NTNU – Trondheim
Norwegian University of
Science and Technology

Recitation lecture: problem set 3

Some words about PS 2

- Making the grammar LR(0)

  $S \rightarrow w\ X\ Y\ z$
  $X \rightarrow M\ B \mid M\ B\ e\ X$ ==>
  $Y \rightarrow e\ B \mid \varepsilon$
  $M \rightarrow m$
  $B \rightarrow b$

  $S \rightarrow w\ X\ z$
  $X \rightarrow M\ B\ X'$
  $X' \rightarrow e\ Y \mid \varepsilon$
  $Y \rightarrow X \mid B$
  $M \rightarrow m$
  $B \rightarrow b$

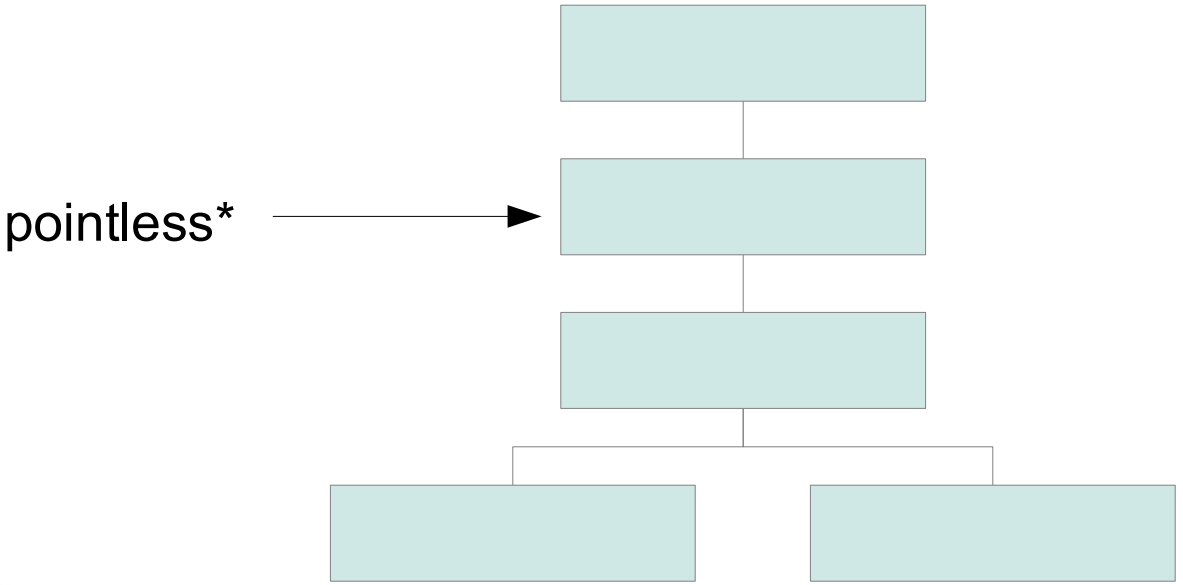
- Write some legal statements to look for alternative patterns
- Obvious for most: left factoring of X
- Less obvious: Remove Y from S and rewrite X

Intro to PS 3: Simplifying trees

- Recursive traversal of syntax tree, removing nodes that are unneeded for different reasons
- Should be easier (or at least less typing) than PS 2
- Compiler construction is your chance to finally get really comfortable with recursion.

Single nodes (syntax artifacts)

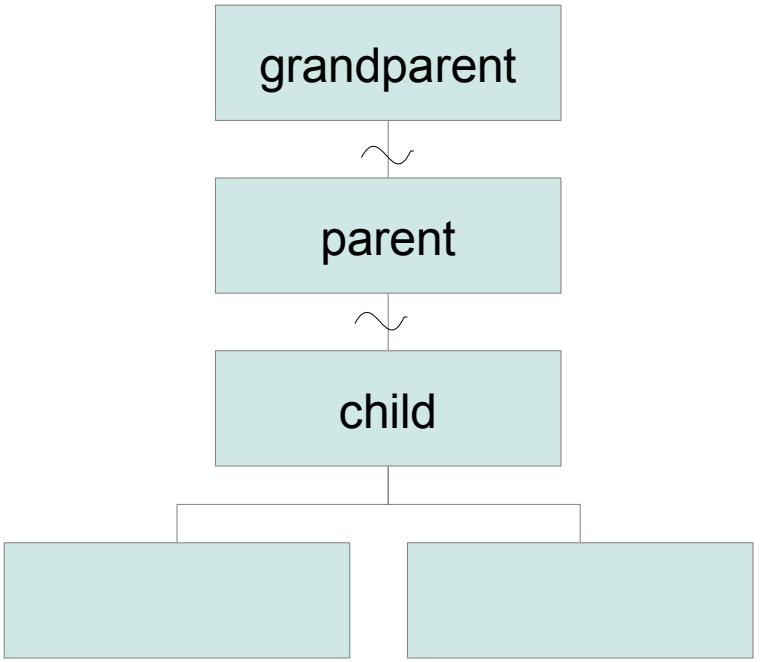
- Some nodes have no semantic value whatsoever, they just simplify the grammar
  - Global, declaration_list → variable_list

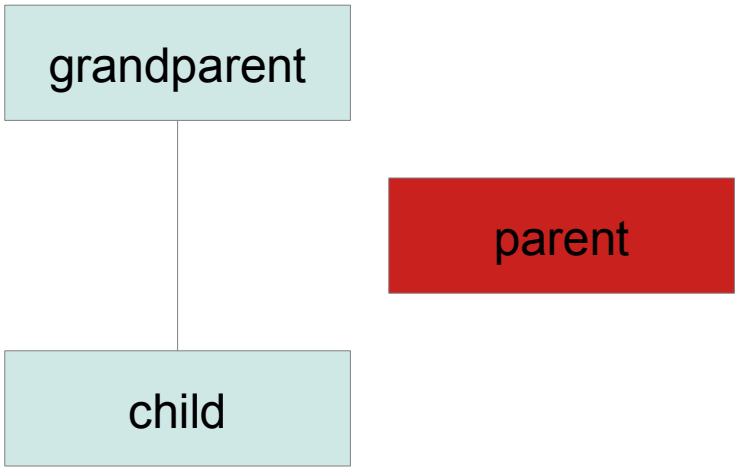


*Also *pointerless*, after we've freed this fellow (sry, bad pun)

Single nodes (syntax artifacts)

- When reaching nodes like this in any kind of traversal, disassociate them, connect the child, delete



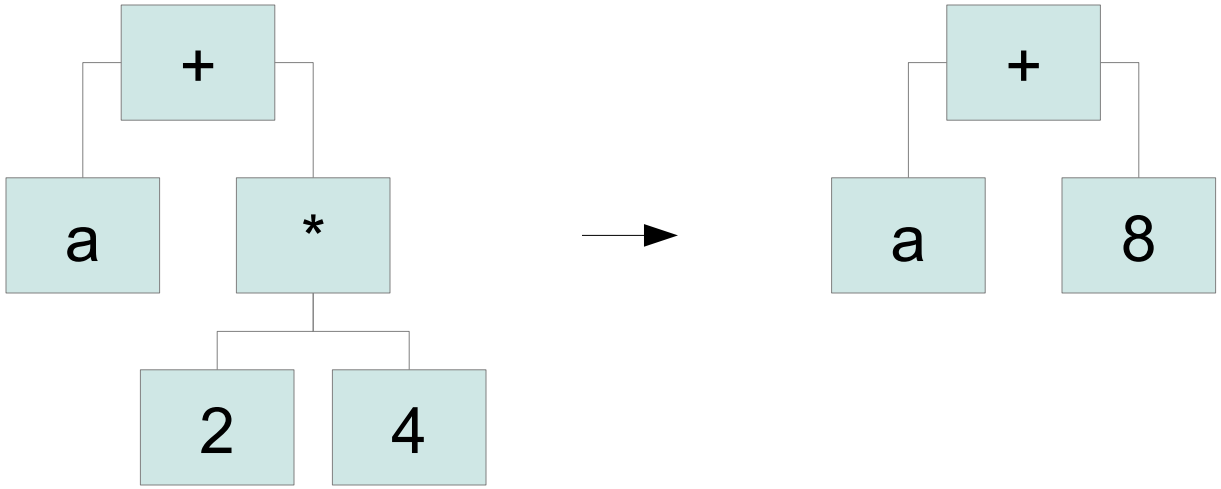


Lists

- Grammar creates deep linked lists
- We prefer them flat
  node->children[2] instead of
  node→children[1]→children[1]→children[1]
- Step 1: Recurse to the bottom (1 child)
- Step 2: When there are two or more children, disconnect the redundant node and bring the list element up one level
- Step 3: Repeat until there is only one list node holding all the list elements

Constant expressions

- Expressions containing only constants can be computed at compile time
- Again, bottom-up to reduce all constant expressions



Library function of the week

- Convenient function to have when editing lists of children
  node_t ****new_array =**
  realloc(old_array, (new_size)*size_of(node_t*));
- Can theoretically fail, if so then old_array remains valid
- (Unsafe) but still okay
  new_array = realloc(old_array, (new_size)*sizeof(node_t*));
- ^ You wouldn't do this in production. If you want to do it the proper way, check new_array for success before trusting it