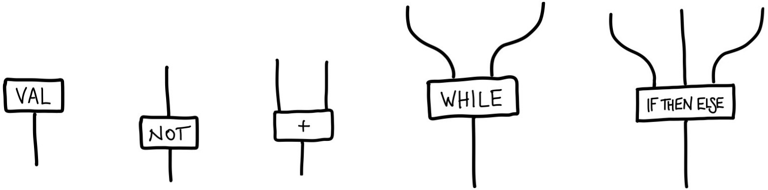
Contour Categories, Control Flow Analysis

Malin Altenmüller

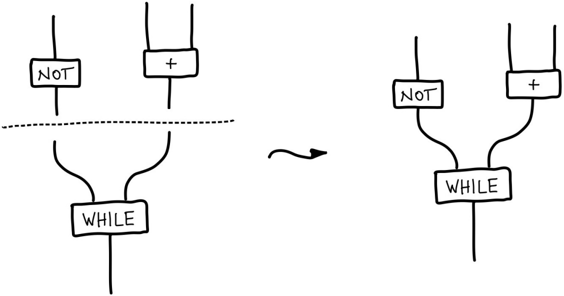
Huawei-Edinburgh Joint Lab Workshop
12 December 2023

Abstract Syntax – Generators

abstract syntax as free trees on a set of generators:

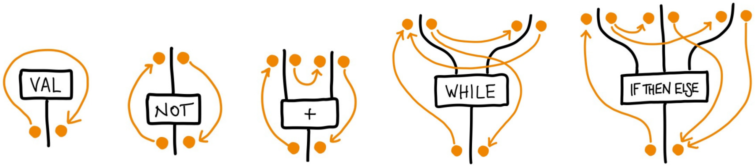


Abstract Syntax Trees



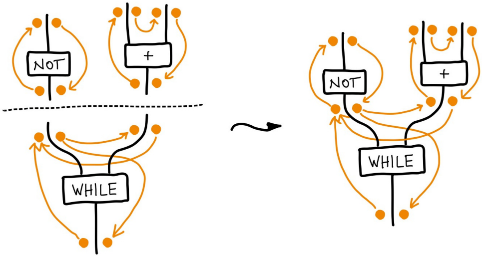
Control Flow – Generators

assign control flow to all generators:



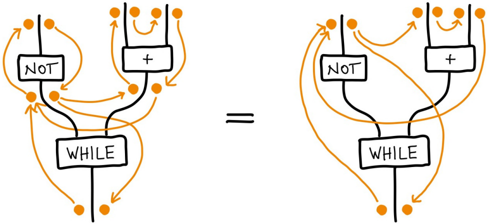
Control Flow – Composition

composition follows from the underlying AST:



Control Flow – Composition

composition follows from the underlying AST:

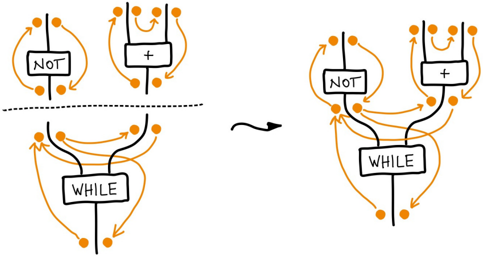


(a closed term without any inputs has very boring control flow)

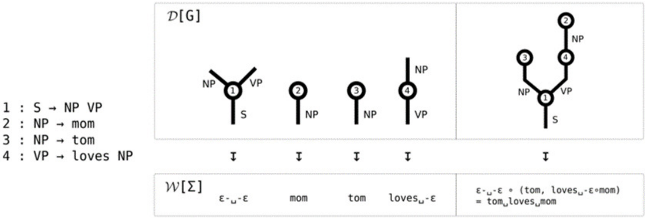
Control Flow

- order of execution of instructions, also traversal of AST
- can go forwards or backwards
- includes all possible cases: overapproximation
- crucially: order of variable accesses
- control flow optimisations mainly about memory access

Focus and Environment



Context-free Languages[1]



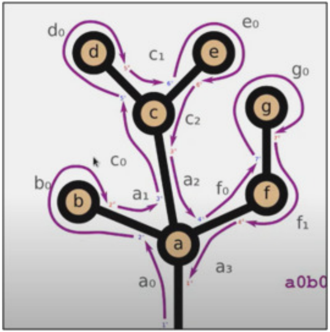
in the context of defining words in a context-free language:

- language generators as nodes
- building free trees from those

[1] Melliès and Zeilberger, "Parsing as a lifting problem and the Chomsky-Schützenberger representation theorem".

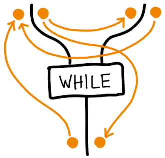
Contour Categories

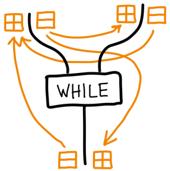
describe a left-to-right traversal of the derivation tree:



Generalising Contour Categories

- not strict about arrows that are allowed, but some rules still apply:





- the notion of contour is not ideal for being overly generalised
- inspired by strategies on games: relations on sets with polarity

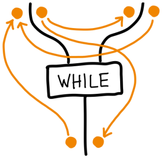
Sets with Polarity

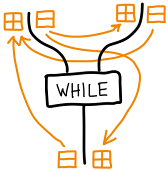
- sets with polarity: $\mathcal{A} = (A, pol_{\mathcal{A}})$ where $pol_{\mathcal{A}} : A \to \{-, +\}$
- negated polarity : $\mathcal{A}^{\perp}$
- union : $\mathcal{A} \| \mathcal{B}$

Relations on Sets with Polarity

we are interested in relations:

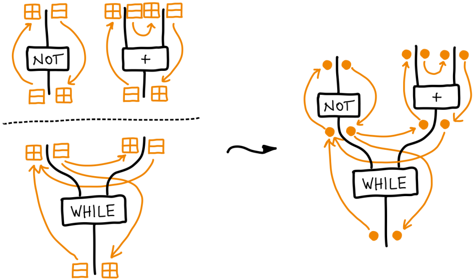
- $R : \mathcal{A}^{\perp} \| \mathcal{B}$
- relations where all arrows are of the form $- \to +$





Interaction and Composition

Compose relations $R : \mathcal{A}^{\perp} \| \mathcal{B}$ and $S : \mathcal{B}^{\perp} \| \mathcal{C}$



- compose along *elements*, not polarities
- interaction: remember intermediate steps
- composition: forget about intermediate states

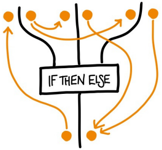
Sets with Polarity

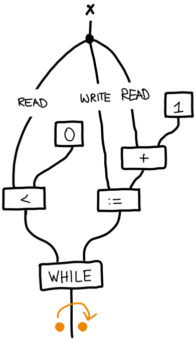
Advantages:

- different notions of composition
- well-known area of game theory
- extendable to represent exception
- extendable to concurrent processes

So far it's all Syntax

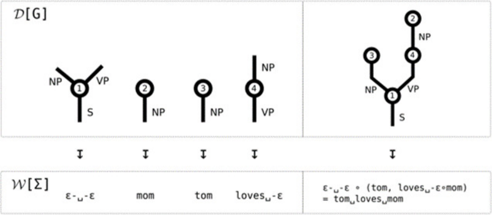
- control flow defined in this way has no access to values
- but we may want to include some semantic information



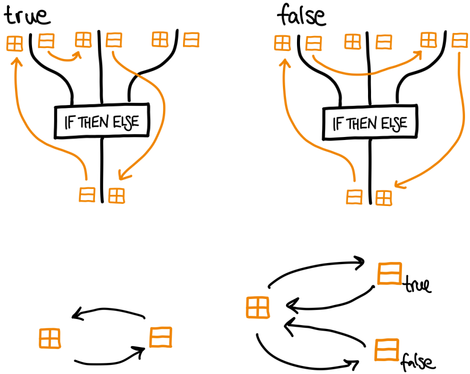


Automata *refining* context-free grammars

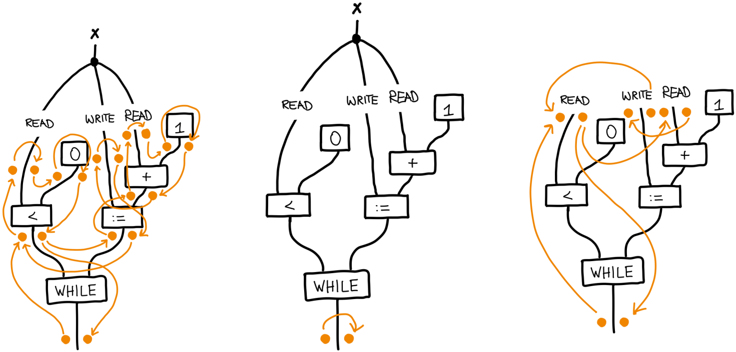
```
1 : S → NP VP
2 : NP → mom
3 : NP → tom
4 : VP → loves NP
```



Automata refining Control Flow



Multiple Options for Refinement?



Summary

- define control flow on top of the AST
- relations on sets with polarities to generalise contour categories
- incorporate semantic information by an automaton *refining* the original definition

Thank You for Your Attention!

Appendix