Disambiguating Grammars

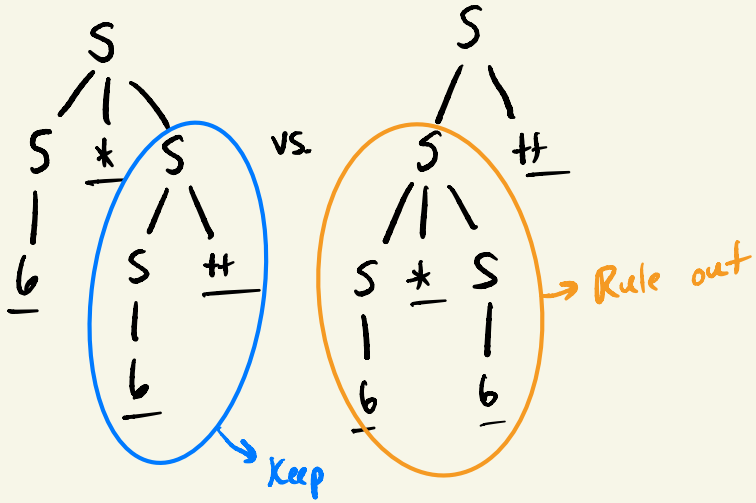
[No algorithm in general
Requires creativity.]

Example 1

Starting grammar:

$$S ::= S * S \mid S\,{+}{+} \mid 6$$

$6 * 6 {+}{+}$

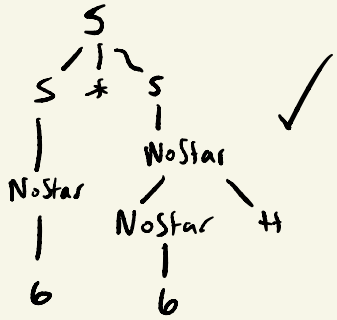


$$S ::= S * S \mid NoStar$$

$$NoStar ::= NoStar\,{+}{+} \mid 6$$

Is this unambiguous

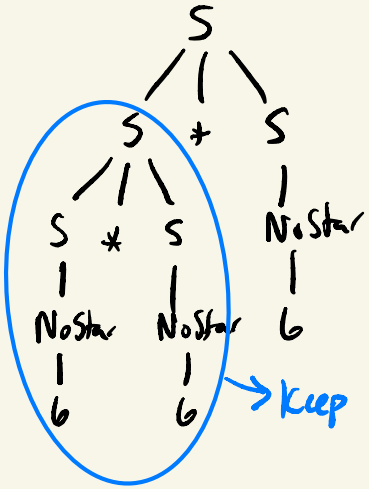
Check $6 * 6{+}{+}$



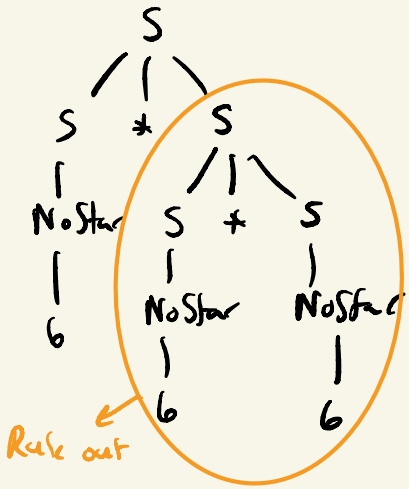
$S ::= S * S \mid NoStar$

$NoStar ::= NoStar\ 6 \mid 6$

$6 * 6 * 6$



vs.



left associativity

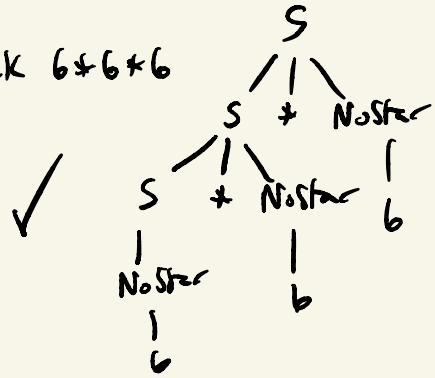
$S ::= S * NoStar \mid NoStar$

$NoStar ::= NoStar\ 6 \mid 6$

Unambiguous? Yes.

Prove it by induction on the depth of parse trees.

Check $6*6*6$



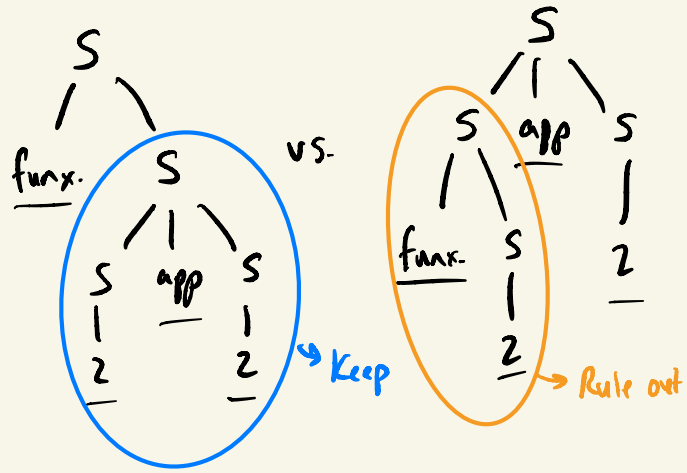
Example 2

Starting grammar:

$$S ::= \text{fun x. } S \mid S \text{ app } S \mid 2 \mid (S)$$

Ambiguous?

fun x. 2 app 2



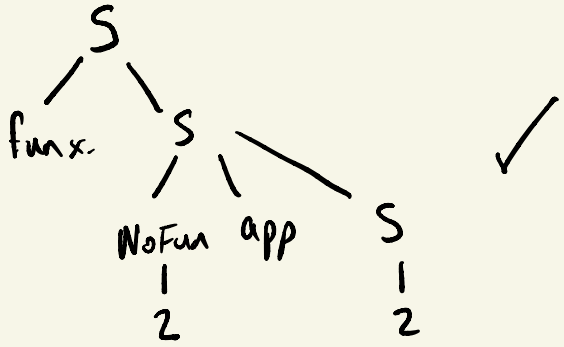
$$S ::= \text{fun x. } S \mid \text{NoFun app } S \mid 2 \mid (S)$$

$$\text{NoFun} ::= \text{NoFun app NoFun} \mid 2 \mid (S)$$

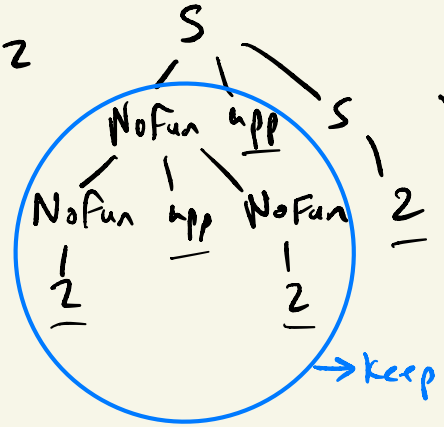
S ::= fun x. S | NoFun app S | 2 | (S)

NoFun ::= NoFun app NoFun | 2 | (S)

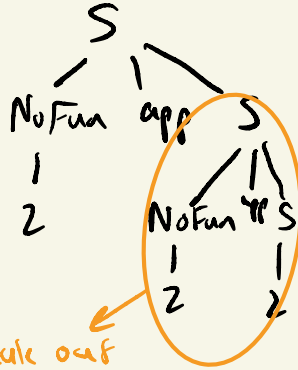
fun x. 2 app 2



2 app 2 app 2



vs.



S ::= fun x. S | NoFun app NoApp | NoFunNoApp

NoFun ::= NoFun app NoFunNoApp | NoFunNoApp

NoApp ::= fun x. S | NoFunNoApp

NoFunNoApp ::= 2 | (S)

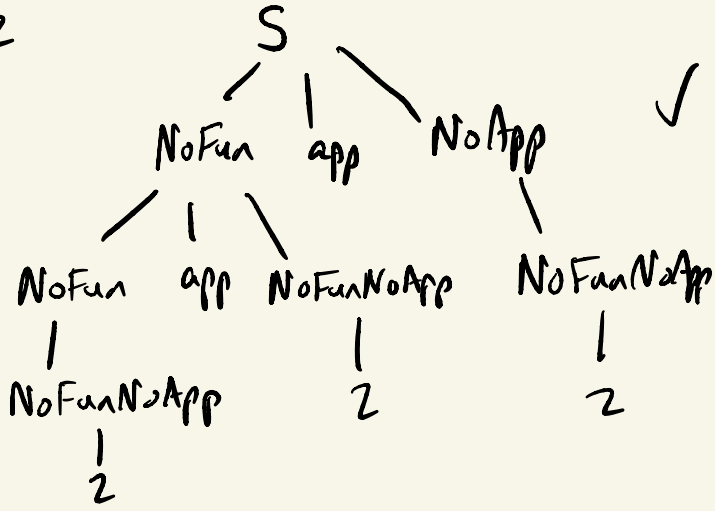
S ::= fun x. S | NoFun app NoApp | NoFunNoApp

NoFun ::= NoFun app NoFunNoApp | NoFunNoApp

NoApp ::= fun x. S | NoFunNoApp

NoFunNoApp ::= 2 | (S)

2 app 2 app 2



Example 3

Starting grammar:

$$S ::= \text{fun } x.\ S \mid S * S \mid S + S \mid 2$$

Precedence & Associativity Specification

right * highest precedence
left +
fun x. lowest precedence

Idea: Start from lowest precedence, and work your way up to higher precedence.

$$S ::= \text{fun } x.\ S \mid Plus$$
$$Plus ::= Plus + Star \mid Star$$
$$Star ::= Atom * Star \mid Atom$$
$$Atom ::= 2$$

Missing things like $2 * \text{fun } x.\ 2$

S ::= fun x. S | Plus

Plus ::= Plus + Star | Star

Star ::= Atom * Star | Atom

Atom ::= 2

Missing things like $2 * \text{fun } x.\ 2$

S ::= NoFun | Fun

NoFun ::= NoFun + NoPlusNoFun | NoPlusNoFun

NoPlusNoFun ::= NoPlusNoFunNoStar * NoPlusNoFun

NoPlusNoFunNoStar ::= 2

Fun ::= NoFun + NoPlus | NoPlus

NoPlus ::= NoPlusNoFunNoStar * NoPlus | FinallyFun

FinallyFun ::= fun x. S

S ::= NoFun | Fun

NoFun ::= NoFun + NoPlusNoFun | NoPlusNoFun

NoPlusNoFun ::= NoPlusNoFunNoStar * NoPlusNoFun | NoPlusNoFunNoStar

NoPlusNoFunNoStar ::= 2

Fun ::= NoFun + NoPlus | NoPlus

NoPlus ::= NoPlusNoFunNoStar * NoPlus | FinallyFun

FinallyFun ::= fun x. S

2 * Fun x. 2