Lecture 15: Treemaps
Stock Market Viz

S&P 500 Stock Market Treemap

size represents 'weight' of company
color represents 'change' in stock price

Price Change
- > 8
- 6 - 8
- 4 - 6
- 2 - 4
- 0 - 2
- 0 - 2
- 2 - 4
- 4 - 6
- 6 - 8
< - 8

Mouse over tiles to see detailed info. Click on tiles to see current price (NYSE)
Building Treemaps with a Greedy Algorithm

Algorithm 1 BuildTreeMap($data$)

Require: A list of $n$ data items. For simplicity, we assume each item is a weight, but it might be a more complex object in reality.

$T \leftarrow$ a list of $n$ trees ($T = T_1 \ldots T_n$)
Sort $T$ from highest weight to lowest weight

while $|T| > 1$ do
  $Z_1 \leftarrow$ the last tree in $T$
  $Z_2 \leftarrow$ the second-to-last tree in $T$
  $Z \leftarrow$ Tree($weight(Z_1) + weight(Z_2)$, $Z_1$, $Z_2$)
  Replace $Z_1$ and $Z_2$ with $Z$ in $T$.
  Sort $T$ form highest weight to lowest weight

end while

return The final tree in $T$
From Trees to Treemaps

[Diagram of a tree structure with nodes labeled 56, 20, 26, 11, 15, 10, and 5, and corresponding treemap visualizations]
A recursive procedure for generating rectangles

leaf If the $t$ is a leaf, then return the list containing $r$; and

non-leaf if $t$ is not a leaf, then

1. split $r$ into two smaller rectangles $r_1$ and $r_2$ along the axis given by $o$ using weight proportional to the left and right subtrees respectively;
2. recursively find the partition of $r_1$ by making a recursive call on the left subtree, passing $r_1$ and the opposite orientation of $o$;
3. recursively find the partition of $r_2$ by making a recursive call on the right subtree, passing $r_2$ and the opposite orientation of $o$;
4. return the concatenation of these two partitions.