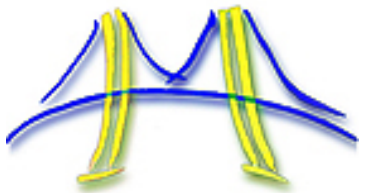
Parallel Graphical Layout

Ras Bodik, Thibaud Hottelier, Edward Lu, Leo Meyerovich,
Ali Sinan Koksal, **Matt Torok**, Eric Atkinson
Doug Kimelman *IBM*, Kimmo Kuusilinna and Per Ljung *Nokia*



Why parallel layout? It's part of web browser.

Web browser has grown into *the* app platform

- True on laptops. Not (yet) true on phones.

Energy constraints make phones too slow

- Browser apps practically unusable on phones
- Hence the faster iOS, Android, Win SDKs dominate

Matters to be worse on post-phone devices

Parallelism helps energy efficiency

Programs run faster

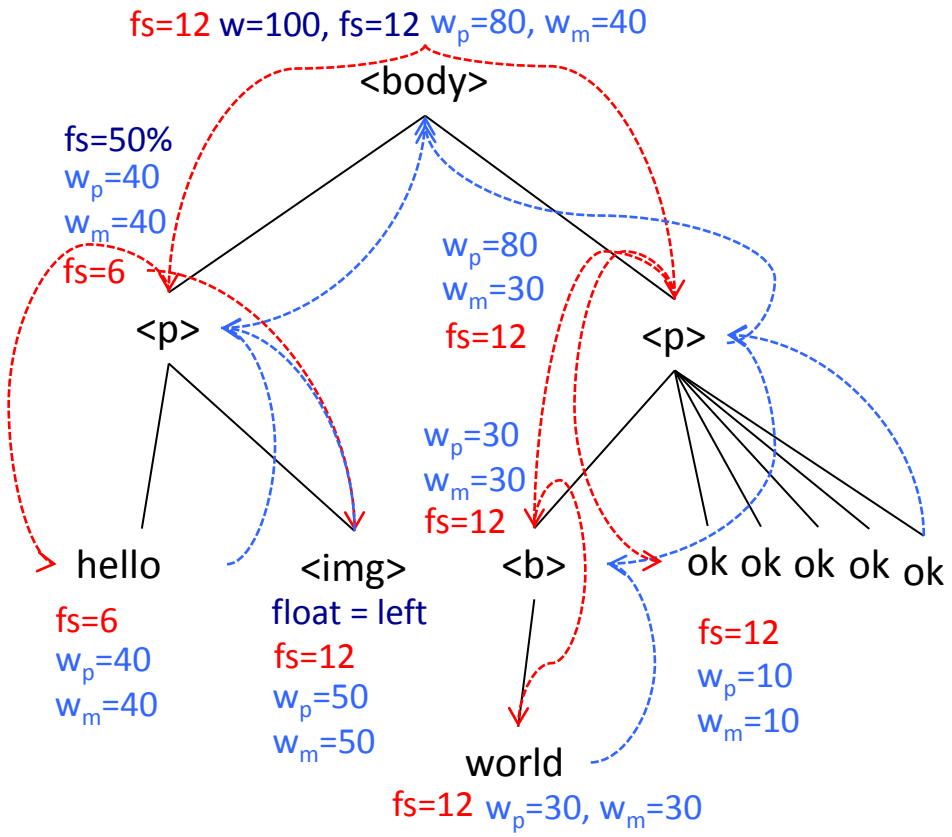
==> CPU sleeps sooner

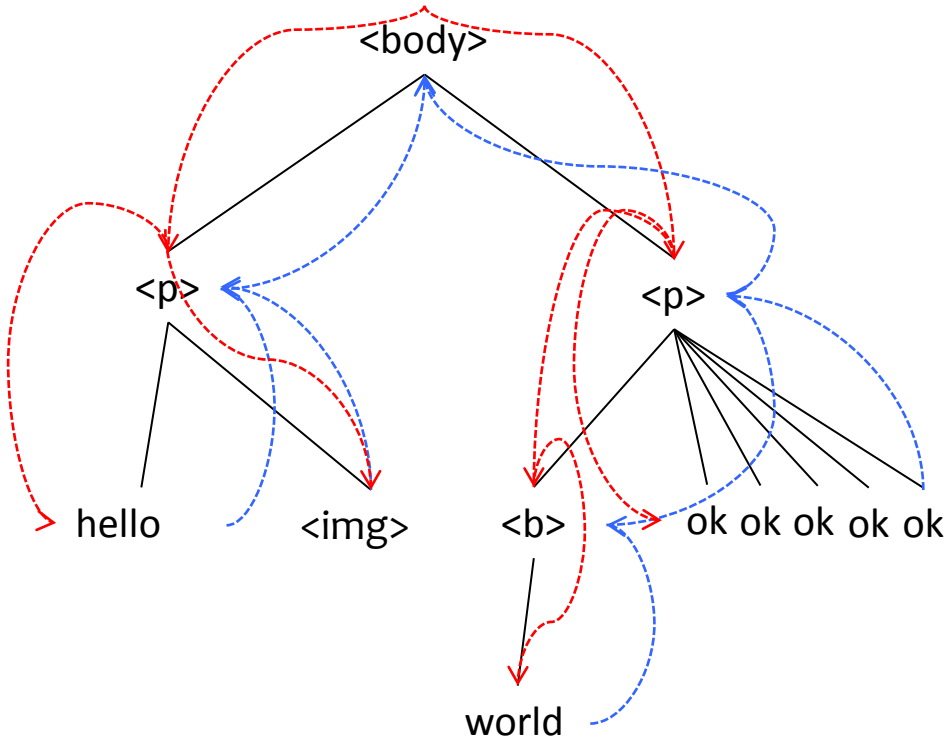
Simple cores more energy efficient

==> use many small cores rather a few big cores

So we need to parallelize layout

Five passes for laying out CSS documents





Phase 1: font size, temporary width

Phase 2: preferred max & min width

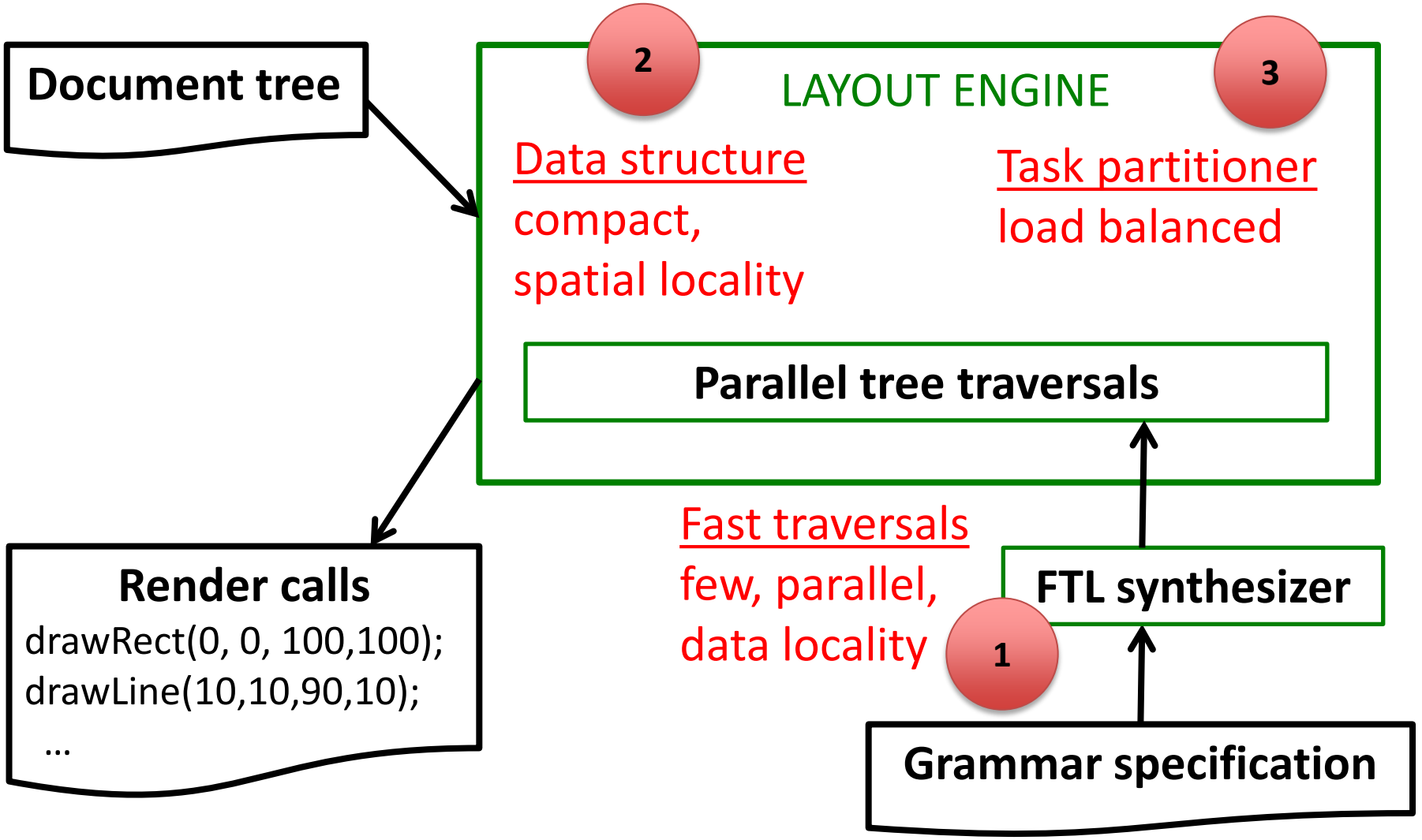
Phase 3: solved width

Phase 4: height, relative x/y position

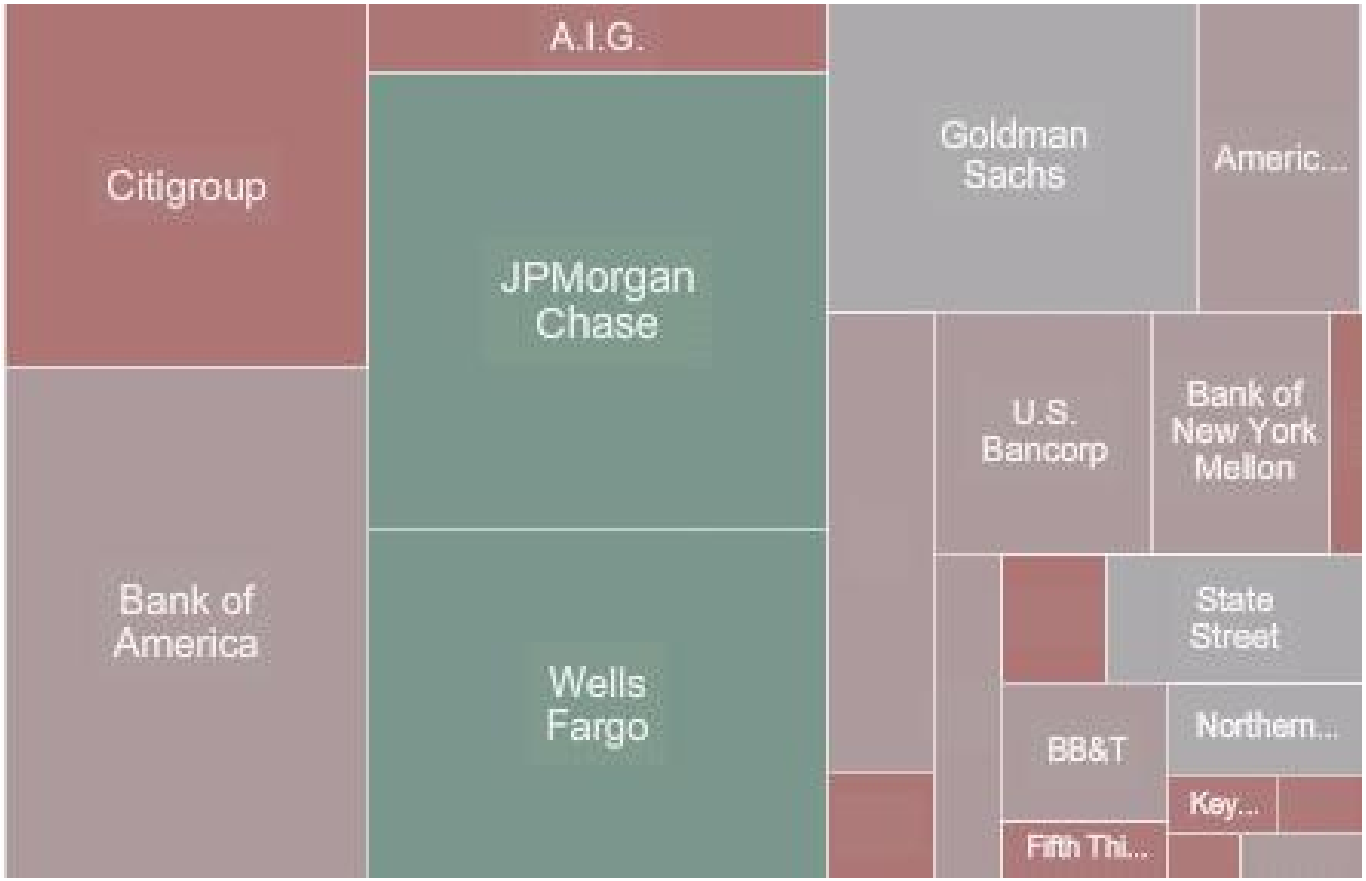
Phase 5: absolute x/y position

5 Phases: Each Exhibits Tree Parallelism

Parallel Layout Engine Generator



Demo: TreeMap



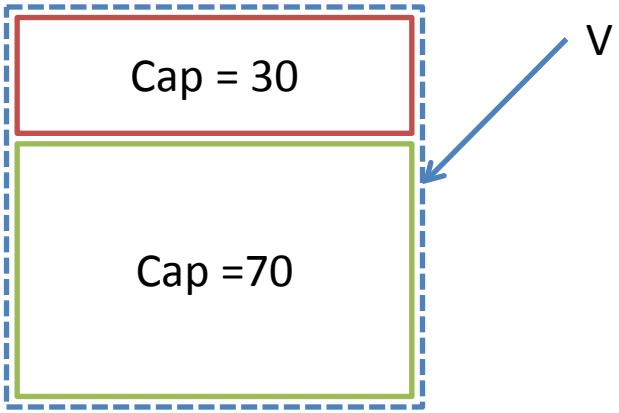
TreeMap of Financial Industry (NY Times)

http://www.nytimes.com/interactive/2010/02/01/us/budget.html

Specification of V

TreeMap spec in English:

1. V is a rectangle with some style.
2. V area is divided vertically among its children
3. V's children are stacked on top of each other.
4. V area is proportional the sum its children's capitalization



The Three Constraints

```
trait VDiv(h, w) {    // vertical division
  h = children[0].h + children[1].h
  w = children[0].w = children[1].w
}
trait VStack() {      // vertical stacking
  children.left = 0
  children[0].top = 0
  children[0].h = children[1].top
}
trait TreeMap(h, w, cap) { // area =~ cap
  SCALE * cap = h * w
  cap = children[0].cap + children[1].cap
}
```