Decidable Problems for Counter Systems

Day 2

Linear-Time Temporal Logics

Stéphane Demri
demri@lsv.ens-cachan.fr

LSV, ENS Cachan, CNRS, INRIA

ESSLLI 2010, Copenhagen, August 2010
Plan of the talk

• Yesterday’s lecture:
  • Classes of counter systems (Minsky machines, CA, VASS, VAS, relational CA, etc.) and standard decision problems.
  • Presburger arithmetic and semilinear sets.

• Standard LTL

• Logic $LTL^{CS}(PrA)$ for counter systems

• Presburger LTL

• LTL with registers
Specifying existence of runs in temporal logic

- Repeated reachability can be obviously expressed by $G F q_f$.

- Initialized VASS $(\mathcal{V}, (q, \vec{z}))$ is unbounded iff there is a run $(q, \vec{z}) \xrightarrow{*} (q', \vec{y}) \xrightarrow{*} (q', \vec{y}')$ with $\vec{y} \prec \vec{y}'$ for some $q'$.

- In temporal logic lingua:

$$\models E \exists y_1, \ldots, y_n F(\bigwedge_{i=1}^{n} x_i = y_i \land F(\bigwedge_{i=1}^{n} x_i \geq y_i \land \bigvee_{i=1}^{n} x_i > y_i))$$

- Linear-time temporal logics offer genericity and fragments can be easily designed.
Temporal Modalities in Computations
Modalities

- Temporal logics contain modalities with a temporal interpretation.

- A modality is a syntactic object (term) that modifies the relationships between a predicate and a subject.

- In the sentence “Tomorrow, it will rain”, the term “Tomorrow” is a temporal modality.

- The temporal modalities allow one to speak about the sequencing of configurations along a run.

- Standard temporal modalities:
  - $\mathcal{X}$ (“neXt”),
  - $\mathcal{F}$ (“sometimes”),
  - $\mathcal{G}$ (“always”).

- We also use the Boolean connectives: $\neg$ (negation), $\lor$ (disjunction), $\land$ (conjunction) and $\Rightarrow$ (material implication).
“next” and “sometimes”

- \( X\varphi \) states that the next state satisfies \( \varphi \).
  
  \( Xp: \) next-time \( p \)

  ![Diagram](Diagram)

  For example, \( \varphi \lor X\varphi \) states that \( \varphi \) is satisfied now or in the next state.

- \( Fp \) announces that a future state satisfies \( \varphi \) without specifying which state, and \( G\varphi \) that all the future states satisfy \( \varphi \).

  \( Fp: \) sometimes \( p \)

  ![Diagram](Diagram)
“always” and “until”

- The operator $G$ is the dual of $F$: whatever the formula $\varphi$ may be, if $\varphi$ is always satisfied, then it is not true that $\neg \varphi$ will some day be satisfied, and conversely. ($G\varphi$ and $\neg F\neg \varphi$ are equivalent.)

  $Gp$: always $p$

  ![Diagram for $Gp$]

  \[Gp, p \rightarrow p \rightarrow p \rightarrow p \rightarrow p\]

- The $\cup$ operator is richer and more complicated than the combinator $F$. $\varphi_1 \cup \varphi_2$ states that $\varphi_1$ is true until $\varphi_2$ is true.

  $p\cup q$: $p$ until $q$

  ![Diagram for $p\cup q$]

  \[p\cup q, p \rightarrow p \rightarrow p \rightarrow p \rightarrow q\]

  $G(\text{alert} \Rightarrow F\text{ halt})$ can be refined with

  \[G(\text{alert} \Rightarrow (\text{alarm} \cup \text{halt})).\]
The $F$ combinator is a special case of $U$: $F\varphi$ and $true \ U \ \varphi$ are equivalent.

Weak until operator $\bar{w}$.

$\varphi_1 \bar{w} \varphi_2$ expresses “$\varphi_1 \ U \varphi_2$”, but without the inevitable occurrence of $\varphi_2$ and if $\varphi_2$ never occurs, then $\varphi_1$ remains true forever.

$\varphi_1 \bar{w} \varphi_2$ is equivalent to $G \varphi_1 \lor (\varphi_1 \ U \varphi_2)$. 
Plain LTL
LTL syntax

- LTL formulae:

\[ \phi, \psi ::= p \mid \neg \phi \mid \phi \land \psi \mid \phi \lor \psi \mid X\phi \mid \phi U \psi \]

- Atomic formulae are propositional variables.

- Later, control states or arithmetical constraints about counter values are considered at the atomic level.

- LTL models \( \rho \) are \( \omega \)-sequences of propositional valuations of the form \( \rho : \mathbb{N} \rightarrow \mathcal{P}(\text{PROP}) \).
Satisfaction relation (formal semantics)

- $\rho, i \models p \iff p \in \rho(i),$

- $\rho, i \models \neg \varphi \iff \rho, i \not\models \varphi,$

- $\rho, i \models \varphi_1 \land \varphi_2 \iff \rho, i \models \varphi_1 \text{ and } \rho, i \models \varphi_2,$

- $\rho, i \models X \varphi \iff \rho, i + 1 \models \varphi,$

- $\rho, i \models \varphi_1 \cup \varphi_2 \iff \text{there is } j \geq i \text{ such that } \rho, j \models \varphi_2 \text{ and } \rho, k \models \varphi_1 \text{ for all } i \leq k < j.$

$F \varphi \overset{\text{def}}{=} T \cup \varphi, \ G \varphi \overset{\text{def}}{=} \neg F \neg \varphi, \ \varphi \Rightarrow \psi \overset{\text{def}}{=} \neg \varphi \lor \psi,$ etc.
About LTL

- Models(φ): set of models ρ such that ρ, 0 ⊨ φ.

- Models can be viewed as ω-words over the alphabet P(PROP).

- Models(φ) can be effectively represented by a Büchi automaton Aφ.

- Satisfiability and model-checking (see later) are PSPACE-complete problems [Sistla & Clarke, JACM 85].
A Brief Introduction to Büchi Automata
Automata-based approach

- Automata-based approach: to reduce logical problems into automata-based decision problems in order to take advantage of known results from automata theory.

- Another view: a means to transform declarative statements (formulae) into operational devices (automata).

- Standard target problems on automata:
  - the nonemptiness problem checks whether an automaton admits at least one accepting computation,
  - the universality problem checking whether an automaton accepts everything,
  - the inclusion problem checking whether the language accepted by automaton $\mathcal{A}$ is included in the language accepted by automaton $\mathcal{B}$. 
Desirable properties

- The reduction should be conceptually simple.

- The computational complexity of the automata-based target problem should be well-characterized.

- Preferrably, the reduction might allow to obtain the optimal complexity for the source logical problem.

- A pioneering work by R. Büchi [Buchi, 62] in which Büchi automata are shown equivalent to formulae in monadic second-order logics (MSO).
Definition for Büchi automata

- Büchi automaton: finite-state automaton except that $\omega$-words are accepted instead of finite words.

- The set of final states is used as an acceptance condition for $\omega$-words.

- Büchi automaton $A = (\Sigma, Q, Q_0, \delta, F)$:
  
  - $\Sigma$ is a finite alphabet,
  
  - $Q$ is a finite set of states,
  
  - $Q_0 \subseteq Q$ is the set of initial states,
  
  - the transition relation $\delta$ is a subset of $Q \times \Sigma \times Q$,
  
  - $F \subseteq Q$ is a set of final states.
Recognizing infinitely many $a$’s

- Run $\rho = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \ldots$: for $i \geq 0$, $(q_i, a_i, q_{i+1}) \in \delta$.

- Run $\rho$ is successful: $q_0 \in Q_0$ and some state of $F$ is repeated infinitely often in $\rho$.

- Label of $\rho$: $\sigma = a_0 a_1 \cdots \in \Sigma^\omega$.

- $A$ accepts $L(A)$ made of $\omega$-words $\sigma \in \Sigma^\omega$ such that there exists a successful run of $A$ with label $\sigma$. 
Generalized Büchi automata

- Standard generalization: conjunctions of classical Büchi conditions.

- Generalized Büchi automaton (GBA)
  \( \mathcal{A} = (\Sigma, Q, Q_0, \delta, \{F_1, \ldots, F_k\}) \) with \( F_1, \ldots, F_k \subseteq Q \).

- Successful run \( \rho \) of \( \mathcal{A} \): the first state is initial and there is a state in \( F_i \) that is repeated infinitely often in \( \rho \) for \( 1 \leq i \leq n \).

- Let \( \mathcal{A} = (\Sigma, Q, Q_0, \delta, \{F_1, \ldots, F_k\}) \) be a GBA. One can compute (in logarithmic space in the size of \( \mathcal{A} \)) a Büchi automaton \( \mathcal{A}^b = (\Sigma, Q^b, Q_0^b, \delta^b, F^b) \) such that \( L(\mathcal{A}^b) = L(\mathcal{A}) \).
Proof

- \( \mathcal{A} = (\Sigma, Q, Q_0, \delta, \{F_1, \ldots, F_k\}) \).

- \( \mathcal{A}^b \) is made of \( k \) copies of \( \mathcal{A} \) and simulates the generalized accepting condition by passing from one copy to another.

- \( Q^b \overset{\text{def}}{=} Q \times \{1, \ldots, k\} \); \( Q_0^b \overset{\text{def}}{=} Q_0 \times \{1\} \); \( F^b \overset{\text{def}}{=} F_1 \times \{1\} \).

- \( \delta^b((q, i), a) \) is the union of the following sets:
  1. \( \{(q', i) : q \xrightarrow{a} q' \in \delta, q \not\in F_i\} \) (stay in the same copy if no final state in \( F_i \) is reached),
  2. \( \{(q', (i \mod k) + 1) : q \xrightarrow{a} q' \in \delta, q \in F_i\} \) (go to the next copy if a final state in \( F_i \) is reached).

- \( \mathcal{A} \) and \( \mathcal{A}^b \) accepts the same language.
Properties about Büchi automata

• The class of languages accepted by Büchi automata corresponds to the class of \( \omega \)-regular languages.

• The family of \( \omega \)-regular languages is closed by intersection, union and complementation.

• Nonemptiness problem for Büchi automata:
  
  **Input:** a Büchi automaton \( \mathcal{A} \),
  
  **Question:** is \( L(\mathcal{A}) \neq \emptyset \)?

• The nonemptiness problem for Büchi automata is \( \text{NLOGSPACE} \)-complete.
  (and can therefore be solved in polynomial time.)

• By contrast, the universality problem for Büchi automata is \( \text{PSPACE} \)-complete [Sistla & Vardi & Wolper, TCS 87].
Recapitulation

- **LTL formulae:**

  \[ \varphi, \psi ::= p \mid \neg \varphi \mid \varphi \land \psi \mid \varphi \lor \psi \mid X \varphi \mid \varphi U \psi \]

- **Models:** \( \omega \)-sequences over \( \mathcal{P}(\text{PROP}) \).

- **Models(\( \varphi \)):** the set of sequences \( \rho \) in \( \mathcal{P}(\text{PROP})^\omega \) such that \( \rho, 0 \models \varphi \).

- Nonemptiness for GBA can be reduced (in logarithmic space) to nonemptiness for BA.

- The nonemptiness problem for Büchi automata can be solved in polynomial-time.
From LTL Formulae to Automata
LTL captured by Büchi automata

- $L(A) = \text{Models}(p \cup p')$.

```
{p, p'}, \{p\}
```

- There is a Büchi automaton $A_\varphi$ s.t.
  1. $L(A_\varphi) = \text{Models}(\varphi)$; $|A_\varphi|$ is in $2^{O(|\varphi|)}$,
  2. $A_\varphi$ can be effectively computed in polynomial space in $|\varphi|$.

[Vardi & Wolper, IC 94]

- Below, we briefly recall how $A_\varphi$ is defined from $\varphi$. 
Closure set

- Closure set contains all the formulae we need to consider to check satisfiability.

- Closure \( \text{cl}(\varphi) \) of \( \varphi \): smallest set
  - containing the subformulae of \( \varphi \),
  - closed under negation (we identify \( \neg\neg\psi \) with \( \psi \)),
  - if \( \chi_1 \cup \chi_2 \in \text{cl}(\varphi) \), then \( x(\chi_1 \cup \chi_2) \in \text{cl}(\varphi) \).

- Cardinal of \( \text{cl}(\varphi) \) is linear in the size of \( \varphi \).
Atom

- An atom is nothing but a maximally consistent subset of \( \text{cl}(\varphi) \).

- Atom \( X \): subset of \( \text{cl}(\varphi) \) s.t.

  1. for all formulae \( \psi \) in \( \text{cl}(\varphi) \), \( \psi \in X \) iff \( \neg \psi \notin X \),

  2. \( \psi_1 \land \psi_2 \in X \) iff \( \psi_1, \psi_2 \in X \),

  3. \( \psi_1 \lor \psi_2 \in X \) iff \( \psi_1 \in X \) or \( \psi_2 \in X \),
One-step consistent pairs

- \((Y, Y')\) is one-step consistent:
  - if \(\psi_1 \cup \psi_2 \in Y\), then \(\psi_2 \in Y\) or \((\psi_1 \in Y \text{ and } \psi_1 \cup \psi_2 \in Y')\).
    \((\psi_1 \cup \psi_2\) is equivalent to \(\psi_2 \lor (\psi_1 \land X\psi_1 \cup \psi_2)\).
  - for \(X\psi \in \text{cl}(\varphi)\), \(X\psi \in Y\) iff \(\psi \in Y'\).

- The states of \(A_\varphi\) are atoms and the transition relation contains one-step consistent pairs.

- Each state \(X\) of \(A_\varphi\) is a set of formulae that are intended to be satisfied from the current position.

- Either this satisfaction can be checked locally or the transition relation of \(A_\varphi\) allows us to propagate the constraints.
Defining $A_\varphi$

- GBA $A_\varphi = (\Sigma, Q, Q_0, \delta, F_1, \ldots, F_\alpha)$:
  - $\Sigma = \mathcal{P}\{p_1, \ldots, p_N\}$,
  - $Q$ is the set of atoms,
  - $Q_0$ is the subset of atoms containing $\varphi$,
  - $X \xrightarrow{a} Y \in \delta$ iff $a = \{p_1, \ldots, p_N\} \cap X$ and $(X, Y)$ is one-step consistent,
  - for $\psi_1 \cup \psi_2 \in \text{cl}(\varphi)$, there is exactly one set $F_i$ such that $F_i = \{X \in Q : \text{either } \psi_1 \cup \psi_2 \notin X \text{ or } \psi_2 \in X\}$.

- Accepting conditions $F_1, \ldots, F_\alpha$ guarantee that the search for witnesses is not delayed forever.
Main properties for $\mathbb{A}_\varphi$

- $L(\mathbb{A}_\varphi) = \text{Models}(\varphi)$.

- Given a model $\rho : \mathbb{N} \rightarrow \mathcal{P}(\{p_1, \ldots, p_N\}) \in \text{Models}(\varphi)$, there is a unique accepting run $X_0 \xrightarrow{\rho(0)} X_1 \xrightarrow{\rho(1)} X_2 \xrightarrow{\rho(2)} \cdots$ in $\mathbb{A}_\varphi$ s.t. (⋆) for $i \geq 0$ and $\psi \in \text{cl}(\varphi)$, we have
  $$\psi \in X_i \iff \rho, i \models \psi.$$  

- Conversely, for each accepting run $X_0 \xrightarrow{a_0} X_1 \xrightarrow{a_1} X_2 \xrightarrow{a_2} \cdots$ in $\mathbb{A}_\varphi$, the model $\rho$ defined by $\rho(i) = a_i$ for $i \geq 0$ satisfies (⋆).

- In the sequel, we write $\mathbb{A}_\varphi$ to denote the Büchi automaton recognizing $\text{Models}(\varphi)$. 

Full Presburger LTL for Counter Systems
New ingredients

- Models of the form \((q_0, \vec{x}_0), (q_1, \vec{x}_1), \ldots\).

- Control states \(q\) as atomic formulae.

- Arithmetical constraints about counter values \((x_1 > x_2)\).

- Comparing values at successive positions, e.g. \(x_1 > xx_2\).

- First-order quantification over counter values, e.g.
  \[\exists y \ G(x_1 \leq y) \approx \text{“Along the run, counter 1 is bounded.”}\]
LTL\textsuperscript{CS}(PrA) syntax

- Queen logic LTL\textsuperscript{CS}(PrA) (fragments are defined from it).
- Giving up the standard abstraction: propositional variables understood as properties about the current configuration.

- $\text{VAR}^p = \{y_1, y_2, \ldots\}$: set of integer variables.
- $\text{VAR} = \{x_1, x_2, \ldots\}$: set of counter variables.
- $\mathcal{Q} = \{q_1, q_2, \ldots\}$: set of control state symbols.

- LTL\textsuperscript{CS}(PrA) formulae:
  \[
  \varphi ::= \psi \mid q \mid \varphi \land \varphi \mid \neg \varphi \mid X \varphi \mid \varphi \vee \varphi \mid \exists y \varphi
  \]

  - $\psi$ is a Presburger formula with free variables included in $\text{VAR}^p \cup \text{VAR}$,
  - $q \in \mathcal{Q}$. 

Satisfaction relation

- Model $\rho$ of dimension $n$: element of $(Q \times \mathbb{N}^n)\omega$ with finite $Q \subseteq Q$.

- Environment $\mathcal{E}$: partial map $\text{VAR}^p \to \mathbb{N}$.

- $\rho, i \models \mathcal{E} \ q \overset{\text{def}}{\iff} q = q_i$.

- $\rho, i \models \mathcal{E} \ \psi \overset{\text{def}}{\iff} \mathbf{v}_i \models \psi$ in $\text{PrA}$
  with $\mathbf{v}_i$ extends $\mathcal{E}$ s.t. $\mathbf{v}_i(x_j) = \overrightarrow{x_i}(j) \ (j \in [1, n])$,
  assuming $\psi$ is a Presburger formula with free variables in
  $\text{VAR}^p \cup \{x_1, \ldots, x_n\}$.

- $\rho, i \models \mathcal{E} \ \exists y \ \varphi \iff$ there is $k \in \mathbb{N}$ such that $\rho, i \models \mathcal{E}[y \mapsto k] \ \varphi$. 

- $\rho, i \models \mathcal{E} \ x \varphi \overset{\text{def}}{\iff} \rho, i + 1 \models \mathcal{E} \ \varphi$. 

- $\rho, i \models \mathcal{E} \ y \varphi$.
Decision problems for $\text{LTL}^{\text{CS}}(\text{PrA})$

- Semi-closed formula: no variable from $\text{VAR}^p$ is free. $F(x_1 = y)$ is not semi-closed unlike $G(x_1 > x_2)$ and $\exists y \ G(x_1 \leq y)$.

- **Satisfiability Problem**
  
  **Input:** An $\text{LTL}^{\text{CS}}(\text{PrA})$ semi-closed formula $\varphi$ with free counter variables $x_1, \ldots, x_n$.
  
  **Question:** Is there a model $\rho \in (Q \times \mathbb{N}^n)$ s.t. $\rho, 0 \models_\emptyset \varphi$?

- **Existential Model-Checking Problem**
  
  **Input:** CS $S = (Q, n, \delta)$, $(q_0, \vec{x}_0)$ and semi-closed formula $\varphi$ with free variables in $\{x_1, \ldots, x_n\}$.
  
  **Question:** Is there an infinite run $\rho$ starting at $(q_0, \vec{x}_0)$ such that $\rho, 0 \models_\emptyset \varphi$?

  (Infinite runs of CS are $\text{LTL}^{\text{CS}}(\text{PrA})$ models)
A simple reduction

- The control state repeated reachability problem can be reduced to the model-checking problem for $\text{LTL}^{CS}(\text{PrA})$.

- Let $S$, $(q, \vec{x})$ and $q_f$ be an instance.

- Equivalence:
  - there is an infinite run from $(q, \vec{x})$ such that $q_f$ is repeated infinitely often,
  - there is an infinite run $\rho$ from $(q, \vec{x})$ such that $\rho, 0 \models G F q_f$.

- This can be extended to the sequence $F_1, \ldots, F_N$ understood conjunctively and each $F_i$ disjunctively.

\[ \bigwedge_{i=1}^{N} \left( \bigvee_{q' \in F_i} G F q' \right) \]
Temporal logics with Presburger constraints

- Constraints on the number of event occurrences.
  [Bouajjani et al., LICS’95; Laroussinie et al., TIME’10]

- Constraints on XML documents.
  [Dal Zilio & Lugiez, RTA’03; Seidl et al, ICALP’04]

- LTL with first-order variables for log auditing.
  [Roger & Goubault-Larrecq, CSFW’01]

- Fairness logic for VASS [Jančar, TCS 90].

- and many many others . . .
**LTL^{CS}(PrA) is sometimes too expressive!**

- Model-checking restricted to LTL^{(Q)} is already undecidable …

- … but LTL^{CS}(PrA) restricted to formulae in which temporal operators are not in the scope of first-order quantification has a decidable satisfiability problem.

- Let us restrict first-order quantification on LTL^{CS}(PrA) to regain decidability for satisfiability.

- Two types of restrictions:
  - First-order quantification only to speak about successive counter values.
  - First-order quantification only to store a counter value and possibly perform equality tests later.
Fragments

\[
\text{LTL} \rightarrow \text{LTL}^\text{CS(PrA)} \rightarrow \text{CLTL(PrA)} \rightarrow \text{CLTL(QFP)} \rightarrow \text{CLTL(DL^+)} \rightarrow \text{CLTL(DL)} \rightarrow \text{CLTL(IPC*)} \rightarrow \text{LTL(Q)} \approx \text{LTL}
\]
Presburger LTL \( \text{CLTL}(\text{PrA}) \)
A macro for comparing successive counter values

• From PrA formula \( \psi(z_1, \ldots, z_k) \),

\[
\psi(X^{i_1}x_{j_1}, \ldots, X^{i_k}x_{j_k}) \overset{\text{def}}{=} \\
(\exists y_1, \ldots, y_k \ X^{i_1}(y_1 = x_{j_1}) \land \cdots \land X^{i_k}(y_k = x_{j_k}) \land \psi(y_1, \ldots, y_k)
\]

• For instance, \( x_1 = Xx_2 \) is obtained from \( z_1 = z_2 \).

• \( G(x_1 = Xx_1) \) states that first counter has constant value.

• CLTL(PrA): fragment of LTL^{CS}(PrA) by allowing quantification only in \( \psi(X^{i_1}x_{j_1}, \ldots, X^{i_k}x_{j_k}) \).

• \( X \) and \( x \) are of distinct nature but both refer to the next position.
The fragment $\text{CLTL(PrA)}$

- **Terms and formulae:**
  
  $t ::= 0 \mid 1 \mid X^i x \mid t + t$

  $\varphi ::= t \equiv_k t \mid t < t \mid q \mid \varphi \land \varphi \mid \neg \varphi \mid X \varphi \mid \varphi \cup \varphi.$

  with $x \in \text{VAR}$, $k > 1$ and $q \in \mathbb{Q}$.

- **NB:** no integer variables, quantifier-free Presburger formulae.

- $G(x_1 < X x_1)$ is satisfiable; $G(x_1 > X x_1)$ is not.

- $\sigma, i \models \psi(X^{l_1} x_1, \ldots, X^{l_n} x_n)$ iff
  
  $(\sigma_1(i + l_1)(x_1), \ldots, \sigma_1(i + l_n)(x_n)) \in \text{REL}(\psi)$ with

  - $\sigma = (\sigma_1, \sigma_2),$
  - $\sigma_1 : \mathbb{N} \rightarrow (\text{VAR} \rightarrow \mathbb{N}),$
  - $\sigma_2 : \mathbb{N} \rightarrow Q$ for some finite subset $Q \subseteq \mathbb{Q}$.

- Model-checking and satisfiability problems as subproblems for $\text{LTL}^{CS}(\text{PrA})$. 
X-length

- **X-length of** $\varphi$: maximal $l$ such that $X^lx$ occurs in $\varphi$.

- **CLTL$^l_n$(PrA):** restriction of **CLTL(PrA)** to formulae with at most $n$ variables and X-length less or equal to $l$.

- There is a logspace reduction from the satisfiability problem for **CLTL(PrA)** to its restriction with X-length 1.
  - Expressions $x_1, \ldots, X^3x_1$ are encoded by
  
  $G(x'' = Xx' \land x' = Xx \land x = Xx_1)$

  - $Xx_1$ [resp. $X^2x_1, X^3x_1$] is replaced by $x$ [resp. $x'$, $x''$].
  - Take the conjunction.
Two fragments of \( \text{PrA} \)

- **Difference logic DL:**
  \[
  \varphi ::= x \sim y + d \mid x \sim d \mid \varphi \land \varphi \mid \neg \varphi
  \]
  with \( d \in \mathbb{Z}, \sim \in \{<, >, =\} \).

- **Quantifier-free Presburger arithmetic QFP:**
  \[
  \varphi ::= \sum_{i \in I} a_i x_i \sim d \mid \sum_{i \in I} a_i x_i \equiv_k c \mid \varphi \land \varphi \mid \neg \varphi
  \]
  \((c \in \mathbb{N}, k \geq 1, a_i’s \text{ in } \mathbb{Z})\).

- Satisfiability problem for QFP is NP-complete.

- **CLTL(\(L\)):** fragment of CLTL(\(\text{PrA}\)) to atomic formulae of the form \( \psi(X^{i_1}x_{j_1}, \ldots, X^{i_k}x_{j_k}) \) with \( \psi \in L \).
A bunch of complexity results

- DL-counter system: counter system such that the Presburger formulae labelling the transitions are in DL.

<table>
<thead>
<tr>
<th></th>
<th>MC (DL)</th>
<th>SAT</th>
<th>MC (CA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{CLTL}^1_3(\text{DL})$</td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>$\text{CLTL}^{\omega}_2(\text{DL})$</td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>$\text{CLTL}^2_1(\text{DL})$</td>
<td>U</td>
<td>U</td>
<td>$\text{PSpace}$-c.</td>
</tr>
<tr>
<td>$\text{CLTL}^1_2(\text{DL})$</td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>$\text{CLTL}^1_1(\text{DL or DL}^+)$</td>
<td>$\text{PSpace}$-c.</td>
<td>$\text{PSpace}$-c.</td>
<td>$\text{PSpace}$-c.</td>
</tr>
<tr>
<td>$\text{CLTL}^1_1(\text{QFP})$</td>
<td>U</td>
<td>U</td>
<td>$\text{PSpace}$-c.</td>
</tr>
<tr>
<td>$\text{CLTL}^{\omega}_1(\text{QFP})$</td>
<td>U</td>
<td>U</td>
<td>$\text{PSpace}$-c.</td>
</tr>
</tbody>
</table>
Satisfiability problem for $\text{CLTL}^{1}_3(\text{DL})$ is undecidable

- Simple reduction from halting problem for Minsky machines.

- Formulae in $\text{CLTL}^{1}_3(\text{DL})$ can easily internalize the instructions of Minsky machines.

- $S$ has $n$ instructions, the $n$th one halts.

- $x_1$ and $x_2$ encode counter values, $x_3$ encodes the instruction ordinal.

- $S$ does not halt iff formula below is satisfiable:

$$\left( \bigwedge_{i \in [1, n]} \psi_i \right) \land x_1 = 0 \land x_2 = 0 \land x_3 = 1$$
Internalizing instructions from Minsky machines

- For “i: increment counter j and goto i’”

\[ \psi_i \overset{\text{def}}{=} G(x_3 = i \Rightarrow (Xx_3 = i') \land (x_j + 1 = Xx_j) \land (x_{3-j} = Xx_{3-j})). \]

- For “i: if counter j equals zero then goto i’ else (decrement counter j; goto i’’):”

\[ G(x_3 = i \Rightarrow ((x_j = 0 \Rightarrow (Xx_3 = i') \land (x_j = Xx_j) \land (x_{3-j} = Xx_{3-j})) \land (x_j \neq 0 \Rightarrow (Xx_3 = i'') \land (x_j - 1 = Xx_j) \land (x_{3-j} = Xx_{3-j})))) \]

- Satisfiability problem for $\text{CLTL}^1_2(\text{DL})$ is undecidable too.
LTL with Registers (a.k.a. Freeze LTL)
• A register can store a counter value and equality tests are performed between registers and counters.

\[ j_r \varphi \overset{\text{def}}{=} \exists y_r \ (y_r = x_j \land \varphi) ;\]
\[ j_r \overset{\text{def}}{=} y_r = x_j. \]

• Generalized disjunction

\[ \bigvee_{i \geq 1} x_1 = x_i^i x_2 \]

\[ \downarrow_1 x_F \uparrow_1^2 \]

\[ \mathcal{G}(\downarrow^1 y \mathcal{G} \neg \uparrow_1^1) : \text{first counter has distinct values at distinct positions.} \]
Freeze operator

- Freeze quantifier in hybrid logics.
  
  \[ \downarrow x \varphi: \varphi \text{ holds true in the variant model where } x \text{ is true only at the current state/world.} \]
  
  [Goranko 94; Blackburn & Seligman, JOLLI 95]

- Freeze quantifier in real-time logics.
  
  \[ x \cdot \varphi(x) \text{ binds the variable } x \text{ to the current time } t. \]
  
  [Alur & Henzinger, JACM 94]

- Predicate \( \lambda \)-abstraction [Fitting, JLC 02].
  
  \[ \langle x \cdot Fp(x) \rangle(c): \text{ current value of constant } c \text{ satisfies the predicate } p. \]

In LTL with registers, FO quantification is restricted to \( \downarrow \) and \( \uparrow \).
LTL with registers

- For $n \geq 1$, LTL$^\downarrow[n]$ formulae:
  
  $$
  \varphi ::= q \mid \uparrow_r^j \mid \neg \varphi \mid \varphi \land \varphi \mid \varphi \lor \varphi \mid \varphi \cup \varphi \mid \mathbf{X} \varphi \mid \downarrow_r^j \varphi
  $$

  where $q \in Q$, $j \in \{1, \ldots, n\}$ and $r \in \mathbb{N}^+$. 

- Register valuation $f$: finite partial map from $\mathbb{N}^+$ to $\mathbb{N}$.

- Models: element of $(Q \times \mathbb{N}^n)\omega$ with finite $Q \subseteq Q$.

- Satisfiability relation:
  
  $$
  \rho, i \models_f \uparrow_r^j \iff r \in \text{dom}(f) \text{ and } f(r) = \vec{x}_i(j)
  $$
  
  $$
  \rho, i \models_f \downarrow_r^j \varphi \iff \rho, i \models_f[r \mapsto \vec{x}_i(j)] \varphi
  $$
Helpful properties

- **Nonce property:**
  \[G(\downarrow_1 XG \neg \uparrow_1)\]

- **There is a value of counter 1 such that infinitely often counter 2 takes that value iff infinitely often counter 3 takes that value:**
  \[F \downarrow_1 (GF \uparrow_2 \iff GF \uparrow_3)\]

- **Value of counter 1 is repeated on counter 2 but not on counter 3:**
  \[\downarrow_1 (F \uparrow_2 \land G \neg \uparrow_3)\]
Complexity of satisfiability problems

- Infinitary satisfiability problem for LTL$\downarrow[1]$ restricted to $x$ and $F$ and to a single register is undecidable.

- Finitary satisfiability problem for LTL$\downarrow[1]$ restricted to a single register is decidable but nonprimitive recursive. [Demri & Lazić, TOCL 09].

- Finitary satisfiability problem for LTL$\downarrow[1]$ restricted to a single register and to $F$ is nonprimitive recursive too.

- Finitary satisfiability problem for LTL$\downarrow[1]$ restricted to $F$ and to two registers is undecidable. [Figueira & Segoufin, MFCS’09]

- Nonprimitive recursiveness uses [Schnoebelen, IPL 02].
Gainy counter automata

- Gainy counter automata: standard counter automaton 
  \((Q, n, \delta)\) s.t. for \(q \in Q\) and \(i \in [1, n]\), \(q \xrightarrow{\text{inc}(i)} q \in \delta\)

- Alternative one-step relation: \((q, \vec{x}) \xrightarrow{t} g (q', \vec{x}')\) iff there are \(\vec{y}, \vec{y}'\) in \(\mathbb{N}^n\) s.t.
  \[
  \vec{x} \preceq \vec{y} \quad (q, \vec{y}) \xrightarrow{t} (q', \vec{y}') \quad (\text{exact step}) \quad \vec{y}' \preceq \vec{x}'
  \]

- Gains can occur in a lazy way: decrement on zero has no effect.
• Control state reachability problem is decidable.

• Control state repeated reachability problem is undecidable (see Day 5).

• Reduction of the latter problem to satisfiability problem for $\text{LTL}^{\downarrow}[1]$ restricted to $\mathsf{X}$ and $\mathsf{F}$ and to a single register.
Simulating Gainy CA

- Gainy CA $S$ with initial configuration $(q_0, \bar{0})$.
- For $t \in \delta$, $\Sigma(t)$ denotes the instruction labelling it in
  $\Sigma = \{\text{inc}(i), \text{dec}(i), \text{zero}(i) : i \in [1, n]\}$.
- Let us build $\varphi$ in $\text{LTL}^\downarrow[1]$ s.t. $\varphi$ is satisfiable iff $(S, (q_0, \bar{0}))$
  has an infinite run with $q_f$ occurring infinitely often.
- $\varphi$ is satisfiable only in models in which each position is
  labelled by a transition and by a value in $\mathbb{N}$.
- Each increment or decrement is associated to a unique value.
- Infinite models of $\varphi$ are of the form
  
  $t_0 \quad t_1 \quad t_2 \quad t_3 \quad \ldots$

  $d_0 \quad d_1 \quad d_2 \quad d_3 \quad \ldots$

  with $t_i \in \delta$ (finite alphabet) and $d_i \in \mathbb{N}$ (infinite domain).
Simulating Gainy CA (II)

- Let us explain how the run from \((q_0, \vec{0})\) below is encoded.

\[
(q_0, \vec{x}_0) \xrightarrow{a_0} (q_1, \vec{x}_1) \xrightarrow{a_1} \cdots \xrightarrow{a_{K-1}} (q_K, \vec{x}_K) \cdots
\]

- Projection of the model over \(\delta\) is

\[
t_0\ t_1\ t_2\ \cdots = q_0 \xrightarrow{a_0} q_1,\ q_1 \xrightarrow{a_1} q_2, \cdots
\]

and \(q_f\) is repeated infinitely often.

- Initial state is \(q_0\):

\[
\bigvee_t \ t \quad t=q_0 \xrightarrow{a}\ q
\]

- The sequence of transitions respects \(S\):

\[
G(\bigwedge_{t=q \xrightarrow{a} q' \in \delta} (t \Rightarrow X \bigvee t'))
\]

\[
t'=q' \xrightarrow{a} q''
\]
Simulating Gainy CA (III)

- Control state $q_f$ is visited infinitely often:

$$\mathsf{GF} \bigvee t$$

$$t=q^a \rightarrow q_f$$

- Below, $\downarrow$ [resp. $\uparrow$] denotes $\downarrow_1$ [resp. $\uparrow_1$].

- For $a \in \Sigma$, $a$ is also used as a shortcut for

$$\bigvee t$$

$$t=q^b \rightarrow q' \in \delta, \ a=b$$

- For $i, j \in [1, n]$, there are no two positions for increments having the same value:

$$\mathsf{G}(\mathsf{inc}(i) \Rightarrow \neg (\downarrow \mathsf{XF}(\uparrow \wedge \mathsf{inc}(j))))$$

- For $i, j \in [1, n]$, there are no two positions for decrements having the same value:

$$\mathsf{G}(\mathsf{dec}(i) \Rightarrow \neg (\downarrow \mathsf{XF}(\uparrow \wedge \mathsf{dec}(j))))$$
Simulating Gainy CA (IV)

- For each zero-test on $i$ at position $K$, each increment on $i$ before position $K$ has a corresponding decrement on $i$ before $K$.

- The two next conditions are formulated in such a way to avoid using the until operator $\mathcal{U}$.

- $I \sim K$: data value at position $I$ is equal to data value at position $K$.

- For $i \in [1, n]$ and $J > I$, if $\Sigma(t_i) = \text{inc}(i)$ and $\Sigma(t_J) = \text{zero}(i)$, then there is no $K > J$ such that $\Sigma(t_K) = \text{dec}(i)$ and $I \sim K$:

$$G(\text{inc}(i) \Rightarrow \downarrow \neg (F(\text{zero}(i) \land (F(\uparrow \land \text{dec}(i)))))))$$
Simulating Gainy CA (V)

- For \( i \in [1, n] \) and \( J > I \), if \( \Sigma(t_I) = \text{inc}(i) \) and \( \Sigma(t_J) = \text{zero}(i) \), then there is no \( K > J \) such that \( \Sigma(t_K) = \text{dec}(i) \) and \( I \sim K \):

\[
G(\text{inc}(i) \Rightarrow \downarrow \neg (F(\text{zero}(i) \land (F(\uparrow \land \text{dec}(i)))))
\]

- For \( i \in [1, n] \), if there are \( J > I \) such that \( \Sigma(t_I) = \text{inc}(i) \) and \( \Sigma(t_J) = \text{zero}(i) \), then there is \( K > I \) such that \( \Sigma(t_K) = \text{dec}(i) \) and \( I \sim K \).

\[
G((\text{inc}(i) \land F \text{zero}(i)) \Rightarrow \downarrow (F(\text{dec}(i) \land \uparrow)))
\]

- \( \varphi \) is satisfiable iff \( (S, (q_0, \vec{0})) \) has an infinite run such that \( q_f \) occurs infinitely often.
Related formalisms

• Register automata
  • Register automata [Kaminski & Francez, TCS 94]
  • Data automata [Bouyer & Petit & Thérien, IC 03]
  • See the survey [Segoufin, CSL 06]

• First-order languages [Bojańczyk et al., LICS 06]

• Temporal logics
  • Real-time logic TPTL [Alur & Henzinger, JACM 94]
  • LTL with freeze [D. & Lazić & Nowak, TIME 05]

• Many other formalisms
  • Rewriting systems with data [Bouajjani et al., FCT 07]
  • Hybrid logics [Schwentick & Weber, STACS 07]
  • . . .
Conclusion

• Today’s lecture:
  • Standard LTL
  • Logic $\text{LTL}^\text{CS}(\text{PrA})$ for counter systems
  • Presburger LTL
  • LTL with registers

• Tomorrow’s lecture:
  • VASS and FO2 over data words
  • Relationships between VASS, VAS and Petri nets
  • Coverability graphs in a nutshell
  • Covering problem in $\text{EXPSPACE}$