Parallel Rendering: A Sorting Classification
Why Parallelism

Applications need:
- High frame rates
- High resolution
- Large geometric models
- Stereo
- Antialiasing
- etc.

Performance implications:
- Hundreds of MFLOPS compute power
- Gigabytes per second memory bandwidth
Stages of Parallelism

Graphics database traversal

Geometry processing

Rasterization

Display
Processing Tasks

Geometry Processors
- Each processor gets a subset of primitives
- Transformation
- (Lighting)
- Set-up for Rasterization

Rasterization Processors
- Each processor gets a subset of pixels
- Visibility computation
- Shading
Rendering as Sorting

Primitives may lie anywhere on or off screen

Determine effect of each primitive on each pixel

Primitives are sorted onto screen

Sorting affects distribution of data on geometry and rasterization processors
Primitives in Screen-space
Regions
Where to sort

Sort First
- Sort during geometry processing

Sort Middle
- Sort between geometry processing and rasterization

Sort Last
- Sort during rasterization
Sort First

Graphics database
(arbitrarily partitioned)

Redistribute "raw" primitives

(Pre-transform)

Geometry processing

Rasterization

Display
Sort First: Data Arrangement

Distribute both geometry and rasterization work according to position of primitives on screen

Load balancing difficult

Different screen regions of equal sizes may contain different numbers of primitives

May need dynamic region sizes
Sort First: Communications

Must determine primitive screen coverage before full transformation

Exploit frame-to-frame coherence

- Shuffle primitives between geometry processors only if screen coverage changes

Possibly employ primitive clustering and bounding volumes

- Pre-transform bounding volumes for small groups of primitives
Sort Middle

Graphics database (arbitrarily partitioned)

Redistribute screen-space primitives

Geometry processing

Rasterization

Display
Sort Middle: Data Arrangement

Geometry processors
- Arbitrary (random) distribution of primitives
- Good for load balancing

Rasterization processors
- Screen-space distribution of primitives
- Load balancing difficult
All geometry transformed primitives must be communicated every frame

All geometry processors must communicate with all rasterization processors

$O(n^2)$ communications paths
Sort Last

Graphics database (arbitrarily partitioned)

Geometry processing

Rasterization

Redistribute pixels, samples, or fragments

(Compositing)

Display
Arbitrary (random) arrangement of data on both geometry and rasterization processors

Great for load balancing

Each rasterization processor makes image of entire screen, with subset of primitives
Rasterization processors must communicate final pixel data.

Composition of pixel data may take place along linear or tree-shaped network.

Requires high bandwidth, assuming pixel data is much larger than primitive data.