Synthesis of Compact Strategies for Coordination Programs

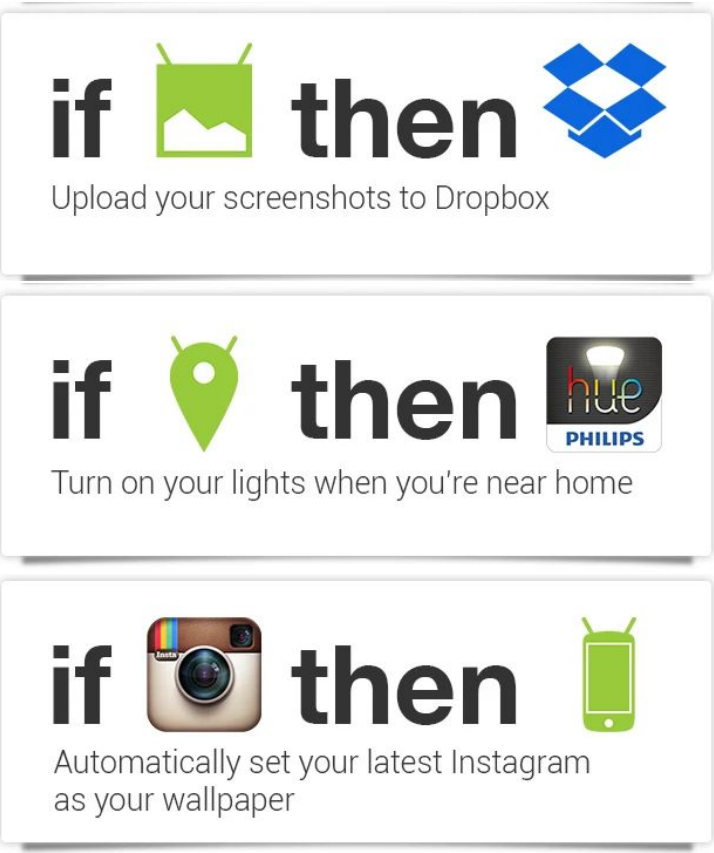
Kedar Namjoshi

Nokia Bell Labs

Nisarg Patel

New York University

Motivation



IFTTT

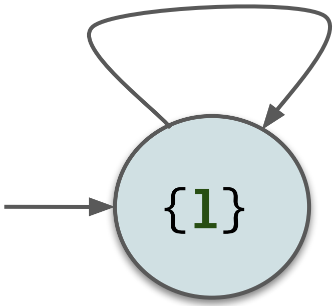
If This Then That

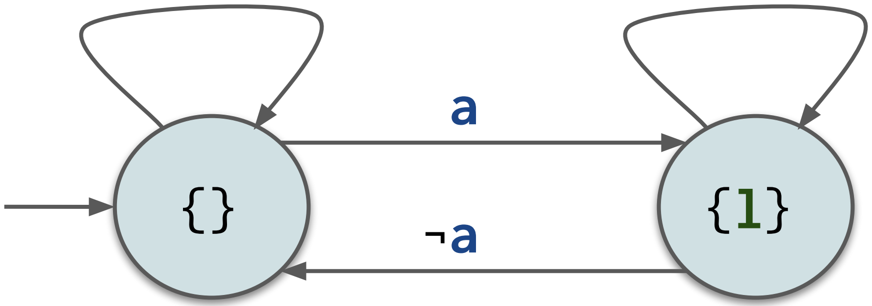
Apple Shortcuts

Motivation

G(at-home ⇒ X light-on)

compact





Multi-robot Setting


```
r1:goto(basement) || r2:goto(basement)
```

Motivation

Unnatural

Non-compositional

```
G(at-home ⇒ X light-on)
```

```
G(at-home ⇒ X light-on) &&
G(!at-home ⇒ X !light-on)
```

```
r1:goto(basement) || r2:goto(basement)
```

```
   r1:goto(basement) && !r2:goto(basement)
|| !r1:goto(basement) && r2:goto(basement)
```

Our contribution

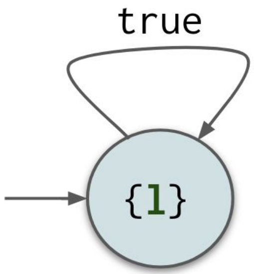
- Bringing attention to compactness, and its formalization.
- Specification transformation ($\mathcal{C}$) to enforce compactness.
  - **Theorem**: φ is compactly realizable iff $\mathcal{C}$(φ) is realizable.
- Prototype tool that offers:
  - **Compact Realizability** of an LTL specification.
  - **Compactness Test** for a model of an LTL specification.

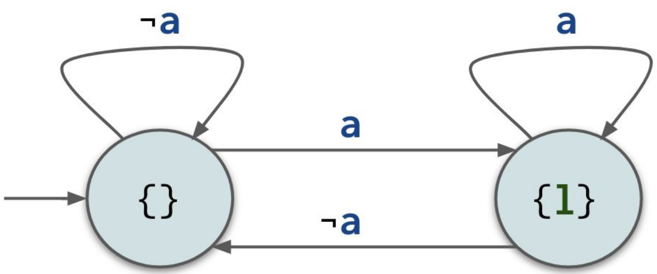
Compactness with Existing Techniques

- **Classical approach**: through connection between programs, strategies and tree automata.

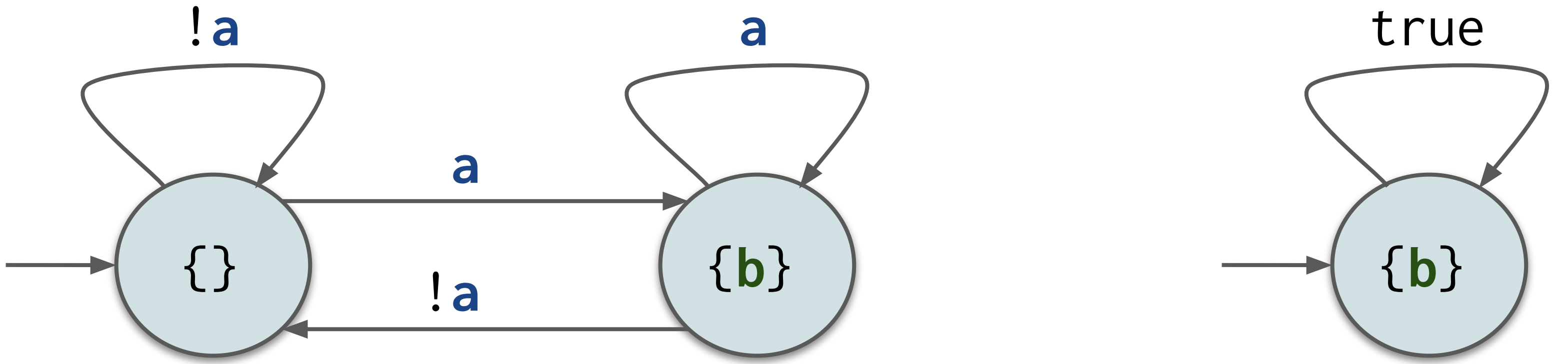
- **Bounded Synthesis**: produces the smallest machine satisfying the specification.

- **Quantitative Synthesis**: Aims to produce a program with minimum worst-case or average-case cost.

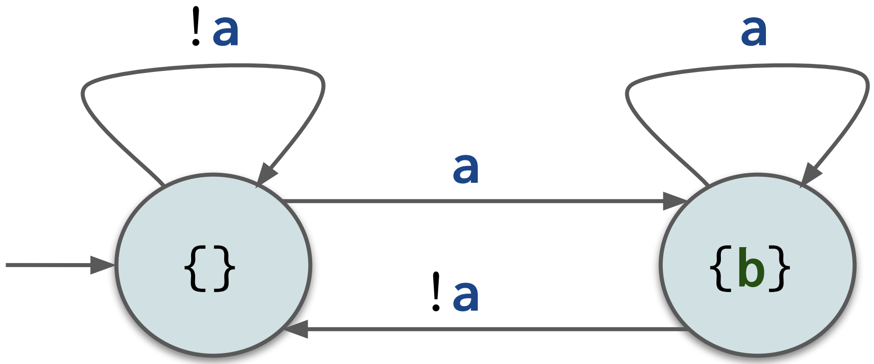


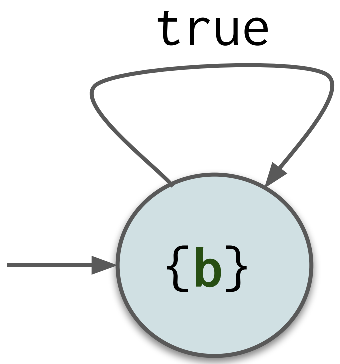


$G(a \Rightarrow X\ b)$



$G(a \Rightarrow X\ b)$





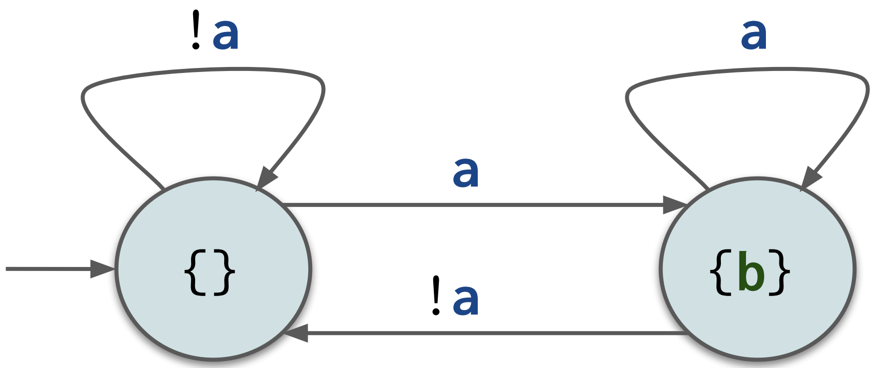
a	!a	a	!a	...
{b}	{}	{b}	{}	

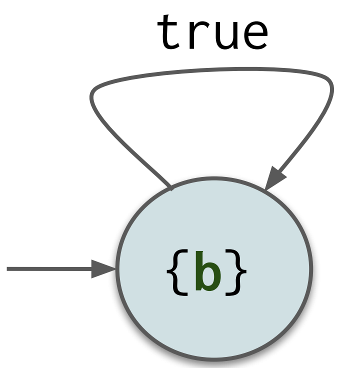
w

a	!a	a	!a	...
{b}	{b}	{b}	{b}	

w’

$$G(a \Rightarrow X\ b)$$





a	!a	a	!a	…
{b}	{}	{b}	{}	

w

$\prec$ "better than"

a	!a	a	!a	…
{b}	{b}	{b}	{b}	

w'

Compactness

- For input sequence $i = i0, i1, \ldots$, output sequence $o = o0, o1, \ldots$,
  An i/o - word $w = (i, o)$.

- $(i, o) \prec (i', o')$ iff $i = i'$ and
  $o < o'$, $<$ is transitive, irreflexive.

Compactness

- For input sequence $i = i0, i1, \ldots$, output sequence $o = o0, o1, \ldots$,
  An i/o - word $w = (i, o)$.

- $(i, o) \prec (i', o')$ iff $i = i'$ and
  $o < o'$, $<$ is transitive, irreflexive.

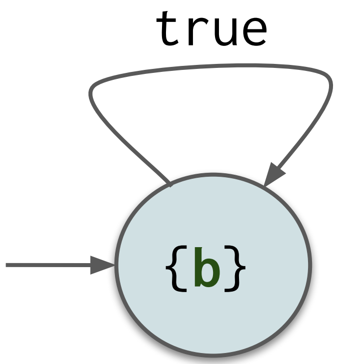
- **Point-wise subset**
- **Point-wise cardinality**
- **#Bit-flips**
- **. . .**

$$G(a \Rightarrow X\ b)$$

P is compact iff for all inputs i,
there is no w ∈ L st. w ≺ (i, P(i)).

$$G(a \Rightarrow X\ b)$$

P is compact iff for all inputs i, there is no $w \in L$ st. $w \prec (i, P(i))$.



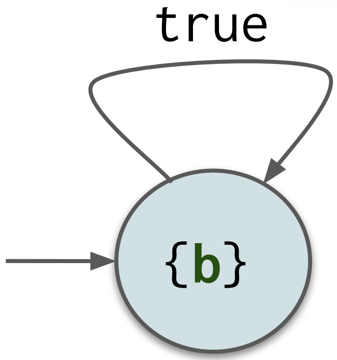
a	!a	a	!a	…
{b}	{b}	{b}	{b}	

w’

$G(a \Rightarrow X\ b)$

✗ **Not compact**

P is compact iff for all inputs i, there is no w ∈ L st. w ≺ (i, P(i)).



a	!a	a	!a	…
{b}	{}	{b}	{}	…

w

≺

a	!a	a	!a	…
{b}	{b}	{b}	{b}	…

w’

Compactness

- For input sequence $i = i0, i1, \ldots$, output sequence $o = o0, o1, \ldots$,
  An i/o - word $w = (i, 0)$.

- $(i, 0) \prec (i', o')$ iff $i = i'$ and
  $o < o'$, $<$ is transitive, irreflexive.

- $\min(L, \prec) = \{ w \mid w \in L$ and not$(\exists\, w'.\ w' \in L$ and $w' \prec w) \}$

Compactness

- For input sequence $i = i0, i1, \ldots$, output sequence $o = o0, o1, \ldots$,
  An i/o - word $w = (i, 0)$.

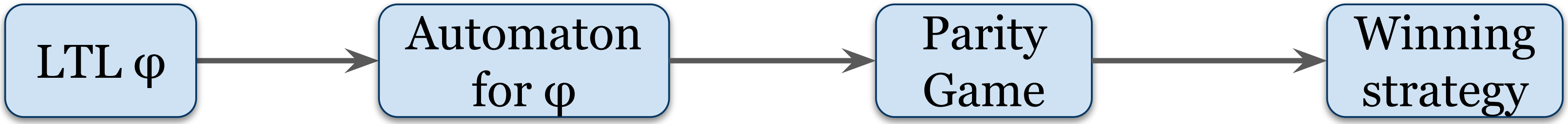
- $(i, 0) \prec (i', o')$ iff $i = i'$ and
  $o < o'$, $<$ is transitive, irreflexive.

- $\min(L, \prec) = \{ w \mid w \in L \text{ and not}(\exists\, w'.\ w' \in L \text{ and } w' \prec w) \}$

Central Theorem: L is compactly realizable iff
$\min(L, \prec)$ is realizable.

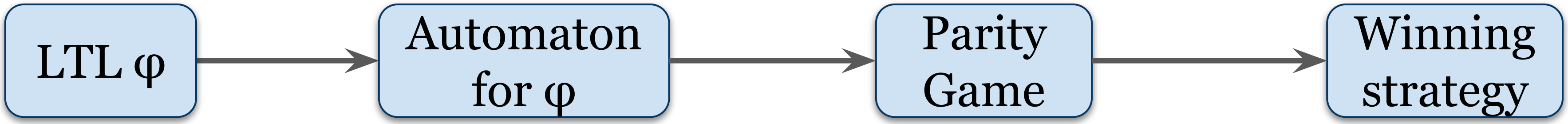
Recipe for a compact program

- Synthesis pipeline:

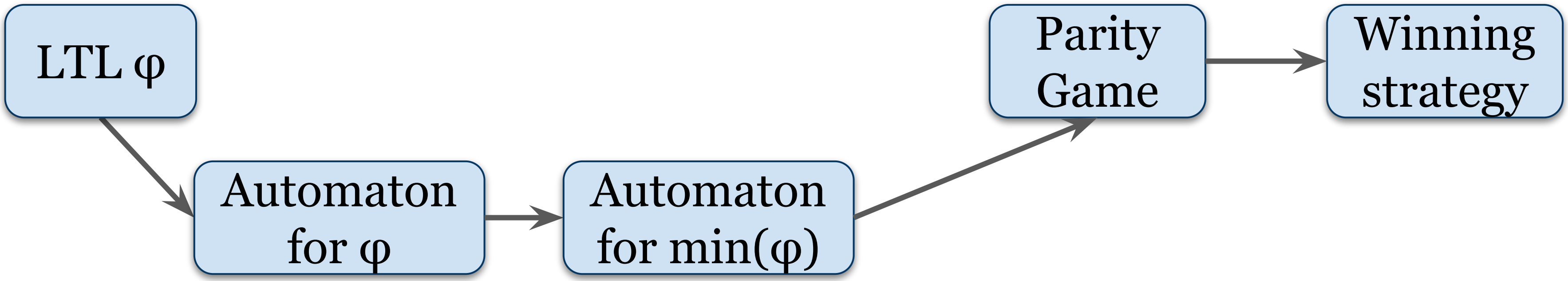


Recipe for a compact program

- Synthesis pipeline:

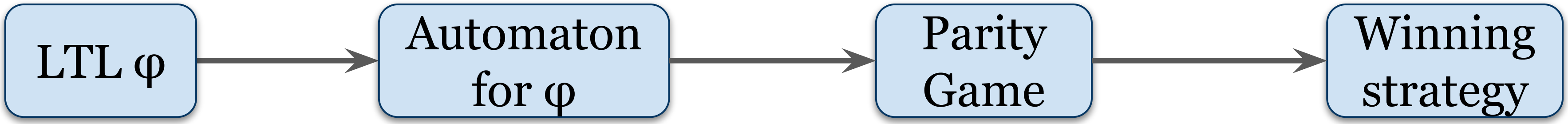


- Compact synthesis pipeline:

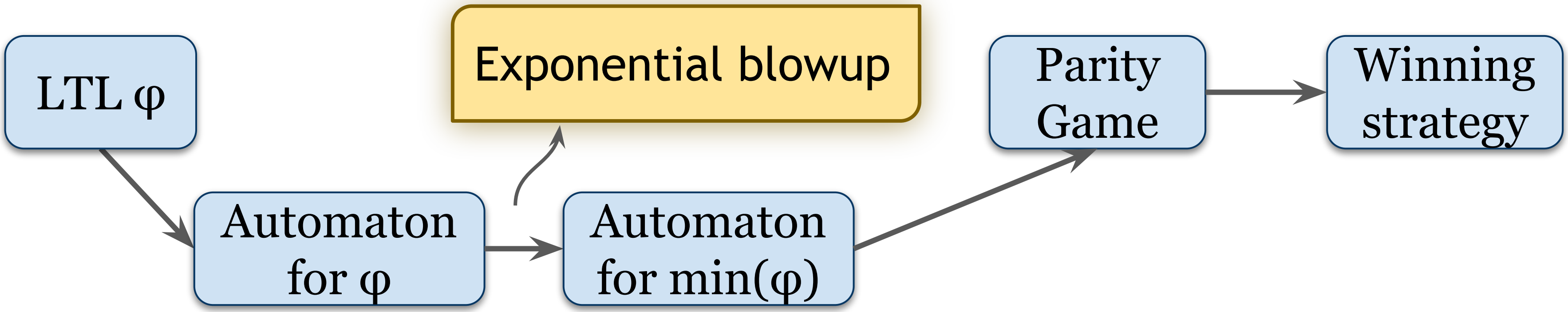


Recipe for a compact program

- Synthesis pipeline:



- Compact synthesis pipeline:

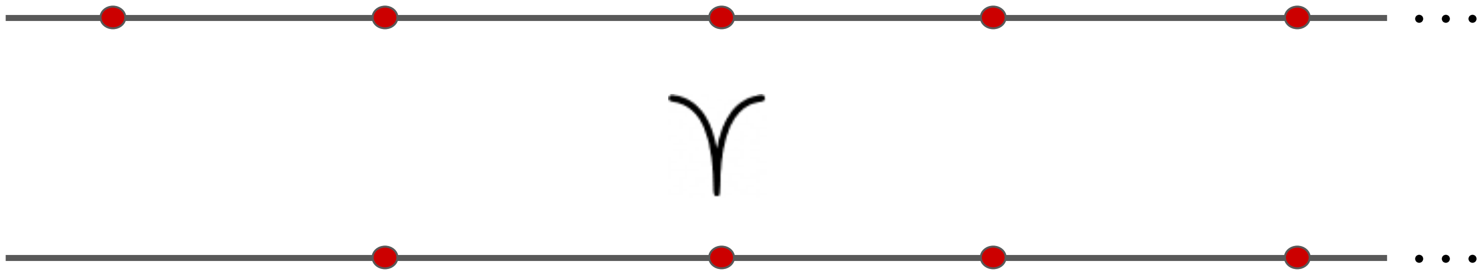


Realizability ⇏ Compact Realizability

- Consider `GF(b)` with pointwise subset ordering.

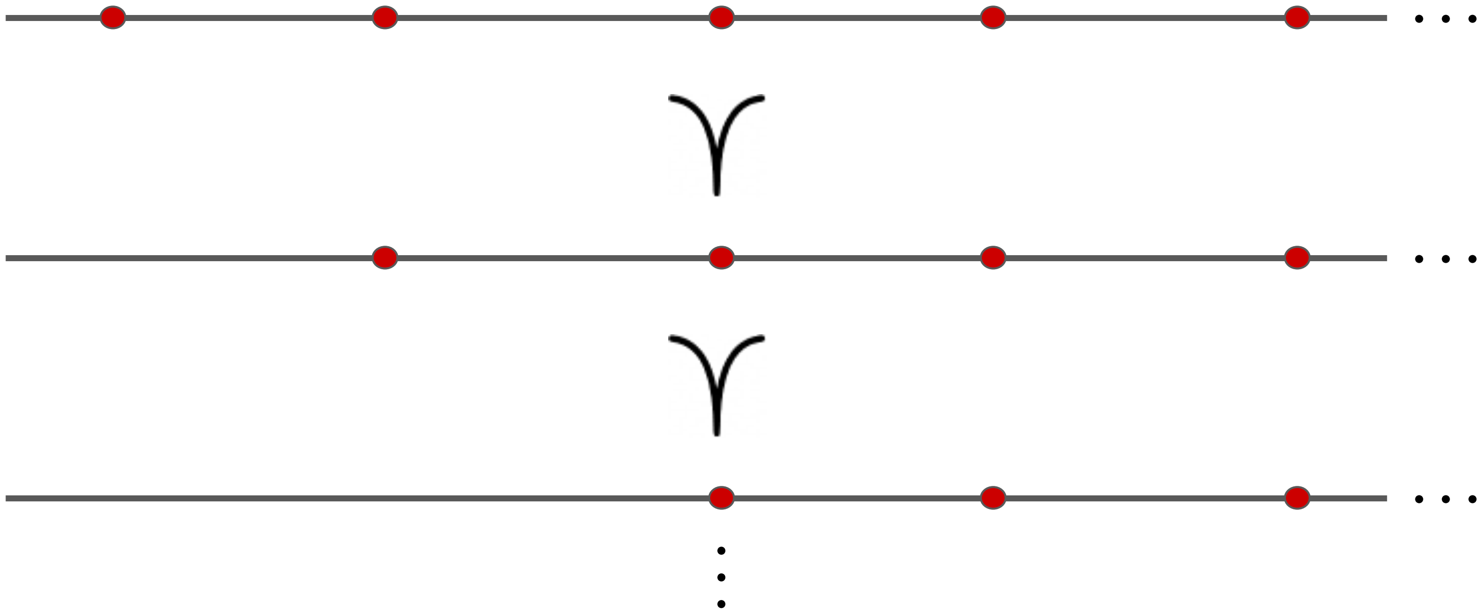
Realizability $\not\Rightarrow$ Compact Realizability

- Consider `GF(b)` with pointwise subset ordering.



Realizability $\not\Rightarrow$ Compact Realizability

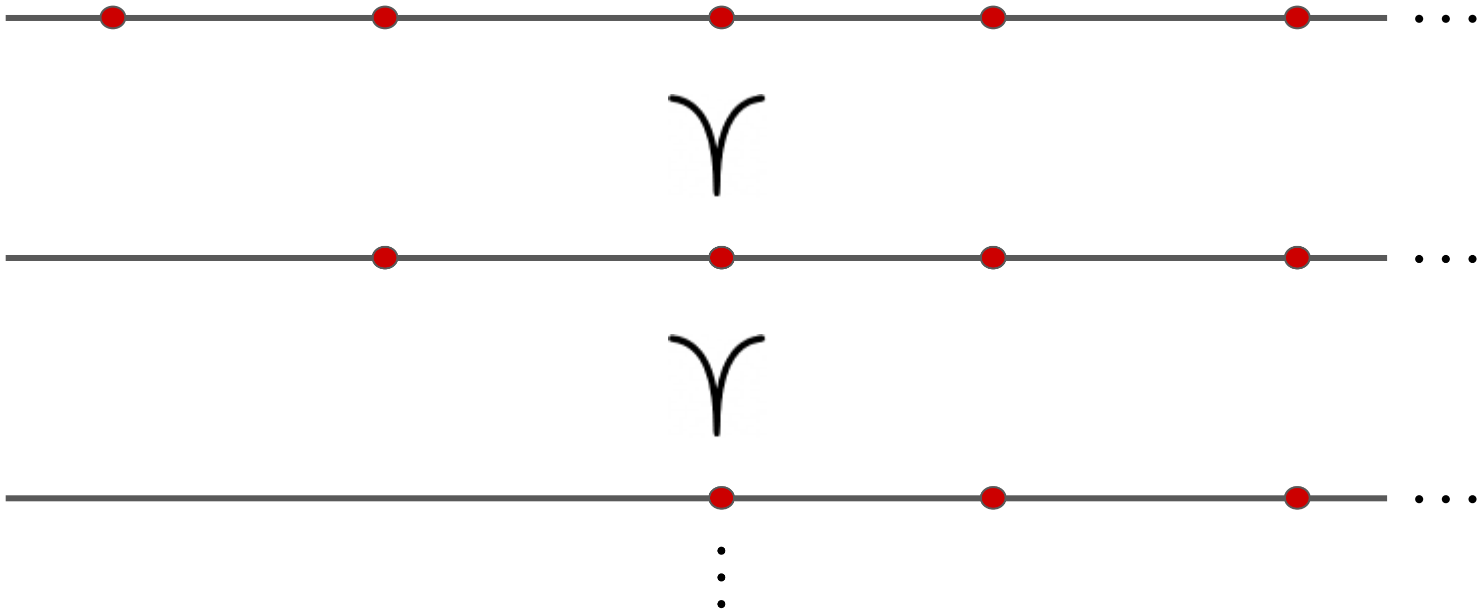
- Consider `GF(b)` with pointwise subset ordering.



Realizability $\not\Rightarrow$ Compact Realizability

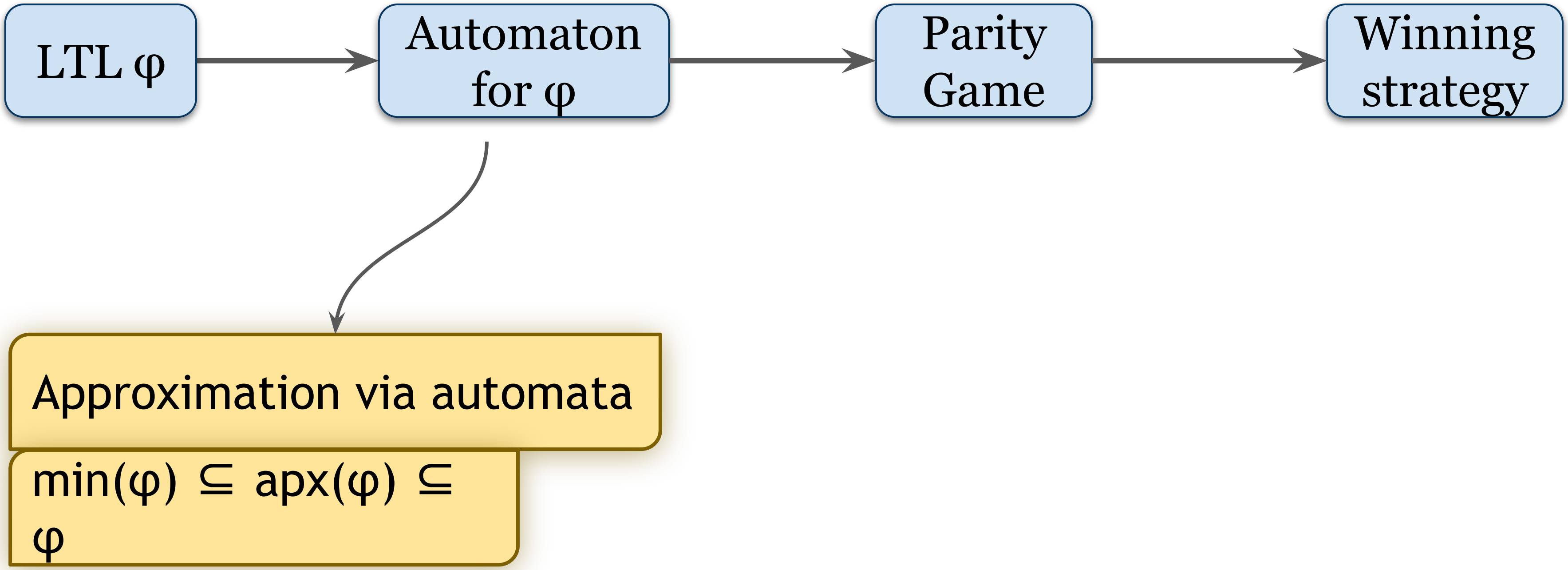
G(**b** ∨ X**b** ∨ XX**b**): compactly realizable

- Consider ~~GF(**b**)~~ with pointwise subset ordering.

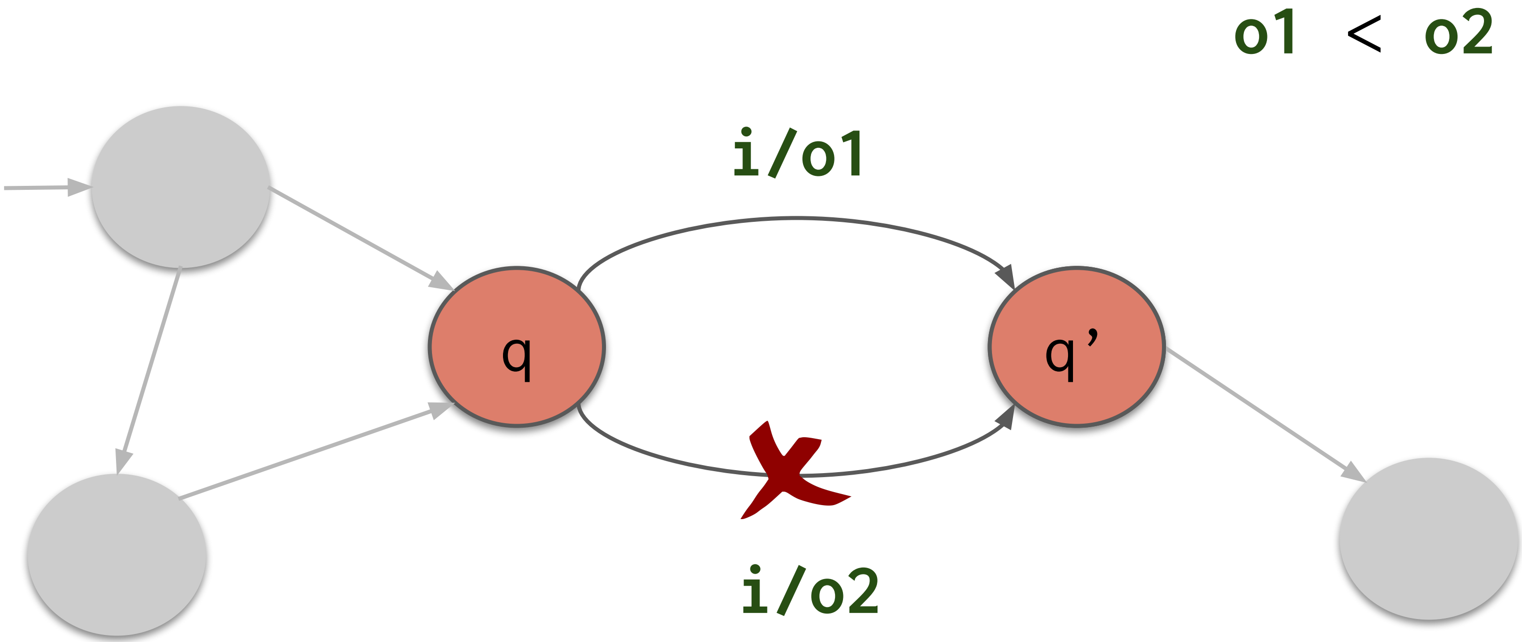


Approximate Compactness (pointwise orderings)

- Synthesis pipeline:

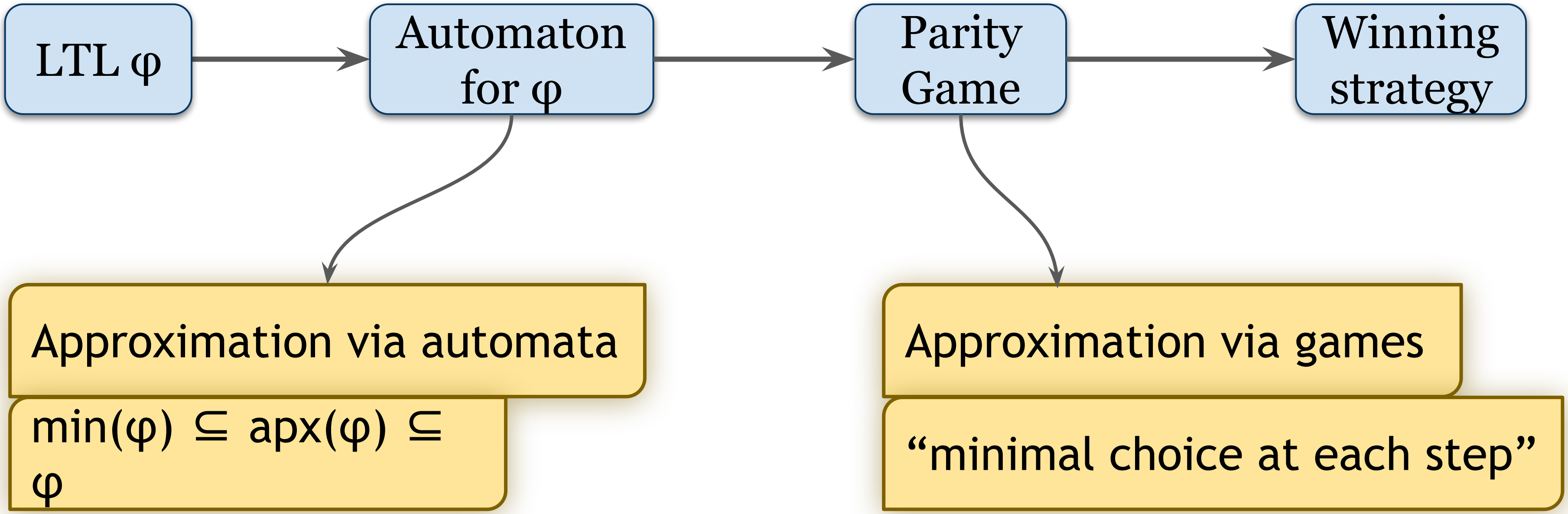


Approximation via automata

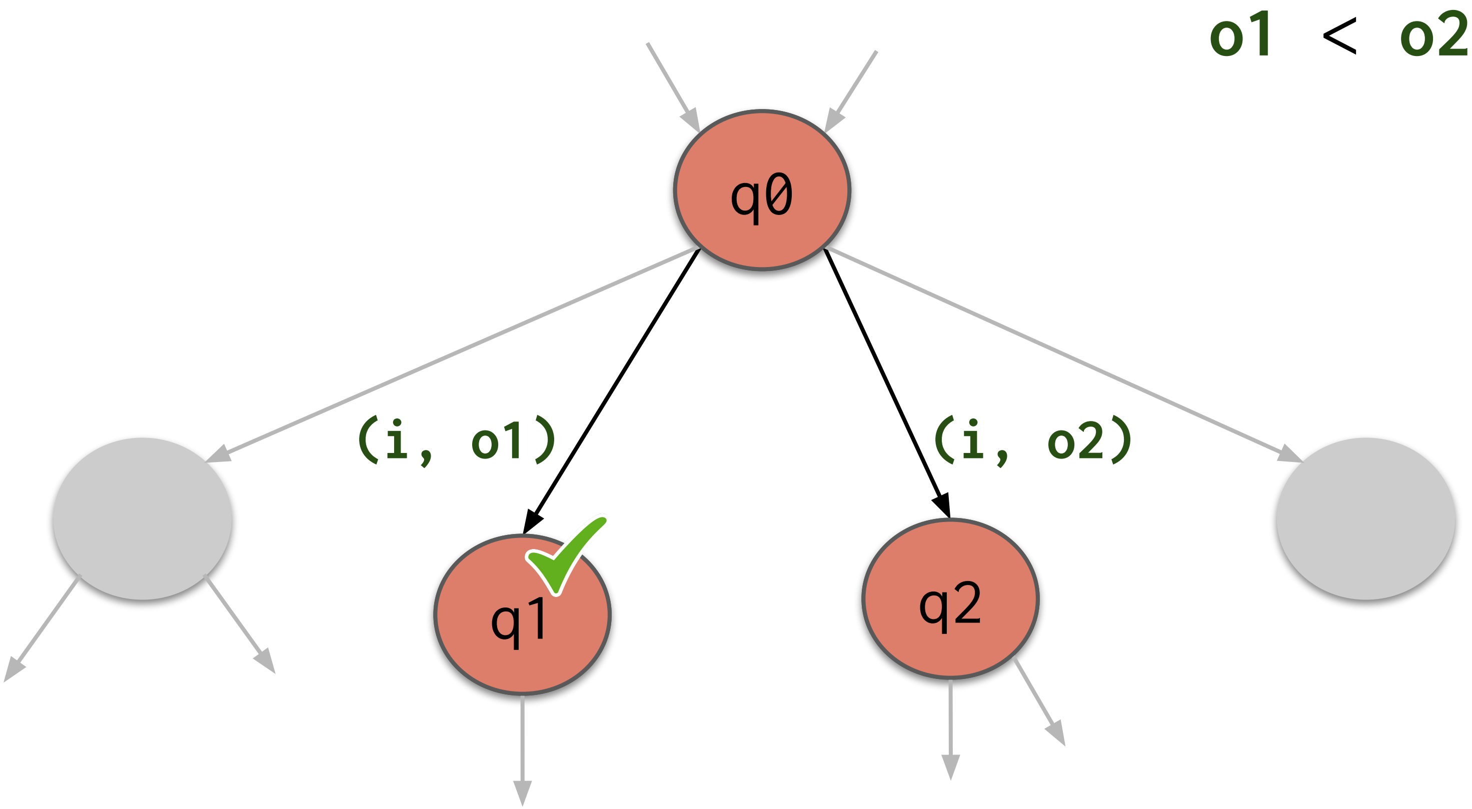


Approximate Compactness (pointwise orderings)

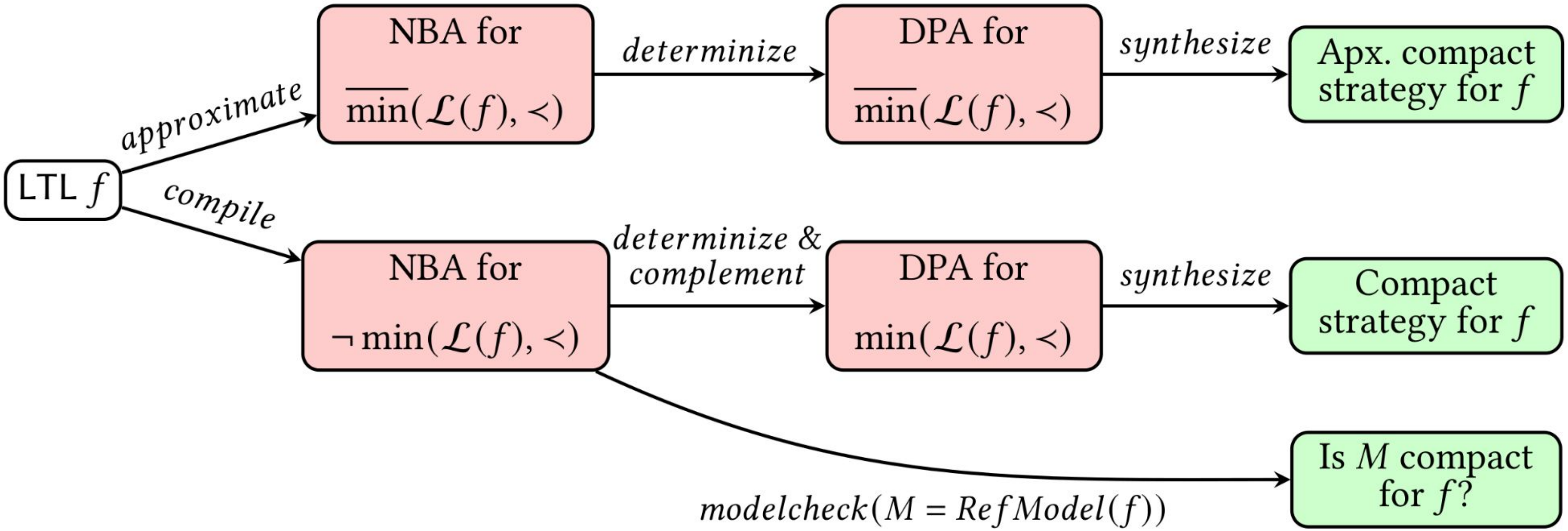
- Synthesis pipeline:



Approximation via games



Prototype tool



Tools: Spot, Owl, Strix, NuSMV

Evaluation

- Evaluated on 246 **realizable** specifications from the SYNTCOMP benchmarks.

- Performance compared to standard synthesis?
  - Within 10 mins:
    - Compact synthesis can solve 50% specifications
    - Standard synthesis can solve 94%.

- Do approximate constructions produce compact strategies?
  - 42% specifications are compact
  - As time-efficient as standard synthesis

Summary

- Desirable to synthesize compact programs; especially where actions have consequences.

- Formalization of compactness parameterized by a preference order.

- Developed notions of “approximate compactness”.

- Prototype tool that offers:
  - Compact Synthesis
  - Approximate Compact Synthesis
  - Compactness Test

Summary

- Desired program: correct + **compact**
  + fault-tolerant + time-efficient + ...

- (?) Relation to the Frame Problem: how to automatically determine scope of an action.
  - Solution to Frame Problem: scope as small as possible
    E.g. Circumscription [McCarthy 1980]
  - Compactness: necessary actions as few as possible.

Thank you!

Backup slides

Compactness vs Avg. case Quantitative Synthesis

$I = \{0\} \quad O = \{a, b\}$

$L = (\{a, b\} \cup \{a\})(0 \cdot \{a, b\})^*$