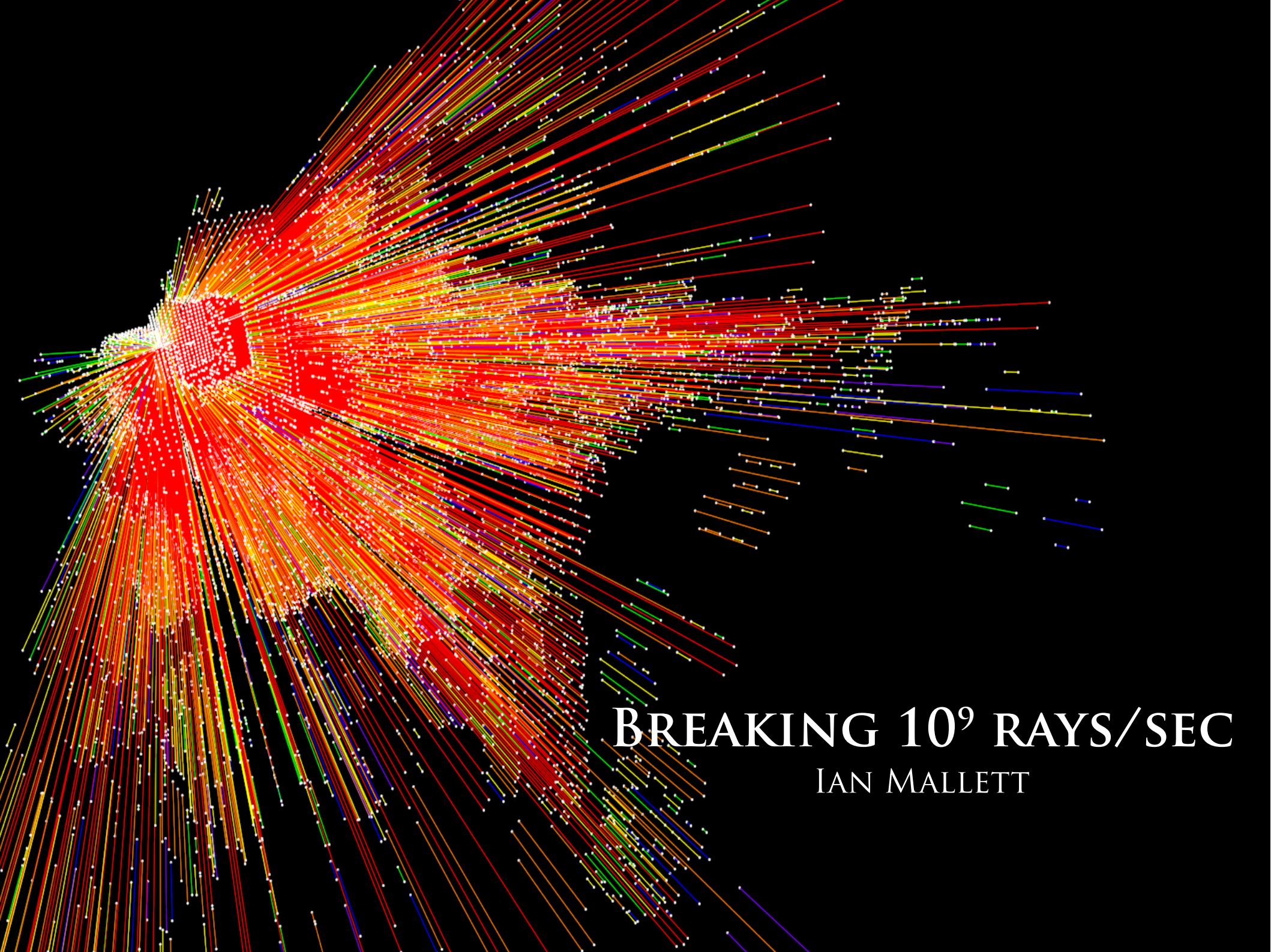
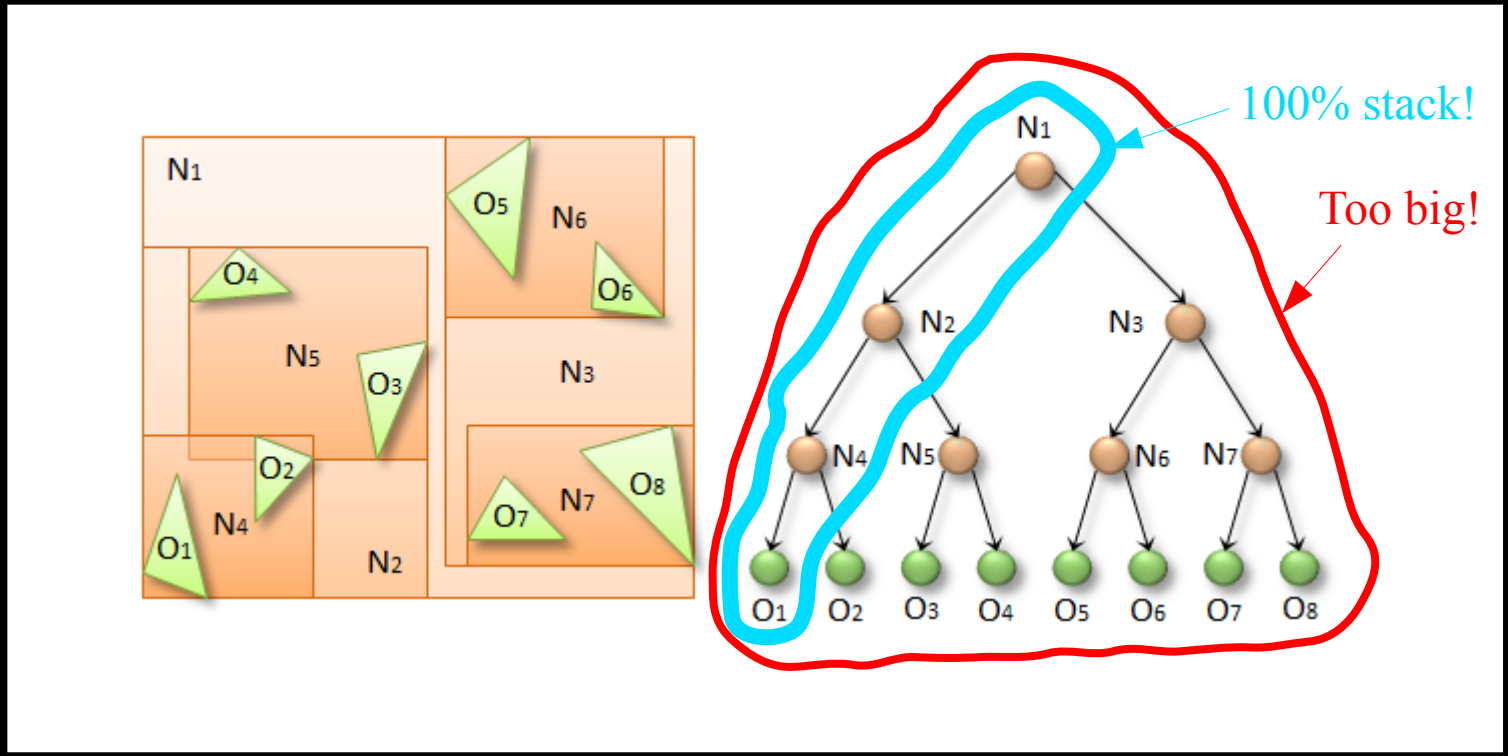


Breaking 10^9 rays/sec

Ian Mallett

Overview


N1
O5
N6
O6
O4
N5
O3
N3
O2
N4
O1
N2
O7
N7
O8
100% stack!
Too big!
N1
N2
N3
N4
N5
N6
N7
O1
O2
O3
O4
O5
O6
O7
O8

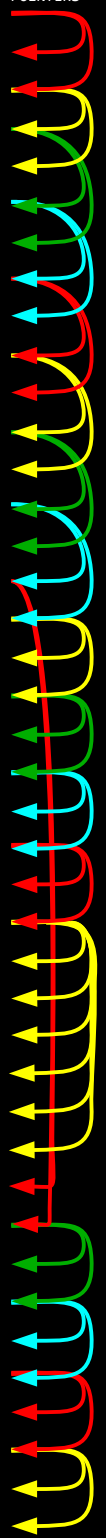
Overview

- Goal: gigaray/sec (1e9 rays/second) on reasonably sized chip (< 700mm^2, preferably smaller)

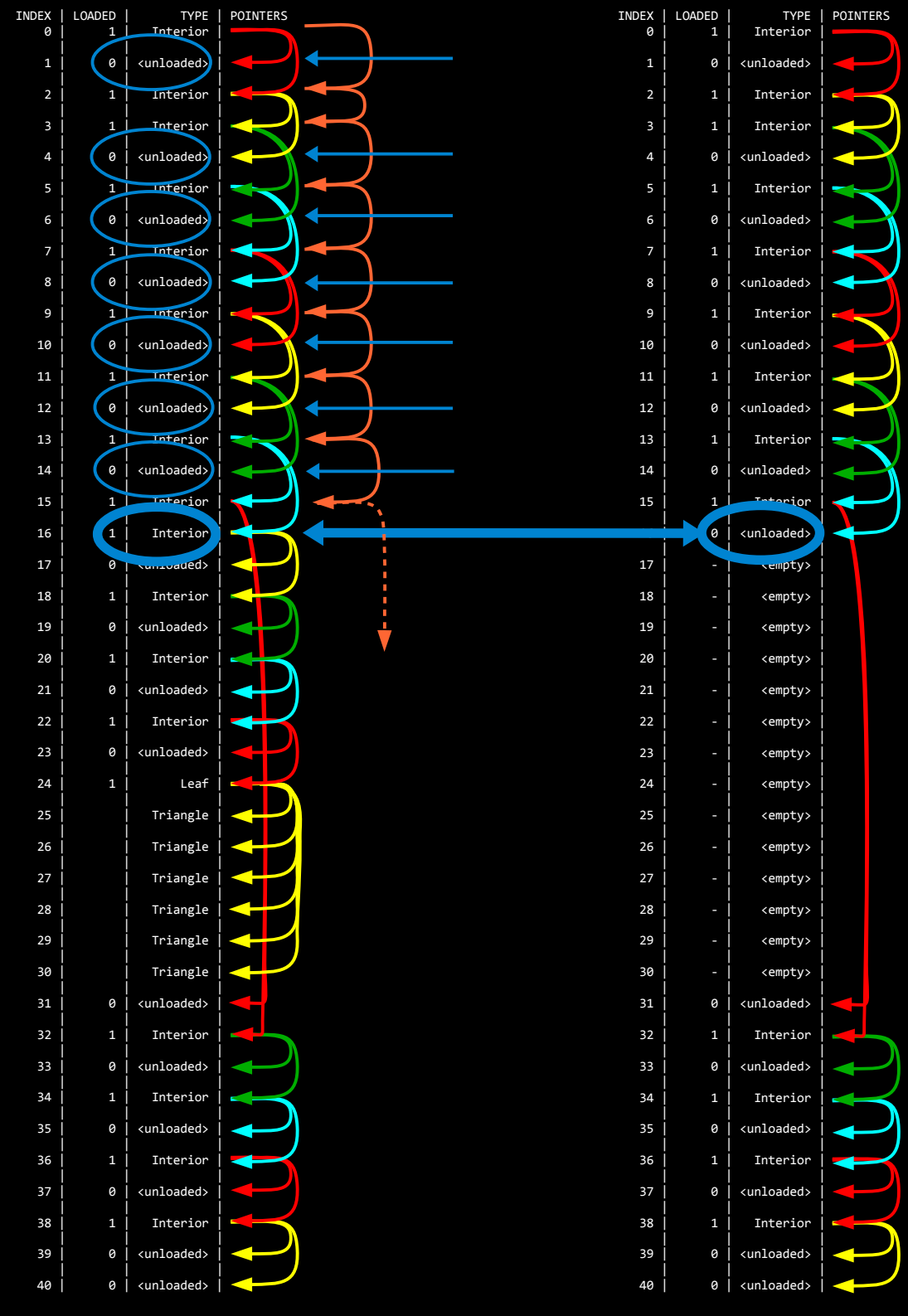
- Store current traversal on local stack
- Also as much of several traversals as possible
- **Think of this as a better L1-cache**

- Also optimize stuff/write really good pathtracer

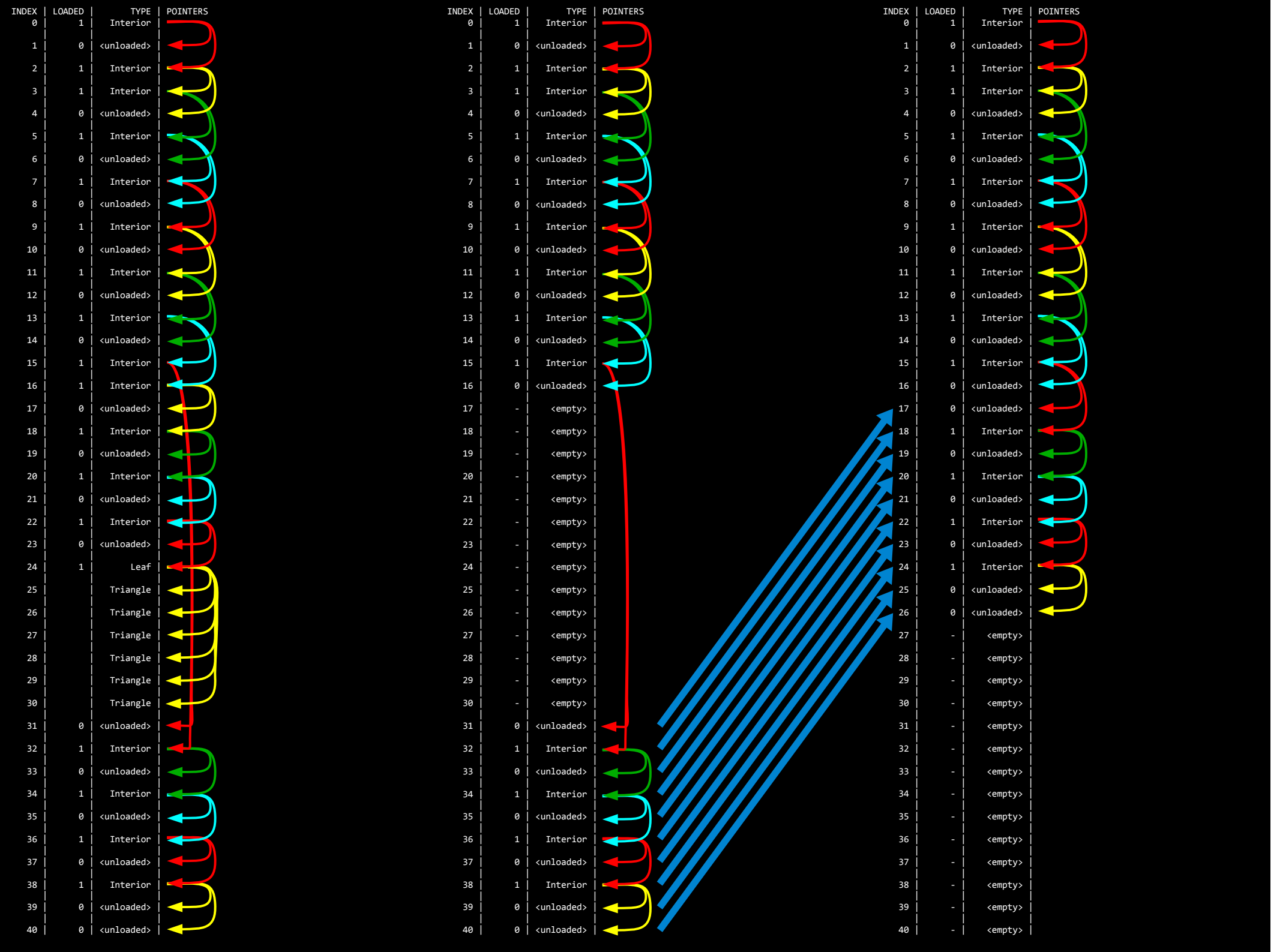
INDEX	LOADED	TYPE	POINTERS
0	1	Interior	
1	0	<unloaded>	
2	1	Interior	
3	1	Interior	
4	0	<unloaded>	
5	1	Interior	
6	0	<unloaded>	
7	1	Interior	
8	0	<unloaded>	
9	1	Interior	
10	0	<unloaded>	
11	1	Interior	
12	0	<unloaded>	
13	1	Interior	
14	0	<unloaded>	
15	1	Interior	
16	1	Interior	
17	0	<unloaded>	
18	1	Interior	
19	0	<unloaded>	
20	1	Interior	
21	0	<unloaded>	
22	1	Interior	
23	0	<unloaded>	
24	1	Leaf	
25		Triangle	
26		Triangle	
27		Triangle	
28		Triangle	
29		Triangle	
30		Triangle	
31	0	<unloaded>	
32	1	Interior	
33	0	<unloaded>	
34	1	Interior	
35	0	<unloaded>	
36	1	Interior	
37	0	<unloaded>	
38	1	Interior	
39	0	<unloaded>	
40	0	<unloaded>	



A Full Stack


INDEX | LOADED | TYPE | POINTERS
0 | 1 | Interior
1 | 0 | <unloaded>
2 | 1 | Interior
3 | 1 | Interior
4 | 0 | <unloaded>
5 | 1 | Interior
6 | 0 | <unloaded>
7 | 1 | Interior
8 | 0 | <unloaded>
9 | 1 | Interior
10 | 0 | <unloaded>
11 | 1 | Interior
12 | 0 | <unloaded>
13 | 1 | Interior
14 | 0 | <unloaded>
15 | 1 | Interior
16 | 1 | Interior
17 | 0 | <unloaded>
18 | 1 | Interior
19 | 0 | <unloaded>
20 | 1 | Interior
21 | 0 | <unloaded>
22 | 1 | Interior
23 | 0 | <unloaded>
24 | 1 | Leaf
25 | | Triangle
26 | | Triangle
27 | | Triangle
28 | | Triangle
29 | | Triangle
30 | | Triangle
31 | 0 | <unloaded>
32 | 1 | Interior
33 | 0 | <unloaded>
34 | 1 | Interior
35 | 0 | <unloaded>
36 | 1 | Interior
37 | 0 | <unloaded>
38 | 1 | Interior
39 | 0 | <unloaded>
40 | 0 | <unloaded>

INDEX | LOADED | TYPE | POINTERS
0 | 1 | Interior
1 | 0 | <unloaded>
2 | 1 | Interior
3 | 1 | Interior
4 | 0 | <unloaded>
5 | 1 | Interior
6 | 0 | <unloaded>
7 | 1 | Interior
8 | 0 | <unloaded>
9 | 1 | Interior
10 | 0 | <unloaded>
11 | 1 | Interior
12 | 0 | <unloaded>
13 | 1 | Interior
14 | 0 | <unloaded>
15 | 1 | Interior
16 | 0 | <unloaded>
17 | - | <empty>
18 | - | <empty>
19 | - | <empty>
20 | - | <empty>
21 | - | <empty>
22 | - | <empty>
23 | - | <empty>
24 | - | <empty>
25 | - | <empty>
26 | - | <empty>
27 | - | <empty>
28 | - | <empty>
29 | - | <empty>
30 | - | <empty>
31 | 0 | <unloaded>
32 | 1 | Interior
33 | 0 | <unloaded>
34 | 1 | Interior
35 | 0 | <unloaded>
36 | 1 | Interior
37 | 0 | <unloaded>
38 | 1 | Interior
39 | 0 | <unloaded>
40 | 0 | <unloaded>


INDEX | LOADED | TYPE | POINTERS
0 | 1 | Interior |
1 | 0 | <unloaded> |
2 | 1 | Interior |
3 | 1 | Interior |
4 | 0 | <unloaded> |
5 | 1 | Interior |
6 | 0 | <unloaded> |
7 | 1 | Interior |
8 | 0 | <unloaded> |
9 | 1 | Interior |
10 | 0 | <unloaded> |
11 | 1 | Interior |
12 | 0 | <unloaded> |
13 | 1 | Interior |
14 | 0 | <unloaded> |
15 | 1 | Interior |
16 | 1 | Interior |
17 | 0 | <unloaded> |
18 | 1 | Interior |
19 | 0 | <unloaded> |
20 | 1 | Interior |
21 | 0 | <unloaded> |
22 | 1 | Interior |
23 | 0 | <unloaded> |
24 | 1 | Leaf |
25 | | Triangle |
26 | | Triangle |
27 | | Triangle |
28 | | Triangle |
29 | | Triangle |
30 | | Triangle |
31 | 0 | <unloaded> |
32 | 1 | Interior |
33 | 0 | <unloaded> |
34 | 1 | Interior |
35 | 0 | <unloaded> |
36 | 1 | Interior |
37 | 0 | <unloaded> |
38 | 1 | Interior |
39 | 0 | <unloaded> |
40 | 0 | <unloaded> |
INDEX | LOADED | TYPE | POINTERS
0 | 1 | Interior |
1 | 0 | <unloaded> |
2 | 1 | Interior |
3 | 1 | Interior |
4 | 0 | <unloaded> |
5 | 1 | Interior |
6 | 0 | <unloaded> |
7 | 1 | Interior |
8 | 0 | <unloaded> |
9 | 1 | Interior |
10 | 0 | <unloaded> |
11 | 1 | Interior |
12 | 0 | <unloaded> |
13 | 1 | Interior |
14 | 0 | <unloaded> |
15 | 1 | Interior |
16 | 0 | <unloaded> |
17 | - | <empty> |
18 | - | <empty> |
19 | - | <empty> |
20 | - | <empty> |
21 | - | <empty> |
22 | - | <empty> |
23 | - | <empty> |
24 | - | <empty> |
25 | - | <empty> |
26 | - | <empty> |
27 | - | <empty> |
28 | - | <empty> |
29 | - | <empty> |
30 | - | <empty> |
31 | 0 | <unloaded> |
32 | 1 | Interior |
33 | 0 | <unloaded> |
34 | 1 | Interior |
35 | 0 | <unloaded> |
36 | 1 | Interior |
37 | 0 | <unloaded> |
38 | 1 | Interior |
39 | 0 | <unloaded> |
40 | 0 | <unloaded> |
INDEX | LOADED | TYPE | POINTERS
0 | 1 | Interior |
1 | 0 | <unloaded> |
2 | 1 | Interior |
3 | 1 | Interior |
4 | 0 | <unloaded> |
5 | 1 | Interior |
6 | 0 | <unloaded> |
7 | 1 | Interior |
8 | 0 | <unloaded> |
9 | 1 | Interior |
10 | 0 | <unloaded> |
11 | 1 | Interior |
12 | 0 | <unloaded> |
13 | 1 | Interior |
14 | 0 | <unloaded> |
15 | 1 | Interior |
16 | 0 | <unloaded> |
17 | 0 | <unloaded> |
18 | 1 | Interior |
19 | 0 | <unloaded> |
20 | 1 | Interior |
21 | 0 | <unloaded> |
22 | 1 | Interior |
23 | 0 | <unloaded> |
24 | 1 | Interior |
25 | 0 | <unloaded> |
26 | 0 | <unloaded> |
27 | - | <empty> |
28 | - | <empty> |
29 | - | <empty> |
30 | - | <empty> |
31 | - | <empty> |
32 | - | <empty> |
33 | - | <empty> |
34 | - | <empty> |
35 | - | <empty> |
36 | - | <empty> |
37 | - | <empty> |
38 | - | <empty> |
39 | - | <empty> |
40 | - | <empty> |

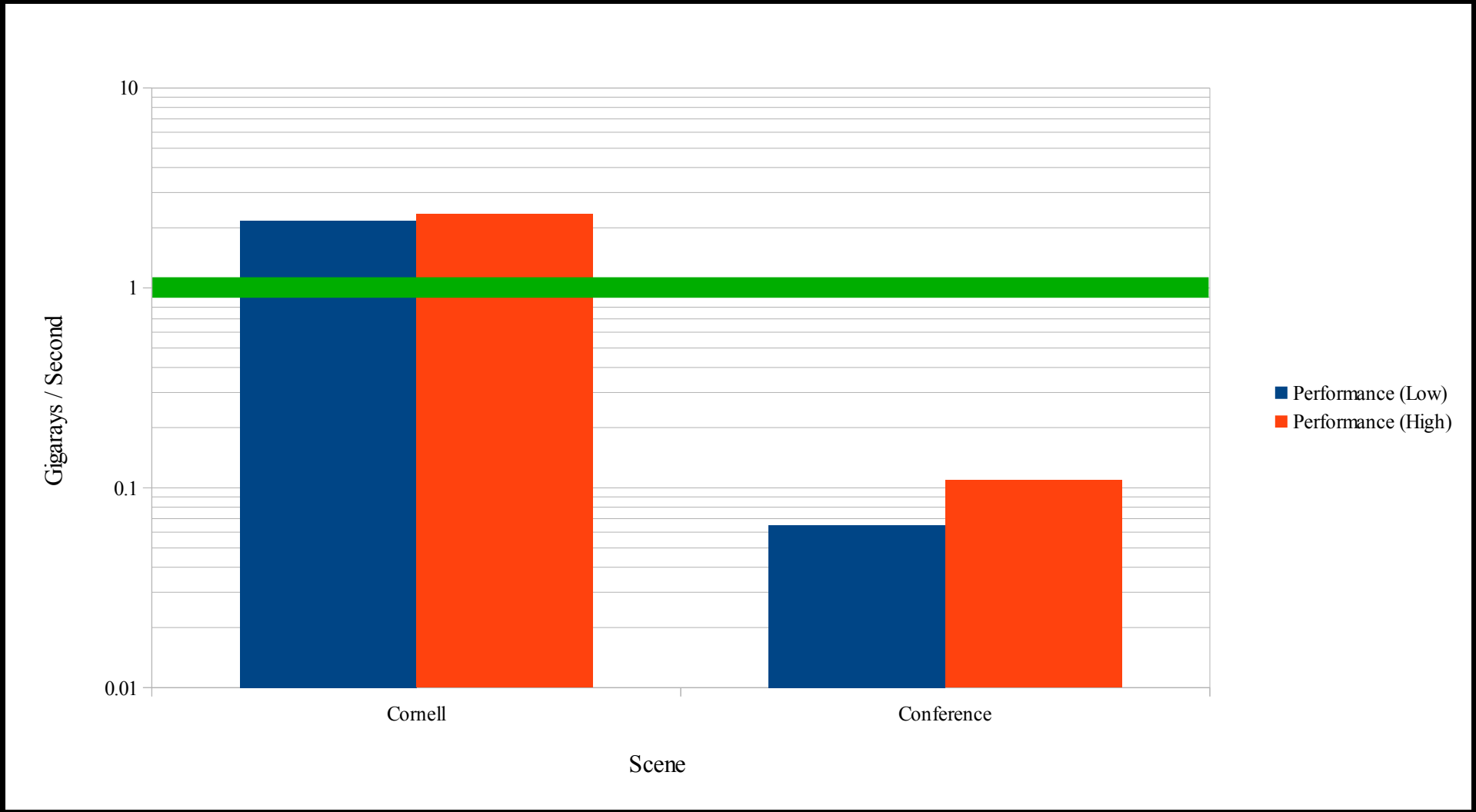
Results (Major Factors)

- A lot of scene fits on the stack (reuse)
- Shadow rays (coherent, occluders usually nearby)

- Very little of scene fits (less reuse)
- Threads jumping in screen-space with shallow depth (incoherency, less reuse)

- Motivating example . . .

Results (Min and Max)



Results

- Cornell
  - Everything fits on stack, and each processor can reuse
- Conference
  - Not everything fits on stack and processors jump in image space a lot → much less reuse
- Hairball/Dragon (simulation too large → too long)

Performance vs. Area

- Area calculation bogus for localstore size (constant)
- Used default size (with allowances for simulator/debug info.), map to same size stack
- $297mm^2 \rightarrow$ reasonably sized chip!
  - Inaccurate; but probably not by too much

Performance Comparison

- Reference path tracer runs . . . differently
  - Use L1 and no L2
  - Try to match areas
  - Unsuccessful; and area calculations bogus anyway
- Certainly looks like my way is faster overall
- Weird slowdowns though.
  Might be ray profiling code.

Difficulties

- Simulator
  - Very slow
  - No debugging
  - Bugs and unimplemented features
- Clever mistakes
  - ~2,000 lines of deep pointer arithmetic
  - Fancy compiler options (gah)

Conclusion

- Mixed results
- Should be combined with ray rescheduling of some kind (even just one thread gets 16 pixels → big win)
- Lots more comparison work is needed → weakest part of project
- *Still* room for optimization . . .

Questions

Image Credits

- me
- others