NSWI101: System Behaviour Models and Verification

6. Symbolic CTL Model Checking

Jan Kofroň

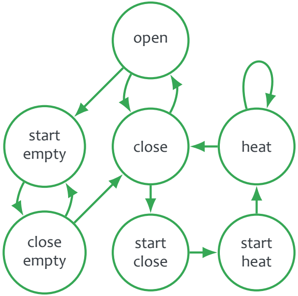
Faculty of Mathematics and Physics
Charles University

Department of Distributed and Dependable Systems

TODAY

- Symbolic CTL model checking using
  - OBDD
  - lattices
  - fixpoints

Model Checking



System model

AG (start $\rightarrow$ AF heat)

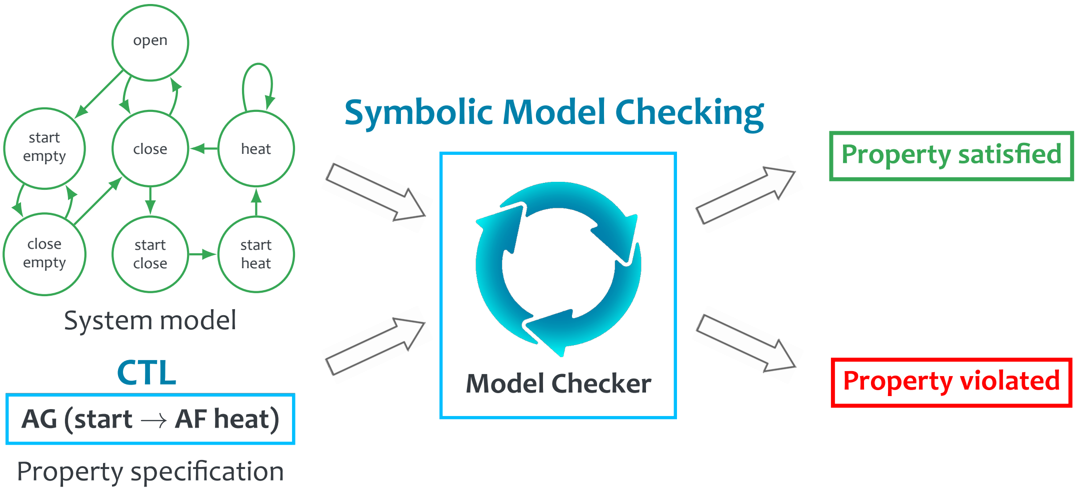
Property specification

Model Checker

Property satisfied

Property violated

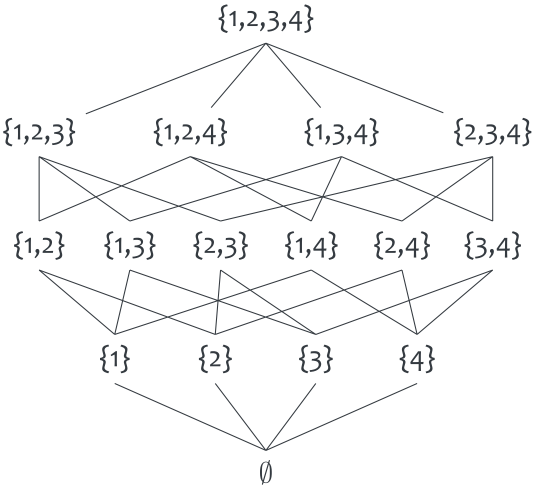
Model Checking



Recall: Lattice

- *Lattice* L is structure consisting of partially ordered set S of elements where every two elements have
  - unique *supremum* (least upper bound or join) and
  - unique *infimum* (greatest lower bound or meet)
- Set $P(S)$ of all subsets of S forms complete lattice
- Each element $E \in L$ can also be thought as *predicate* on S
- Greatest element of L is S ($\top$, true)
- Least element of L is $\emptyset$ ($\bot$, false)
- $\tau : P(S) \mapsto P(S)$ is called *predicate transformer*

Example: Subset Lattice of {1, 2, 3, 4}



Fixpoints

Let $\tau : P(S) \mapsto P(S)$ be predicate transformer

- τ is *monotonic* $\equiv Q \subseteq R \implies \tau(Q) \subseteq \tau(R)$
- Q is *fixpoint* of $\tau \equiv \tau(Q) = Q$

Fixpoint Computation

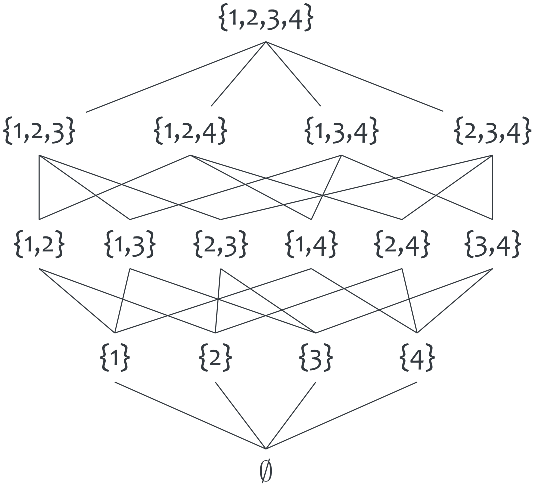
function LFP(τ : *PredicateTransformer*): Predicate
  $Q := \mathit{false}$
  $Q' := \tau(Q)$
  while $Q \neq Q'$ **do**
    $Q := Q'$
    $Q' := \tau(Q)$
  end while
  $\mathit{return}(Q)$
end function

Function Gfp differs just in initialization $Q := \mathit{true}$

Example of Fixpoints

Let $\tau(Q) = Q \cup \{1\}$.

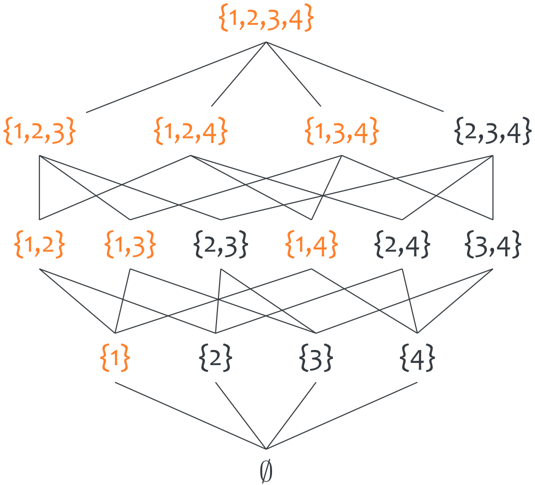
What are fixpoints of τ?



Example of Fixpoints

Let $\tau(Q) = Q \cup \{1\}$.

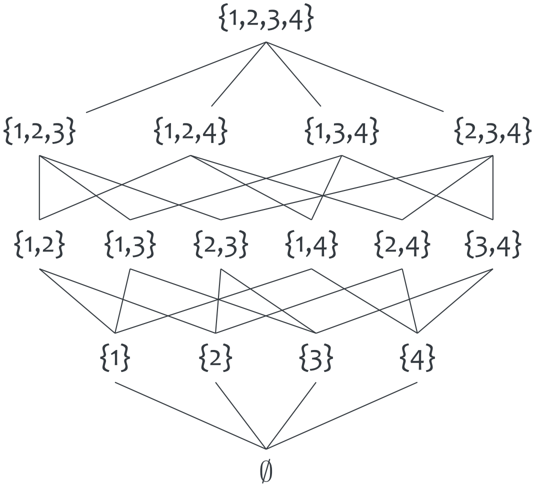
What are fixpoints of τ?



EXAMPLE OF FIXPOINTS

Let $\tau(Q) = Q \cup \{1\}$.

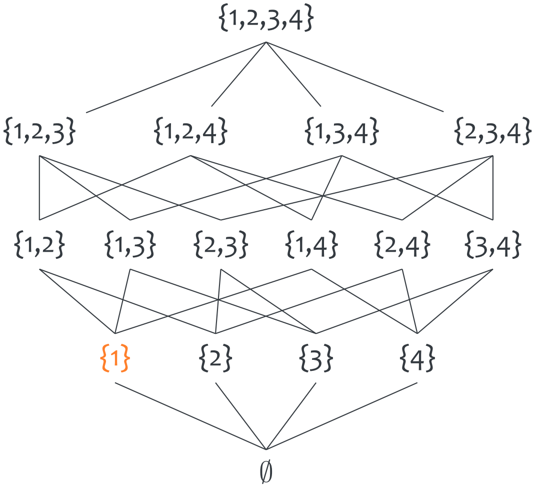
What is the least fixpoint of τ?



Example of Fixpoints

Let $\tau(Q) = Q \cup \{1\}$.

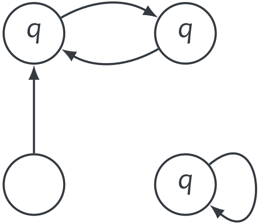
What is the least fixpoint of τ?



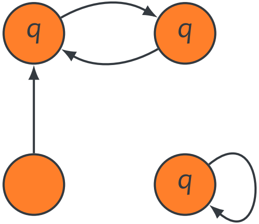
CTL Operators as Fixpoints

- We identify CTL formula f with set/predicate $\{s | M, s \models f\}$ in $P(S)$
- EG and EU may be characterized as least or greatest fixpoints of an appropriate predicate transformer:
  - $\mathrm{EG}\, q = \nu Z.(q \wedge \mathrm{EX}\, Z)$
  - $\mathrm{E}[p \,\mathrm{U}\, q] = \mu Z.\big(q \vee (p \wedge \mathrm{EX}\, Z)\big)$
- The same holds for EF, AG, AF, AU, however, those operators can be expressed using EG, EU
- Intuitively:
  - least fixpoints correspond to eventualities
  - greatest fixpoints correspond to properties that should hold forever

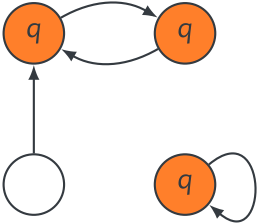
EG AS FIXPOINT



Input KS M



$\tau^0(\top)$



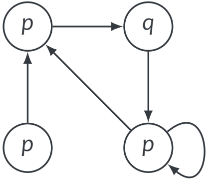
$\tau^1(\top)$

$M, s_0 \models \mathrm{EG}\, q$

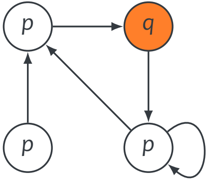
$\mathrm{EG}\, q = \nu Z.(q \wedge \mathrm{EX}\, Z)$

$\tau(Z) = \{s : s \models q \wedge (\exists t : s \to t \wedge t \in Z)\}$

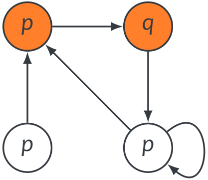
EU as Fixpoint



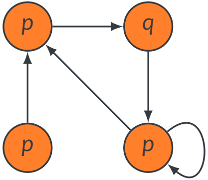
Input KS M



$\tau^1(\perp)$



$\tau^2(\perp)$



$\tau^3(\perp)$

$$M, s_0 \models \mathrm{E}[p \,\mathrm{U}\, q]$$

$$\mathrm{E}[p \,\mathrm{U}\, q] = \mu Z.(q \vee (p \wedge \mathrm{EX}\, Z))$$

$$\tau(Z) = \{s : s \models q\} \vee \{s : s \models p \wedge (\exists t : s \rightarrow t \wedge t \in Z)\}$$

Symbolic CTL Model Checking

Explicit model checking—e.g., Spin—is linear in size of generated state space

- usually exponential in size of input model
- resulting in state space explosion

Symbolic model checking operates on sets of states in each step of algorithm

- can mitigate state-space-explosion impact substantially

Quantified Boolean Formulae

QBFs are useful in symbolic CTL model checking

Quantification does not introduce greater expressive power:

- $\exists x\, f \equiv f|_{x=\bot} \vee f|_{x=\top}$
- $\forall x\, f \equiv f|_{x=\bot} \wedge f|_{x=\top}$

Symbolic CTL Model Checking

General approach identical to explicit model checking

- decomposing formula into sub-formulae
- identifying sets of states satisfying particular sub-formulae

Computing states satisfying particular formula types based on manipulation with OBDDs

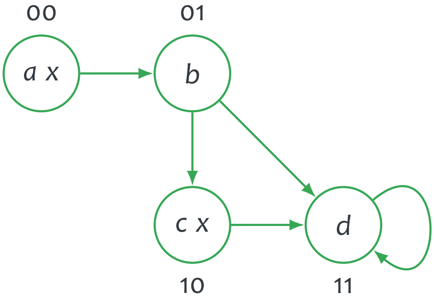
Symbolic CTL Model Checking

Computing OBDD(f) for formula f depends on top-most operand

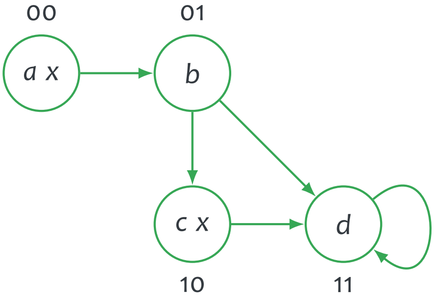
- note that only $\neg$, $\wedge$, $\vee$, EX, EG, and EU are needed, others can be eliminated

- $f \in AP$: return OBDD defined for f
- $f : \neg g, f \wedge g$, or $f \vee g$: use logical operation upon OBDD
  - described in previous lecture
- $f = \mathrm{EX}\, g$: OBDD for $\exists \langle v' \rangle \big(o(\langle v' \rangle) \wedge R(\langle v \rangle, \langle v' \rangle) \big)$
  - $o(\langle v \rangle)$ stands for OBDD representing states satisfying formula g
- $f = \mathrm{E}[f\, \mathrm{U}\, g]$: compute least fixpoint $\mathrm{E}[f\, \mathrm{U}\, g] = \mu Z. \big(g \vee (f \wedge \mathrm{EX}\, Z) \big)$
  - using LfP procedure
- $f = \mathrm{EG}\, f$: compute greatest fixpoint $\mathrm{EG}\, f = \nu Z.(f \wedge \mathrm{EX}\, Z)$
  - using GfP procedure

Example



EXAMPLE



$$\mathrm{AF}\,x = \neg\mathrm{EG}\,(\neg x)$$

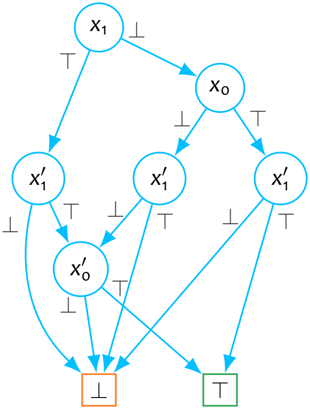
$$\mathrm{EG}\,(\neg x)$$

$$\neg x$$

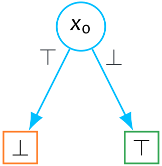
$$x$$

Example

TR:

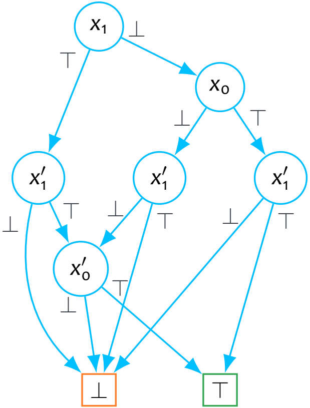


OBDD for states satisfying *x*:

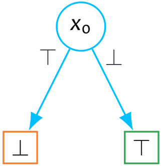


Example

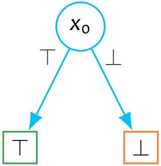
TR:



OBDD for states satisfying *x*:



OBDD for states satisfying ¬*x*:



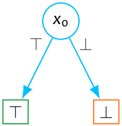
Example

- We have OBDD for states satisfying $\neg x$ and now, we can proceed to EG $(\neg x)$ and compute OBDD for it.
- We compute *greatest fixpoint* of predicate transformer: EG $(\neg x) : \nu Z.(\neg x \wedge \mathrm{EX}\, Z)$.
  - computation starts with trivial OBDD for $\top$ (Z).
  - single step: $Z = \neg x \wedge (\exists x'_0, x'_1 : Z' \wedge TR)$
    - Z' denotes OBDD Z where all variables get primed $(x \rightarrow x')$
  - if Z changes, repeat previous step, otherwise fixpoint reached and computation is over

Example

- We have OBDD for states satisfying $\neg x$ and now, we can proceed to $\mathrm{EG}\,(\neg x)$ and compute OBDD for it.
- We compute *greatest fixpoint* of predicate transformer: $\mathrm{EG}\,(\neg x) : \nu Z.(\neg x \wedge \mathrm{EX}\, Z)$.
  - computation starts with trivial OBDD for $\top$ (Z).
  - single step: $Z = \neg x \wedge (\exists x'_0, x'_1 : Z' \wedge TR)$
    - Z' denotes OBDD Z where all variables get primed $(x \to x')$
  - if Z changes, repeat previous step, otherwise fixpoint reached and computation is over

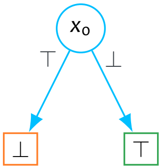
$\mathrm{EG}\,(\neg x) :$

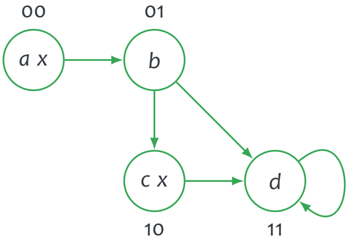


Example

We have OBDD for states satisfying $\mathrm{EG}\,(\neg x)$ and now, we can trivially compute its negation $\neg\mathrm{EG}\,(\neg x) = \mathrm{AF}\,x$.
This corresponds to states 00 and 10 of Kripke structure.

$\neg\mathrm{EG}\,(\neg x) = \mathrm{AF}\,x$:





Conclusion

- During symbolic CTL model checking, all operation performed just upon OBDDs as application of logical operations and fixpoint computations.
- Usually highly efficient comparing to explicit model checking.