

```

<div style="float: left; width: 200px">
  <div>
    <div style="float: left; ">
      inner float
      <div style="float: left">inner inner float</div>
    </div>
    the div
  </div>
  outer float
</div>

```



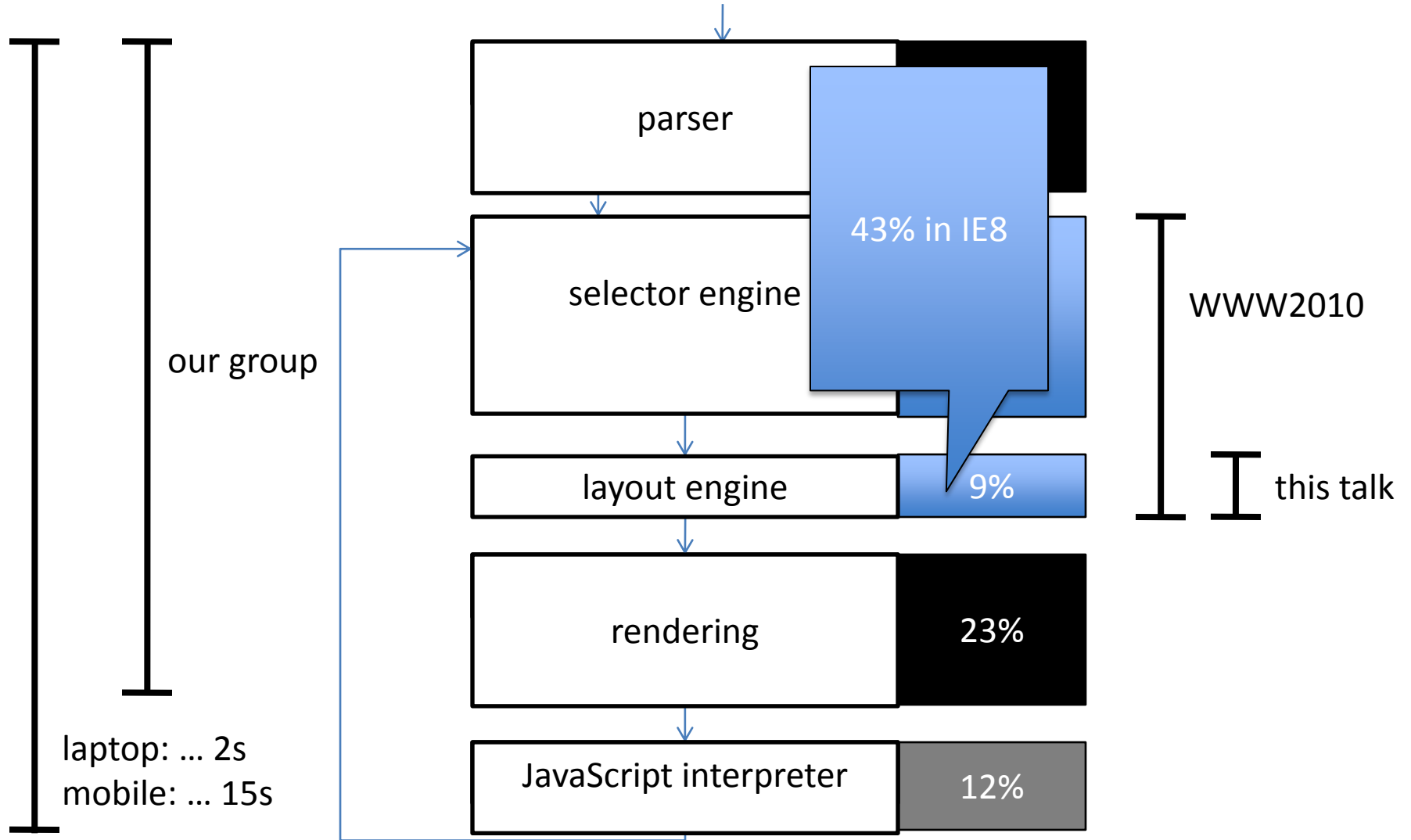
Rich Layout from First Principles

Specification, Generation, and Parallelization

Adam Jiang, Leo Meyerovich
with Seth Fowler, John Ng, and RasBodik

Hot Par 2009, WWW 2010

Why Layout?



The Layout Problem

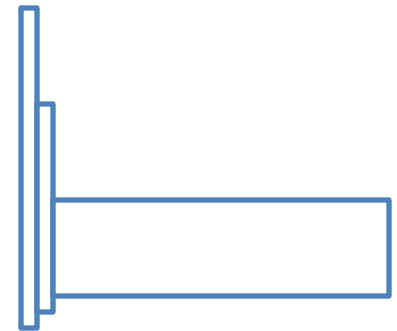


width=shrinkToFit

width=50% of parent

width=200px

constraints satisfied

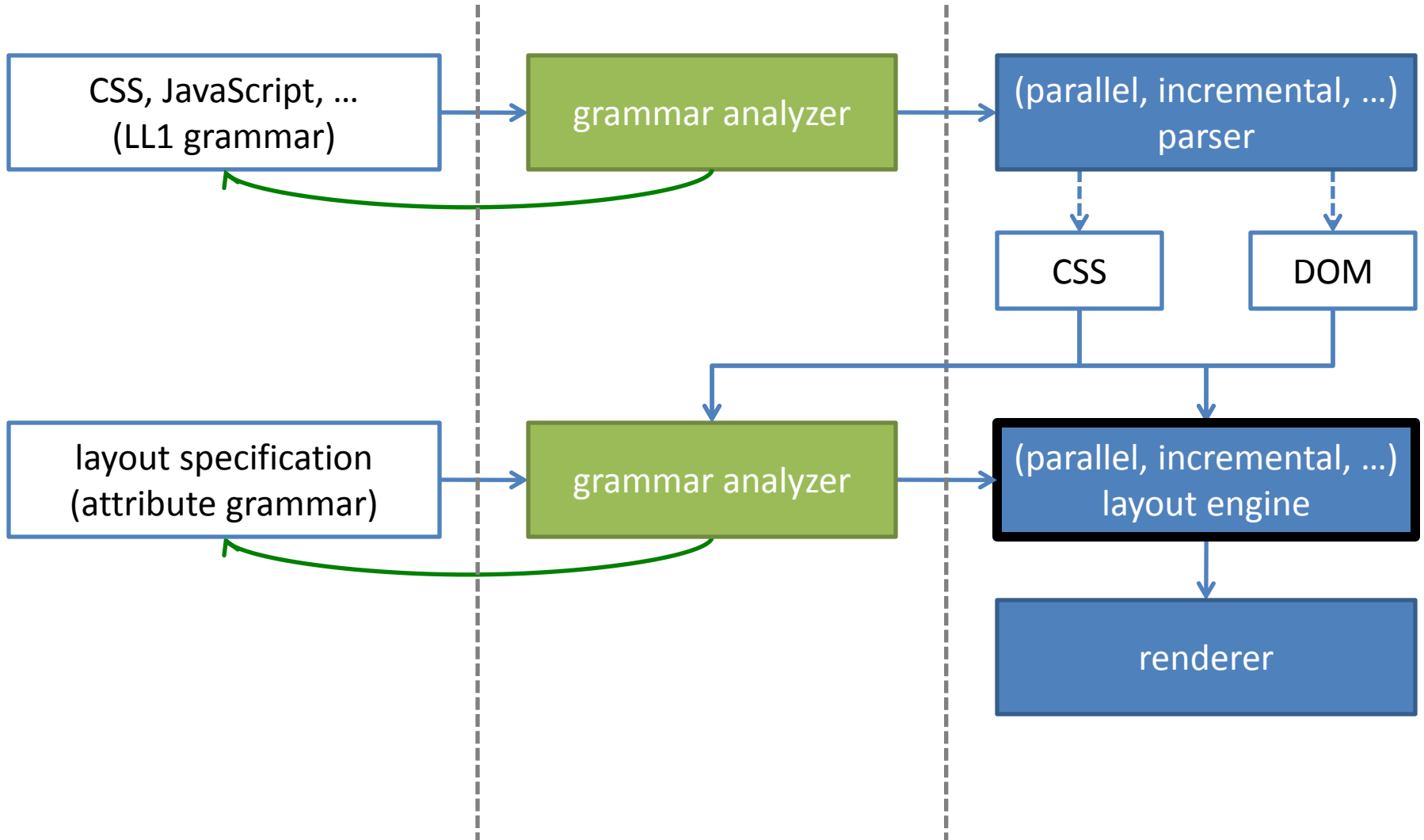


... but strange



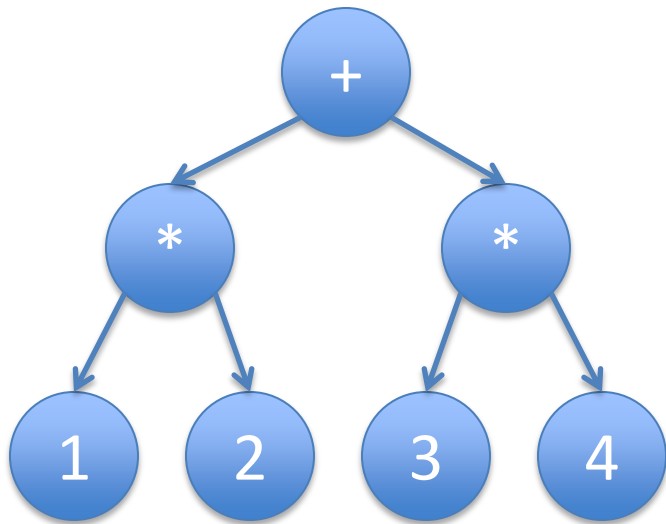
Standard and implementation strategy are unclear

Towards a Mechanized Layout Engine

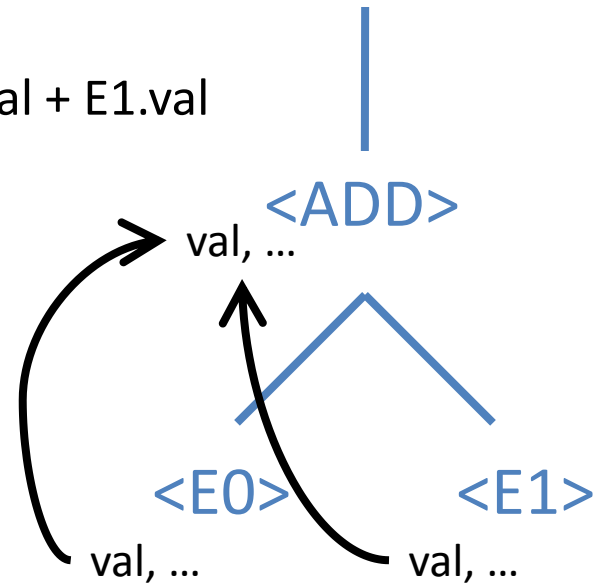


Attribute Grammars

- Knuth '67: executable language semantics

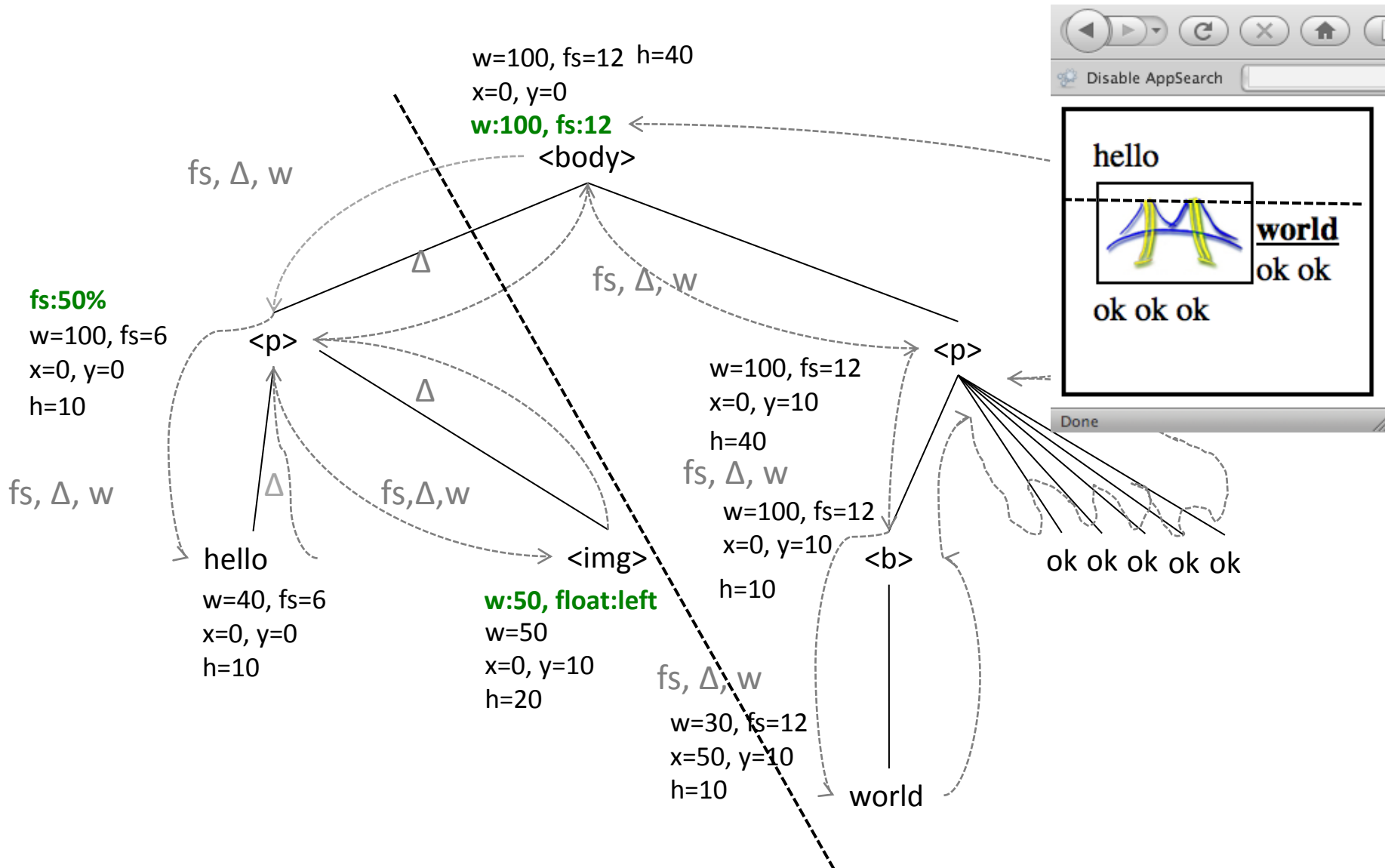


$\text{this.val} = E0.\text{val} + E1.\text{val}$

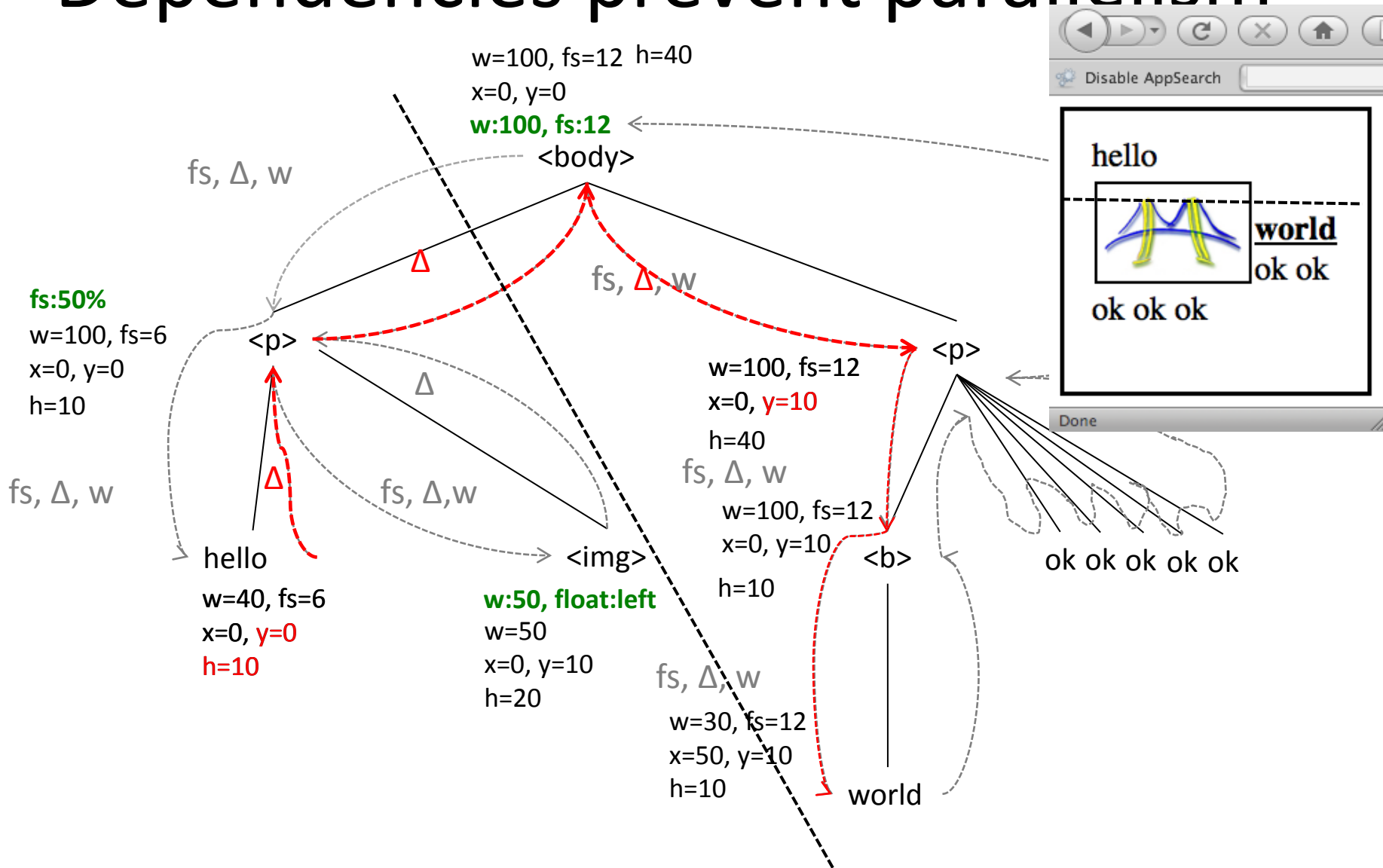


- Automation: parallel, incremental, ...
- Good for IDEs, compiling Pascal-like languages

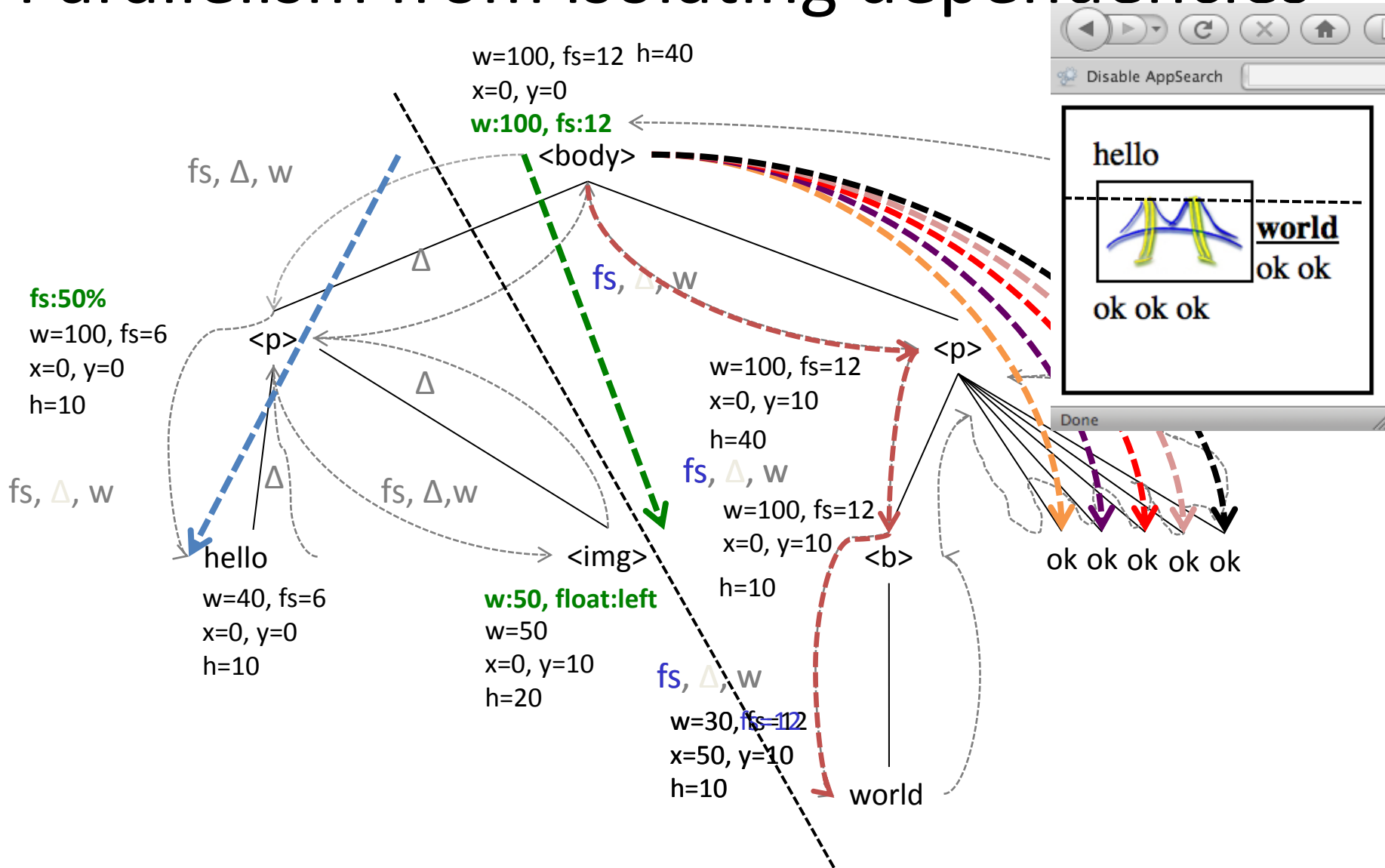
Is Layout Inherently Sequential?



Dependencies prevent parallelism



Parallelism from isolating dependencies

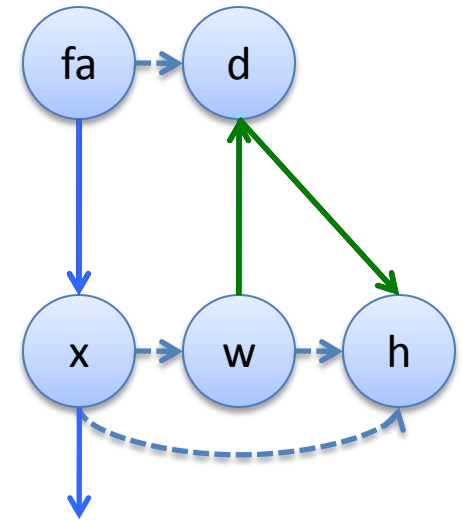
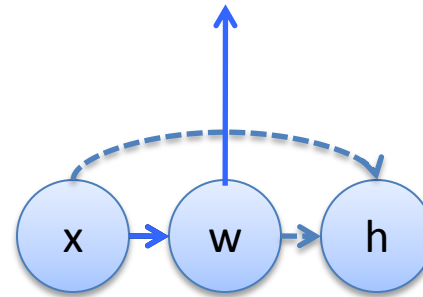
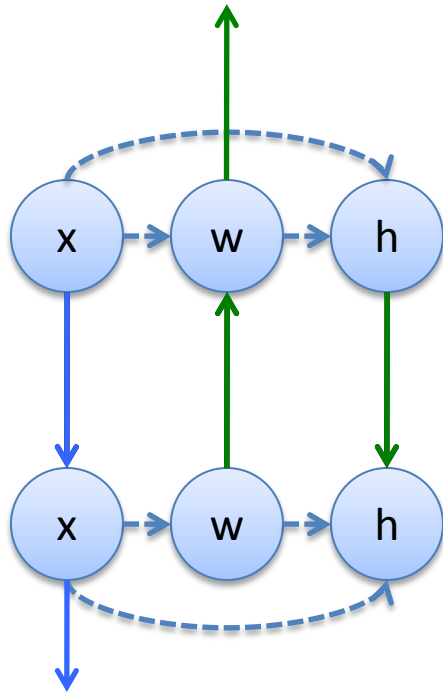


innerNode

leafNode

topNode

self



(child1.x :=
self.x + 23)

child1 : Node

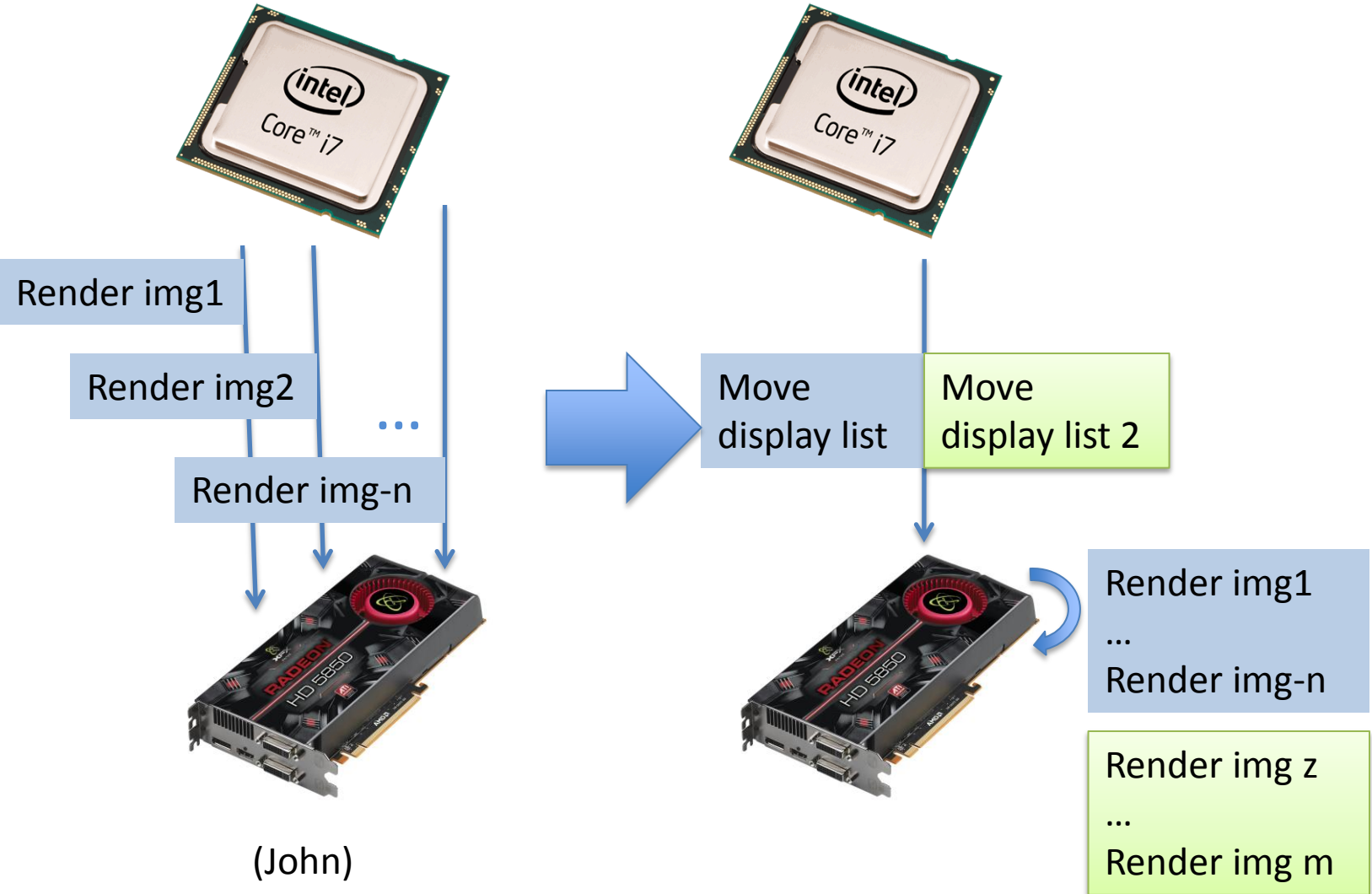
1. Compute local dependencies

2. Compute transitive dependencies

for all layouts in this grammar, if $dep(a, b, \dots, z)$, then $E(a, b), E(b, c), \dots$

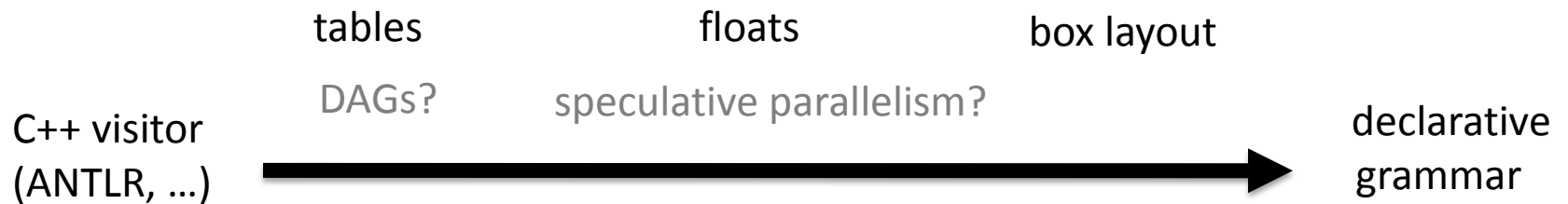
3. Schedule by stitching together (topological sort)

Automatic Incrementalization of Rendering



Takeaway

- **Attribute grammar specification promising**



- **New attribute grammar language**

- **Big benefits**

Users: fast and conformant browser

Designers: analysis tools

Browser developers: separate feature from optimization

Standards: verification (well-defined, backwards compat..)

Questions?