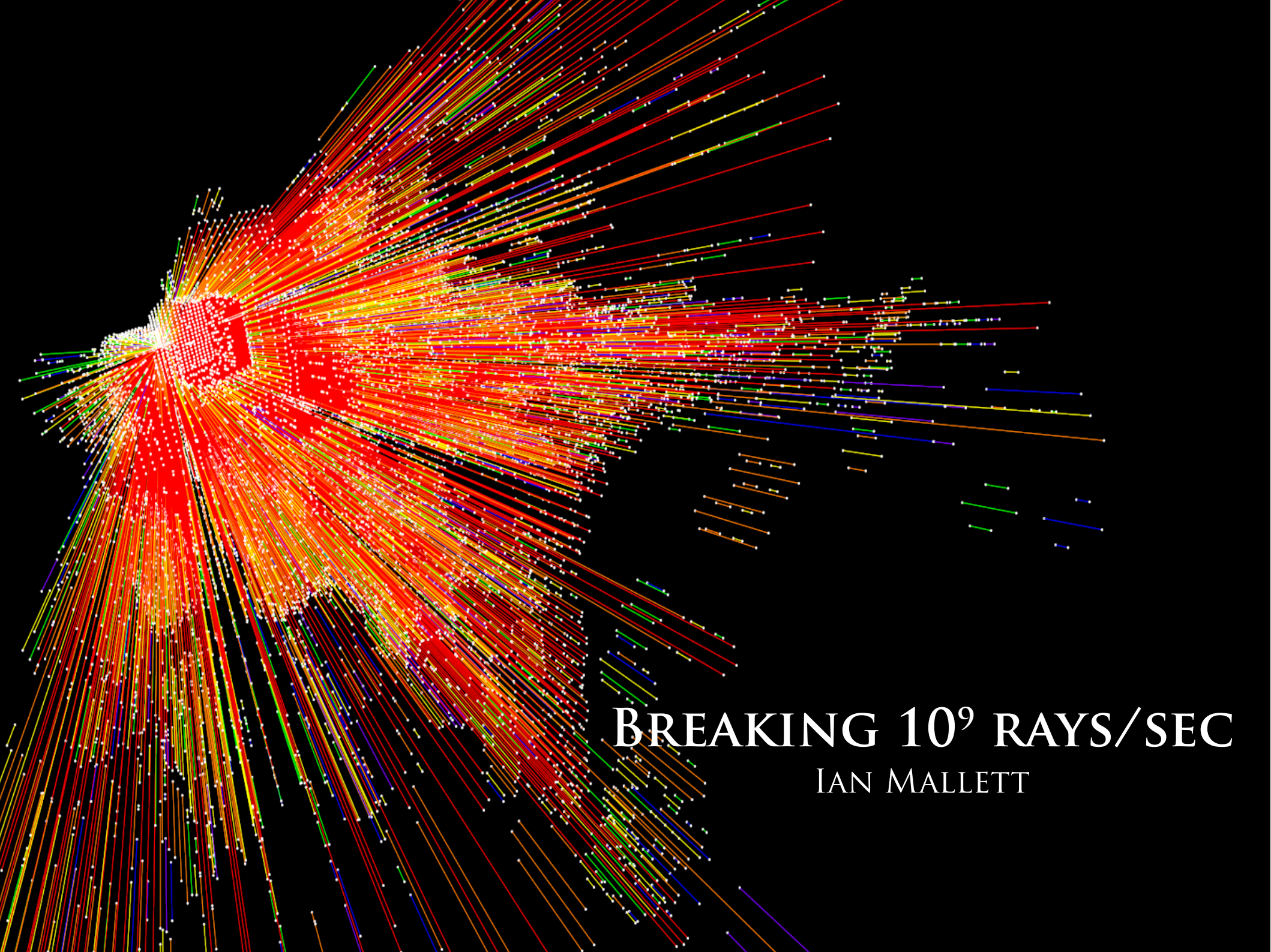


Breaking 10^9 rays/sec

Ian Mallett

Breaking 10^9 rays/sec

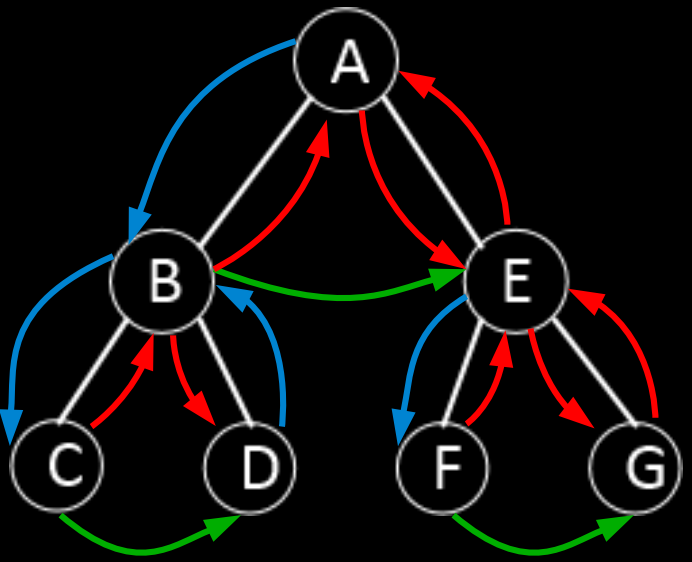
Ian Mallett

Overview

- Lots of fancy optimizations & lots of challenges
- Finished “idea 1” from original proposal: the stack-based scene-caching thing
- Gotten 1,000,000,000 rays/second with reasonably sized chip! Mission accomplished?

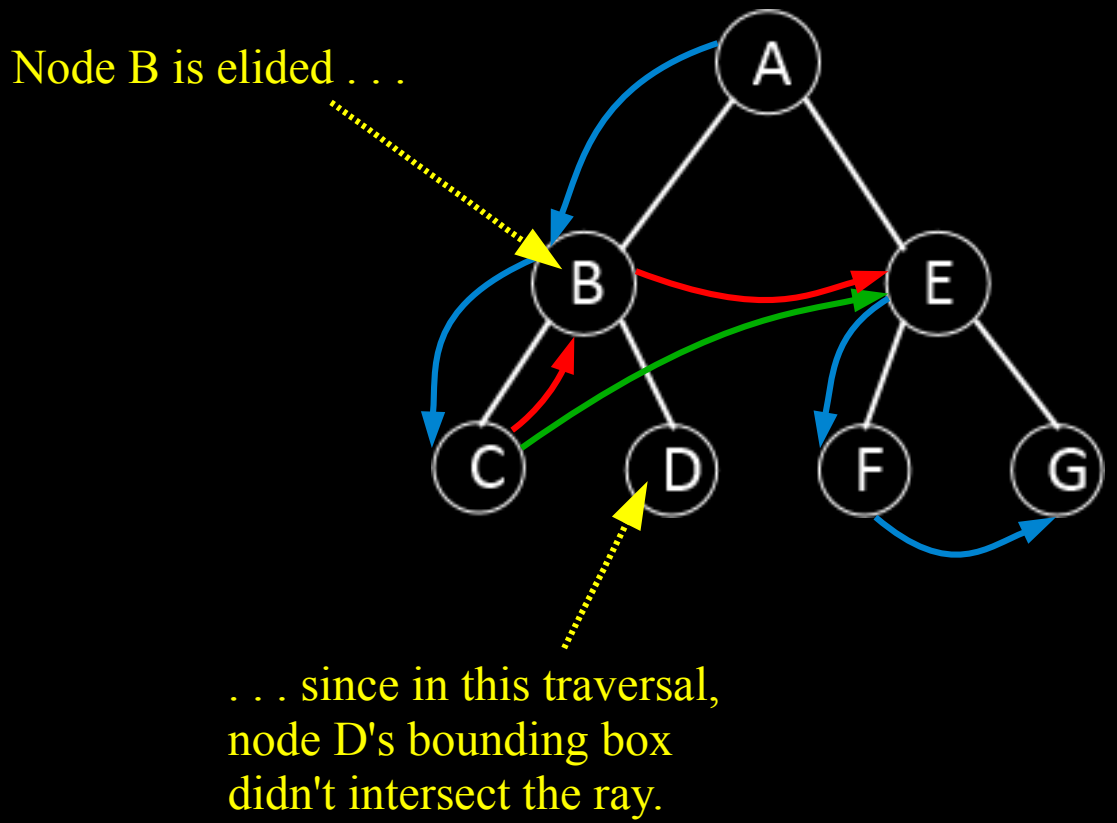
Important Optimizations

- Sidestep traversal (skip parent nodes)



Important Optimizations

- Elision of nodes where only one child hit in traversal (subtract one from depth of later nodes)



Important Optimizations

- Miscellaneous other for BVH:
  - Miscellaneous mentioned last time
    - Faster ray/box intersection
    - Nearest first & early termination
    - Non-recursive traversal
  - Miscellaneous fancy unmentioned stuff
    - Triangle edges, not verts
    - Relative pointers (TODO?)
    - Special shadow ray traversal (TODO)

Important Optimizations

- Impact of optimizations:
  - Recursive (and so no sidestep or elision):
  -
  - Nonrecursive (with sidestep and elision):
  -

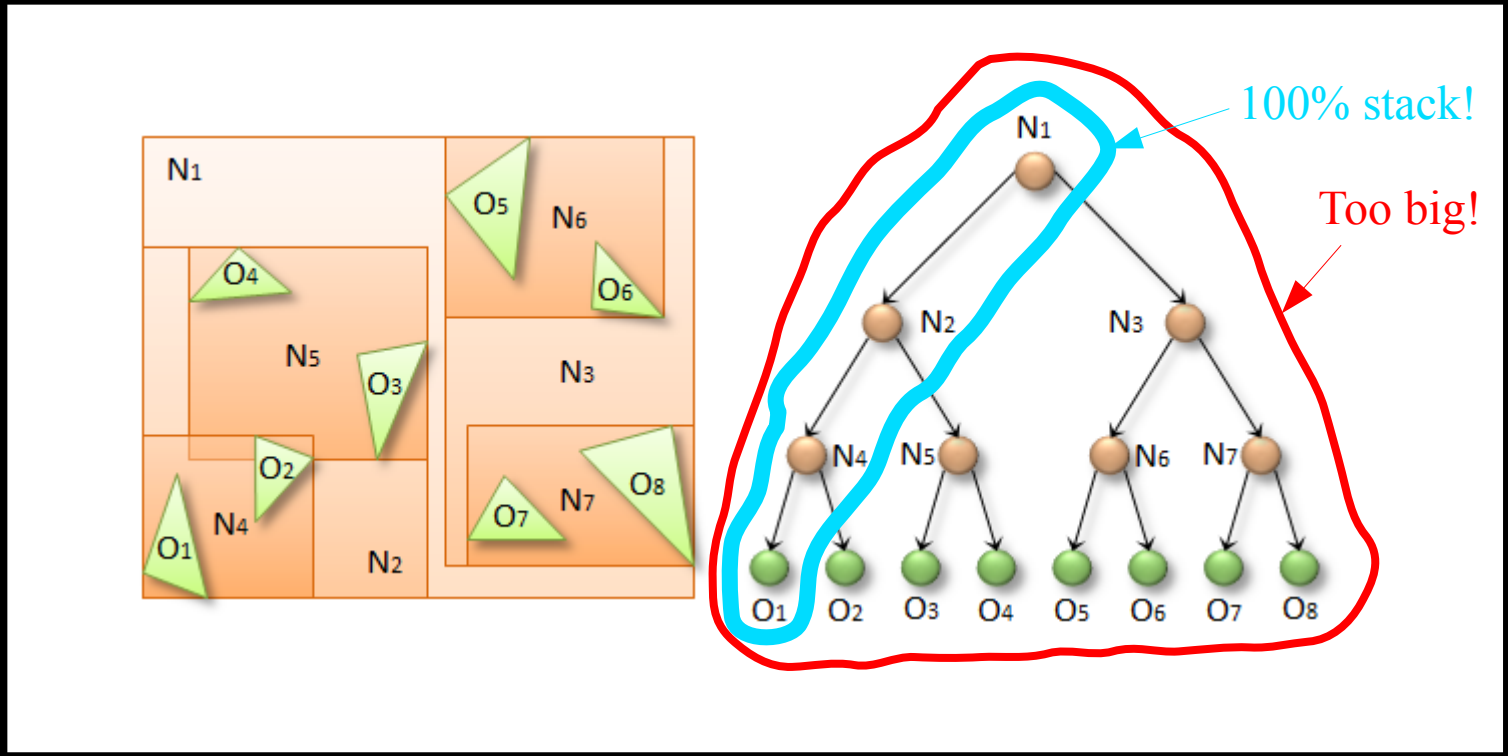
Important Optimizations

- Miscellaneous micro-optimization
  - Not sorry about this. Worth it.
- Sometimes didn't play nicely with compiler / simulator . . .
  - Placement new, std:: library stuff
  - Unimplemented instructions
  - Cycle counter
  - Danny fixes all! Thanks!

Challenges

- Pointer arithmetic nightmare!
- Concurrent deletion within thread (i.e., node A's load triggers a delete, which requires node A to move within memory)
- Deletion in general (surprisingly difficult algorithm!)
- Implementation issues
- Slow compiles/runs (esp. larger scenes)
- EGSR paper deadline
- Ran out of coffee

Finished Idea 1 of 3!

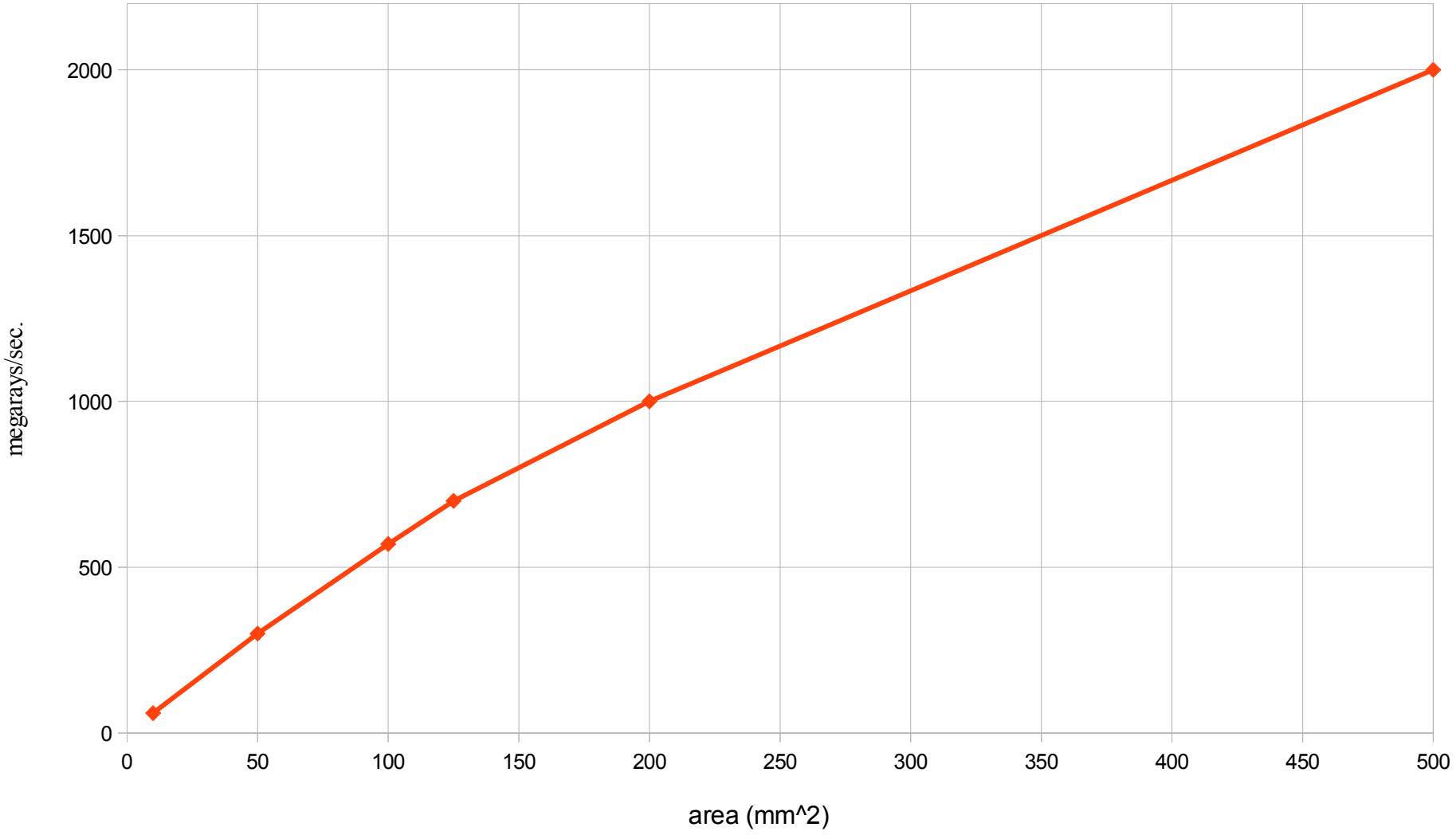


The Performance/Area Graph

- Main result: broke 10^9 rays/second with reasonably sized chip on decently sized dynamically-loaded scene (conference)

- Hand-generated some configurations to produce performance vs. area graph (next page)

The Performance/Area Graph



TODO

- Now what? Remaining (original) two ideas:
  - 2: Bitmap coarse-traversal
    - Traversal is already ludicrously complicated. Well, *very*, anyway.
    - Requires hacking BVH building. Thoughts?
  - 3: Ray rescheduling (req. 2) (was “stretch goal”)
- Still have room for optimization
  - Ray packets
  - Other datastructure (I changed node size)
- New ideas? 10^{10} rays/second???

Questions?

Image Credits

- http://bryanwagstaff.com/wp-content/uploads/2013/10/d
- http://childrenshungerfund.org/missiondh/images/missio
- me
- other?