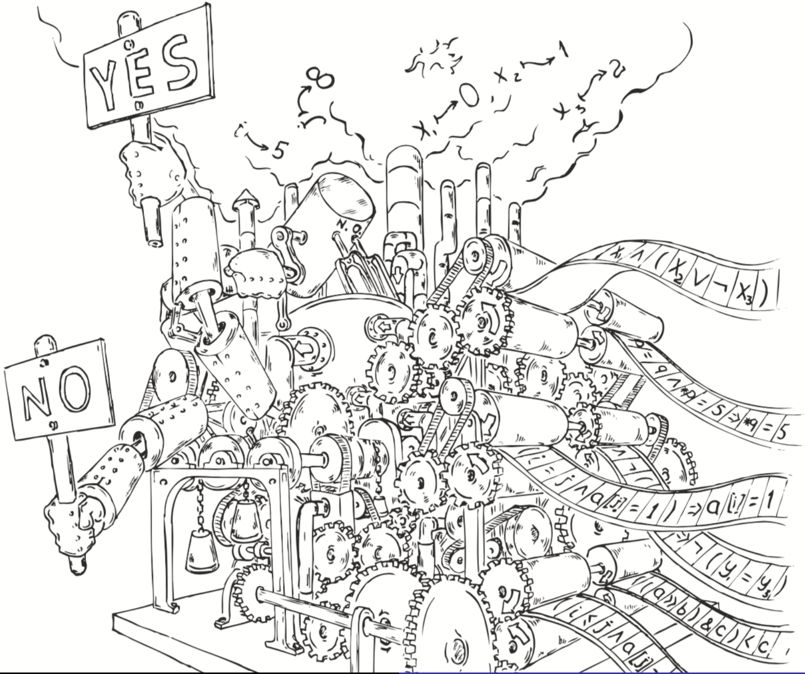

YES
NO

SAT/SMT solvers
13. Bounded Model Checking

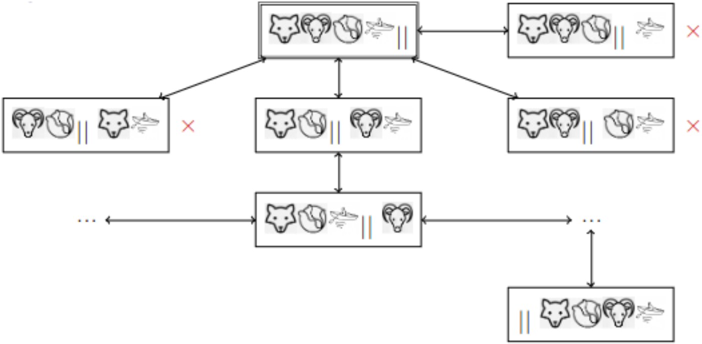
Roman Kholin

Lomonosov Moscow State University

Moscow, 2023

Kripke Structures

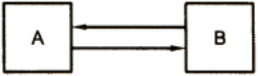
- $K = (S, S_0, L, T)$
- S is a (finite) set of states
- S_0 is a start state
- $L : S \to 2^V$ is a labelling function that maps each state to the set of propositional variables that hold in it
- $T \subseteq S \times S$ is a total transition relation



Big number of states

2 Variables

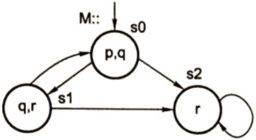
2 Buffers



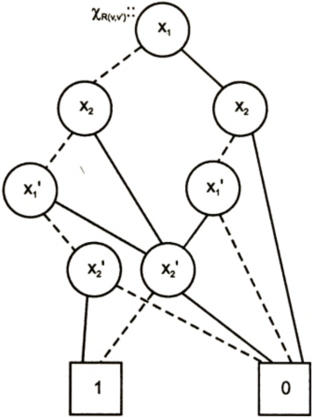
Symbolic Model Checking

- $M = (S, S_0, L, T)$
- $S = \{s_0, s_1, s_2\}$
- $S_0 = \{s_0\}$
- $T = \{(s_0, s_1), (s_0, s_2), (s_1, s_0), (s_1, s_2), (s_2, s_2)\}$
- predicats: $\{p, q, r\}$
- $Sat_p = \{s_0\}, Sat_q = \{s_0, s_1\}, Sat_r = \{s_1, s_2\}$

Symbolic Model Checking



a



x_1	x_2	x_1'	x_2'	$\chi_{R(v,v')}$
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Symbolic Model Checking

$$\chi_{s0} = \neg x_1 \neg x_2$$

$$\chi_{r(v,v')} = \neg x_1 \neg x_2 \left(x_1{}' \oplus x_2{}' \right) \vee \neg x_1 x_2 \neg x_2{}' \vee x_1 \neg x_2 x_1{}' \neg x_2{}'$$

$$\chi_{Sat\ p}\left(v\right) = \neg x_1 \neg x_2$$

$$\chi_{Sat\ q}\left(v\right) = \neg x_1$$

$$\chi_{Sat\ r}\left(v\right) = x_1 \oplus x_2$$

Symbolic Model Checking

- Initial state: $S_0 : \neg l \land \neg r$
- Transition: $T : (l' = (l \neq r)) \land (r' = \neg r)$
- Property: $\neg l \lor \neg r$

Path: $\pi = (s_0, s_1, \dots)$
Suffix: $\pi^i = (s_i, s_1, \dots)$
Temporal operators are the: «next time» operator **X**, the «finally» operator **F**, the «globally» operator **G**

$\pi \models p$	iff	$p \in L(\pi(0))$	$\pi \models \neg p$	iff	$p \notin L(\pi(0))$
$\pi \models g \vee h$	iff	$\pi \models g$ or $\pi \models h$	$\pi \models g \wedge h$	iff	$\pi \models g$ and $\pi \models h$
$\pi \models \mathbf{F}g$	iff	$\exists j \in \mathbb{N}\colon \pi^j \models g$	$\pi \models \mathbf{G}g$	iff	$\forall j \in \mathbb{N}\colon \pi^j \models g$
$\pi \models \mathbf{X}g$	iff	$\pi^1 \models g$			

$\neg \mathbf{F}g =$
$\neg \mathbf{G}g =$
$\neg \mathbf{X}g =$

Temporal properties

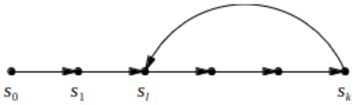
- "Safety"properties
  - "Always x=y"
    ($\mathbf{G}(x = y)$)
  - "Every Send is followed by Ack"
    ($\mathbf{G}(Send \to \mathbf{F}Ack)$)
- "Liveness"properties
  - "Reset can always be reached"
    ($\mathbf{GF}Reset$)
  - "From some point on, always switch_on"
    ($\mathbf{FG}switch\_on$)

Bounded Model Checking

- Based on SAT
- $\exists$ Counterexample of length $k \iff$ Propositional Formula is satisifiable
- BMC for LTL reduced to SAT in poly time
- Advantages:
  - CounterExamples – found fast, minimal length
  - Less space, No manual ordering (vs BDD)
  - The best SAT solvers are capable of handling thousands of state variables
- Disadvantages
  - with the limit k, completeness is naturally sacrificed

Bounded Semantics

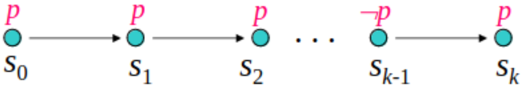
$$\begin{array}{llll}
\pi^i \models p & \text{iff} \quad p \in L(\pi(i)) & \pi^i \models \neg p & \text{iff} \quad p \notin L(\pi(i)) \\
\pi^i \models g \vee h & \text{iff} \quad \pi^i \models g \text{ or } \pi^i \models h & \pi^i \models g \wedge h & \text{iff} \quad \pi^i \models g \text{ and } \pi^i \models h \\
\pi^i \models \mathbf{F}g & \text{iff} \quad \exists j \in \mathbb{N}\colon \pi^{i+j} \models g & \pi^i \models \mathbf{G}g & \text{iff} \quad \forall j \in \mathbb{N}\colon \pi^{i+j} \models g \\
\pi^i \models \mathbf{X}g & \text{iff} \quad \pi^{i+1} \models g & &
\end{array}$$



$$\begin{array}{lll}
\pi^i \models \mathbf{F}g & \text{iff} & \exists j \in \{\min(i,l), \ldots, k\}\colon \pi^j \models g \\
\pi^i \models \mathbf{G}g & \text{iff} & \forall j \in \{\min(i,l), \ldots, k\}\colon \pi^j \models g \\
\pi^i \models \mathbf{X}g & \text{iff} & \begin{cases} \pi^{i+1} \models g & \text{if } i < k \\ \pi^{l} \models g & \text{if } i = k \end{cases}
\end{array}$$

Example

Most safety properties can be reduced to "Always p"where p is propositional



SAT

satSolver.ASSERT(*Init*($\overline{p}$));
satSolver.PUSH(*Err*($\overline{p}$));
for $k \in [0..B]$ **do**
  if *satSolver*.CHECKSAT() = SAT **then**
    return *Небезопасно, контрпример solver*.GETMODEL();
  satSolver.POP();
  if $k < B$ **then**
    satSolver.ASSERT($T(\overline{p}^k, \overline{p}^{k+1})$);
    satSolver.PUSH(*Err*($\overline{p}^{k+1}$));

Encoding

With loop:

$$_l[p]_k^i \equiv p_i \qquad _l[\neg p]_k^i \equiv \neg p_i$$

$$_l[g \vee h]_k^i \equiv {}_l[g]_k^i \vee {}_l[h]_k^i \qquad _l[g \wedge h]_k^i \equiv {}_l[g]_k^i \wedge {}_l[h]_k^i$$

$$_l[\mathbf{F}g]_k^i \equiv \bigvee_{j=\min(l,i)}^{k} {}_l[g]_k^j \qquad _l[\mathbf{G}g]_k^i \equiv \bigwedge_{j=\min(l,i)}^{k} {}_l[g]_k^j$$

$$_l[\mathbf{X}g]_k^i \equiv {}_l[g]_k^j \text{ with } j = i+1 \text{ if } i < k \text{ else } j = l$$

Without loop:

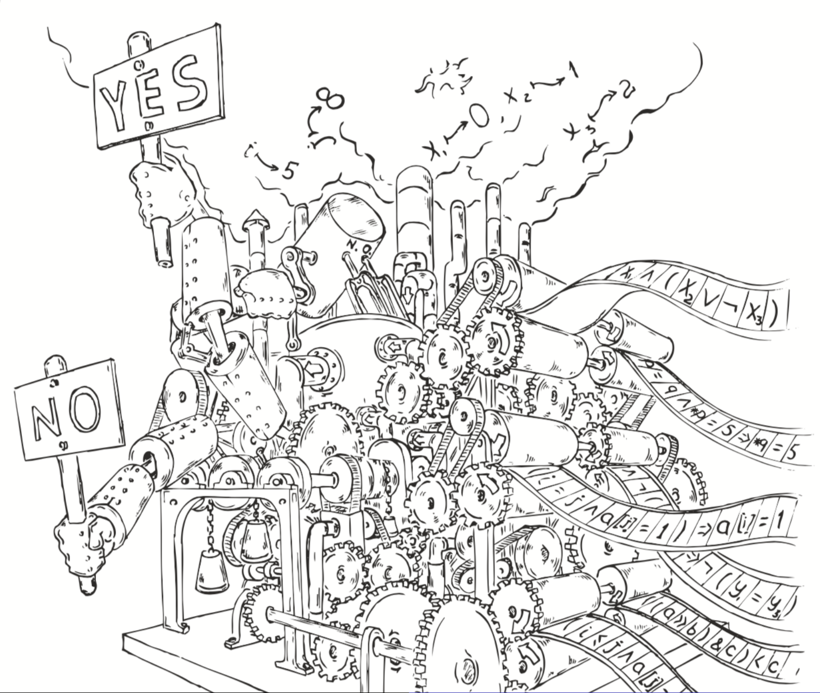
$$[\mathbf{F}g]_k^i \equiv \bigvee_{j=i}^{k} [g]_k^j \qquad [\mathbf{G}g]_k^i \equiv \bot \qquad [\mathbf{X}g]_k^i \equiv \begin{cases} [g]_k^{i+1} & \text{if } i < k \\ \bot & \text{if } j = k \end{cases}$$

Result:

$$[f]_k \quad \equiv \quad [f]_k^0 \vee \bigvee_{l=0}^{k} \lambda_l \wedge {}_l[f]_k^0$$

Determining the Bound

- Theorem: for **G**p properties Completeness Threshold is Diameter - longest "shortest path"from an initial state to any other reachable state
- Theorem: for **F**p properties Completeness Threshold is Recurrence Diameter - longest loop-free path
- Open Problem: The value of Completeness Threshold for general Linear Temporal Logic properties is unknown


YES
NO