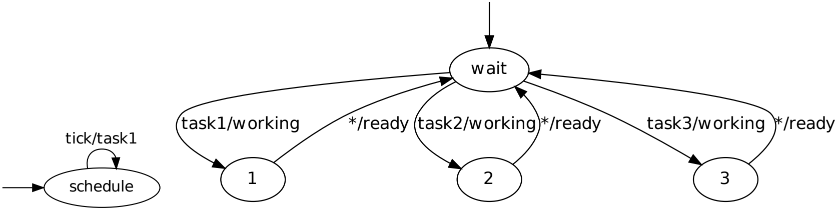
Testing, Bounded Model Checking

Stefan Ratschan

Katedra číslicového návrhu
Fakulta informačních technologií
České vysoké učení technické v Praze

Evropský sociální fond Praha & EU: Investujeme do vaší budoucnosti

Example:



After cascade composition:

$\mathbf{G}$ ok, where $\mathcal{I}(\text{ok}) = \{(u, v) \mid v = \textit{wait} \lor v = 1 \lor v = 2\}$?

For simple automata directly visible from state graph

But: In practice we have hundreds/thousands of interacting components

The set of all states may not even fit in memory!

General Problem

We have:

- System models based on automata and their interaction
- Formal specification of system behavior based on temporal logic

How to check that a certain model fulfills a specification?
(i.e., *model checking*)

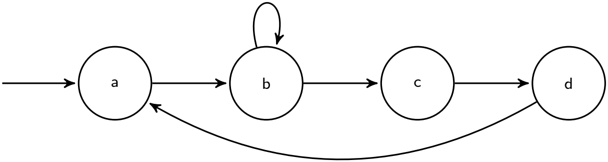
Examples: $\mathbf{G}\,\mathrm{ok}$, $\mathbf{F}\,\mathrm{goal}$, ...

Rest of semester:

- Model checking (3 lectures)
- Modeling time, probabilities, physical behavior etc.

Example:

Given: Transition system:



Question: $\models \mathbf{F}p$, where $\mathcal{I}(p) = \{d\}$?

What does this mean?

We have to check, whether
for all paths π of the transition system, $\pi \models \mathbf{F}p$

If $\not\models \mathbf{F}p$, information for debugging?
counter-example (path π s.t. $\pi \not\models \mathbf{F}p$)

Model Checking Problem

- Given: a transition system and an LTL formula ϕ
- Output:
  - o.k., if $\models \phi$ (i.e., for all paths π, $\pi \models \phi$)
  - a counter-example (a path π s.t. $\pi \not\models \phi$), otherwise.

Let us try to automatize this ...

Model: Transition System

- Set of states S
- Set $S_0 \subseteq S$ of initial states
- Transition Relation $R \subseteq S \times S$

A *path* of a transition system (S, S_0, R) is
an infinite sequence of states $s_0 s_1 s_2 \dots$ s.t.

- $s_0 \in S_0$,
- for all $i \in \{0, 1, \dots\}$, $(s_i, s_{i+1}) \in R$.

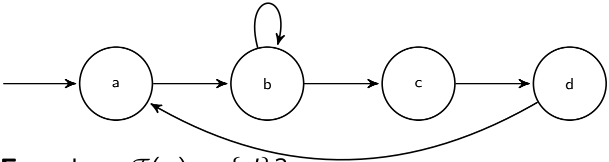
Specification: Temporal Logic

For a path π and LTL formulas p, q,

- $\pi \models p$ where p is a state property
  iff $\pi(0) \models p$
- $\pi \models \mathbf{X}p$ iff $\pi^1 \models p$
- $\pi \models \mathbf{F}p$ iff there is k s.t. $\pi^k \models p$
- $\pi \models \mathbf{G}p$ iff for all k, $\pi^k \models p$
- $\pi \models p\mathbf{U}q$ iff there is i s.t. $\pi^i \models q$ and for all $j < i$, $\pi^j \models p$
- $\pi \models p\mathbf{R}q$ iff for all j, if [for all $i < j$, $\pi^i \not\models p$] then $\pi^j \models q$
- $\pi \models \neg p$ iff not $\pi \models p$
- $\pi \models p \wedge q$ iff $\pi \models p$ and $\pi \models q$
- $\pi \models p \vee q$ iff $\pi \models p$ or $\pi \models q$

$\models p$ iff for all paths π of the transition system, $\pi \models p$

Example:



Question: $\models \mathbf{F}p$, where $\mathcal{I}(p) = \{d\}$?

We need to check, whether for all paths π, $\pi \models \mathbf{F}p$. Let us try:
$(a, b, c, d, a, b, c, d, a, \dots)$?
$(a, b, b, c, d, a, b, c, d, a, \dots)$?
$(a, b, b, b, c, d, a, b, c, d, a, \dots)$?
$(a, b, b, b, b, c, d, a, b, c, d, a, \dots)$?
$(a, b, b, b, b, b, c, d, a, b, c, d, a, \dots)$?

Problem: Even in finite state case,
transition systems can have infinitely many paths of infinite length!

Traditional Solution

How to check correctness of usual programs?

Instead of checking correctness for all inputs,
check correctness for some inputs (*test cases*)

Testing correctness of transition systems (first attempt):

- Given: a transition system and an LTL formula ϕ
- Output:
  - o.k., if for all test cases π, $\pi \models \phi$
  - a path π s.t. $\pi \not\models \phi$, otherwise.

o.k. means that system is correct?

o.k. does not imply correctness any more!

Testing Transition Systems (First Attempt)

We have check:

for all test cases π, $\pi \models \phi$

Testing finitely many paths of infinite length

Remaining problem: How to check $\pi \models \phi$ for one path π?

Example: $(a, b, c, d, a, b, c, d, \dots)$

A *finite path* of length n of a transition system (S, S_0, R) is a finite sequence of states $s_0 s_1 s_2 \dots s_{n-1}$ s.t.

- $s_0 \in S_0$,
- for all $i \in \{0, 1, \dots, n-2\}$, $(s_i, s_{i+1}) \in R$.

Example: (a, b, c, d, a, b, c, d)

Case: Formula **G** ok, Where ok Is a State Property

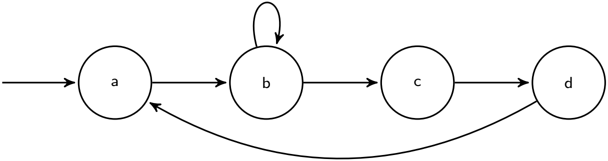
$\pi \models \mathbf{G}\,\mathsf{ok}$ iff for all $k \geq 0$, $\pi(k) \models \mathsf{ok}$

Path has infinite length! so just test against finite prefix:

for a finite path t of length n

$t \models \mathbf{G}\,\mathsf{ok} :\Leftrightarrow$

for all $k \in \{0, \ldots, n-1\}$, $t(k) \models \mathsf{ok}$



Question: $\models \mathbf{G} p$, where $\mathcal{I}(p) = \{a, b\}$?

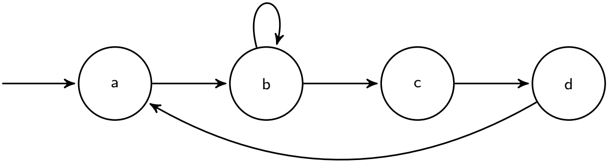
$(a, b, b) \models \mathbf{G} p$, $(a, b, c) \not\models \mathbf{G} p$

Case: Formula $\mathbf{G}\,\mathrm{ok}$, Where ok Is a State Property

for a finite path t of length n

$t \models \mathbf{G}\,\mathrm{ok} :\Leftrightarrow$

for all $k \in \{0, \dots, n-1\}, t(k) \models \mathrm{ok}$



For $\mathcal{I}(p) = \{a, b\}$: $(a, b, b) \models \mathbf{G}p$, $(a, b, c) \not\models \mathbf{G}p$

Necessary condition for correctness:

for all finite paths t of the given transition system

$\models \mathbf{G}\,\mathrm{ok}$ implies $t \models \mathbf{G}\,\mathrm{ok}$

$t \not\models \mathbf{G}\,\mathrm{ok}$ implies $\not\models \mathbf{G}\,\mathrm{ok}$

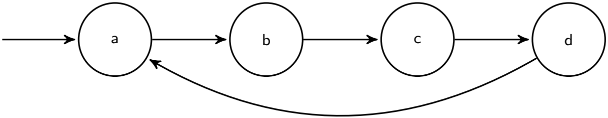
Resulting counter-example: any continuation of t

Case: Formula **F** goal, Where goal Is a State Property

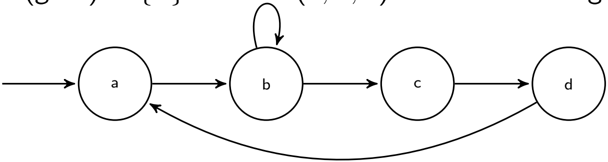
$\pi \models \mathbf{F}\,\text{goal}$ iff there is $k \geq 0$, $\pi(k) \models \text{goal}$

Again we cannot check all k! Does it suffice to test finitely many?

Example:



$\models \mathbf{F}\,\text{goal}$, for $\mathcal{I}(goal) = \{d\}$? ✓ But (a, b, c) does not reach goal!



$\models \mathbf{F}\,\text{goal}$? ↯ Finite path (a, b, b, b, c, d) reaches goal, detects loop!

Finite path (a, b, b) already detects the loop.

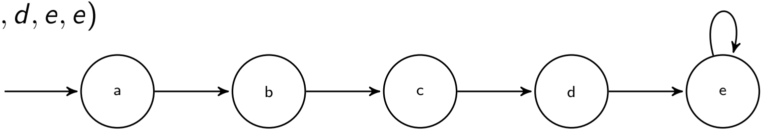
Case: Formula **F** goal: Loop Detection

for a finite path t of length n

$t \models \mathbf{F}\,\text{goal} :\Leftrightarrow$

not there is $l \in \{0, \dots, n-2\}$ s.t. $t(n-1) = t(l)$???

(a, b, c, d, e, e)



$\models \mathbf{F}\,\text{goal}$ where $\mathcal{I}(\text{goal}) = \{d\}$? ✓ (a, b, c, d, e, e) contains a loop!

for a finite path t of length n

$t \models \mathbf{F}\,\text{goal} :\Leftrightarrow$

there is $l \in \{0, \dots, n-2\}$ s.t. $t(n-1) = t(l)$ implies

there is $k \in \{0, \dots, n-1\}$, $t(k) \models \text{goal}$

Case: Formula F goal: Loop Detection

for a finite path t of length n

$t \models \mathbf{F}\,\text{goal} :\Leftrightarrow$

there is $l \in \{0, \dots, n-2\}$ s.t. $t(n-1) = t(l)$ implies

there is $k \in \{0, \dots, n-1\}$, $t(k) \models \text{goal}$

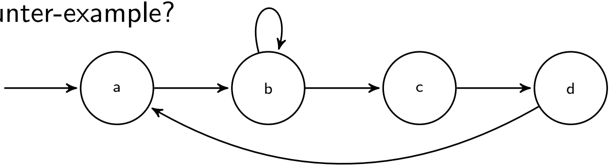
Necessary condition for correctness:

for every finite path t of the given transition system

$\models \mathbf{F}\,\text{goal}$ implies $t \models \mathbf{F}\,\text{goal}$

$t \not\models \mathbf{F}\,\text{goal}$ implies $\not\models \mathbf{F}\,\text{goal}$

Resulting counter-example?



Finite path: (a, b, b), corresponding counter-example $(a, b, b, b, \dots)$

Path leading into and staying in loop

Preliminary Summary

for a finite path t of length n

$t \models \mathbf{G}\,\mathrm{ok} :\Leftrightarrow$
for all $k \in \{0, \dots, n-1\}, t(k) \models \mathrm{ok}$

$t \models \mathbf{F}\,\mathrm{goal} :\Leftrightarrow$
there is $l \in \{0, \dots, n-2\}$ s.t. $t(n-1) = t(l)$ implies
there is $k \in \{0, \dots, n-1\}$, $t(k) \models \mathrm{goal}$

Generalize to full LTL?

Testing of Paths: Bounded Semantics of LTL

For a finite path t of length n and LTL formulas p, q,

- $t \models p$ if t has length 0, and otherwise,
- $t \models p$ where p is a state property iff
$$t(0) \models p.$$
- $t \models \neg p$ where p is a state property iff
$$t(0) \not\models p.$$
- $t \models \mathbf{X}p$ iff $t^1 \models p$
- $t \models \mathbf{F}p$ iff

  there is $l \in \{0, \ldots, n-2\}$ s.t. $t(l) = t(n-1)$

  implies

  there is $k \in \{0, \ldots, n-1\}$ s.t. $t^k \models p$
- $t \models \mathbf{G}p$ iff for all $k \in \{0, \ldots, n-1\}$, $t^k \models p$
- $t \models p \wedge q$ iff $t \models p$ and $t \models q$
- $t \models p \vee q$ iff $t \models p$ or $t \models q$

So: negation only allowed before state property,
if not, push down (e.g., from $\neg\mathbf{F}\ \mathbf{X}p$ to $\mathbf{G}\ \mathbf{X}\neg p$)

Necessary Condition for Correctness

For every *LTL* formula p,

for every finite path t of the given transition system

$\models p$ implies $t \models p$

$t \not\models p$ implies $\not\models p$

Example: $t \models \mathbf{GF}p$

$(a, b, c, d, e, c, d) \models \mathbf{GF}p$, where $\mathcal{I}(p) = \{b\}$

for all $k \in \{0, \dots, 6\}$, $(a, b, c, d, e, c, d)^k \models \mathbf{F}p$

$(a, b, c, d, e, c, d) \models \mathbf{F}p$
$(b, c, d, e, c, d) \models \mathbf{F}p$
$(c, d, e, c, d) \models \mathbf{F}p$

there is $l \in \{0, \dots, n-2\}$ s.t. $t(l) = t(n-1)$ implies
there is $k \in \{0, \dots, n-1\}$ s.t. $t^k \models p$

Counter-example: $(a, b, c, d, e, c, d, e, c, d, e, \dots)$

Example: $t \models \mathbf{FG}p$

$(a, b, c, d, e) \models \mathbf{FG}p$, where $\mathcal{I}(p) = \{e\}$

there is $l \in \{0, \dots, 3\}$ s.t. $t(l) = t(4)$ then
there is $k \in \{0, \dots, 4\}$ s.t. $(a, b, c, d, e)^k \models \mathbf{G}p$

$(a, b, e, d, e) \models \mathbf{FG}p$, where $\mathcal{I}(p) = \{e\}$

Reason: $(a, b, e, d, e)^4 = (e)$ and $(e) \models \mathbf{G}p$

But: $(a, b, e, d, e, d, e, d, e, \dots) \not\models \mathbf{FG}p$!

$(a, b, e, d, e, d) \not\models \mathbf{FG}p$!

Literature: stronger, but more complicated tests
[Latvala et al., 2004, Biere et al., 2009]

Testing Transition Systems (Definitive Version)

Instead of checking correctness for all paths,
check correctness for some finite paths (*test cases*).

$\mathit{Test}(\phi, T)$:⇔ for all finite paths $t \in T$, $t \models \phi$

Task:

- Given:
  - a transition system,
  - an LTL formula ϕ,
  - and a set of finite paths T
- Output:
  - o.k., if $\mathit{Test}(\phi, T)$
  - a path π s.t. $\pi \not\models \phi$, otherwise.

Simulation: generation of finite paths.

Again we have:

$\models \phi$ implies $\mathit{Test}(\phi, T)$

$\neg \mathit{Test}(\phi, T)$ implies $\not\models \phi$

Which Cases to Test?

Transition systems can have infinitely many paths of infinite length.

Systematic methods for choosing finite paths for testing:
black-box testing, white-box testing, coverage criteria, etc.

Field on its own, see MI-TSP (testing and reliability), ...

Problems of testing:

- Can easily miss bugs
- Not reliable enough for many safety critical systems

Costs of Intel FDIV Bug (1994): 1/2 billion dollars

Airbus: Testing almost more expensive than development itself

Because of this industry has more and more interest in methods for proving correctness (*model checking*).

Bounded Model Checking

Original problem:

$\models \phi \quad \Leftrightarrow \quad$ for all paths π, $\quad \pi \models \phi$

infinitely many paths of infinite length

Simpler problem:

$\mathit{Test}(\phi, T) \Leftrightarrow \quad$ for all finite paths $t \in T$, $\quad t \models \phi$

Observation: only finitely many paths of finite length

$\mathit{BMC}(\phi, n) :\Leftrightarrow \quad$ for all finite paths t of length n, $\quad t \models \phi$

$\Leftrightarrow \quad \mathit{Test}(\phi, \{t \mid \text{is a finite path of length } n\})$

BMC(**G** ok): Example:

Instead of $\models \mathbf{G}\,\mathrm{ok}$:

for all paths π,

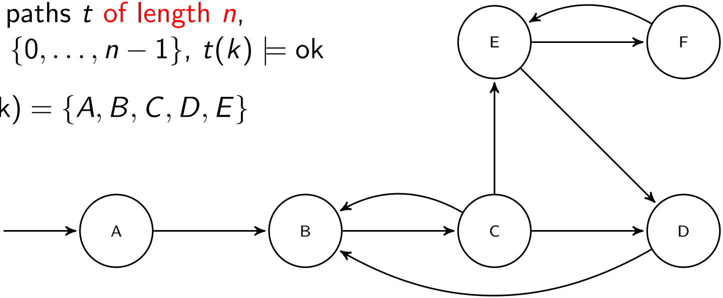
for all $k \geq 0$, $\pi(k) \models \mathrm{ok}$

we have: $BMC(\mathbf{G}\,\mathrm{ok}, n)$

for all finite paths t of length n,

for all $k \in \{0, \ldots, n-1\}$, $t(k) \models \mathrm{ok}$

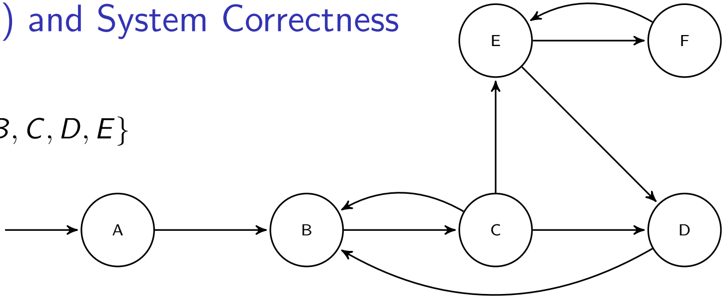
Example: $\mathcal{I}(\mathrm{ok}) = \{A, B, C, D, E\}$



$BMC(\mathbf{G}\,\mathrm{ok}, n)$: holds for $n \leq 4$, does not hold for $n > 4$.

BMC($\mathbf{G}$ ok) and System Correctness

$\mathcal{I}(\text{ok}) = \{A, B, C, D, E\}$



We already know:

- $\models \mathbf{G}\,\text{ok}$ implies BMC($\mathbf{G}\,\text{ok}, n$), i.e.,
- $\neg$BMC($\mathbf{G}\,\text{ok}, n$) implies $\not\models \mathbf{G}\,\text{ok}$

But: BMC($\mathbf{G}\,\text{ok}, n$) does not imply $\models \mathbf{G}\,\text{ok}$.

BMC($\mathbf{G}\,\text{ok}, n$) only ensures correctness within n steps, i.e.,
non-existence of a counter-example of length n

Special Case

BMC($\mathbf{G}\,\mathrm{ok}, 0$)?:

for all finite paths t of length 0,
for all $k \in \{0, \ldots, -1\}$, $t(k) \models \mathrm{ok}$

for all finite paths t of length 0,
for all $k \in \emptyset$, $t(k) \models \mathrm{ok}$

for all finite paths t of length 0,
for all k . $k \in \emptyset \Rightarrow t(k) \models \mathrm{ok}$

for all finite paths t of length 0,
for all k . $\bot \Rightarrow t(k) \models \mathrm{ok}$

for all finite paths t of length 0,
for all k . $\neg\bot \vee t(k) \models \mathrm{ok}$

...

$BMC(\mathbf{F}\,\mathrm{goal})$

Instead of $\models \mathbf{F}\,\mathrm{goal}$ iff

for all paths π there is $k \geq 0$ s.t. $\pi(k) \models \mathrm{goal}$

BMC($\mathbf{F}\,\mathrm{goal}, n$):

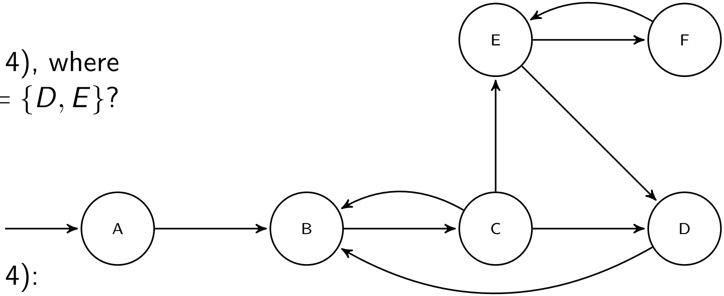
for all finite paths t of length n

if there is $l \in \{0, \dots, n-2\}$ s.t. $t(n-1) = t(l)$

then there is $k \in \{0, \dots, n-1\}$ s.t. $t(k) \models \mathrm{goal}$.

Example

BMC($\mathbf{F}\, \mathtt{goal}, 4$), where
$\mathcal{I}(\mathtt{goal}) = \{D, E\}$?



BMC($\mathbf{F}\, \mathtt{goal}, 4$):

for all finite paths t of length 4
if there is $l \in \{0, \dots, 2\}$ s.t. $t(3) = t(l)$
then there is $k \in \{0, \dots, 3\}$ s.t. $t(k) \models \text{goal}$.

for all $t \in \{(A, B, C, D), (A, B, C, E), (A, B, C, B)\}$
if there is $l \in \{0, \dots, 2\}$ s.t. $t(3) = t(l)$
then there is $k \in \{0, \dots, 3\}$ s.t. $t(k) \in \{D, E\}$.

Will BMC($\mathbf{F}\, \mathtt{goal}, 5$), BMC($\mathbf{F}\, \mathtt{goal}, 6$), ... find the same cycle?

BMC($\mathbf{F}\, \mathtt{goal}, 0$), BMC($\mathbf{F}\, \mathtt{goal}, 1$): similar to the case BMC($\mathbf{G}\, \mathtt{ok}, 0$)

Bounded Model Checking for $\mathbf{F}\,\text{goal}$

$\text{BMC}(\mathbf{F}\,\text{goal}, n)$:
  for all finite paths t of length n
    if there is $l \in \{0, \ldots, n-2\}$ s.t. $t(n-1) = t(l)$
    then there is $k \in \{0, \ldots, n-1\}$ s.t. $t(k) \models \text{goal}$.

Again: $\models \mathbf{F}\,\text{goal}$ implies $\text{BMC}(\mathbf{F}\,\text{goal}, n)$, but not the other way round.

BMC(ϕ, n) and System Correctness

In general: For every n, n' with $n \geq n'$,

- $\models \phi$ implies BMC(ϕ, n) implies BMC(ϕ, n') i.e.,
- $\neg$BMC(ϕ, n') implies $\neg$BMC(ϕ, n) implies $\not\models \phi$.

In other words:
If BMC finds a bug, then
it will also find a bug for higher n and the system really is incorrect.

But: BMC(ϕ, n) does not imply $\models \phi$.

Never?

In finite case, transition relation forms a graph.

For big n, the paths of length n reach all reachable states!

General Observation

- Every finite path that is longer than $|S|$ contains a cycle.
- If there exists a counter-example for $\mathbf{G}\,\mathrm{ok}$ that contains a cycle, then there also exists a counter-example that is shorter.
- Hence:
  If there exists a counter-example for $\mathbf{G}\,\mathrm{ok}$ that is longer than $|S|$, then there also exists a counter-example that is shorter.
- Hence: $\models \mathbf{G}\,\mathrm{ok}$ iff $\mathrm{BMC}(\mathbf{G}\,\mathrm{ok}, |S|)$,

Theorem ([Biere et al., 2003])

for all finite transition system, for all LTL formulas ϕ
there is a bound n s.t.
for all $n' \geq n$, $BMC(\phi, n')$ iff $\models \phi$

Independent of used method.

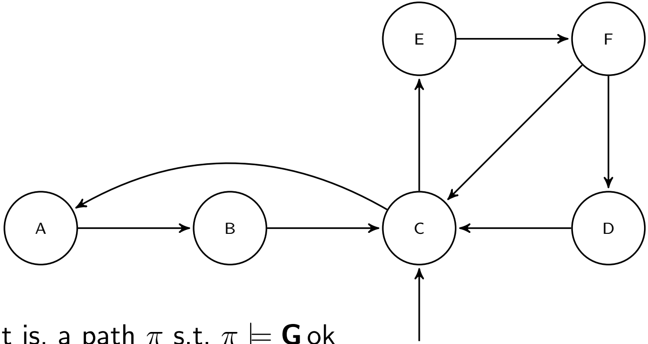
But: Bound may be huge!

Temporal Logic for Planning: Example

Example: Smart European power grid

Here: Non-determinism corresponds to decisions we can take.
Example: Should we switch on coal-fired power station? Yes/no

$\mathbf{G}\,\mathrm{ok}$, $\mathcal{I}(ok) = \{A, B, C, E, F\}$, current state: C



Find: plan, that is, a path π s.t. $\pi \models \mathbf{G}\,\mathrm{ok}$

Temporal Logic for Planning

Given:

- Transition system $(S, \{s_0\}, R)$, where
  s_0 is the current state of the system, and
- LTL formula ϕ,

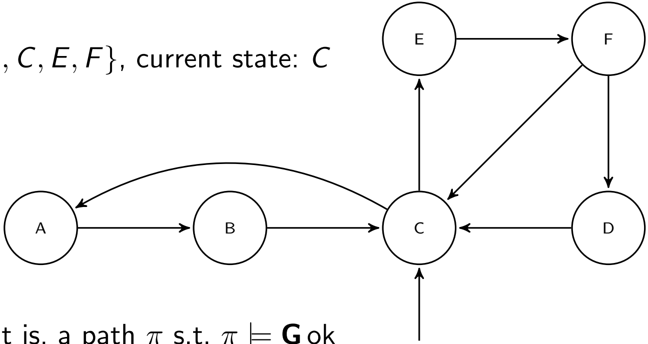
Find: path π, s.t. $\pi \models \phi$
(plan what to do from current state s_0)

Idea: Such a path is a counter-example to $\models \neg\phi$

Find plan by applying a method for checking $\models \neg\phi$

Example: **G** ok

$\mathcal{I}(ok) = \{A, B, C, E, F\}$, current state: C



Find: plan, that is, a path π s.t. $\pi \models \mathbf{G}\,\text{ok}$

Method: find counter-example to $\neg\mathbf{G}\,\text{ok}$

$\neg\mathbf{G}\,\text{ok}$ is equivalent to $\mathbf{F}\neg\text{ok}$

Counter-examples: $(C, A, B, C, \dots)$, $(C, E, F, C, \dots)$

Result: path, that will cycle outside of $\{s \mid s \models \neg\text{ok}\}$, in $\mathcal{I}(\text{ok})$

Example: $\mathbf{F}$goal

Plan: counter-example to $\neg\mathbf{F}\,\text{goal}$

$\neg\mathbf{F}\,\text{goal}$ is equivalent to $\mathbf{G}\neg\text{goal}$

Result: counter-example to $\mathbf{G}\neg\text{goal}$, that is,
path, that will reach $\mathcal{I}(\text{goal})$

Planning vs. Model Checking

Previous lecture:

Check $\models p\,\mathbf{R}\,q$ by searching for a counter-example

path π s.t. $\pi \not\models p\,\mathbf{R}\,q$
path π s.t. $\pi \models \neg[p\,\mathbf{R}\,q]$
path π s.t. $\pi \models \neg p\,\mathbf{U}\,\neg q$

Opposite direction!

Possibilities:

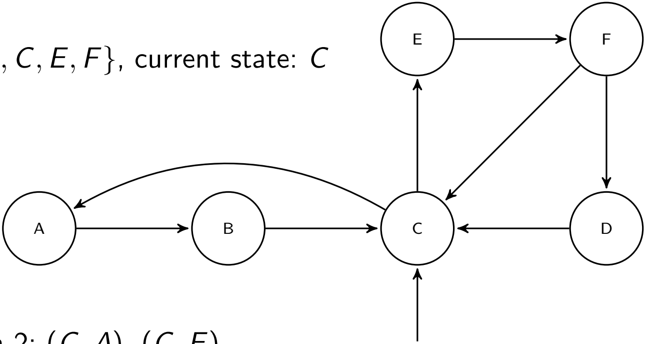
- Model checking problem:
  Check $\models \phi$: yes, or counter-example ($\pi \not\models \phi$)
- Planning problem:
  Find π s.t. $\pi \models \phi$ or "does not exist" (i.e., $\models \neg\phi$)

One is reducible to the other

BMC is weaker!

Example: **G** ok

$\mathcal{I}(ok) = \{A, B, C, E, F\}$, current state: C



paths of length 2: (C, A), (C, E)
$BMC(\mathbf{F}\neg\text{ok}, 2)$, no counter-example, no plan

paths of length 3: (C, A, B), (C, E, F)
$BMC(\mathbf{F}\neg\text{ok}, 3)$, no counter-example, no plan

paths of length 4: (C, A, B, C), (C, E, F, C), (C, E, F, D)
$BMC(\mathbf{F}\neg\text{ok}, 4)$, counter-example, plan (C, A, B, C)

Efficient Checking of BMC(ϕ, n)

Example: 10 states, $n = 10$, $10^{10} = 10000000000$ paths to check!

Next lecture: Methods that are able to handle this.

Farewell demo (BMC needs 1 second to find a bug in the Needham-Schroeder protocol that took humans 17 years to find)

Literature

Armin Biere, Alessandro Cimatti, Edmund M. Clarke, Ofer Strichman, and Yunshan Zhu. Bounded model checking. *Advances in Computers*, 58: 117 – 148, 2003. ISSN 0065-2458. doi: DOI:10.1016/S0065-2458(03)58003-2.

Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, February 2009. ISBN 978-1-58603-929-5.

Timo Latvala, Armin Biere, Keijo Heljanko, and Tommi Junttila. Simple bounded ltl model checking. In *FMCAD*, volume 4, pages 186–200. Springer, 2004.