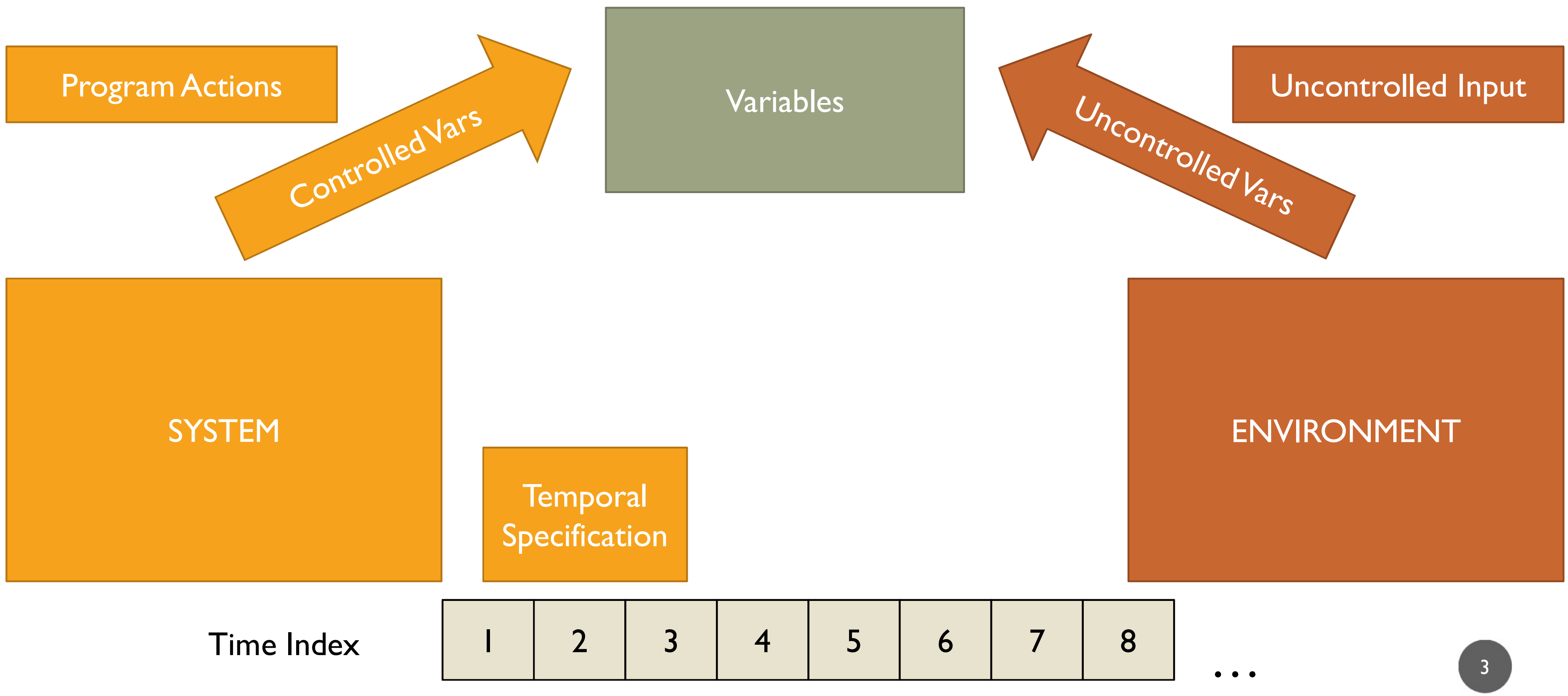
MULTIAGENT SYSTEMS WITH FINITE-HORIZON GOALS

Senthil Rajasekaran
Joint work with Moshe Y. Vardi
Rice University

BACKGROUND

REACTIVE SYSTEMS

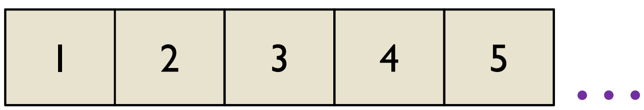


TEMPORAL LOGICS

Temporal logic formulae are often used to model program specifications with a temporal element

Infinite Horizon

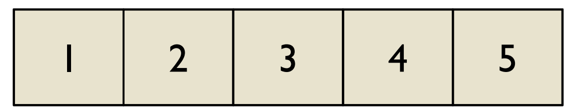
LTL

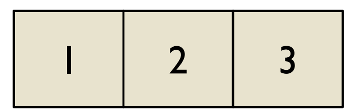


- Represents specifications that must hold over an infinite model of time.
- Very popular in previous literature

Finite Horizon

LTLf

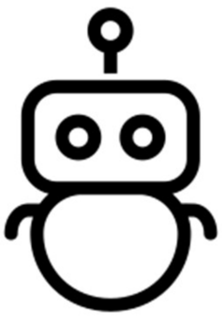




- Represent specifications that hold over a sequence of finite discrete time-steps
- Recently developed but gaining lots of popularity

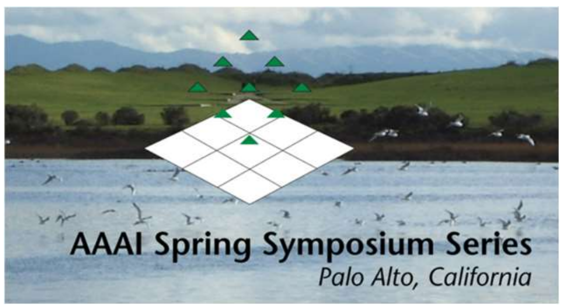
WHY CONSIDER FINITE HORIZONS?

Many tasks are truly finite-horizon in nature.



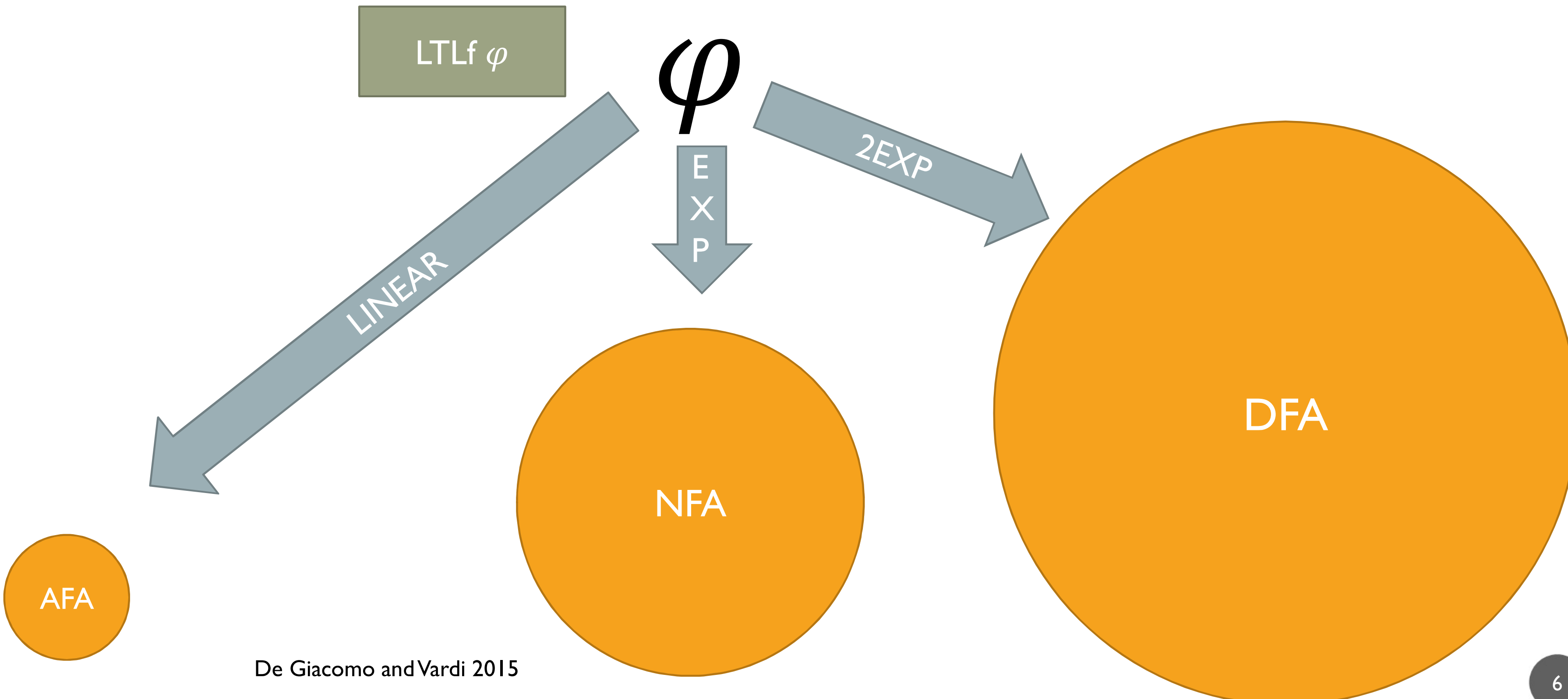
Tasks that involve completion, like "reach the goal" or "reach a final configuration" are more accurately modeled by finite-horizon logic.

Finite-horizon logics are reasoned about through finite-word automata, which are easier to reason about and admit better algorithms than their infinite-word counterparts.



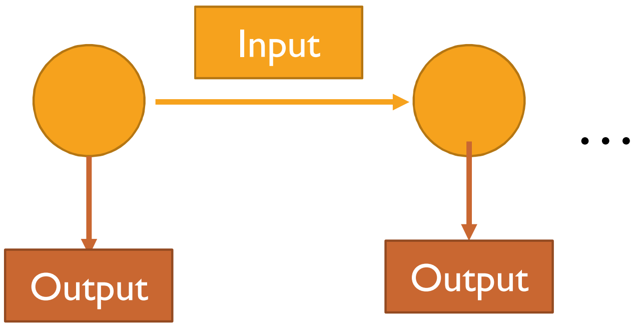
"On the Effectiveness of Temporal Logics on Finite Traces in AI" 2023

AUTOMATON REPRESENTATIONS



De Giacomo and Vardi 2015

STRATEGIES AND WINNING STRATEGIES



A strategy is represented by a finite-state transducer. This is an automaton with an input alphabet of the previously observed actions (the collective settings of the variables from both the system and the environment) that associates each state with an output. For example, a system strategy would have outputs corresponding to settings of the controlled variables.

For reactive systems, the system agent wants to satisfy the temporal specification, while the environment agent wants to ensure it is never satisfied. A winning strategy is a system strategy that enforces the specification regardless of the environment's choice of strategy.

VERIFICATION AND REALIZABILITY

Verification:
Given a strategy for the system, is it winning?

Realizability:
Determine whether the system has a winning strategy or not.

COMPLEXITY RESULTS FOR REACTIVE SYSTEMS

	Verification	**Realizability**
Deterministic Finite Automata	NL-complete (s-t reachability)	PTIME-complete (reachability game)
Nondeterministic Finite Automata	PSPACE-complete (subset constr.)	EXPTIME upper bound PSPACE lower bound
Alternating Finite Automata	EXPSPACE-complete	2EXPTIME-complete

The NFA hardness results come from unpublished reductions from the NFA universality problem. The AFA results are largely inherited from previously known LTLf results since there is a linear transformation between LTLf formulae and AFA.

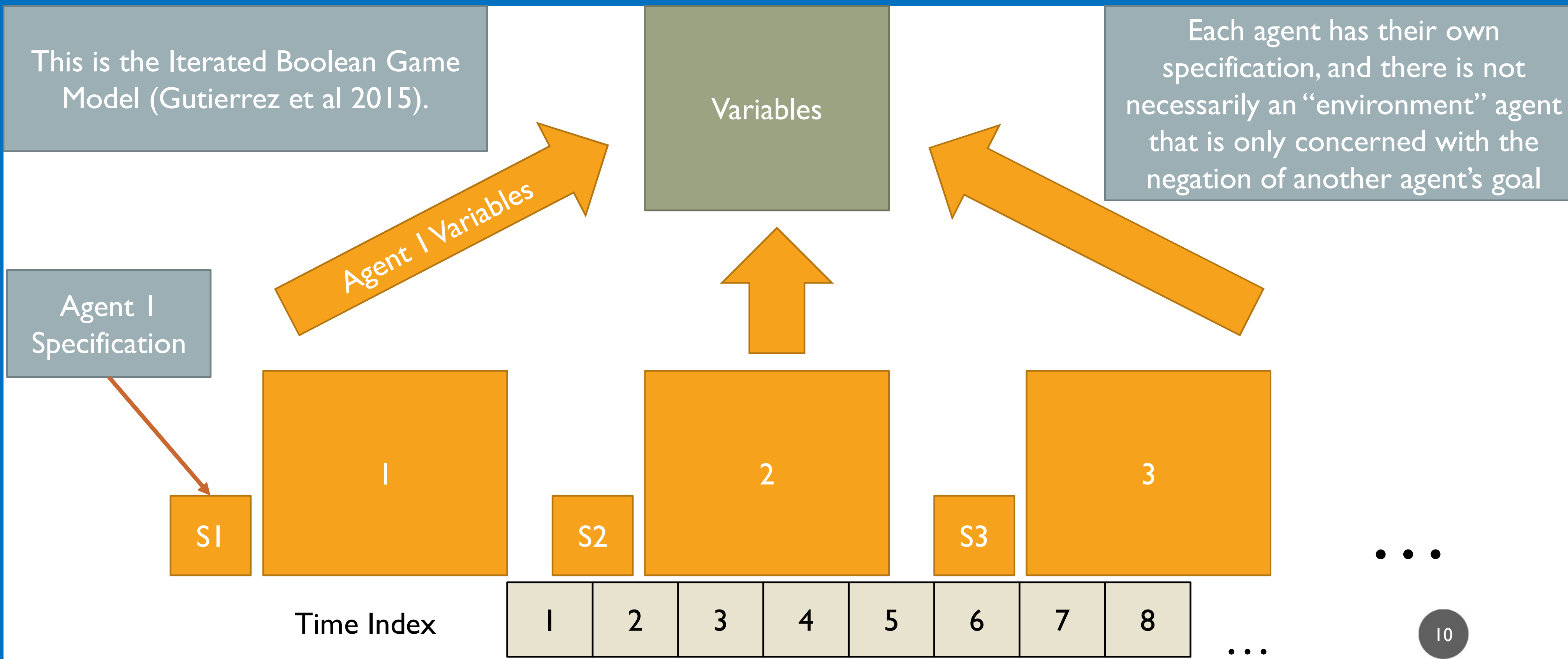
S-T Connectivity is NL-complete – Papadimitriou 1994

Reachability Games are PTIME-complete – Immerman 1981

Complexity of Realizability, Satisfiability of LTLf – De Giacomo and Vardi 2015

Verification of AFA is EXPSPACE-complete – Bansal et al 2023

MULTIAGENT SYSTEMS



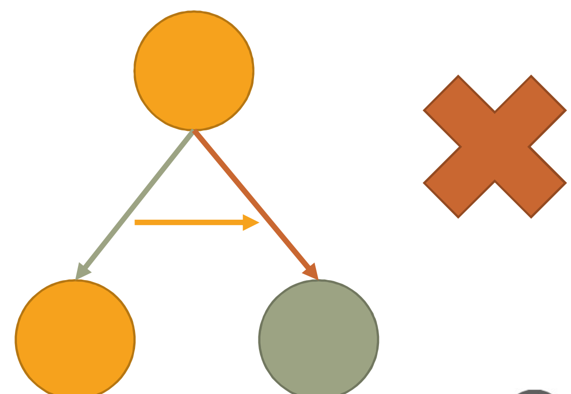
NASH EQUILIBRIUM – MULTIAGENT SOLUTION CONCEPT

A profile of strategies is a Nash Equilibrium if no agent can unilaterally deviate from it to increase their own payoff.

Can an agent that does not have its goal met change its strategy to meet its goal?

Agents that meet their goal on the deterministic execution outlined by the agent strategies have received the highest possible payoff and are therefore not interested in deviation.

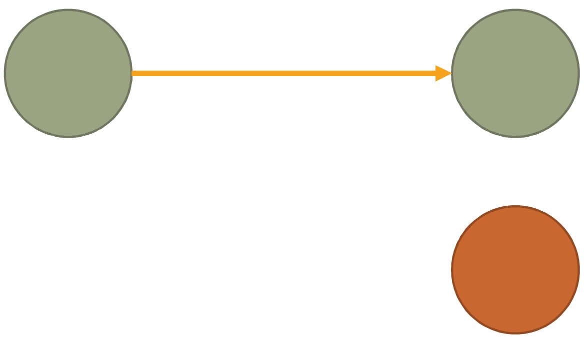
Agents that do not meet their goal when the strategies are followed should not be able to change their strategy in order to create a new profile where their goal is met.



ANALYSIS VIA TWO COMPONENTS

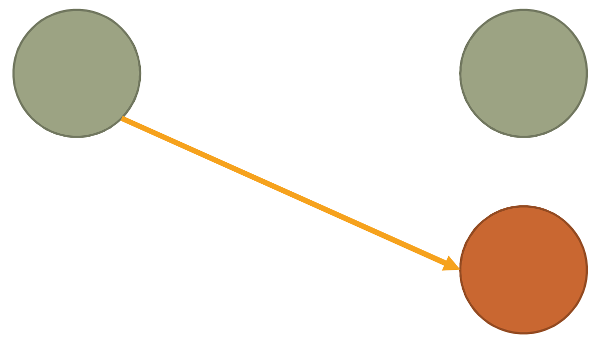
Since we are dealing with the deterministic setting in which goals can either be met or not, we specify the set of agents W that we expect to meet their goal as part of the input – we ask if a profile is a "W-NE" for a subset of agents W.

Primary Trace Condition



If all agents behave according to the strategy assigned, do we get the behavior we expect?

Deviation Condition



If an agent decides to deviate, are they able to improve their own payoff?

VERIFICATION AND REALIZABILITY

Verification:
Given a strategy for all agents, **check** if it's a Nash equilibrium.

Realizability:
Determine whether a Nash equilibrium **exists**.

What drives the complexity of analyzing multiagent systems?

REALIZABILITY RESULTS FOR MULTIAGENT SYSTEMS

	Compleixty of Realizability
Deterministic Finite Automaton	PSPACE-complete
Nondeterministic Finite Automaton	EXPTIME-complete
Alternating Finite Automaton	2EXPTIME-complete

Previous results by Gutierrez et.al. 2017 started with an LDLf formula and showed that the realizability problem belonged to 2EXPTIME. This result was driven by the cost of reasoning about LDLf formulae, i.e., the cost of converting them into doubly exponentially large DFAs. Therefore, the complexity of the strategic reasoning was overpowered by the complexity of the translation.

Results in table due to Rajasekaran and Vardi, 2021, 2022

REALIZABILITY RESULTS FOR MULTIAGENT SYSTEMS

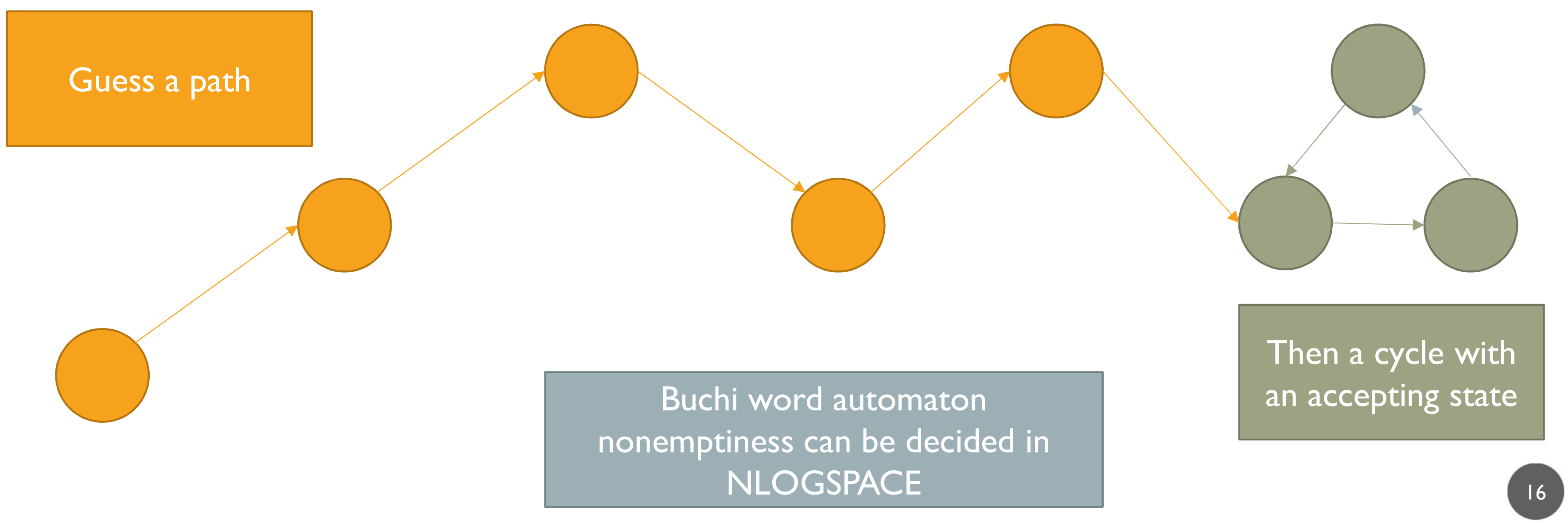
	Compleixty of Realizability
Deterministic Finite Automaton	PSPACE-complete
Nondeterministic Finite Automaton	EXPTIME-complete
Alternating Finite Automaton	2EXPTIME-complete

The EXPTIME-hard result for Nondeterministic Finite Automata was shown for two-agent systems. Therefore, it also provides a solution to the open problem shown in the reactive systems result slide.

Results in table due to Rajasekaran and Vardi, 2021, 2022

DFA REALIZABILITY UPPER BOUND

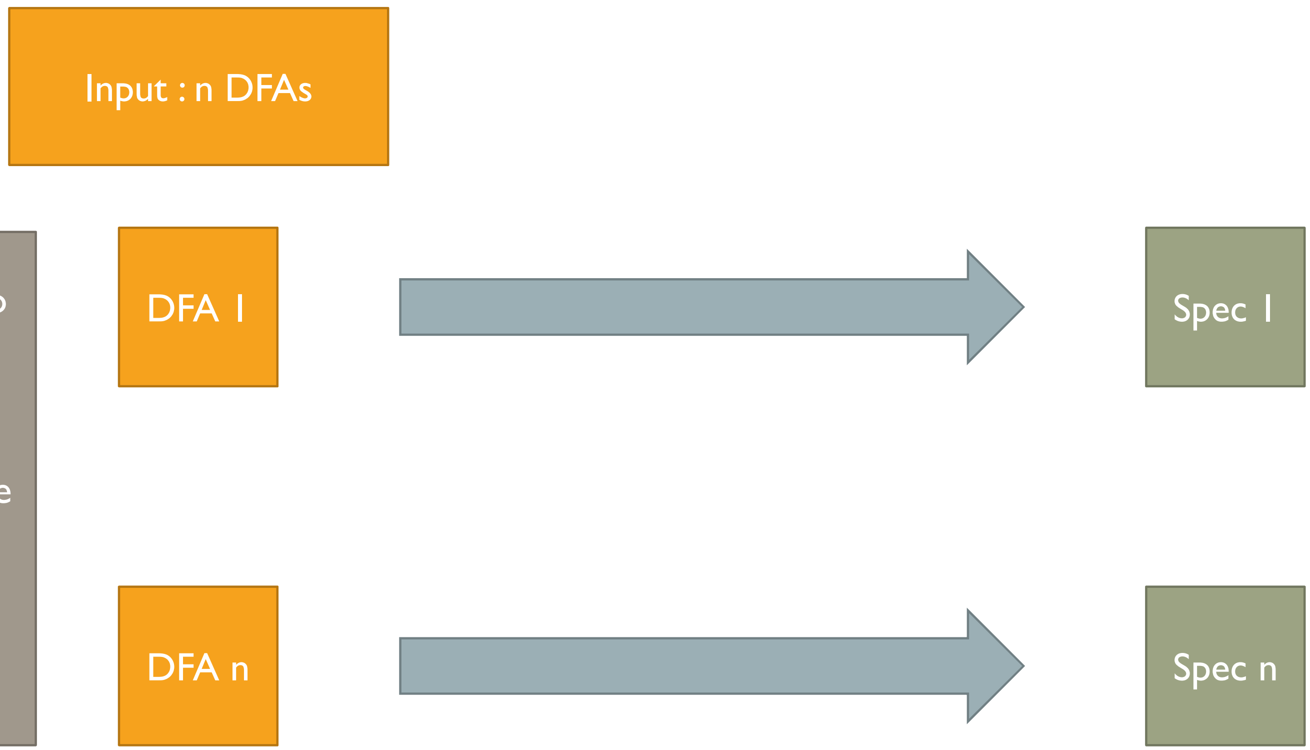
The separability of the primary-trace and deviating-trace conditions allowed us to build a special top-down deterministic Buchi tree automaton. Even though this automaton had an exponential state-space, the special nature of the automaton allowed us to reduce tree automaton nonemptiness into nonemptiness of an exponential-size Buchi word automaton.



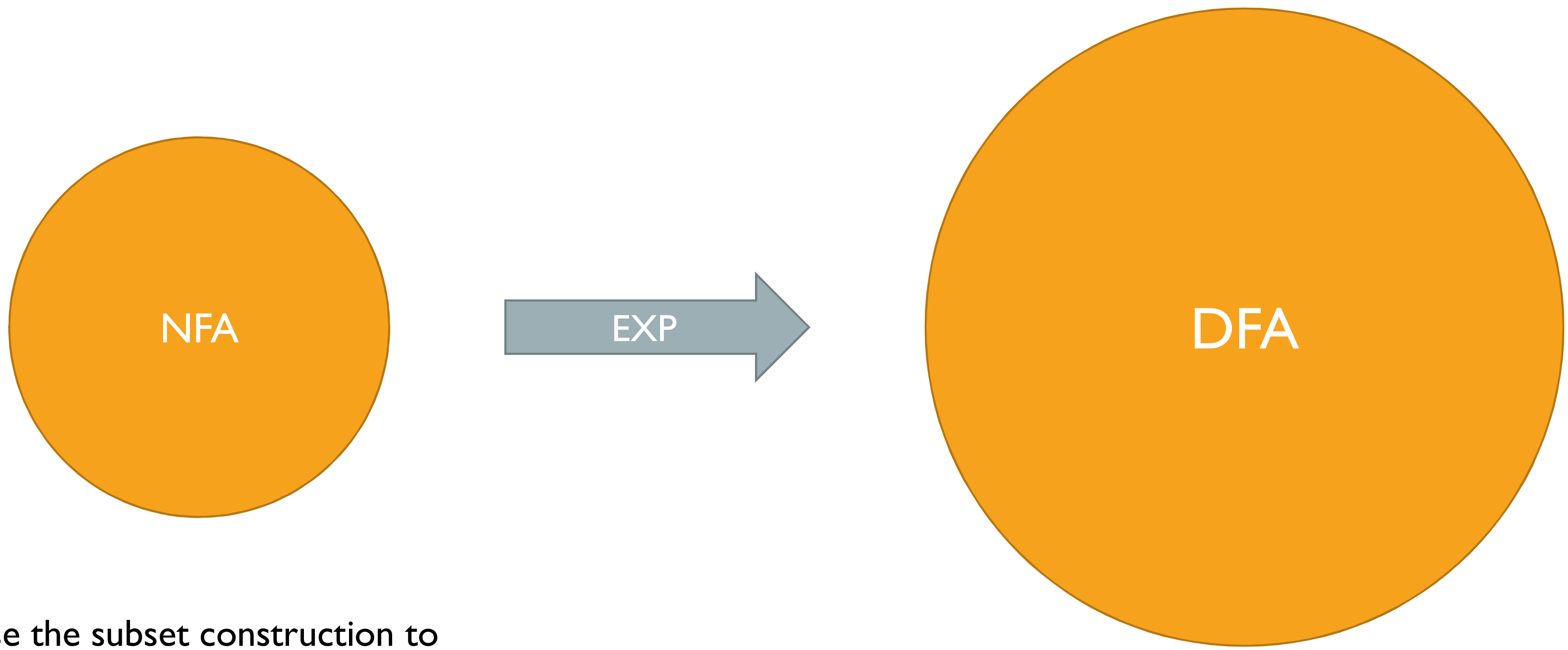
DFA REALIZABILITY LOWER BOUND

A reduction from DFA intersection nonemptiness : Given n DFAs, do they accept a common word?

DFA goals are modified to create prefix-satisfaction compliant specifications Since we specify the behavior we expect on the primary trace, we can specify that we expect all these specifications to be met.



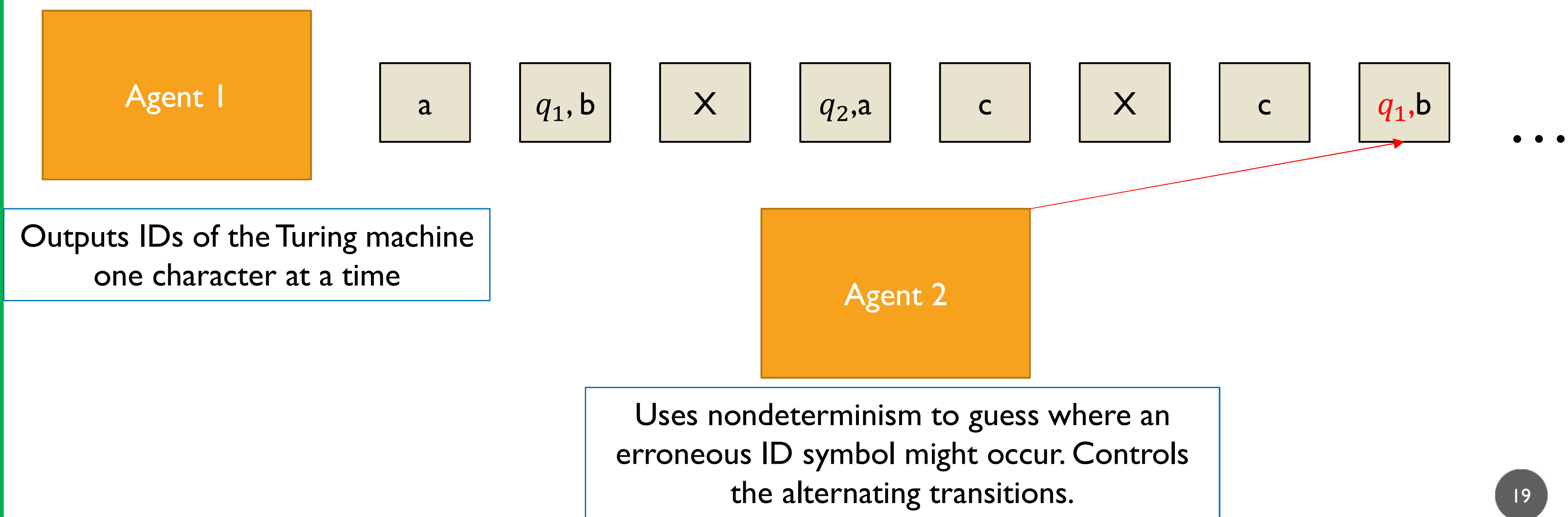
NFA REALIZABILITY – UPPER BOUND



Use the subset construction to translate from NFA to DFA.

NFA REALIZABILITY – LOWER BOUND

Reduce from the decision problem of whether a bounded-space alternating Turing machine accepts the empty tape



REALIZABILITY RESULTS FOR MULTIAGENT SYSTEMS

	Compleixty of Realizability
Deterministic Finite Automaton	PSPACE-complete
Nondeterministic Finite Automaton	EXPTIME-complete
Alternating Finite Automaton	2EXPTIME-complete

For DFA goals, the deviation condition can be checked in polynomial time. However, the inherent cross-product that arises when dealing with multiple agents means it is PSPACE-complete. For NFA and AFA goals, the inherent complexity involved with these more succinct representations dominates the overall complexity. We can show hardness with just the two-agent games that arise when deviations are analyzed.

Results in table due to Rajasekaran and Vardi, 2021, 2022

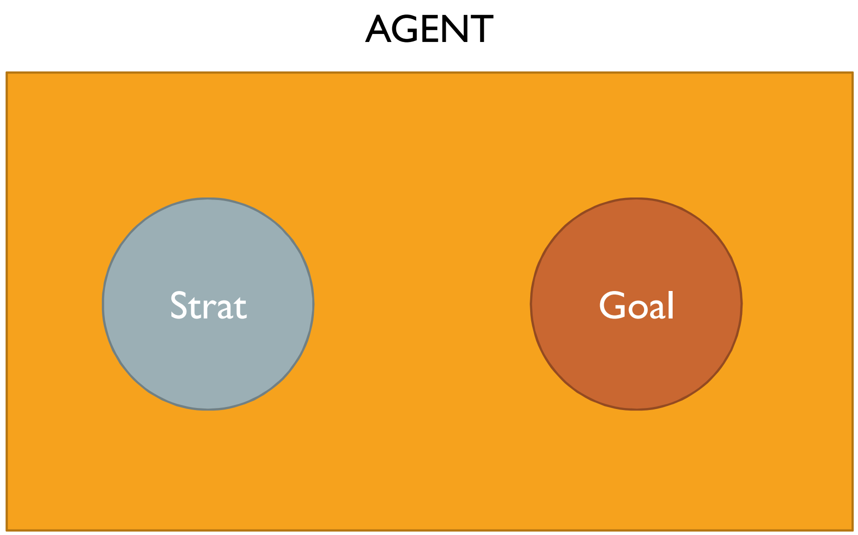
RESULTS FOR MULTIAGENT SYSTEMS

	Verification	Realizability
Deterministic Finite Automata	PSPACE-complete	PSPACE-complete
Nondeterministic Finite Automata	PSPACE-complete	EXPTIME-complete
Alternating Finite Automata	PSPACE-complete	2EXPTIME-complete

While goal representation matters for Realizability, it does not matter for Verification!

Results in table due to Rajasekaran and Vardi, 2021, 2022

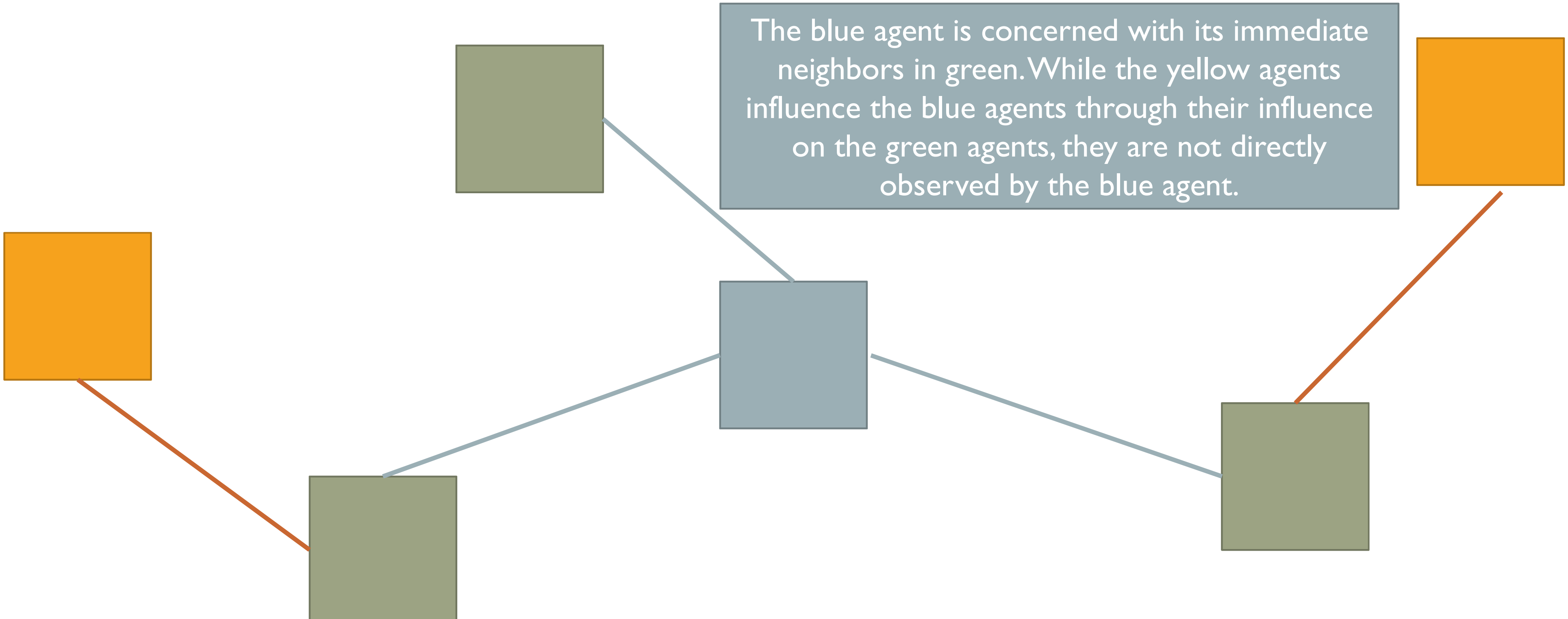
SETUP FOR VERIFICATION PROBLEM



Agents come equipped with a goal (as before) and a finite-state transducer representing their assigned strategy. Both components (strategy transducer and goal automaton) normally consider the actions of every other agent.

For an unbounded number of agents, this means that the input alphabet to these components itself can be seen as an exponential construction. This makes it hard to create polynomial-time reductions to prove the problem PSPACE-hard.

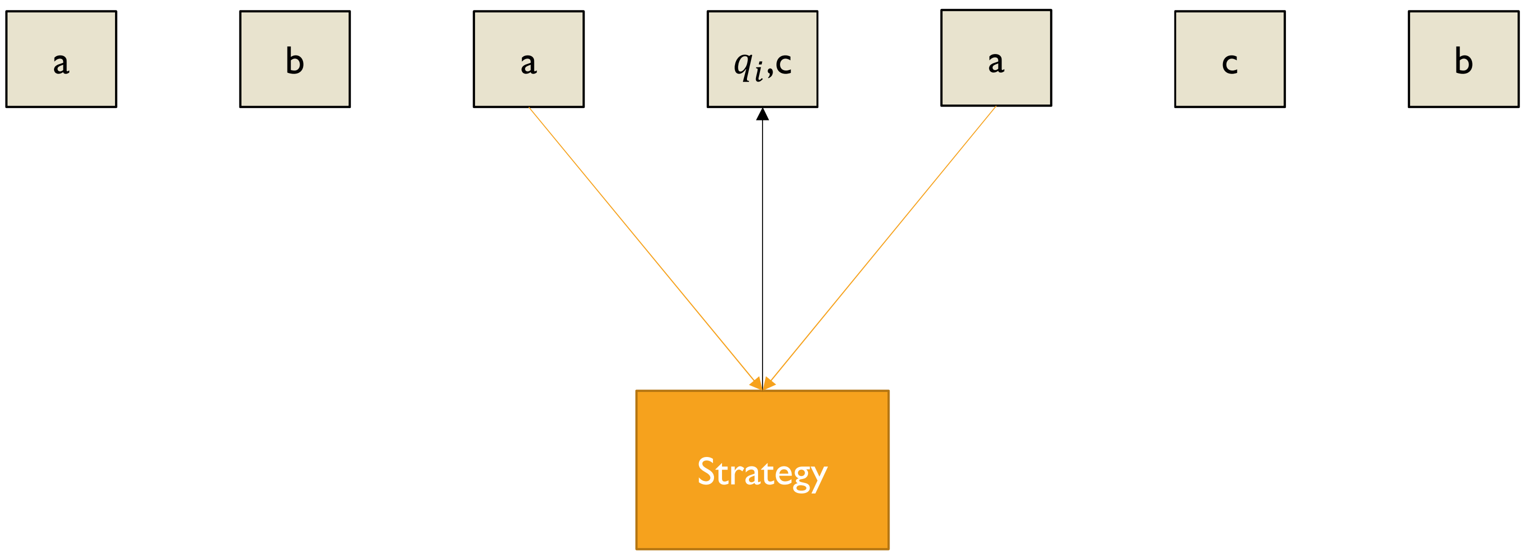
BOUNDED-CHANNEL PROPERTY



REDUCTION

We reduce from the problem of deciding whether a bounded space deterministic Turing machine accepts the empty tape.

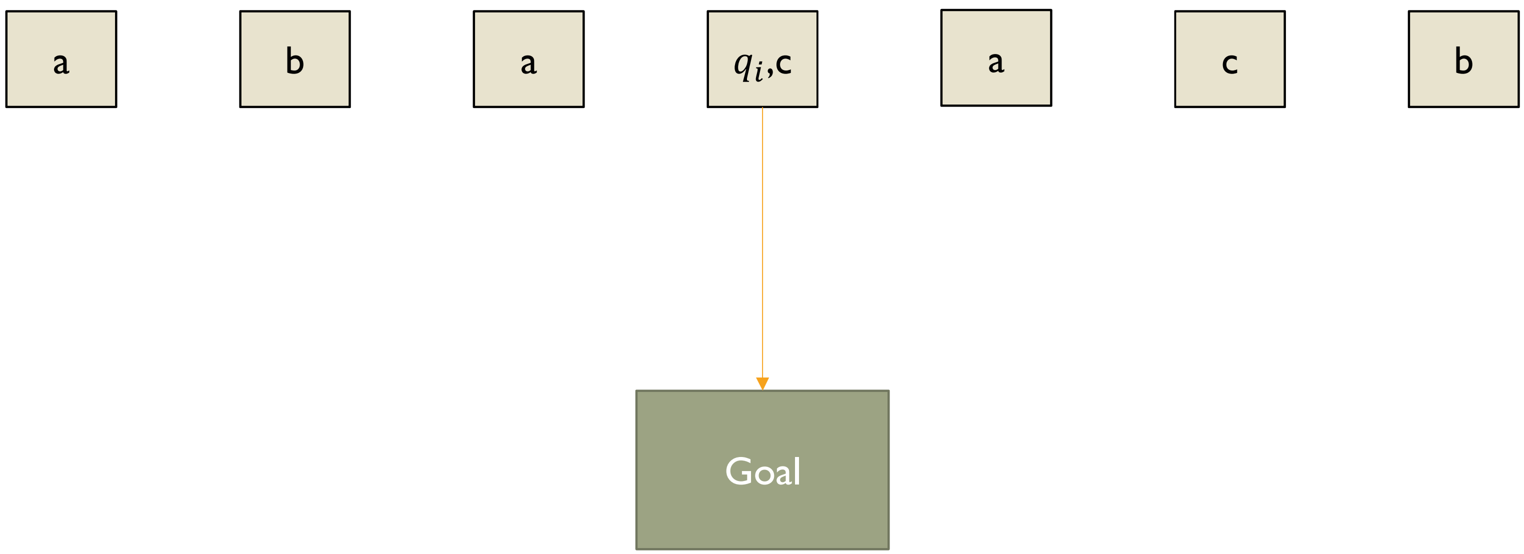
Strategies output the contents of the tape. To know the contents of a cell at time i, you only need the knowledge of its (at most) two neighbors at time $i - 1$. This naturally gives us the bounded-channel property.



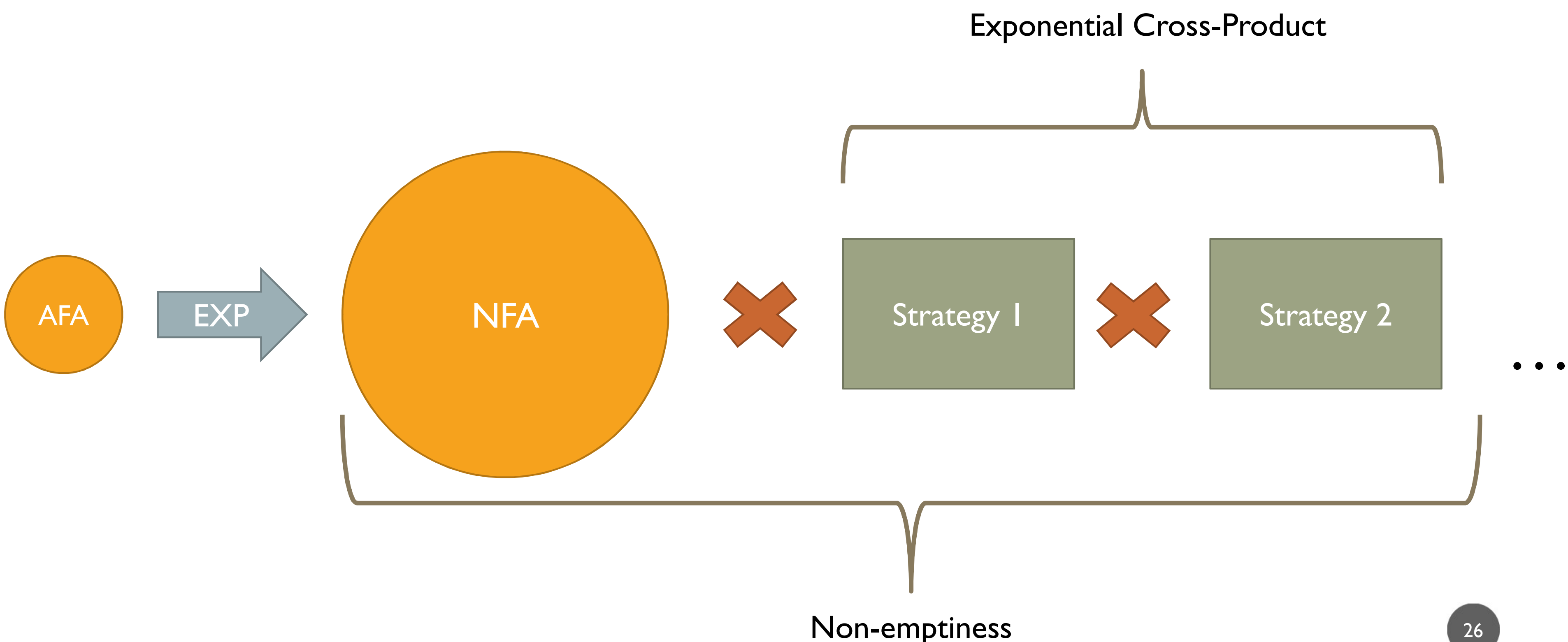
REDUCTION II

The goal is then to observe a character that corresponds to the unique final accepting state of the computation.

This can easily be done with any automaton goal.



UPPER BOUND



Exponential Cross-Product
AFA
EXP
NFA
Strategy 1
Strategy 2
. . .
Non-emptiness

RESULTS FOR MULTIAGENT SYSTEMS

	Verification	**Realizability**
Deterministic Finite Automata	PSPACE-complete	PSPACE-complete
Nondeterministic Finite Automata	PSPACE-complete	EXPTIME-complete
Alternating Finite Automata	PSPACE-complete	2EXPTIME-complete

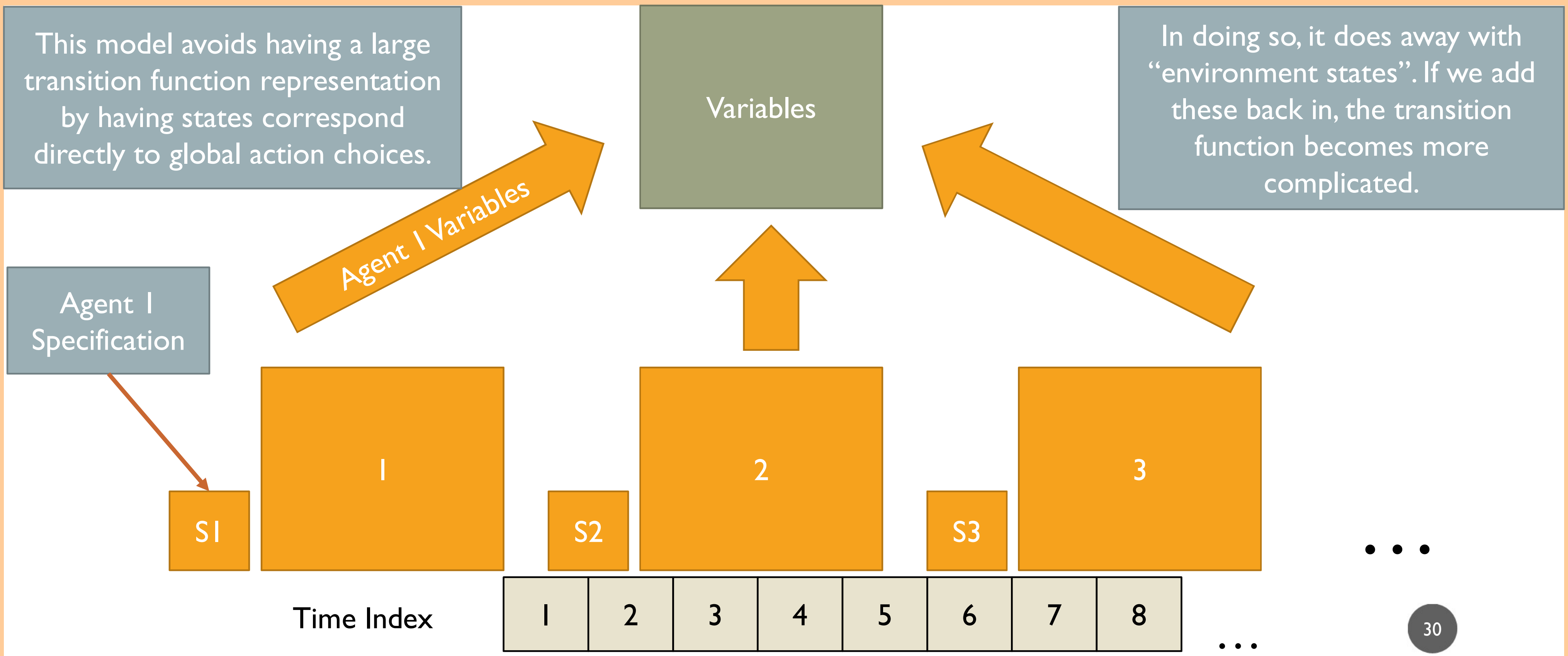
This gives us our verification results. The lower bound is shown through the use of bounded-channel models, a special subset of the general model with a naturally succinct representation.

SUMMARY

	Verification	Realizability
Deterministic Finite Automata	PSPACE-complete	PSPACE-complete
Nondeterministic Finite Automata	PSPACE-complete	EXPTIME-complete
Alternating Finite Automata	PSPACE-complete	2EXPTIME-complete

SUCCINCT MODELS

THE IBG MODEL



REPRESENTATION

PSPACE-complete

$$n \in L$$

Polynomial?

Actions

Agent 1 — 1, 2

Agent 2 — 1, 2

$\cdots$

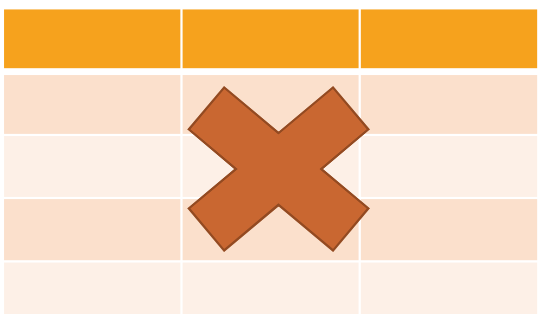
Agent $|n|$ — 1, 2

We want to reduce from L and use a linear number of agents w.r.t. the input size. But if each agent has a non-trivial number of actions and we are dealing with a set of environment states, this implicitly gives us an exponential transition table.

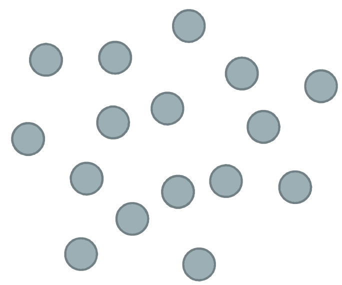
REPRESENTATION

The iBG model got around this somewhat. By not including environment states, the model did not need to create a large transition table. The state of the game directly corresponded to the last collective action taken.

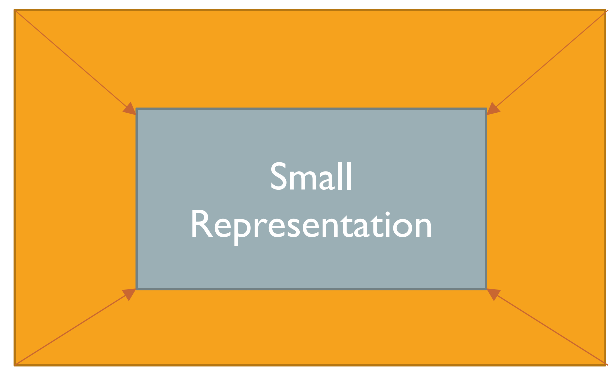
However, environment states are desirable. Is there a way to create succinct multiagent systems with environment states?



No huge transition table

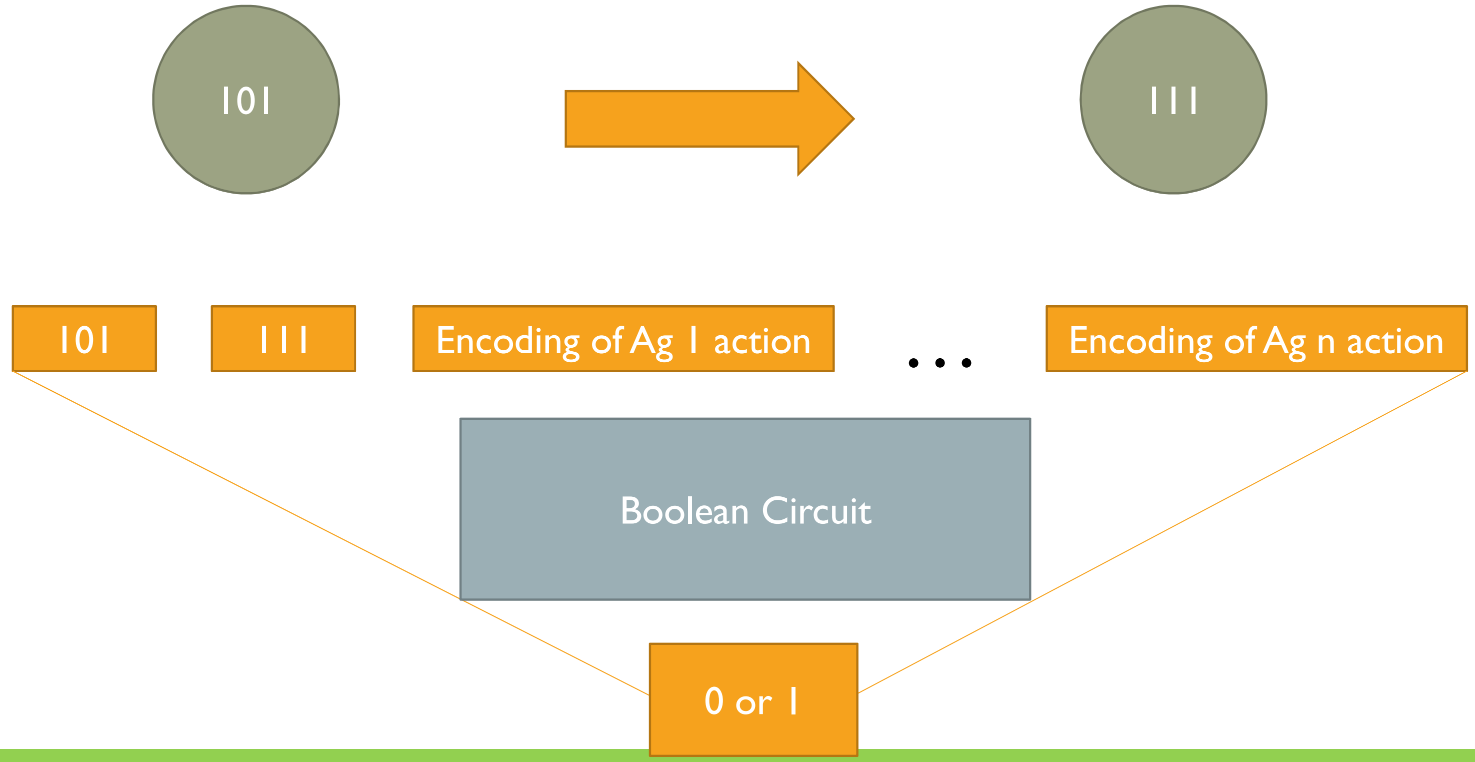


Environment States



CIRCUIT-BASED MODEL

We can represent states, transitions, and actions through a circuit-based model in the same vein as succinct graphs.



CONCLUSION

	Verification	**Realizability**
Deterministic Finite Automata	PSPACE-complete	PSPACE-complete
Nondeterministic Finite Automata	PSPACE-complete	EXPTIME-complete
Alternating Finite Automata	PSPACE-complete	2EXPTIME-complete