Model-checking FCTL is hard

F. LAROUSSINIE, N. MARKEY, PH. SCHNOEBELEN

Lab. Specification & Verification
CNRS UMR 8643 – ENS Cachan
61, av. President Wilson
94235 Cachan Cedex – FRANCE
What is model-checking?

Does this system satisfy that property?

\[ M \models \varphi \]

\( M \) is a finite Kripke structure modeling the system,
\( \varphi \) is a temporal logic formula expressing the property.
What is model-checking?

When you call the lift, you want it to come eventually.
What is model-checking?

finite state model $M$:

When you call the lift, you want it to come eventually.

temporal logic formula $\phi$:

$$\forall G(\text{call} \rightarrow \forall F \text{ servicing})$$

Question: does $M \models \phi$?
<table>
<thead>
<tr>
<th>Temporal Logic</th>
<th>Model checking complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTL [Pnu77]</td>
<td>PSPACE-complete [SC85]</td>
</tr>
<tr>
<td>CTL [CE81]</td>
<td>P-complete [CES86]</td>
</tr>
<tr>
<td>CTL * [EH86]</td>
<td>PSPACE-complete [CES86]</td>
</tr>
</tbody>
</table>
Complexity of model checking

<table>
<thead>
<tr>
<th>Temporal Logic</th>
<th>Model checking complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTL ([Pnu77])</td>
<td>PSPACE-complete ([SC85])</td>
</tr>
<tr>
<td>CTL ([CE81])</td>
<td>P-complete ([CES86])</td>
</tr>
<tr>
<td>FCTL ([EH85])</td>
<td>?</td>
</tr>
<tr>
<td>CTL * ([EH86])</td>
<td>PSPACE-complete ([CES86])</td>
</tr>
</tbody>
</table>
Complexity of Model-Checking

Importance of complexity:

- important criterion for comparing temporal logics,
- optimal algorithms,
- understand why model checking is difficult,
- reach a compromise between expressivity, succinctness, and complexity.
Outline of the talk

- What is FCTL precisely?
- How hard is FCTL model checking?
  - it is $\textbf{NP}$- and $\textbf{co-NP}$-hard,
  - it is $\Delta_2$-easy \cite{CES86},
- What is there between $\textbf{NP}$ and $\Delta_2$?
- New result: model-checking FCTL is $\Delta_2$-hard.
CTL (Computation Tree Logic) [CE81]

\[ \begin{align*}
E(a \lor c) & \mathcal{U} b \\
A(a \lor c) & \mathcal{U} b \\
AX & \neg d
\end{align*} \]
CTL (Computation Tree Logic) [CE81]

\[ \mathbf{E}(a \vee c) \mathbf{U} b \]
\[ \mathbf{A}(a \vee c) \mathbf{U} b \]
\[ \mathbf{AX} \neg d \]

**Abbreviations:**

\[ \mathbf{EF} b = \mathbf{E} \mathbf{T} \mathbf{U} b \] (Potentially)
\[ \mathbf{AG} \neg c = \neg \mathbf{EF} c \] (Always)
\[ \mathbf{AF}(b \vee d) = \mathbf{A} \mathbf{T} \mathbf{U} (b \vee d) \] (Inevitably)
CTL (Computation Tree Logic) \[\text{[CE81]}\]

\[\text{EF}(\neg a \land \text{AF}a)\]

\[\text{AG}(a \rightarrow \text{EF}(\text{EG}\neg a))\]

CTL cannot express \textit{fairness} properties, not even “there exists a path going infinitely often through \(a\)”. 
FCTL (Fair Computation Tree Logic) [EL85]

FCTL specification: \((\varphi, \Phi)\)

\(\Phi\) is a fairness constraint, i.e. a boolean combination of \(\mathcal{F} a\), where \(\mathcal{F} a\) means "infinitely often \(a\)".

\(\varphi\) is a CTL formula where path quantifiers are relativised to fair paths (i.e. paths verifying \(\Phi\)). This is emphasized by writing \(E_f\) and \(A_f\) for path quantifiers.
FCTL (Fair Computation Tree Logic) [EL85]

\[
\begin{align*}
\varphi & : A_f G (E_f F d) \\
\Phi & : F a \\
\end{align*}
\]

\[
\begin{align*}
\varphi & : A_f F (d \land A_f F \neg a) \\
\Phi & : F a \\
\end{align*}
\]

The right expressive power.
Model Checking FCTL is NP-hard

\[ SAT \,: \text{is } (x \lor y \lor z) \land (\overline{x} \lor t \lor \overline{z}) \land (\overline{x} \lor \overline{t} \lor \overline{y}) \text{ satisfiable?} \]
Model Checking FCTL is NP-hard

\[ \text{SAT} : \text{is } (x \lor y \lor z) \land (\overline{x} \lor t \lor \overline{z}) \land (\overline{x} \lor \overline{t} \lor \overline{y}) \text{ satisfiable?} \]

\[ \Phi = (Fx \lor Fy \lor Fz) \land (\overline{F}x \lor Ft \lor F\overline{z}) \land (\overline{F}x \lor F\overline{t} \lor F\overline{y}) \]

Does \( M \models E\Phi \)?
Model Checking FCTL is NP-hard

\[ \text{SAT} : \text{is } (x \lor y \lor z) \land (\overline{x} \lor t \lor \overline{z}) \land (\overline{x} \lor \overline{t} \lor y) \text{ satisfiable?} \]

\[ \Phi = ((\mathcal{F}x \lor \mathcal{F}y \lor \mathcal{F}z) \land (\mathcal{F}\overline{x} \lor \mathcal{F}t \lor \mathcal{F}\overline{z}) \land (\mathcal{F}\overline{x} \lor \mathcal{F}\overline{t} \lor \mathcal{F}y)) \land \]

\[ \land_{u \in \text{Var}} \neg (\mathcal{F}u \lor \mathcal{F}\overline{u}) \]

Does \( M \models (\mathcal{E}_t \mathcal{F}\text{True}, \Phi) \)?
Δ₂, also denoted $P^{NP}$, is the class of problems that can be solved in *polynomial* time by a deterministic Turing machine with an $NP$ oracle (subroutine).

Model-checking FCTL is $Δ₂$-easy: for each $E_{f} \phi_1 U \phi_2$, we use the $NP$ oracle to check whether there exists a fair path satisfying $\phi_1 U \phi_2$. The same holds for the other operators. Quantifiers can be nested, but a $P^{NP}$ algorithm can ask a polynomial number of dependant queries.
Polynomial time hierarchy

\[ \text{PSPACE} \quad \vdash \quad \text{CTL}^*, \text{LTL} \]

\[ \Pi_2 = \text{co-NP}^{\text{NP}} \quad \Sigma_2 = \text{NP}^{\text{NP}} \]

\[ \Delta_2 = \text{P}^{\text{NP}} \]

\[ \Theta \text{P} = \text{P}^{\text{NP}(O(\log n))} \]

\[ \text{DP} = \{ L \cap L' \mid L \in \text{NP}, L' \in \text{co-NP} \} \]

\[ \text{co-NP} = \Pi_1 \quad \text{NP} = \Sigma_1 \]

\[ \text{P} = \Delta_1 \]

\[ \text{L(F)} \quad \text{CTL} \]
(I) \[
\begin{align*}
x_1 & := \exists \{z_{1,1}, \ldots, z_{1,p}\} \quad (z_{1,1} \lor \overline{z_{1,2}} \lor \ldots) \land (z_{1,2} \lor \ldots) \land \ldots \\
x_2 & := \exists \{z_{2,1}, \ldots, z_{2,p}\} \quad (\overline{z_{2,1}} \lor \overline{x_1} \lor \ldots) \land (z_{2,2} \lor \ldots) \land \ldots \\
x_3 & := \exists \{z_{3,1}, \ldots, z_{3,p}\} \quad (z_{3,1} \lor \overline{x_2} \lor \ldots) \land (x_1 \lor \ldots) \land \ldots \\
& \quad \ldots \\
x_r & := \exists \{z_{r,1}, \ldots, z_{r,p}\} \quad \land \left( \lor_{l \leq m} \alpha_{r,l,m} \right) \\
\end{align*}
\]

This defines a valuation $v_I : \{x_1, \ldots, x_r\} \to \{0, 1\}$.

Question: Does $v_I(x_r) = 1$?

**SNSAT** is $\Delta_2$-complete.
Model checking FCTL is $\Delta_2$-hard

\[
\begin{align*}
\phi_k &= E_f G[\neg C \land ((\lor_i P_{\overline{x_i}}) \rightarrow E_f X(C \land E_f X \neg \phi_{k-1}))] \\
\Phi &= \land_i \left((\land_m \land \infty P_{\alpha_i,l,m}) \land \land_{u \in Var} \neg \left(\land_{l,m} \land \infty P_u \land \land_{u \in Var} \right)\right)
\end{align*}
\]
More $\Delta_2$-complete verification problems

\[ A(\text{Fa} \rightarrow b \mathcal{U} a) \quad \text{CTL}^+ \quad [EH85] \]
\[ A(\text{F}(a) \rightarrow \text{Ga}) \quad \text{ECTL}^+ \quad [EH86] \]
\[ E(\text{Fa} \land \text{G}\neg b) \quad \text{BTL}^+ \]

Model-checking ECTL$^+$, CTL$^+$ and BTL$^+$ is $\Delta_2$-complete.
Conclusion

• answers a longstanding open problem,

• provides the first model-checking problem that is complete for $\Delta_2$,

• looking for other verification problems in $\Delta_2$. 