LTL$_f$ Synthesis on First-Order Agent Programs in Nondeterministic Environments

Till Hofmann and Jens Claßen

Chair of Computer Science 6 (Machine Learning and Reasoning) · RWTH Aachen University · Roskilde University

At a Glance

Golog: agent programming language based on **first-order logic** with **nondeterministic operators**

Program realization: Resolve nondeterministic operators to determine successful program execution

So far: Nondeterminism assumed to be **angelic**

Key idea: Environment controls some actions + **temporal LTL$_f$ goal** → realization becomes a **synthesis task**

Result: Sound and complete procedure to determine a **policy**

Background

LTL$_f$ Synthesis

Given:

- A **finite** set of **propositional symbols** $\mathcal{P} = \mathcal{X} \dot\cup \mathcal{Y}$ where $\mathcal{X}$ is uncontrollable and $\mathcal{Y}$ is controllable
- LTL$_f$ formula Φ

Task: Control $\mathcal{Y}$ such that for all values of $\mathcal{X}$, Φ is satisfied

Limitations:

- Fixed, finite set propositions
- Agent and environment always alternate
- User preferences must be encoded into Φ

Agent Programming with Golog

- *Situation calculus:* first-order axiomatization of dynamic worlds
- $\mathcal{ES}$: modal variant, e.g.,

  $[a]\phi \mathrel{\hat=}$ "*ϕ holds after doing action a*"
- Golog: agent programming language based on the situation calculus
- program may use nondeterministic operators → partial behavior specification

Limitation: Nondeterminism is assumed to be **angelic**

LTL$_f$ Synthesis on Golog Programs

Proposal: Combine LTL$_f$ synthesis with agent programs

Given:

- Golog program $\mathcal{G} = (\mathcal{D}, \delta)$
- Partitioning of primitive actions $\mathcal{A} = \mathcal{A}_C \dot\cup \mathcal{A}_E$ into controllable $\mathcal{A}_C$ and environment actions $\mathcal{A}_E$
- First-order LTL$_f$ temporal goal Φ:

  $$\Phi ::= \phi \mid \Phi \wedge \Phi \mid \mathcal{X}\,\Phi \mid \Phi\,\mathcal{U}\,\Phi$$

  where ϕ is an $\mathcal{ES}$ fluent sentence

Task: Determine **execution policy** π such that

- π may only choose actions according to program δ
- π may not restrict environment actions
- π must be *non-blocking*
- every trace must satisfy Φ

A Decidable Fragment

The synthesis problem is **undecidable** in general

Decidable fragment (Zarrieß and Claßen 2016):

- Base logic restricted to C^2 (two variables + counting)
- Successor state axioms must be **acyclic**
- Pick operator restricted to finite domains

Finite abstraction:

- **Characteristic graph** is a finite representation of δ
- Finitely many (accumulated) effects $\mathfrak{E}^{\mathcal{D},\mathcal{A}}$
- Finitely many **world types** Types($\mathcal{G}$)

$$\mathrm{type}(w) \doteq \{(\underbrace{\psi}_{\text{context condition from } \mathcal{C}(\mathcal{G})}, \underbrace{E}_{\text{effect from } \mathfrak{E}^{\mathcal{D},\mathcal{A}}}) \mid \underbrace{w \models \mathcal{R}[E,\psi]}_{\text{"}\psi \text{ holds in } w \text{ after applying } E\text{"}}\}$$

Example: Dish Robot

Initial situation:

$Dish(x) \equiv (x = d_1),\ Room(x) \equiv (x = r_1)$

$\forall x.\, At(x) \equiv x = kitchen$

$\forall x.\, New(x) \equiv Dish(x)$

$\quad \wedge \forall y.\, \neg DirtyDish(x,y) \wedge \neg OnRobot(x)$

$OnRobot(x) \supset Dish(x) \wedge \neg\exists y\, DirtyDish(x,y)$

$DirtyDish(x,y) \supset Dish(x) \wedge Room(y) \wedge \neg OnRobot(x)$

Precondition axioms:

$\Box\,\mathrm{Poss}(load(x,y)) \equiv DirtyDish(x,y) \wedge At(y)$

$\Box\,\mathrm{Poss}(unload(x)) \equiv OnRobot(x) \wedge At(kitchen)$

$\Box\,\mathrm{Poss}(addDish(x,y)) \equiv New(x) \wedge Room(y)$

$\Box\,\mathrm{Poss}(goto(x)) \equiv Room(x) \vee x = kitchen$

Successor state axioms:

$\Box[a]DirtyDish(x,y) \equiv a = addDish(x,y)$

$\quad \vee DirtyDish(x,y) \wedge a \neq load(x,y)$

$\Box[a]OnRobot(x) \equiv \exists y.\, a = load(x,y)$

$\quad \vee OnRobot(x) \wedge a \neq unload(x)$

$\Box[a]New(x) \equiv New(x) \wedge \neg\exists y.\, a = addDish(x,y)$

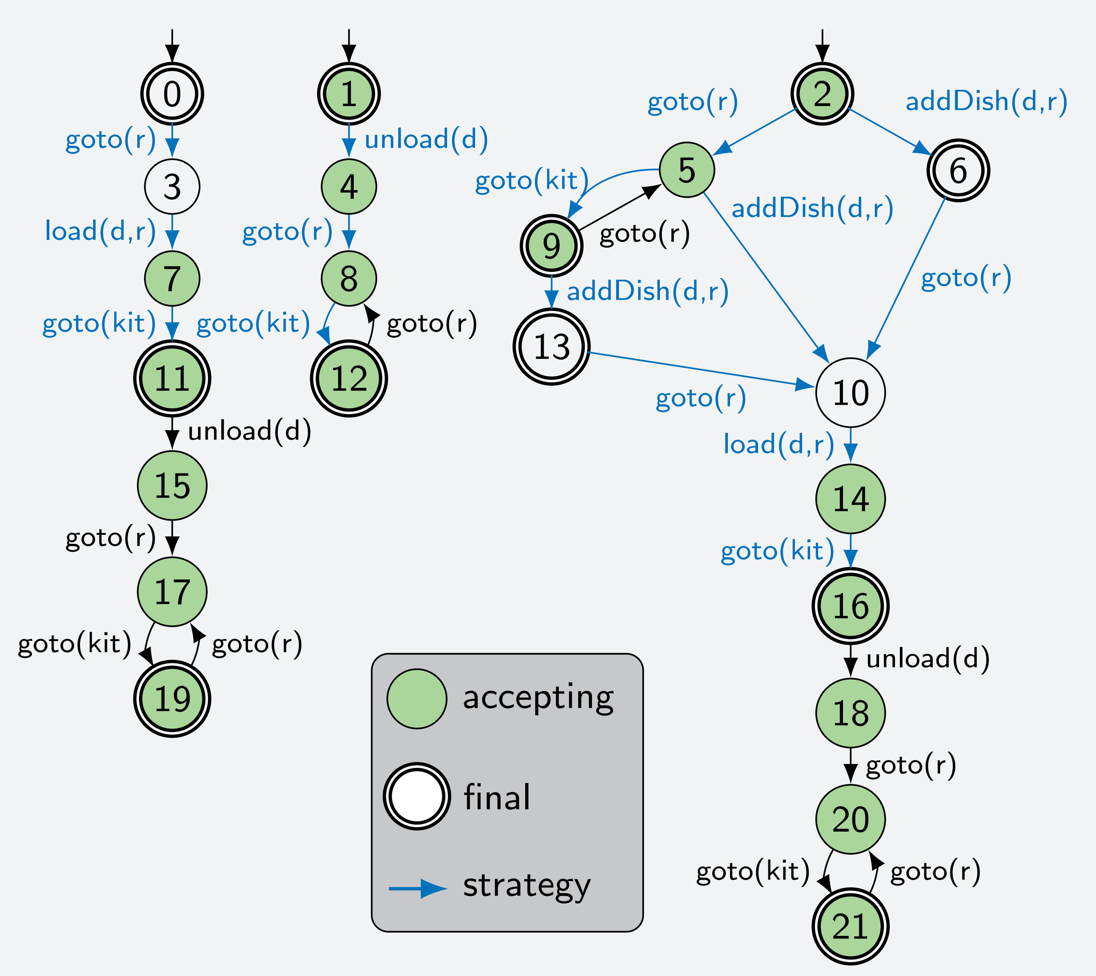
$\Box[a]At(x) \equiv a = goto(x) \vee At(x) \wedge \neg\exists y. a = goto(y)$

Program:

loop:
  while $\exists x.\, OnRobot(x)$ **do**
    $\pi x : \{d_1\}.\, unload(x)$;
    $\pi y : \{r_1\}.\, goto(y)$; *agent*
  while $\exists x.\, DirtyDish(x,y)$ **do**
    $\pi x : \{d_1\}.\, load(x,y)$;
  $goto(kitchen)$
$\|$
loop: $\pi x : \{d_1\}, y : \{r_1\}.\, addDish(x,y)$ *environment*

Specification: $\mathcal{F}\,\mathcal{G}\,\neg\exists x,y.\, DirtyDish(x,y)$

Game Arena for Dish Robot



Evaluation: Dish Robot

R	D	Nodes (TS)	Edges (TS)	Nodes (St)	Edges (St)	Time [s]
1	1	22	25	16	19	2.6
1	2	150	203	128	176	155.4
1	3	–	–	–	–	–
2	1	109	168	87	110	69.0
2	2	–	–	–	–	–
3	1	483	857	413	543	1885.7
3	2	–	–	–	–	–

LTL$_f$ Normal Forms: TNF and XNF

TNF: Introduce *Tail* to mark the last state of a trace

XNF: Convert formula such that the only outermost temporal connective is $\mathcal{X}$

⇒ Treat Φ as **propositional formula**, where subformulas of the form $\mathcal{X}\,\Psi$ are propositions

⇒ Split propositional assignment P into three parts: **local** $L(P)$, **next** $X(P)$, **tail** $T(P)$

The Game Arena

Given:

- Golog program $\mathcal{G} = (\mathcal{D}, \delta)$
- Temporal formula Φ

Game arena $\mathbb{A}^{\Phi}_{\mathcal{G}} = (\mathcal{S}, \mathcal{S}_0, \rightarrow, \mathcal{S}_F, \mathcal{S}_A)$:

- Each state $s \in \mathcal{S}$ is of the form $s = (\tau, E, A, \rho)$ where
  1. $\tau \in \mathrm{Types}(\mathcal{G})$;
  2. $\rho \in \mathrm{sub}(\delta)$ is a node of the characteristic graph;
  3. $E \subseteq \mathfrak{E}^{\mathcal{D},\mathcal{A}}$;
  4. $A = \{(\chi_i, \theta_i)\}_i$, where $\chi_i \subseteq \mathrm{cl}(\Phi)$, $\theta_i \in \{\top, \bot\}$.
- A state $s = (\tau, E, A, \rho)$ is an initial state $s \in \mathcal{S}_0$ if
  1. $\tau = \mathrm{type}(w)$ for some w with $w \models \mathcal{D}$;
  2. $\rho = \delta$ is the initial program expression;
  3. $E = \emptyset$;
  4. $(\chi, \theta) \in A$ iff there is a propositional assignment P of $\mathrm{xnf}(\Phi)^p$ such that
     (a) $\{(\psi, E) \mid \psi \in L(P)\} \subseteq \tau$
     (b) $\chi = X(P)$
     (c) $\theta = T(P)$
- There is a transition $s_1 \xrightarrow{\alpha} s_2$ from $s_1 = (\tau, E_1, A_1, \rho_1)$ to $s_2 = (\tau, E_2, A_2, \rho_2)$ if
  1. there is an edge $\rho_1 \xrightarrow{\alpha:\psi} \rho_2$ in $\mathcal{C}_\delta$ with $(\psi, E_1) \in \tau$;
  2. $E_2 = E_1 \rhd \mathcal{E}_{\mathcal{D}}(\tau, E_1, \alpha)$;
  3. $(\chi_2, \theta_2) \in A_2$ if there is a propositional assignment P of $\mathrm{xnf}(\bigwedge \chi_1^p)$ for some $(\chi_1, \theta_1) \in A_1$ such that
     (a) $\theta_1 = \bot$
     (b) $\{(\psi, E_2) \mid \psi \in L(P)\} \subseteq \tau$
     (c) $\chi_2 = X(P)$
     (d) $\theta_2 = T(P)$

A state $s = (\tau, E, A, \rho)$ is

- *final* if $(\varphi(\rho), E) \in \tau$, and • *accepting* if $(\emptyset, \top) \in A$.

Finding a Strategy

for all $H \in 2^{\mathcal{S}_F \cap \mathcal{S}_A}$ **do**
  $G \leftarrow H$; $R \leftarrow \{s \in G \mid \mathrm{Succ}_E(s) = \emptyset\}$; $\sigma \leftarrow \emptyset$
  $Q \leftarrow \{s \in \mathcal{S} \mid \mathrm{Succ}(s) \cap G \neq \emptyset\}$
  while $Q \neq \emptyset$ **do**
    $s \leftarrow \mathrm{POP}(Q)$
    if $s \in \mathcal{S}_F \setminus \mathcal{S}_A \wedge \mathrm{Succ}_C(s) = \emptyset$ **then continue**
    if $s \in R$ **then continue**
    if $\mathrm{Succ}_E(s) \neq \emptyset \wedge \forall s' \in \mathrm{Succ}_E(s) : s' \in G \vee \mathrm{Succ}_E(s) = \emptyset \wedge \exists s' \in \mathrm{Succ}_C(s) : s' \in G$ **then**
      $G \leftarrow G \cup \{s\}$; $R \leftarrow R \cup \{s\}$
      if $s \in \mathcal{S}_F \cap \mathcal{S}_A$ **then**
        $\sigma(s) \leftarrow \{\alpha \mid \exists s' \in \mathrm{Succ}_E(s).\ s \xrightarrow{\alpha} s'\}$
      else $\sigma(s) \leftarrow \{\alpha \mid \exists s' \in G.\ s \xrightarrow{\alpha} s'\}$
      $Q \leftarrow Q \cup \{s' \mid s \in \mathrm{Succ}(s')\}$
  if $H \cup \mathcal{S}_0 \subseteq R$ **then return** σ

Paper   Poster   Code