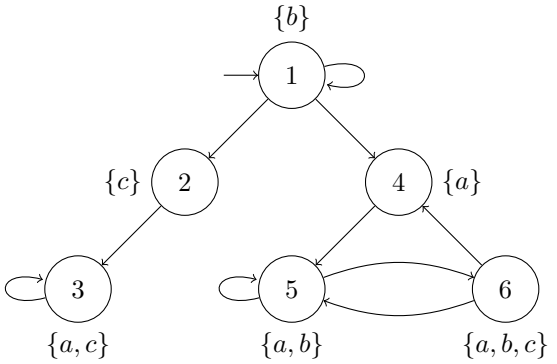
Software verification - Computation Tree Logic

assignments for week 2

Assignment 3 should be handed in in the post box at floor 1 of Mercator 1, behind the couches, before monday 24-04-2017, 12:30.

1 Model checking by hand

Consider the following transition system:



For each of the CTL-formulas below, check whether the formula holds for this transition system. Argue why (not).

a) $\forall\Box a$

b) $\exists\Diamond\forall\Box a$

c) $\forall\Diamond\forall\Box a$

d) $\forall\bigcirc(a \vee b \vee c)$

e) $\forall\bigcirc(a \wedge b \wedge c)$

f) $(\exists\bigcirc a) \wedge (\exists\bigcirc b) \wedge (\exists\bigcirc c)$

g) $\forall\Box\exists\Diamond c$

h) $\forall\Box\exists\bigcirc a$

i) $\forall((b \vee a)\ \mathbf{U}\ c)$

j) $\forall\Box(\exists(b\ \mathbf{U}\ c) \rightarrow (\exists\bigcirc\neg a))$

2 Equivalences

Do the following equivalences hold? Argue why (proving is generally difficult). In case they do not hold, give a transition system for which the equivalence does not hold:

a) $\forall\bigcirc\forall\Box\psi = \forall\Box\forall\bigcirc\psi$

b) $\forall\bigcirc\forall\Diamond\psi = \forall\Diamond\forall\bigcirc\psi$

c) $\exists\bigcirc\exists\Box\psi = \exists\Box\exists\bigcirc\psi$

d) $\exists\bigcirc\exists\Diamond\psi = \exists\Diamond\exists\bigcirc\psi$

3 CTL-properties of processes

Consider a computer which runs two processes p_1 and p_2 (or p_i with $i \in \{1, 2\}$) We define the following atomic propositions on the processes:

- run_i defines whether p_i is now running
- req_i defines whether p_i requests to be ran
- $stop_i$ defines whether p_i is terminated by the user

Formalize the following properties in CTL, or explain why it cannot be expressed in CTL. If it cannot be expressed in CTL, express it in LTL instead, if possible.

a) p_1 and p_2 cannot run at the same time.

b) p_1 and p_2 can run at the same time.

c) Process p_1 will run over and over again.

d) Process p_1 can run over and over again.

e) Any process can be ran first.

f) if p_2 never runs, then p_1 will run forever.

g) if p_2 never runs, then p_1 may run forever.

h) A user can always terminate a process.

i) If a user never terminates p_1, p_1 will always run over and over again.

j) Always if p_1 does not run, p_2 may first run before p_1 starts running again.