Compiling Techniques
Revision Notes
Ver 0.1

Guy Taylor

April 2011

Contents

1 Introduction

This document is a set of revision notes for the Compiling Techniques[1] course at the Univerisy of Edinbugh.

2 Parsing

Parsing

	Top Down	Bottom Up
Design	TODO	Can create a state machine from the grammer
Efficent	TODO	Efficent
Method	TODO	Start resurse; replace all NT; untill no NT (None Terminal) left;

2.1 Bottom UP

2.1.1 Reduction

As u work backwards you get the parentless leafs. These are the 'upper fringe'. You then parse a new leafe untill they join. This is called reduction. Side Note: 'Finding Reductions' position is referedfrom the most right char possition.

2.1.2 Hurdles

Find the right hand side that is a definate terminal (ie ; in c)
Allows stuff to be streamed efficently (?? maybe ??)

2.1.3 Shift-Reduce

Bad errors: Errors can allways be found but as there is a laxck of contect no good error message can be produced

2.1.4 LR(1)

- Bottom up
- table based
- Shift-replace
- one char look ahead

Extra: There can be more than one char look ahead (ie. LR(2), LR(3) ...)
Note: Although the language is contex free we use DFA (D.. Finite Atomata) to prosses it as it is only done 'line by line'.

References

[1] "Compiler Techinques Home page" by "Bjrn Franke", "http://www.inf.ed.ac.uk/teaching/courses/ct/"