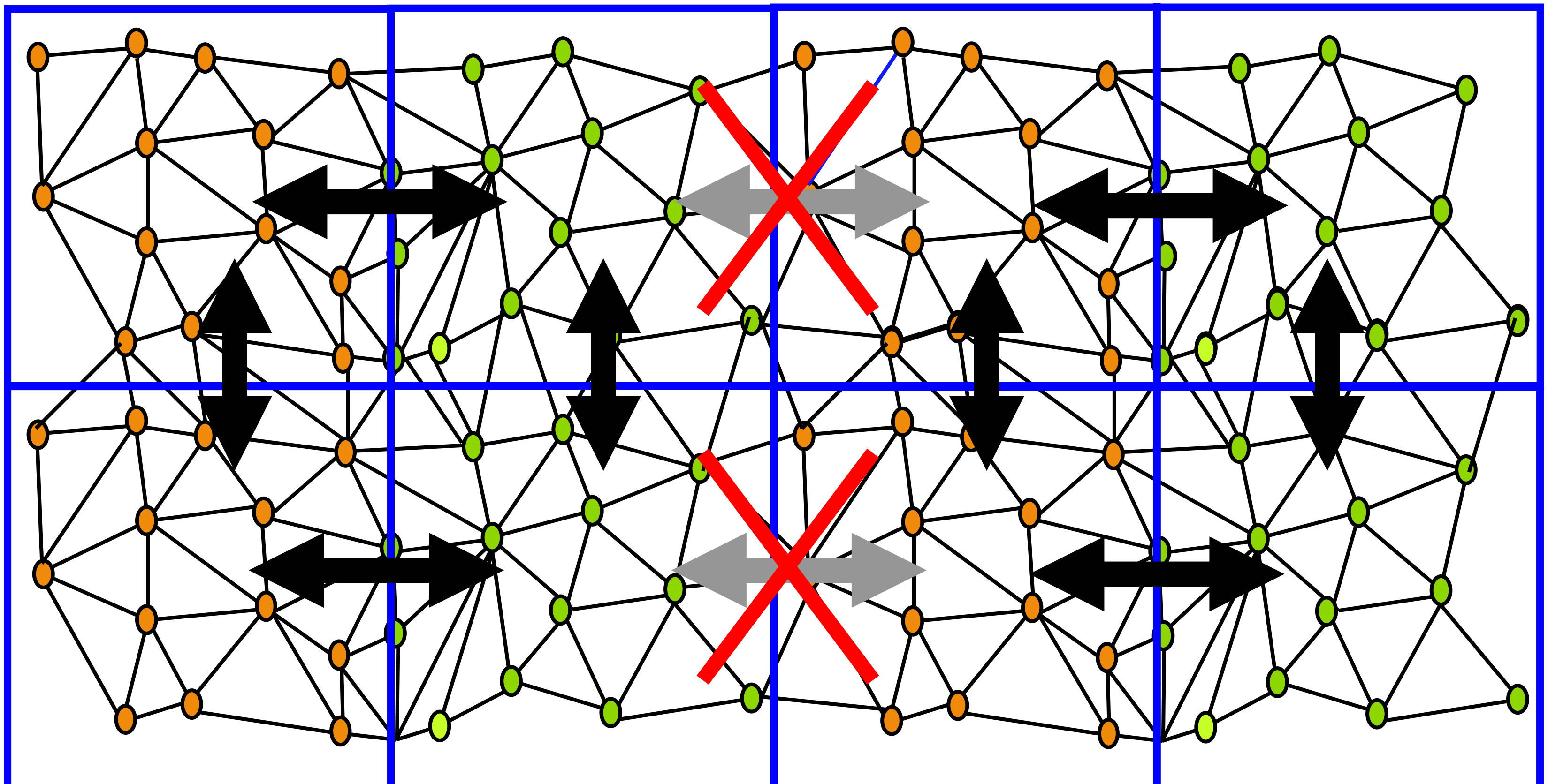


Chained Dependency

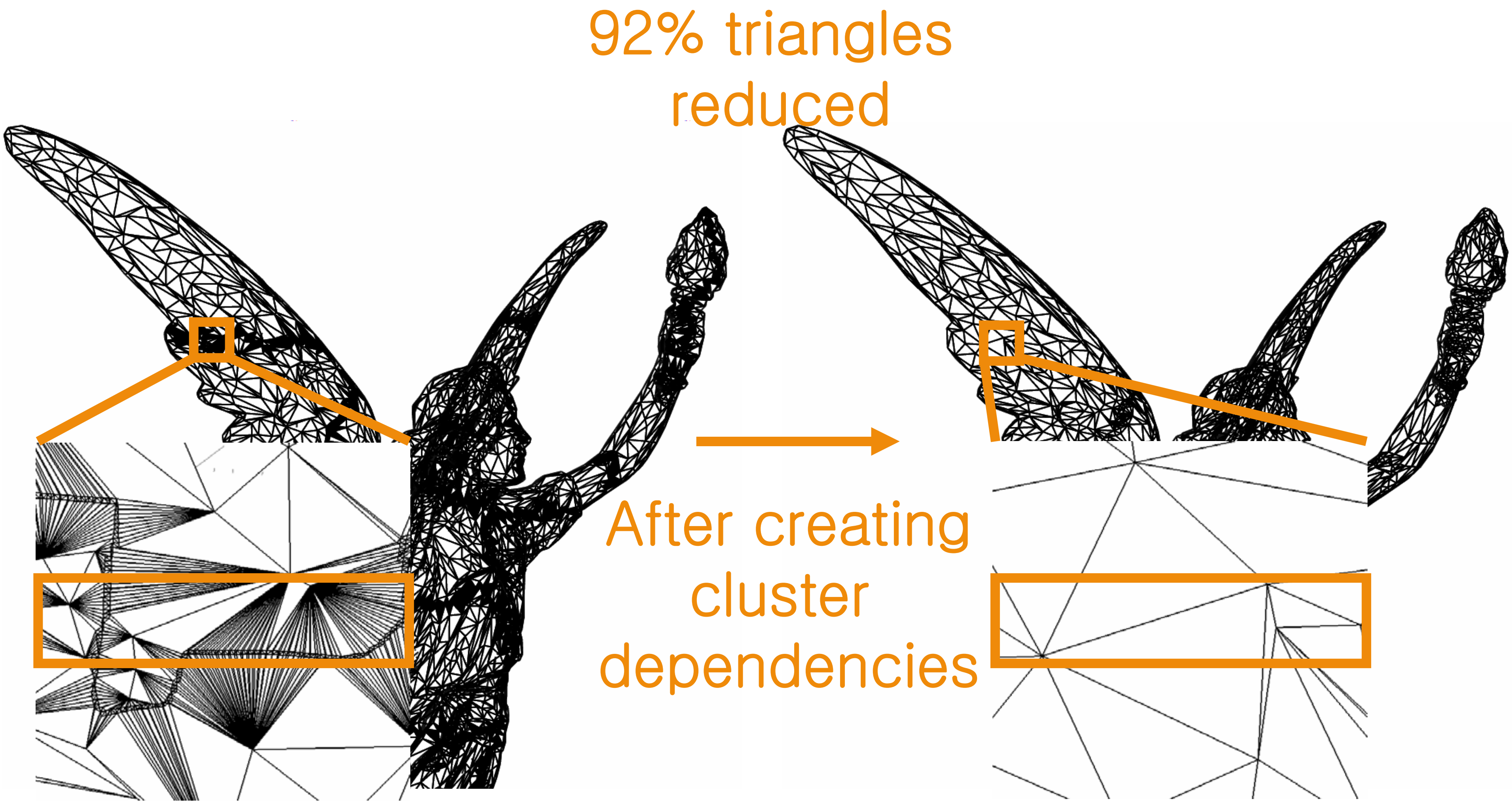
- **Inappropriate for refinements**



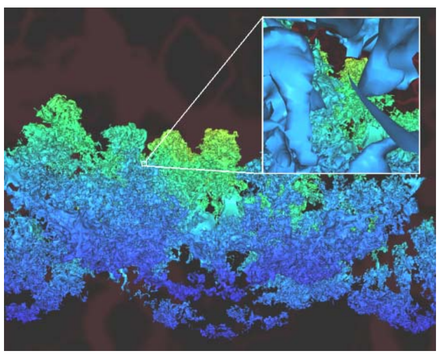
Cluster Dependencies

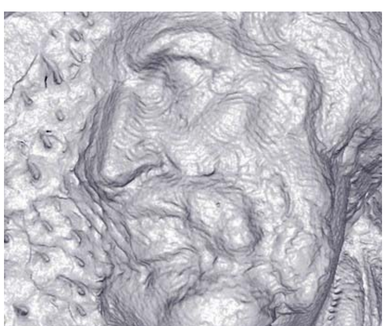


Cluster Dependencies



Runtime Performance





Model	Pixels-of-errors	Frame rate	Model size
Power plant	1	28	1GB
Iso-surface	10	18	2.5GB
St. Matthew	1	29	13GB

Other Forms of LOD

- Image impostors
- Shader LOD
  - Number of shaders
  - Number of textures
- Simulation LOD
  - Time steps
  - Simulation resolution
  - Number of particles
- Lighting
  - Number and type of lights used

Next Time..

- **Study cache-coherent algorithms**