

(These notes contain mostly material discussed in class, but not presented in the handouts.)

## 1. Handout 12-Unwinding Programs: Two Small Examples

- The process of simple loop unwinding is analyzed in the following steps: We draw a flow chart for the original program and do the loop unwinding in the GCD example program in Python. With "if-then-else" clauses we "open the loop" and separate the two paths, making a copy of the step following the loop, for both branches. We see that instead of a DAG, in this case we get a Tree. We can use a heuristic to cut the if's off and stop the program's execution. In the Figures below we see a simple representation of this process.

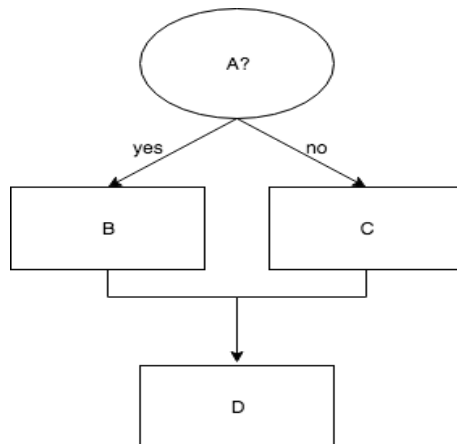


Fig1 : DAG

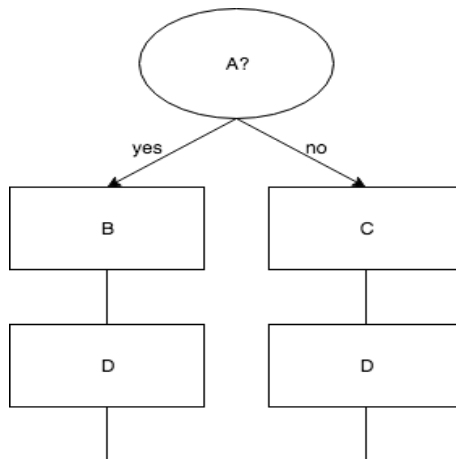


Fig2 : Tree

- Unwinding with timestamps:
  - Timestamps are used to keep track of the evolution of the values assigned to the variables.
  - There is a bounded loop in our GCD program, so we repeat the calculation of the expression 3 times.
  - (p.12) The formulas  $\phi_\pi$  are not in Propositional Logic, but in First Order Logic. They represent the "action" of the program.
  - The numbers of the execution steps written in parentheses as  $()^n$  mean that these steps are repeated n times.

- “=” mathematically means equality, but in programming it is used as the update symbol.
- By using timestamps, we simplify our expressions, and use variables only once as we do not update the same variables multiple times.

## 2. Handout 13-Binary Decision Diagrams (BDDs)

- The Binary Decision Diagrams or BDD's are covered in Chapter 6 of [LCS] and they improve methods covered in the previous Chapters 1-5 of the book.
- With  $\text{table}(\phi_1)$  we are referring to the columns of the Truth Table corresponding to the variables in a formula  $\phi_1$  as well as the last column of the Truth Table, which gives us the value of  $\phi_1$ , as shown here:

x	y	z	$\phi_1$
...	...	...	...