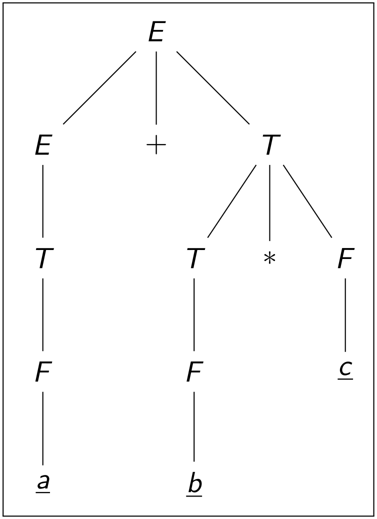
Abstract syntax tree

Marcin Kuta

Theory of Compilation
Laboratory 3

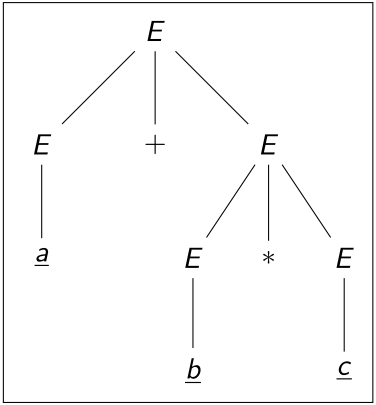
Parse tree and abstract syntax tree

Parse tree in unambiguous grammar:



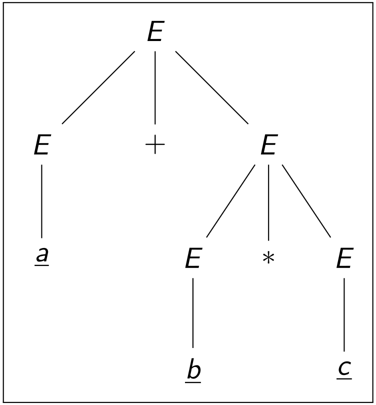
Parse tree and abstract syntax tree

Parse tree in ambiguous grammar:

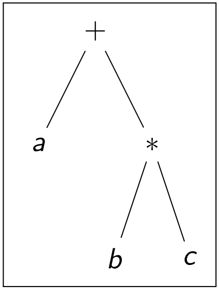


Parse tree and abstract syntax tree

Parse tree in ambiguous grammar:



Abstract syntax tree (AST):



Actions in SLY or PLY

```
@_('expr "+" expr')
def expr(p):
    return AST.BinExpr(p[1], p[0], p[2])
    return AST.BinExpr(p[1], p.expr0, p.expr1)
```

```
def p_expression_1(p):
    """expression: expression '+' expression"""
    p[0] = AST.BinExpr(p[2], p[1], p[3])
...
```

Actions in SLY

```
@_('expr "+" expr',
   'expr "-" expr',
   'expr "*" expr',
   'expr "/" expr')
def expr(p):
    return AST.BinExpr(p[1], p[0], p[2])


@_('"(" expr ")"')
def expr(p):
    return p[1]
```

AST definition

```
...
class BinExpr(Node):
    def __init__(self, op, left, right):
        self.op = op
        self.left = left
        self.right = right
...
```

```
from dataclasses import dataclass
...
@dataclass
class BinExpr(Node):
    op: Any
    left: Any
    right: Any
...
```

Visitors

	TreePrinter	TypeChecker	Optimizer	...
VarExpr				
NumExpr				
BinExpr				

- Object-oriented style
- Phase-oriented style (syntax separate from interpretations)

Tree printer

```
def addToClass(cls):

    def decorator(func):
        setattr(cls,func.__name__,func)
        return func
    return decorator



class TreePrinter:
...
    @addToClass(AST.BinExpr)
    def printTree(self):
        print(self.op)
        self.left.printTree(indent+1)
        self.right.printTree(indent+1)
...
```

References

1. https://sly.readthedocs.io/en/latest/sly.html
2. http://www.dabeaz.com/ply/ply.html
3. https://eli.thegreenplace.net/2009/02/16/abstract-vs-concrete-syntax-trees
4. https://eli.thegreenplace.net/2016/the-expression-problem-and-its-solutions