

Temporal Logic (Goal) Specifications for Automated Planning



THE
MOONSHOT
FACTORY

Alberto Camacho
acamacho@cs.toronto.edu

TLf@AAAI-SSS'23
March 27, 2023

This work was conducted while
affiliated with:



UNIVERSITY OF
TORONTO

It takes a village...



Sheila Mcilraith



Christian Muise



Jorge Baier



Eleni Triantafillou



Anonymous reviewers



Scott Sanner



Meghyn Bienvenu




Toryn Klassen



Rodrigo Toro Icarte




Rick Valenzano



We want to **synthesize** strategies for sequential decision making...

- ... to operate in a world that is potentially **non-deterministic**
- ... to satisfy complex goals that may be **temporally extended**
- ... in a way that we can offer correctness **guarantees**.



I Will Talk About...

How can we specify problems?

- FOND planning
 - Env. Fairness
- Reactive synthesis

How can we specify goals in planning?

- Final-state condition
- Temporal logics
 - on finite traces
 - on infinite traces

How can we solve planning problems?

- Automata goal representations
- FOND planners as a tool

Complexity results (Theory and Practice)

Fully Observable Non-Deterministic (FOND) planning

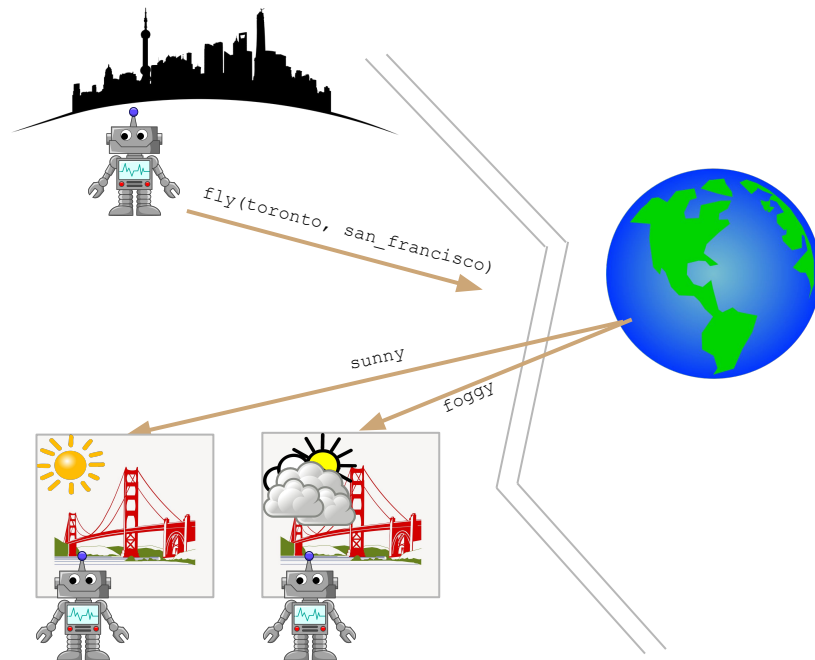
In FOND planning, the agent is given:

- A model of the dynamics of the world.
- The initial state.
- The “goal”.

The **agent** can perform **actions**:

- Actions change properties of the world state.
- Action **effects** are **non-deterministic** (i.e., the effect can be one among many).

Solutions are **policies**, or agent strategies to achieve the goal, regardless of the non-determinism.



Environment Fairness in FOND Planning

FOND planning usually presumes that the environment is “fair”.

Fairness: all the action effects occur if actions are executed infinitely often.

$$\Phi_{\mathcal{P}}^{fair} \stackrel{\text{def}}{=} \bigwedge_{o \in O, e \in Eff_o} (GFo \rightarrow GFe)$$

We consider two types of solutions to FOND planning:

- **Strong solutions** are robust to all the environment non-determinism.
- **Strong-cyclic solutions** presume that the environment is fair.

Specifying FOND planning problems



How can we specify problems?

- Usually, in PDDL or SAS⁺.

PDDL describes actions¹ in terms of:

- Action preconditions.
- Action effects (non-deterministic).

Hey! where is **fairness**?

- Fairness needs not be specified!

```
(:action fly
  :parameters (?orig, ?dest)
  :precondition
    (agent-at ?orig)
  :effect
    (and
      (not (agent-at ?orig))
      (agent-at ?dest)
      (oneof
        (weather-at ?dest ?sunny)
        (weather-at ?dest ?foggy)
      )
    )
)
```

Non-deterministic action pick-up in PDDL.

[1] Note, actions have Markovian preconditions and effects.

I Will Talk About...

How can we specify problems?

- FOND planning
 - Env. Fairness
- Reactive synthesis

How can we specify goals in planning?

- Final-state condition
- Temporal logics
 - on finite traces
 - on infinite traces

How can we solve planning problems?

- Automata goal representations
- FOND planners as a tool

Complexity results (Theory and Practice)

Planning for Temporally Extended Goals

In Planning, goals are typically **final-state** conditions.

- There is an implicit condition that **plans terminate**.
- Existing FOND planning tools are very optimized, but **can only handle final-state goals**.

Is Planning for final-state conditions enough?

- **No!** In many real-world situations we have to deal with **safety, liveness**, and other **temporally extended** properties that refer to the **whole trajectory of visited states**.

Planning for temporally extended goals in *deterministic* domains dates back from the 90s¹. In the last decade, we have studied it in-depth for FOND domains.

[1] Fahiem Bacchus, Froduald Kabanza. *Planning for Temporally Extended Goals*. AAAI/IAAI, Vol. 2 1996.

Goals and Specifications in Linear Temporal Logic

Syntax of Linear Temporal Logic:

Atomic propositions (can be world features)

Logical connectives (\wedge , \vee , \neg)

Basic Temporal modalities:

- Next (\circ)
- Until (\mathcal{U})

Other temporal modalities:

- Always (\Box), Eventually (\Diamond), Release (\mathcal{R}),
Weak-Until (\mathcal{W}), Weak-Next (\bullet), ...

Semantics of Linear Temporal Logic:

- **LTL**: evaluated over infinite-length traces
- **LTLf**: evaluated over finite-length traces

Examples:

In English	"eat until you are not hungry"
LTLf formula	eat $\mathcal{U} \neg$ hungry
Trace	{eat, hungry}, {eat, hungry}, {}

In English	"after each request there is some response"
LTLf formula	$\Box (\text{req} \rightarrow \circ \Diamond \text{resp})$
Trace	{req}, {}, {}, {resp}, {}, {req}, {}, {req}, {resp}

In English	"the lights must be turned on if it is dark outside, except when nobody is inside the room"
LTLf formula	$\Box (\circ \text{lights_on} \leftrightarrow \text{nobody} \wedge \text{dark})$
Trace	{nobody, dark}, {nobody, lights_on}, {}, {}, ...

A variety of Temporal Logics to Choose From

Temporally extended goals can refer to **finite-** or **infinite-length** trajectories.

Languages used to specify goals:

- For **non-terminating** programs: LTL
- For **terminating** programs: f-LTL, LTLf, LDLf, Past LTL, f-LTL-RE, PDDL3 temporal operators, ...

In general, there is no temporal language that is “better than” another in all scenarios.
For example, LDLf is more expressive than LTLf, but LTLf is simpler and easier to interpret.

Can we specify FOND domains with temporal logics as well?

Yes, but...

Let me first introduce the model for reactive synthesis.

I Will Talk About...

How can we specify problems?

- FOND planning
 - Env. Fairness
- **Reactive synthesis**

How can we specify goals in planning?

- Final-state condition
- **Temporal logics**
 - on finite traces
 - on infinite traces

How can we solve planning problems?

- Automata goal representations
- FOND planners as a tool

Complexity results (Theory and Practice)

Reactive Synthesis

Specification: a tuple $\langle X, Y, \varphi \rangle$

X : a set of environment variables

Y : a set of agent variables

φ : a temporally extended **formula** over $X \cup Y$

Problems
are usually
specified in
TLSF.

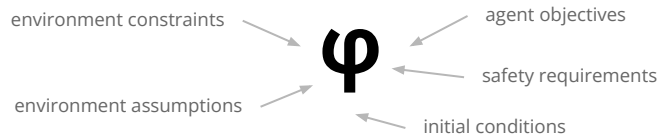
Solutions:

An **agent strategy** $\sigma: (X \cup Y)^* \rightarrow 2^Y$

so that all the generated plays satisfy the specification formula,

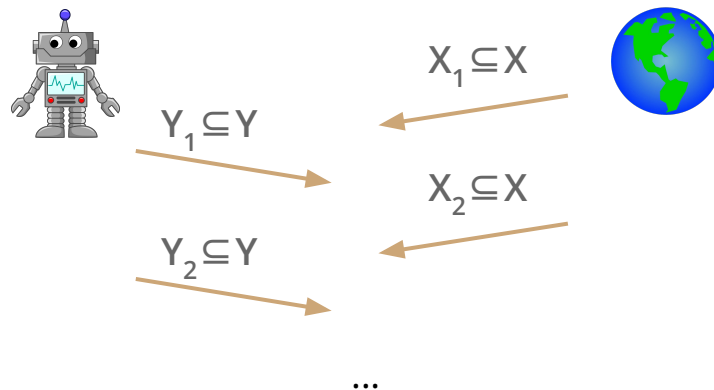
φ , regardless how the environment player moves.

Everything is embedded in a monolithic specification formula, φ .



Synthesis problem: computing a solution.

Realizability problem: determine whether a solution exists.



A **play** is a sequence $(X_1 \cup Y_1), (X_2 \cup Y_2), \dots$

Can we specify FOND domains with temporal logics as well?

Yes, but it is **not practical**.

Encoding the action dynamics:

$$\psi_a^{\text{eff}} := a \rightarrow \bigvee_{e \in \text{Eff}_a} \left(\bigwedge_{f \in \text{Add}_e} \circ f \wedge \bigwedge_{f \in \text{Del}_e} \circ \neg f \wedge \bigwedge_{f \in F \setminus \text{Add}_e \cup \text{Del}_e} f \leftrightarrow \circ f \right)$$

$$\psi_e := \bigwedge_{a \in \mathcal{A}} \psi_a^{\text{eff}} \quad \theta_s := \left(\varphi_{\mathcal{A}}^{\text{one}} \wedge \bigwedge_{a \in \mathcal{A}} \left(\neg a \vee \bigwedge_{p \in \text{Pre}_a} p \right) \right) \quad \psi_s := \circ \theta_s$$

Further, encoding FOND planning as LTL synthesis:¹

$$(\theta_e \rightarrow \theta_s) \wedge (\theta_e \rightarrow (\psi_s \mathbf{W} \neg \psi_e)) \wedge ((\Box \psi_e \wedge \theta_e) \rightarrow \varphi_G)$$

	FOND Planning	Reactive Synthesis
Environment variables	F : set of fluents	X : uncontrollable variables
Agent variables	A : set of actions	Y : controllable variables
Initial environment conds	φ_i : initial state	θ_e : INITIALLY formula
Initial agent conditions	–	θ_s : PRESET formula
Agent constraints	Pre : action preconds	ψ_s : ASSERT formula
Environment constraints	Eff : action effects	ψ_e : REQUIRE formula
Agent objectives	φ_G : goal condition	φ_G : GUARANTEE formula
Environment assumptions	fairness	φ_e : ASSUME formula

[1] Alberto Camacho, Meghyn Bienvenu, Sheila A. McIlraith. *Towards a Unified View of AI Planning and Reactive Synthesis*. ICAPS 2019.

Our lesson: Domains in PDDL; Goals in Temporal Logics

Advantages of PDDL planning:

- Domains are more **compact**.
- **Frame axioms** need not be specified explicitly
 - PDDL actions describe world change, and everything else is assumed to stay the same.
- **Fairness** does not need to be specified explicitly.
 - strong-cyclic FOND planners deal with fairness procedurally.
 - Describing fairness in LTL is involved.

$$\Phi_{\mathcal{P}}^{fair} \stackrel{\text{def}}{=} \bigwedge_{o \in O, e \in Eff_o} (GFo \rightarrow GFe)$$

This LTL formula describes fairness.¹
When goals are temporally extended, describing fairness is more tricky.²

[1] Sebastian Sardiña, Nicolás D'Ippolito. *Towards Fully Observable Non-Deterministic Planning as Assumption-based Automatic Synthesis*. IJCAI 2015.

[2] Benjamin Aminof, Giuseppe De Giacomo, Sasha Rubin. *Stochastic Fairness and Language-Theoretic Fairness in Planning in Nondeterministic Domains*. ICAPS 2020.

```
(:action fly
  :parameters (?orig, ?dest)
  :precondition
    (agent-at ?orig)
  :effect
    (and
      (not (agent-at ?orig))
      (agent-at ?dest)
      (oneof
        (weather-at ?dest ?sunny)
        (weather-at ?dest ?foggy)
      )
    )
)
```

Non-deterministic action pick-up in PDDL.

I Will Talk About...

How can we specify problems?

- FOND planning
 - Env. Fairness
- Reactive synthesis

How can we specify goals in planning?

- Final-state condition
- Temporal logics
 - on finite traces
 - on infinite traces

How can we solve planning problems?

- Automata goal representations
- FOND planners as a tool

Complexity results (Theory and Practice)

Review on Finite State Automata

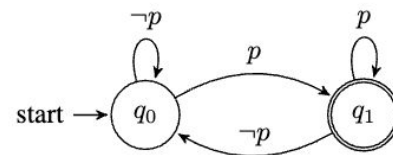
Finite state automata can capture temporally extended properties of finite- and infinite-length traces.

Infinite-word automata:

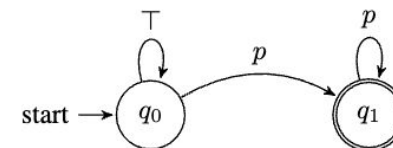
- Non-deterministic Buchi Word (**NBW**) automata
- Deterministic Buchi Word (**DBW**) automata
- Universal Co-Buchi Word (**UCW**) automata

Finite-word automata:

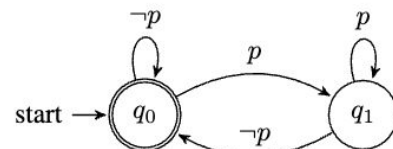
- Non-deterministic Finite Word (**NFW**) automata
- Deterministic Finite Word (**DFW**) automata
- Non-deterministic k-Buchi Word (**NkBW**) automata
- Universal k-Co-Buchi Word (**UkCW**) automata



(a) DFW automaton for LTL formula $FG\ p$

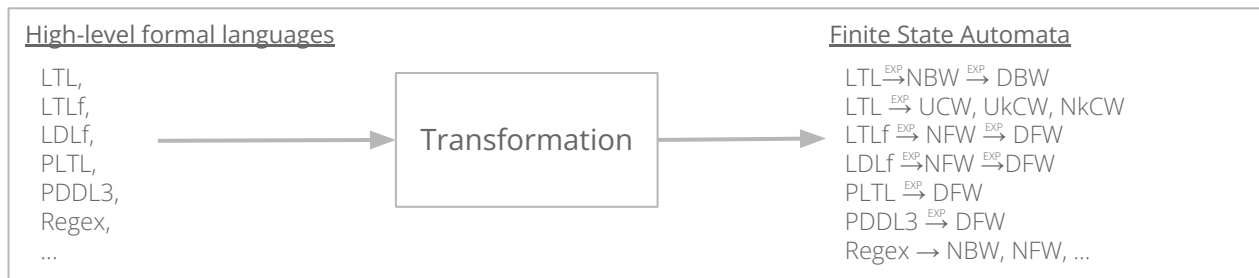


(b) NBW automaton for LTL formula $FG\ p$



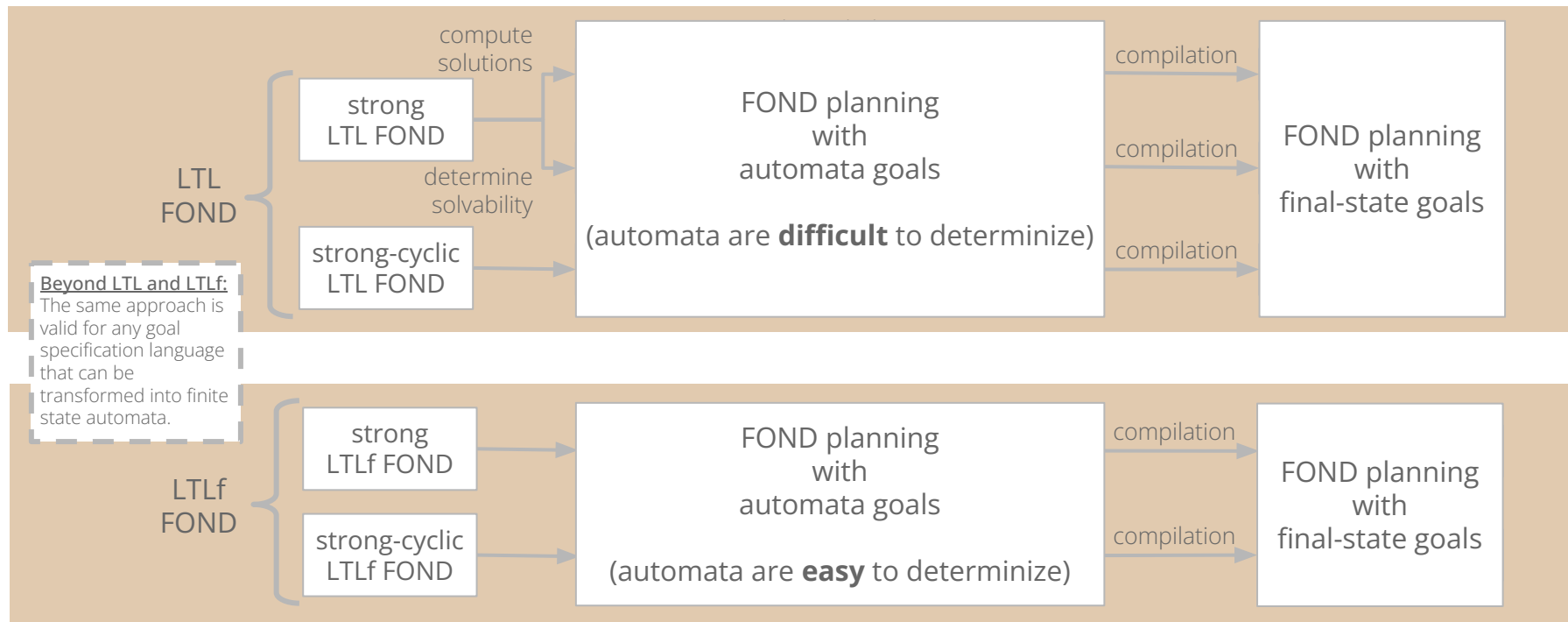
(c) UCW automaton for LTL formula $FG\ p$

Automata Transformations of Temporal Logics

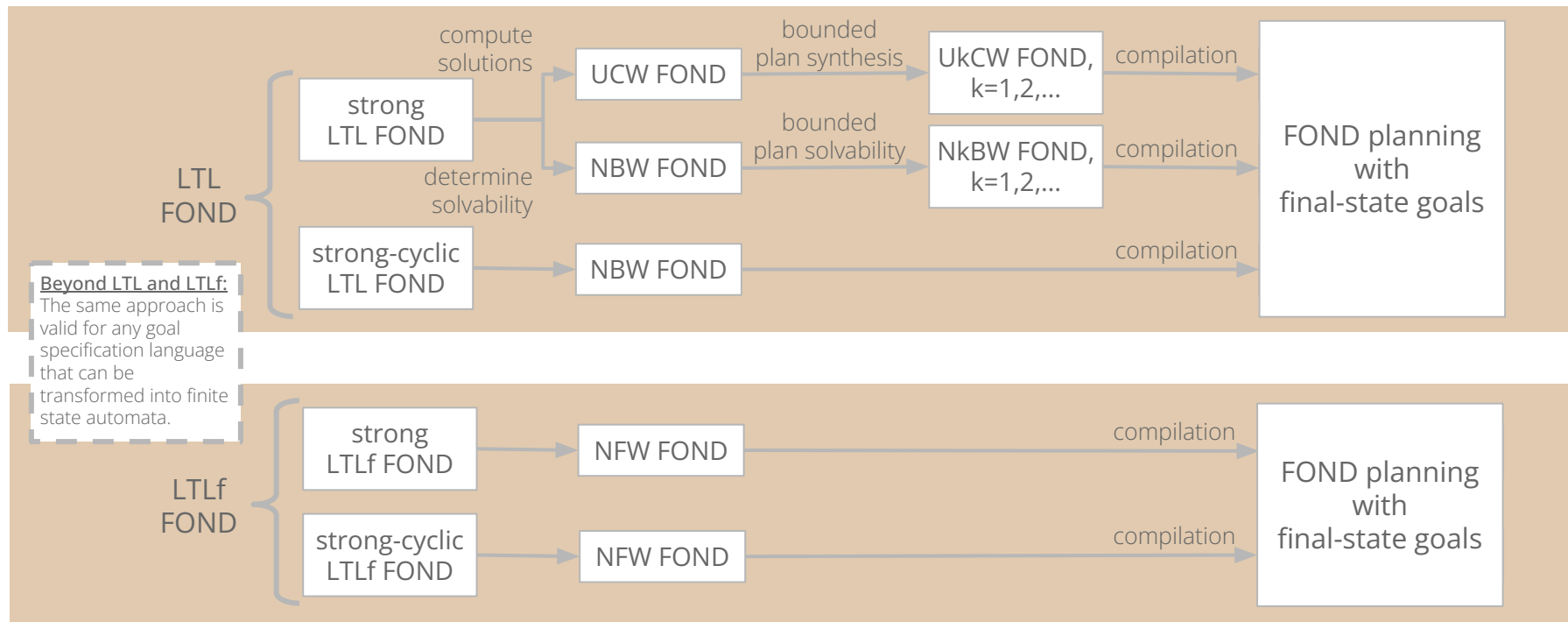


Formula to automata	Time and size	Reference
LTL to NBW	exp.	Vardi and Wolper. In Inf. and Comp. 1994
LTL to UCW	exp.	Kupferman and Vardi. In FOCS 2005
LTL _f to NFW	exp.	Baier and McIlraith. In ICAPS 2006
LTL _f to DFW	2 exp.	De Giacomo and Vardi. In IJCAI 2013
PLTL to DFW	exp.	Zhu, Pu, and Vardi. In TAMC 2019
PDDL3 to DFW	poly	Gerevini and Long, In Technical Report 2005

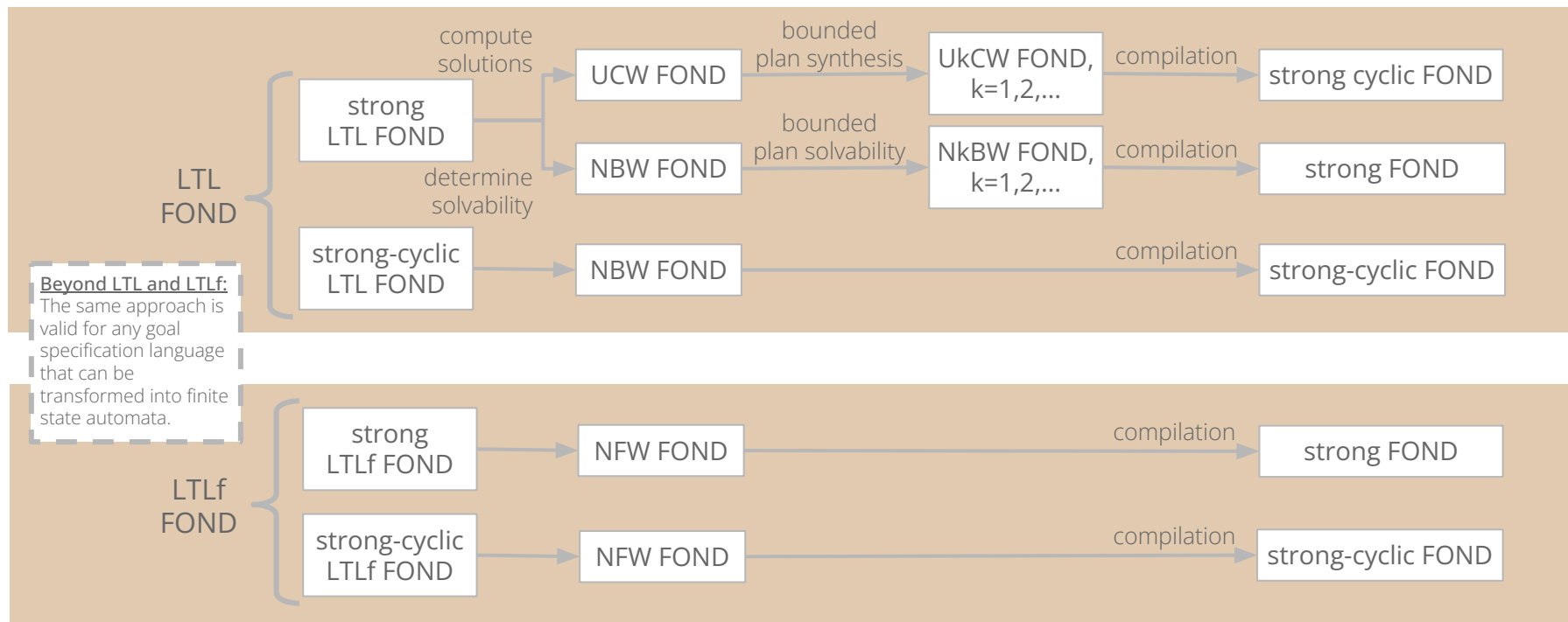
Compilations to FOND planning with final-state goals



Compilations to FOND planning with final-state goals



Compilations to FOND planning with final-state goals



Advantages of our Algorithmic Approach

Automata-based methods can handle a diversity of goal specifications that can be transformed into automata (LTLf, LDLf, ...).

Compilation-based methods enable the use of existing highly-optimized planners (which are only capable of handling final-state goals) as a tool for the broader class of temporally extended goals.

Goal specification	Solution	Reference
UCW, NFW, DFW	strong	Camacho and McIlraith. In IJCAI 2019
NFW and DFW	stochastic-fair	Camacho et al. In AAAI 2017
DBW	stochastic-fair	Patrizi, Lipovetzky, and Geffner. In IJCAI 2013
NBW	stochastic-fair	Camacho et al. In AAAI 2017

Summary of compilation-based approaches to FOND planning with temporally extended goals. They can take automata goal representations.

I Will Talk About...

How can we specify problems?

- FOND planning
 - Env. Fairness
- Reactive synthesis

How can we specify goals in planning?

- Final-state condition
- Temporal logics
 - on finite traces
 - on infinite traces

How can we solve planning problems?

- Automata goal representations
- FOND planners as a tool

Complexity results (Theory and Practice)

Complexity results

We know very well the complexity of FOND planning with temporally extended goals.

In summary:

- **domain complexity:** 1EXP-complete
- **goal complexity:** it depends on the goal representation:
 - 2EXP-complete for LTL_f
 - 1EXP-complete for PLTL
 - it is tied to the worst-case explosion of automata goal transformations.
- **fairness** does not influence complexity.

Goal spec.	Solution	Complexity	Reference
NFW, DFW	strong	EXP-c	Camacho. In PhD thesis 2022
LTL _f /LDL _f	strong	2EXP-c	De Giacomo and Rubin. In IJCAI 2018
PLTL	strong	EXP-c	De Giacomo et al. In IJCAI 2020
UCW	strong	EXP-c	Camacho. In PhD thesis 2022
LTL	strong	2EXP-c	Camacho et al. In ICAPS 2019 Aminof et al. In ICAPS 2020
NFW, DFW	strong-cyclic	EXP-c	Camacho. In PhD thesis 2022
LTL _f /LDL _f	strong-cyclic	2EXP-c	Aminof et al. In ICAPS 2020
NBW	strong-cyclic	EXP-c	Camacho. In PhD thesis 2022
LTL	strong-cyclic	2EXP-c	Camacho. In PhD thesis 2022

Experimental Results

Lessons learned:¹

- **Stochastic fair planning is easier** than strong planning, in practice.
- **Terminating plans are easier** to compute than non-terminating plans, in practice
- Worst-case complexity does not manifest, in practice (MONA's LTL2DFA tool is very effective [Zhu et al., 2018]).

problem	Strong Solutions to LTL _f FOND		Stochastic-Fair Solutions to LTL _f FOND	
	policy size	search runtime	policy size	search runtime
clerk-p1	N/A	0.46	N/A	0.02
clerk-p2	N/A	1.26	N/A	0.08
clerk-p3	N/A	3.62	N/A	0.28
clerk-p4	N/A	12.5	N/A	1.04
clerk-p5	N/A	67.4	N/A	3.12
waldo-p10	50	0.19	18	0.04
waldo-p20	85	0.38	18	0.06
waldo-p30	120	0.61	18	0.08
waldo-p40	155	0.65	14	0.06

Figure 1. Strong LTL_f FOND versus stochastic-fair LTL_f FOND.

problem	Strong Solutions to LTL FOND			Stochastic-Fair Solutions to LTL FOND	
	co-Büchi index	policy size	search runtime	policy size	search runtime
clerk-p1	6	164	0.98	37	0.04
clerk-p2	8	339	1.66	51	0.06
clerk-p3	10	689	18.4	68	0.04
clerk-p4	12	1963	281	88	0.12
clerk-p5	–	–	timeout	111	0.18
waldo-p10	8	250	8.18	51	0.04
waldo-p20	13	477	84.1	71	0.04
waldo-p30	18	687	448	91	0.04
waldo-p40	23	–	timeout	121	0.06
waldo-p100	23	–	timeout	178	0.20

Figure 2. Strong LTL FOND versus stochastic-fair LTL FOND.

¹ <https://bitbucket.org/acamacho/ltlfond2fond>

Summary and Final Thoughts

Temporal Logics can be used to specify:

- Temporally extended goals in planning.
- FOND domains, **although** we prefer using compact PDDL descriptions.
- Fairness assumptions, **although** PDDL planning have them implicit.

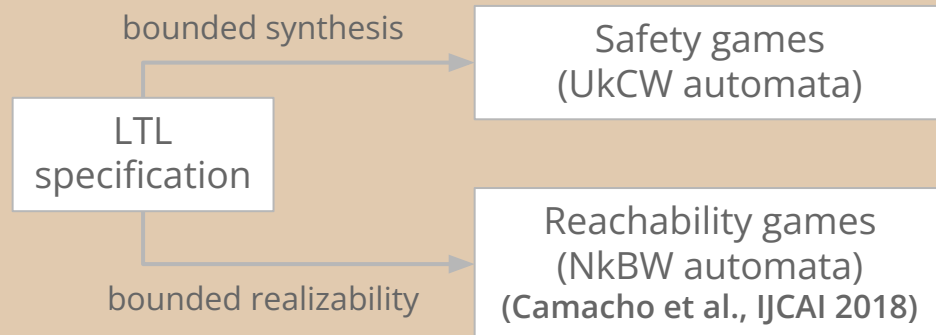
Advantages of Temporal Logics **on Finite Traces** (vs. Infinite Traces):

- Terminating plans are easier to compute than non-terminating plans (the comparison is not apples-to-apples).
- We shall specify goals using Temporal Logics on Finite Traces if programs terminate.

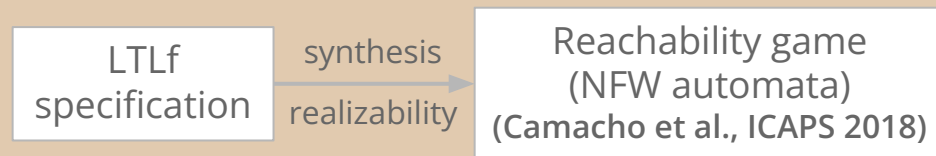
Bonus: Can we also use FOND
planners **as a tool**
for Reactive Synthesis?

Reactive Synthesis *via* Automata Games *via* FOND Planning

Non-terminating programs

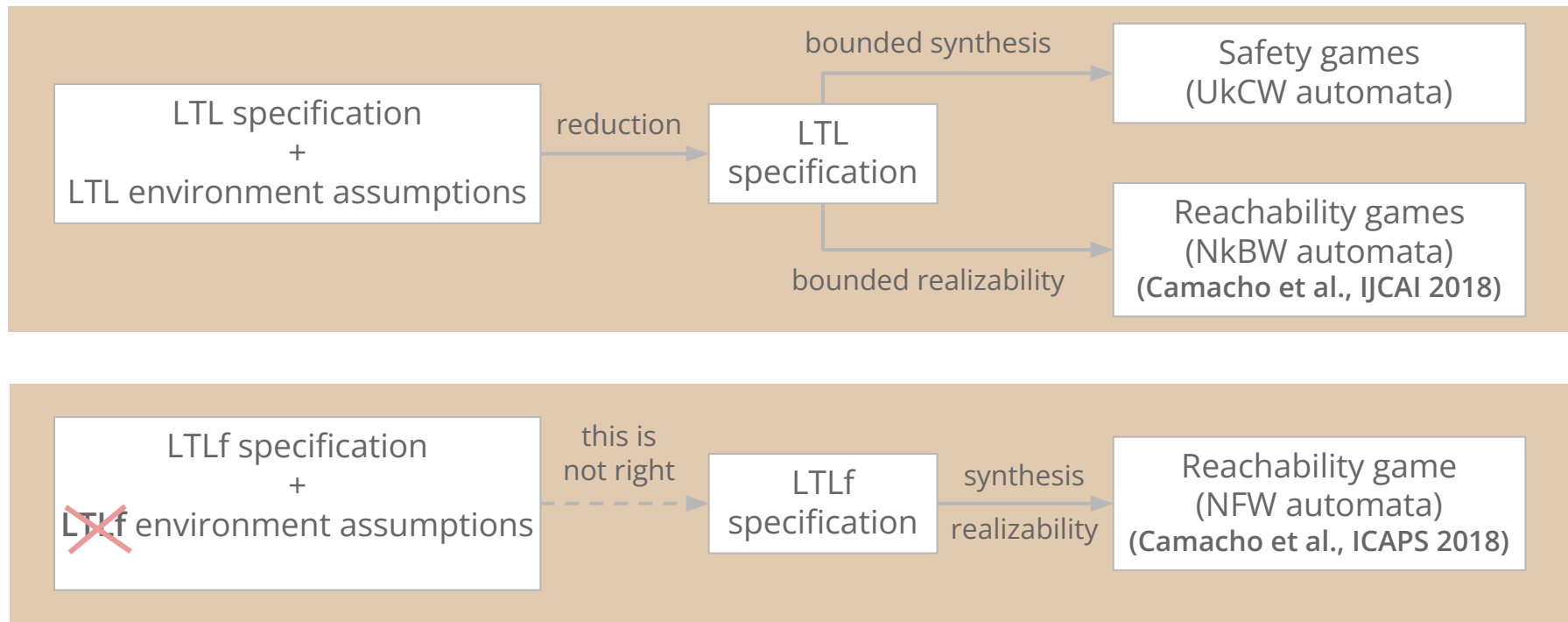


Terminating programs

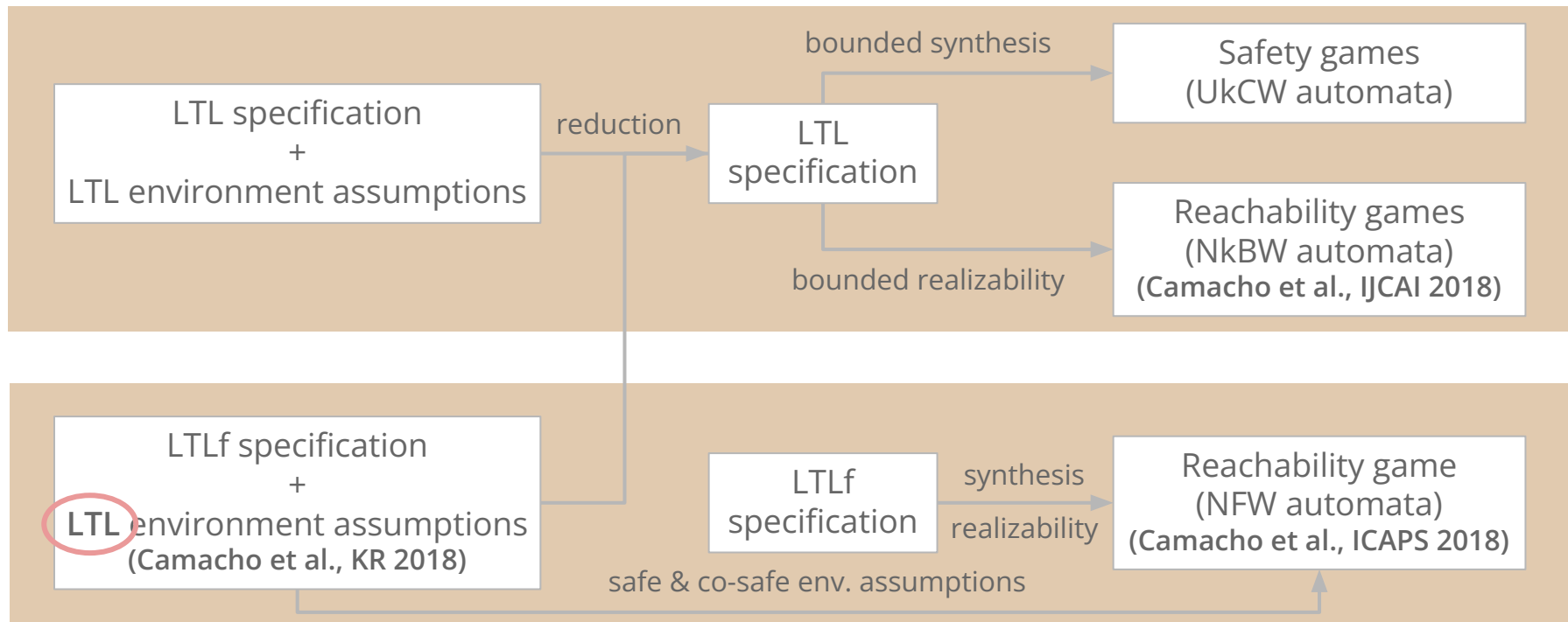


Bonus: What about
Environment Assumptions?

Reactive Synthesis with Environment Assumptions



Reactive Synthesis with Environment Assumptions



Summary and Final Thoughts

Temporal Logics **on Finite Traces** can have a computational advantage.

Environment Assumptions are properties that need to be evaluated **over infinite-length traces**.

It is possible to **decouple** the (finite) LTLf part of specification from the (infinite) LTL assumptions.

- Sometimes, we can stay in the “finite” world (e.g., safe and co-safe env. assumptions).
- That reminds a lot of FOND planning!