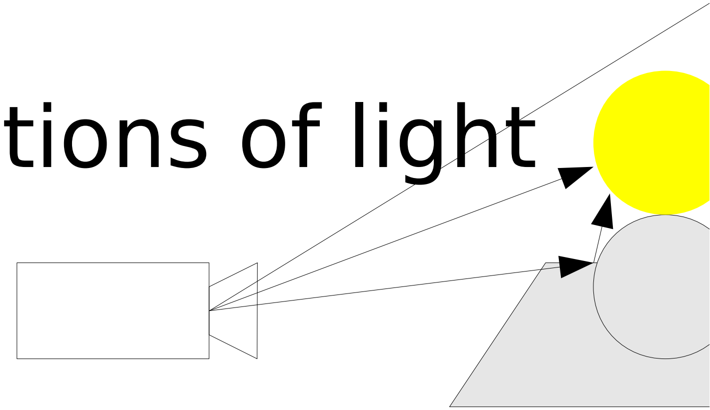
Parallel Ray Tracing

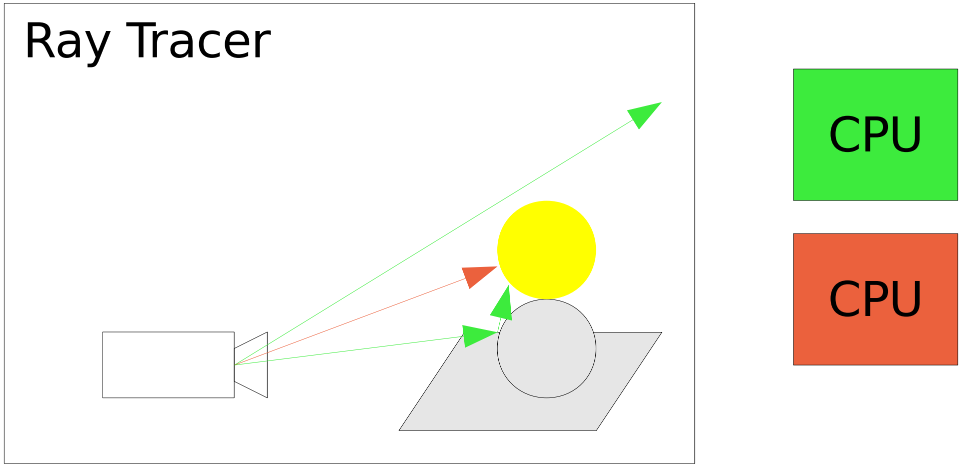
18.337 – Parallelization Project

Jordan Sorensen

Remember . . .

- A ray tracer simulates reflections of light in a scene
- It can be incredibly slow
- But it's embarrassingly parallel





Goals

- Want the ability to trace incredibly complex scenes by utilizing lots of computer power
- Solution must scale to an arbitrary number of processors
- Solution must be extensible – able to add in new sources of computing power

Results

- Implemented framework in which I can theoretically “plug in” different sources of processing power
- Implemented ability to plug in local processors (CPUs on a single machine) – can scale up to arbitrary number
- Did not have time to implement cross-network communication as I hoped

Results

- Thought I could take advantage of type of problem to save data transfer time, but low level details got in the way
- Took much more work to parallelize than expected
- High overhead costs cut back significantly on speed gains

Analysis & Future Work

- Parallelization not a good long term strategy – design with parallel architecture in mind!
- Would like to start ray tracer anew with ray tracing in mind to eliminate overhead
- Would eventually like to fill out wish list from “Goals” - networked processing, GPUs, etc