SMT-based Flat Model-Checking for LTL with Counting

Anton Pirogov

26.10.2017

Contents

Preliminaries (counter systems, flatness, path schemas)

The logic lcLTL

Quantifier-free Presburger arithmetic

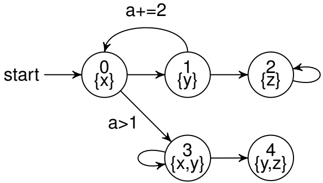
A tour through the translation

Counter systems

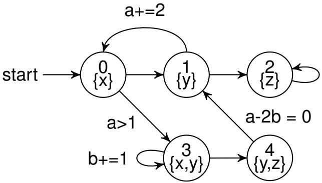
Kripke structure = Directed graph with labelled nodes and initial state

Counter system = Kripke structure + counters + edge guards and updates

Flat:

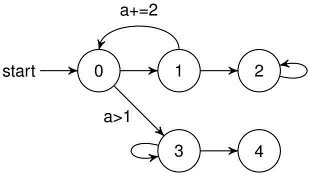


Non-flat:

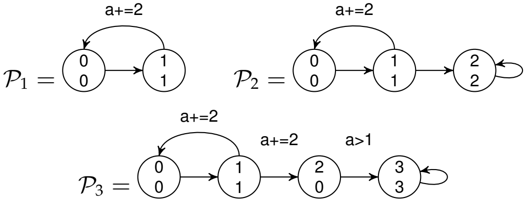


Path schemas

Path schema = linear state sequence with non-overlapping (flat!) backloops

$\mathcal{S} =$ 

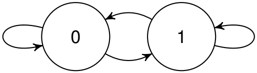
A finite set of path schemas fully describes all runs in the flat system $\mathcal{S}$:



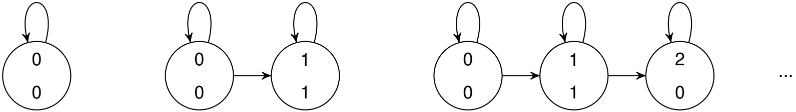
Path schemas

No finite set of path schemas can describe runs in arbitrary counter systems:

$\mathcal{S} =$



$\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \ldots =$



lcLTL **syntax**

$$\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\left[\sum_{i=0}^{n} k_i\varphi \oplus k\right]\varphi$$

where $p \in AP, k_i, k \in \mathbb{Z}, n \in \mathbb{N}, \oplus \in \{<, \geq\}$.

Additional expressions as syntactic sugar:

$$\text{true} := p \vee \neg p \qquad \text{false} := p \wedge \neg p$$

$$\varphi\mathbf{U}\psi := \varphi\mathbf{U}[1 \cdot \text{true} \geq 0]\psi$$

$$\mathbf{F}\varphi := \text{true}\mathbf{U}\varphi \qquad \mathbf{G}\varphi := \neg\mathbf{F}\neg\varphi$$

lcLTL semantics

$$(w,i) \models p \quad :\Leftrightarrow \quad p \in \lambda(w(i))$$

$$(w,i) \models \neg\varphi \quad :\Leftrightarrow \quad (w,i) \not\models \varphi$$

$$(w,i) \models \varphi \wedge \psi \quad :\Leftrightarrow \quad (w,i) \models \varphi \text{ and } (w,i) \models \psi$$

$$(w,i) \models \varphi \vee \psi \quad :\Leftrightarrow \quad (w,i) \models \varphi \text{ or } (w,i) \models \psi$$

$$(w,i) \models \mathbf{X}\varphi \quad :\Leftrightarrow \quad (w,i+1) \models \varphi$$

$$(w,i) \models \varphi \mathbf{U} \left[\sum_{j=0}^{n} k_j \eta_j \oplus k \right] \psi \quad :\Leftrightarrow \quad \exists_{l \geq i} : (w,l) \models \psi \text{ and } \forall_{i \leq m < l} : (w,m) \models \varphi$$

$$\text{and } \sum_{j=1}^{n} k_j \cdot \#^{w}_{[i,l-1]}(\eta_j) \oplus k$$

$$\oplus \in \{<, \leq, \geq, >\}$$

lcLTL semantics

$$(w,i) \models p \quad :\Leftrightarrow \quad p \in \lambda(w(i))$$

$$(w,i) \models \neg\varphi \quad :\Leftrightarrow \quad (w,i) \not\models \varphi$$

$$(w,i) \models \varphi \wedge \psi \quad :\Leftrightarrow \quad (w,i) \models \varphi \text{ and } (w,i) \models \psi$$

$$(w,i) \models \varphi \vee \psi \quad :\Leftrightarrow \quad (w,i) \models \varphi \text{ or } (w,i) \models \psi$$

$$(w,i) \models \mathbf{X}\varphi \quad :\Leftrightarrow \quad (w,i+1) \models \varphi$$

$$(w,i) \models \varphi \mathbf{U} \left[\sum_{j=0}^{n} k_j \eta_j \oplus k \right] \psi \quad :\Leftrightarrow \quad \exists_{l \geq i} : (w,l) \models \psi \text{ and } \forall_{i \leq m < l} : (w,m) \models \varphi$$

$$\text{and } \sum_{j=1}^{n} k_j \cdot \#^{w}_{[i,l-1]}(\eta_j) \oplus k$$

$$\oplus \in \{<, \leq, \geq, >\}$$

$\mathbf{U}[\ldots]$ examples

$[\ 0 + 2 + 0 + 0\ ] = 2 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi$ $\quad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\varphi,\eta\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\psi\}\ \ldots$ $\quad\Rightarrow$ satisfied

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi$ $\quad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\eta,\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\emptyset\rightarrowtail\{\psi\}\ \ldots$ $\quad\Rightarrow \varphi\mathbf{U}\psi$ violated

$[\ 0 + 0 + 0 + 0\ ] = 0 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi$ $\quad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\psi\}\ \ldots$ $\quad\Rightarrow$ violated

$[\ ] = 0 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi$ $\quad w = \rightarrowtail\{\eta,\psi\}\ \ldots$ $\quad\Rightarrow$ violated

$[\ ] = 0 \geq 0$

$\Phi = \varphi\mathbf{U}[2\eta \geq 0]\psi$ $\quad w = \rightarrowtail\{\eta,\psi\}\ \ldots$ $\quad\Rightarrow$ satisfied

$\mathbf{U}[\ldots]$ examples

$[\ 0 + 2 + 0 + 0\ ] = 2 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\varphi,\eta\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\psi\}\ \ldots$ $\Rightarrow$ satisfied

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\eta,\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\emptyset\rightarrowtail\{\psi\}\ \ldots$ $\Rightarrow \varphi\mathbf{U}\psi$ violated

$[\ 0 + 0 + 0 + 0\ ] = 0 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\psi\}\ \ldots$ $\Rightarrow$ violated

$[\,] = 0 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \rightarrowtail\{\eta,\psi\}\ \ldots$ $\Rightarrow$ violated

$[\,] = 0 \geq 0$

$\Phi = \varphi\mathbf{U}[2\eta \geq 0]\psi \qquad w = \rightarrowtail\{\eta,\psi\}\ \ldots$ $\Rightarrow$ satisfied

$\mathbf{U}[\ldots]$ examples

$[\ 0 + 2 + 0 + 0\ ] = 2 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \succ\{\varphi\}\succ\{\varphi,\eta\}\succ\{\varphi\}\succ\{\varphi\}\succ\{\psi\}\ \ldots$ $\Rightarrow$ satisfied

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \succ\{\varphi\}\succ\{\eta,\varphi\}\succ\{\varphi\}\succ\emptyset\succ\{\psi\}\ \ldots$ $\Rightarrow$ $\varphi\mathbf{U}\psi$ violated

$[\ 0 + 0 + 0 + 0\ ] = 0 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \succ\{\varphi\}\succ\{\varphi\}\succ\{\varphi\}\succ\{\varphi\}\succ\{\psi\}\ \ldots$ $\Rightarrow$ violated

$[\,] = 0 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \succ\{\eta,\psi\}\ \ldots$ $\Rightarrow$ violated

$[\,] = 0 \geq 0$

$\Phi = \varphi\mathbf{U}[2\eta \geq 0]\psi \qquad w = \succ\{\eta,\psi\}\ \ldots$ $\Rightarrow$ satisfied

$\mathbf{U}[\ldots]$ examples

$[\ 0 + 2 + 0 + 0\ ] = 2 > 0$

$\Phi = \varphi \mathbf{U}[2\eta > 0]\psi \qquad w = \succ\{\varphi\}\succ\{\varphi,\eta\}\succ\{\varphi\}\succ\{\varphi\}\succ\{\psi\}\ \ldots$ $\Rightarrow$ satisfied

$\Phi = \varphi \mathbf{U}[2\eta > 0]\psi \qquad w = \succ\{\varphi\}\succ\{\eta,\varphi\}\succ\{\varphi\}\succ\emptyset\succ\{\psi\}\ \ldots$ $\Rightarrow$ $\varphi \mathbf{U} \psi$ violated

$[\ 0 + 0 + 0 + 0\ ] = 0 > 0$

$\Phi = \varphi \mathbf{U}[2\eta > 0]\psi \qquad w = \succ\{\varphi\}\succ\{\varphi\}\succ\{\varphi\}\succ\{\varphi\}\succ\{\psi\}\ \ldots$ $\Rightarrow$ violated

$[\,] = 0 > 0$

$\Phi = \varphi \mathbf{U}[2\eta > 0]\psi \qquad w = \succ\{\eta,\psi\}\ \ldots$ $\Rightarrow$ violated

$[\,] = 0 \geq 0$

$\Phi = \varphi \mathbf{U}[2\eta \geq 0]\psi \qquad w = \succ\{\eta,\psi\}\ \ldots$ $\Rightarrow$ satisfied

$\mathbf{U}[\ldots]$ examples

$[\ 0 + 2 + 0 + 0\ ] = 2 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\varphi,\eta\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\psi\}\ \ldots$ ⇒ satisfied

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\eta,\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\emptyset\rightarrowtail\{\psi\}\ \ldots$ ⇒ $\varphi\mathbf{U}\psi$ violated

$[\ 0 + 0 + 0 + 0\ ] = 0 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\varphi\}\rightarrowtail\{\psi\}\ \ldots$ ⇒ violated

$[\,] = 0 > 0$

$\Phi = \varphi\mathbf{U}[2\eta > 0]\psi \qquad w = \rightarrowtail\{\eta,\psi\}\ \ldots$ ⇒ violated

$[\,] = 0 \geq 0$

$\Phi = \varphi\mathbf{U}[2\eta \geq 0]\psi \qquad w = \rightarrowtail\{\eta,\psi\}\ \ldots$ ⇒ satisfied

Model-checking

Definition (Model-checking problems for lcLTL)

Let $\mathcal{S}$ be a counter system and φ an lcLTL formula. We say $\mathcal{S}$ is a *universal model* of φ and write $\mathcal{S} \models \varphi$, iff for all runs w in $\mathcal{S}$ we have $w \models \varphi$. We say $\mathcal{S}$ is an *existential model* of φ and write $\mathcal{S} \models_\exists \varphi$, iff there exists a run w in $\mathcal{S}$ such that $w \models \varphi$.

If $\mathcal{S}$ is flat: $\mathcal{S} \not\models \varphi \quad \Leftrightarrow \quad \mathcal{S} \models_\exists \neg\varphi \quad \Leftrightarrow$

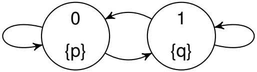
$$\exists n \in \mathbb{N}, \text{path schema } \mathcal{P} \in \mathcal{P}(\mathcal{S}), \text{run } w \text{ in } \mathcal{P} \text{ with } |S_\mathcal{P}| \leq n \ \wedge \ w \models \neg\varphi$$

Flat model-checking ≈ bounded model-checking on path schemas

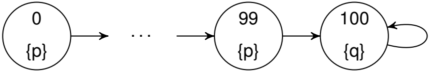
(which are in general flat underapproximations!)

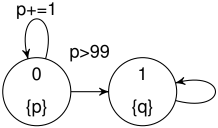
parameter specifies path schema size instead of run prefix length

Why flat model-checking?

$\mathcal{S} =$ 

$\Phi = p\mathbf{U}[p > 100]q$

Bounded MC: $\mathcal{R} =$ 

Flat MC: $\mathcal{P} =$ 

Flat MC allows compact encoding of witnessing runs

for formulas that require counter updates!

- **Goal:** Tool that performs flat model-checking
- **Strategy:**
- **1.** $\overline{\mathrm{MC}}$ problem $\rightarrow$ SAT problem of *Presburger arithmetic*
- **2.** use generic solver to search for a solution
- SAT of quantifier-free(!) PA is in **NP**
- can often be solved quite well by modern SMT-solvers!

Quantifier-free Presburger arithmetic

A minimalistic syntax:

$$\varphi \quad := \quad \tau \leq \tau \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi$$

$$\tau \quad := \quad c \mid c \cdot x \mid \tau + \tau$$

We can write formulas like: $3 \cdot x + 4 \cdot y \leq -7 \cdot z \vee \neg(x \leq 10)$

We **cannot** multiply variables: $x \cdot y \leq 7$

We can also write: $x_2 \in \{3, 5\} \wedge \forall_{i \in [0,1]} : x_i = 0 \vee x_{i+1} > 1$

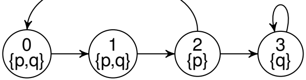
Which can be expanded to:

$$((x_2 \leq 3 \wedge 3 \leq x_2) \vee (x_2 \leq 5 \wedge 5 \leq x_2))$$
$$\wedge\ ((x_0 \leq 0 \wedge 0 \leq x_0) \vee 2 \leq x_1)$$
$$\wedge\ ((x_1 \leq 0 \wedge 0 \leq x_1) \vee 2 \leq x_2)$$

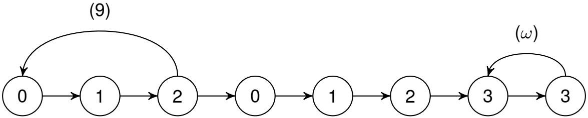
Overview of the translation

- **Input:** Counter system $\mathcal{S}$, lcLTL formula Φ, $n \in \mathbb{N}$
- Encoding is a matrix of variables
- Encode constraints (run + path schema)
- Enforce valid state sequences and edges
- Label each position with sat. subformulas Φ
- **Output:** A witnessing run, if found

First glimpse of the encoding I

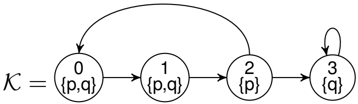
$\mathcal{K} =$ 

$L = \{1, 2, 3\}$ $\Phi = \mathbf{X}(p \wedge q)$

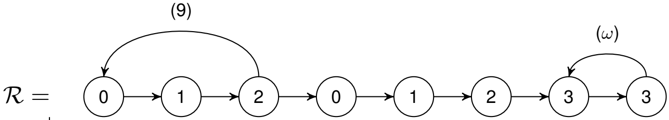
$\mathcal{R} =$ 

i	0		1		2	3	4	5	6		7
id$_i$	0		1		2	0	1	2	3		3
lType$_i$	▷		+		◁	-	-	-	▷		◁
lCount$_i$	9	←	9	←	9	1	1	1	0	←	0
lStart$_i$	0	→	0	→	0	$\bot$	$\bot$	$\bot$	2	→	2
lCtr$_i$	1	→	2	→	3	0	0	0	1	→	2

First glimpse of the encoding II



$L = \{1, 2, 3\}$ $\Phi = \mathbf{X}(p \wedge q)$



i	0	1	2	3	4	5	6	7
lbl_i^p	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$		
lbl_i^q	$\top$	$\top$		$\top$	$\top$		$\top$	$\top$
$\mathsf{lbl}_i^{p \wedge q}$	$\top$	$\top$		$\top$	$\top$			
$\mathsf{lbl}_i^{\mathbf{X}(p \wedge q) = \Phi}$	$\top$		$\top$	$\top$				

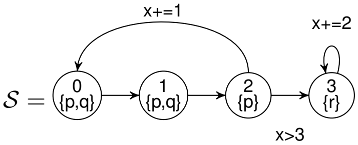
Under the hood: A simple example

Part that encodes the loops in the path schema:

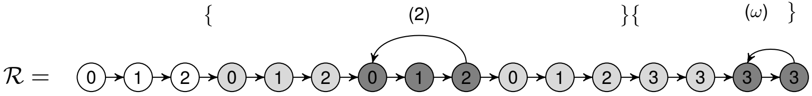
$$
\begin{aligned}
& \mathsf{lCount}_{n-1} = 0 \quad \wedge \\
\forall_{i\in[0,n-1]} :\ & \mathsf{lCount}_i \geq 0 \quad \wedge \quad \big(\mathsf{lType}_i = - \;\Leftrightarrow\; \mathsf{lCount}_i = 1\big) \\
\forall_{i\in[0,n-2]} :\ & \mathsf{lType}_i = \triangleleft \quad \Rightarrow \quad \mathsf{lCount}_i > 1 \\
\forall_{i\in[1,n-1]} :\ & \big(\mathsf{lType}_i \in \{-,\triangleright\} \quad \Rightarrow \quad \mathsf{lType}_{i-1} \in \{-,\triangleleft\}\big) \\
& \wedge \big(\mathsf{lType}_i \in \{+,\triangleleft\} \quad \Rightarrow \quad \mathsf{lType}_{i-1} \in \{+,\triangleright\} \wedge \mathsf{lCount}_i = \mathsf{lCount}_{i-1}\big)
\end{aligned}
$$

(Complete formula: ≈ 3 pages in similar style...)

A second glimpse: Labelling $\mathrm{U}[\ldots]$ and respecting guards

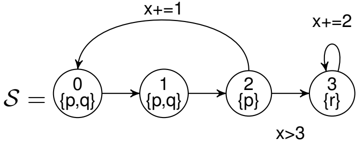


$L = \{1, 2, 3\}$ $\Phi = p\mathbf{U}[q > 8]r$

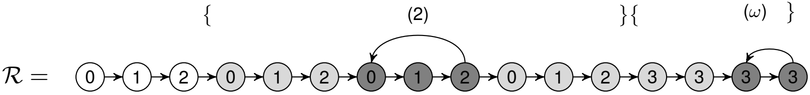


i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
lbls_i^p	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$				
lbls_i^q	$\top$	$\top$		$\top$	$\top$		$\top$	$\top$		$\top$	$\top$					
lbls_i^r													$\top$	$\top$	$\top$	$\top$
$\mathrm{lbls}_i^{p\mathbf{U}r}$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$
lbls_i^{Φ}	$\top$	$\top$														

A second glimpse: Labelling $\mathrm{U}[\ldots]$ and respecting guards



$L = \{1, 2, 3\}$ $\Phi = p\mathbf{U}[q > 8]r$



i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
udCtrs$_i^\Phi$	0	1	2	2	3	4	4	6	8	8	9	10	10	10	10	10
uwCtrs$_i^\Phi$	0	1	2	2	3	4	4	5	6	8	9	10	10	10	10	10
uBest$_i^\Phi$	$\perp^\Phi$	10	10	10	10	10	10	10	10	10	10	10	10	$\perp^\Phi$	$\perp^\Phi$	10
gCtrs$_i^x$	0	0	0	1	1	1	2	2	2	4	4	4	4	6	8	8

ulDelta$_2^\Phi = 0$ allPhi$_\Phi = \perp$ glDelta$_2^{x>3} = 4$

Summary - bag of tricks and upper bounds

- constraint invariance
- forced loop unrollings
- information propagation
- exploit distributivity
- use knowledge about the counter system (*lengths of loops*)
- Result: formula size and number of variables is **linear** in n!

Variables:

$$(8+|\Phi|+3u+c)n+u(\hat{l}+1)+g\hat{l} \overset{\hat{l}\leq n, u\leq|\Phi|}{\in} \mathcal{O}((|S|+|\Phi|)\cdot n)$$

Formula size:

$$\mathcal{O}(n\cdot(|E|+|\Phi|+|L|+u+c+g)+\hat{l}\cdot(u+g)) \overset{\hat{l}\leq n, u\leq|\Phi|}{\subseteq} \mathcal{O}((|S|+|\Phi|+|L|)\cdot n)$$

Summary - bag of tricks and upper bounds

- constraint invariance
- forced loop unrollings
- information propagation
- exploit distributivity
- use knowledge about the counter system (*lengths of loops*)
- Result: formula size and number of variables is **linear** in n!

Variables:

$$(8 + |\Phi| + 3u + c)n + u(\hat{l} + 1) + g\hat{l} \overset{\hat{l} \leq n, u \leq |\Phi|}{\in} \mathcal{O}((|S| + |\Phi|) \cdot n)$$

Formula size:

$$\mathcal{O}(n \cdot (|E| + |\Phi| + |L| + u + c + g) + \hat{l} \cdot (u + g)) \overset{\hat{l} \leq n, u \leq |\Phi|}{\subseteq} \mathcal{O}((|S| + |\Phi| + |L|) \cdot n)$$

Implemented prototype

- `flat-checker` available on GitHub
  (`https://github.com/apirogov/flat-checker`)
- Implemented in Haskell, using Z3 SMT solver
- Command line interface
- Counter system input as file with graph in DOT-format
- Correctly identified all 52 falsifiable formulas in Problem 1 of RERS Challenge 2017!