Plan of the lecture

- Based on Huth and Ryan book.
- the language of CTL
- models of CTL
- truth of CTL formulas
- expressing properties in CTL
- CTL and model checking

CTL

- CTL is a subset of CTL*. In CTL*, path quantifiers and temporal modalities can be combined in any order.
- In CTL, combinations are restricted allowing for special purpose model-checking algorithms.

Syntax of CTL

CTL is a branching-time temporal logic

- a set of atomic propositions p, q, r, …
- standard logical connectives: $\neg$, $\wedge$, $\vee$, $\rightarrow$
- temporal connectives: AX, EX, AF, EF, AG, EG, AU and EU.

Temporal connectives

- AX ϕ : on **A**ll paths, ϕ is true in the ne**X**t state
- EX ϕ : on som**E** path, ϕ is true in the ne**X**t state
- AF ϕ : on **A**ll paths, in some **F**uture state ϕ is true
- EF ϕ : on som**E** path, in some **F**uture state ϕ is true
- AG ϕ : on **A**ll paths, in all future states (**G**lobally) ϕ is true
- EG ϕ : on som**E** path, in all future states (**G**lobally) ϕ is true
- AU(ϕ_1, ϕ_2) : on **A**ll paths, ϕ_1 is true **U**ntil ϕ_2 is true
- EU(ϕ_1, ϕ_2) : on som**E** path, ϕ_1 is true **U**ntil ϕ_2 is true

Models of CTL

A model M = (S, $\rightarrow$, L) for CTL is given by:

- a set of states S
- a transition relation $\rightarrow$, on S, such that for every $s \in$ S there exists an $s' \in$ S such that $s \rightarrow s'$
- a labelling function L(s) specifying the set of atomic propositions which are true at s.

Definition of truth for CTL formulas

Let M = (S, $\rightarrow$, L) be a model of CTL. For any state $s \in$ S, a CTL formula ϕ holds at s iff:

M, $s \models \phi$

1. M, $s \models \top$ and M, $s \not\models \bot$ for all $s \in$ S
2. M, $s \models p$ iff $p \in$ L(s)
3. M, $s \models \neg\phi$ iff M, $s \not\models \phi$
4. M, $s \models \phi_1 \rightarrow \phi_2$ iff M, $s \not\models \phi_1$ or M, $s \models \phi_2$

Definition of truth for CTL formulas 2

5. M, $s \models AX\,\phi$ iff for all s_1 such that $s \rightarrow s_1$, we have M, $s_1 \models \phi$

6. M, $s \models EX\,\phi$ iff for some s_1 such that $s \rightarrow s_1$, we have M, $s_1 \models \phi$

7. M, $s \models AF\,\phi$ iff for all paths $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$, where s_1 equals s and there is some s_i such that M, $s_i \models \phi$

8. M, $s \models EF\,\phi$ iff there exists a path $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$, where s_1 equals s and there is some s_i such that M, $s_i \models \phi$

9. M, $s \models AG\,\phi$ iff for all paths $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$, where s_1 equals s and all s_i along the path we have M, $s_i \models \phi$

10. M, $s \models EG\,\phi$ iff there exists a path $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$, where s_1 equals s and all s_i along the path we have M, $s_i \models \phi$

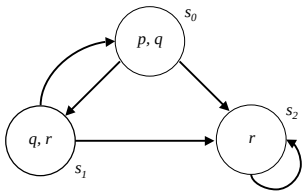
Definition of truth for CTL formulas 3

11. M, $s \models A\,U(\phi_1, \phi_2)$ iff for all paths $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$, where s_1 equals s and that path satisfies $U(\phi_1, \phi_2)$, i.e., there is some s_i along the path such that M, $s_i \models \phi_2$ and for each $j < i$, we have M, $s_j \models \phi_1$

12. M, $s \models EU(\phi_1, \phi_2)$ iff there exists a path $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$, where s_1 equals s and that path satisfies $U(\phi_1, \phi_2)$, i.e., there is some s_i along the path such that M, $s_i \models \phi_2$ and for each $j < i$, we have M, $s_j \models \phi_1$

Graphs as models

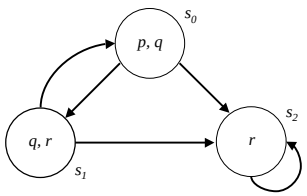
We can represent a model M as a directed graph whose nodes are states containing all the atomic propositions which are true in that particular state, e.g.:



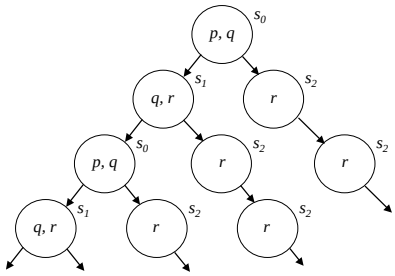
Semantics of CTL

CTL formulas can be evaluated relative to the computation tree which is the unwinding of the labelled transition system describing the system to be modelled, e.g.:

Example:



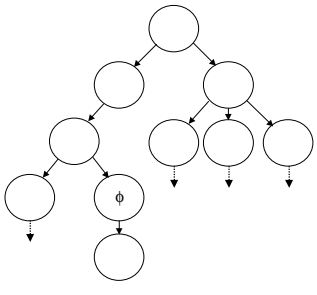
Example: unwinding the graph



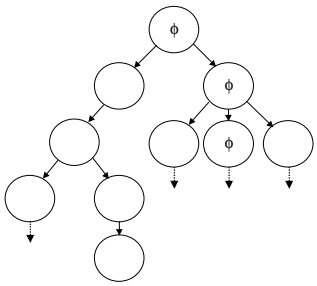
Question

- is the CTL formula AF r true at s_0?

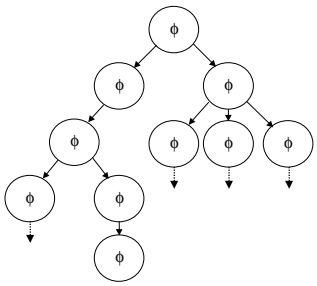
Example: a system which satisfies EF ϕ



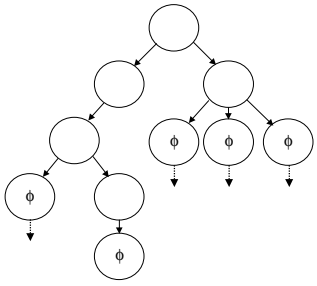
Example: a system which satisfies EG ϕ



Example: a system which satisfies AG ϕ



Example: a system which satisfies AF ϕ

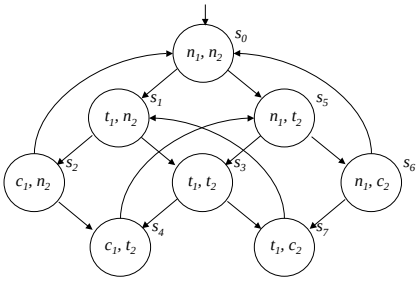


Example: mutual exclusion

Given the following model of a simple mutual exclusion protocol for two processes

- a set of atomic propositions describing the system: n_i (process i is not in its critical section, i.e., it is initialising or in the remainder), t_i (process i is trying to enter its critical section), and c_i (process i is in its critical section)
- each process undergoes transitions in the cycle $n_i \rightarrow t_i \rightarrow c_i \rightarrow n_i \ldots$
- only one process can make a transition at time (e.g., a single processor and the transitions are atomic)
- the two processes start off not in their critical sections, in the initial state s_0

Example: mutual exclusion



Example: mutual exclusion

- Mutual Exclusion: AG $\neg(c_1 \wedge c_2)$ (true in the example above)
- Absence of starvation (false in the example above):

AG($t_i \rightarrow \neg$EG $\neg c_i$) (if a process i is trying to get into a critical section, it is impossible to follow a path from that state where globally c_i is false).

Model checking

- Given a system to verify, generate the state space (transition system).
- This is done automatically provided the system is specified in suitable format. The system is finite (not a tree unravelling!)
- Verify the formula on the resulting transition system. This is also done automatically.
- The simplest algorithm is as follows. Given a formula ϕ to check and a system S
  - generate the set of subformulas of ϕ; order them by complexity (propositional variables first,…, ϕ last)
  - Repeat: take a subformula from the list and annotate states of S which satisfy it, with this subformula.
- When we reach the end of the list we see which states satisfy ϕ.

Annotation

- Since states come with a labelling function, we know how to annotate states with propositional variables.
- When the current subformula is $\neg \psi$, we annotate with it the states which are not annotated with ψ (note that ψ precedes $\neg \psi$ in the list of subformulas, so we already annotated the states with ψ).
- How to do this for $\psi_1 \rightarrow \psi_2$ and $\Box \psi$, is the subject of an exam question.

Annotation

- Here, I'll give a naive algorithm for EX, EU and AF (these three connectives are sufficient to express all CTL connectives.
  - If ϕ is EX ψ, annotate predecessors of any state labelled ψ by EX ψ.
  - If ϕ is EU(ψ_1,ψ_2), first find all states annotated ψ_2 . Then work backwards from those states and so long as we encounter ψ_1 states we label them by EU(ψ_1,ψ_2).
  - If ϕ is AFψ, first annotate all states annotated with ψ with AFψ. Then annotate a state with AFψ if all its successor states are annotated with AFψ. Releat until there is no change.