Synthesizing Locally Symmetric Parameterized Protocols from Temporal Specifications

Ruoxi Zhang[1], Richard Trefler[1] and Kedar S. Namjoshi[2]

[1] University of Waterloo, Waterloo, Canada
[2] Nokia Bell Labs, Murray Hill, USA

FMCAD 2022

DARPA DEFENSE ADVANCED RESEARCH PROJECTS AGENCY

Natural Sciences and Engineering Research Council of Canada
Conseil de recherches en sciences naturelles et en génie du Canada
Canada

Outline

- **Overview**
- Model of Computation
- Parameterized Synthesis Problem
- Tableau Approach
- Application
- Conclusion and Future Work

Problem

- Synthesize parametric protocols from temporal specifications
  - Locally symmetric protocols composed of many isomorphic copies of a representative process
  - Applications: network communication protocols, distributed algorithms, multi-core hardware models, etc
  - Example: token-passing mutual exclusion
- Undecidable in general [Pnueli & Rosner, 1990]
- State explosion

Our Approach: Global to Local

- Automatically construct a (representative) process P_n
  - P_n is closed under interference by neighboring copies of P_n
  - The closure satisfies a temporal specification φ_n
- Implies that the global structure of any instance $||_i P_i$ satisfies $\bigwedge_i \varphi_i$

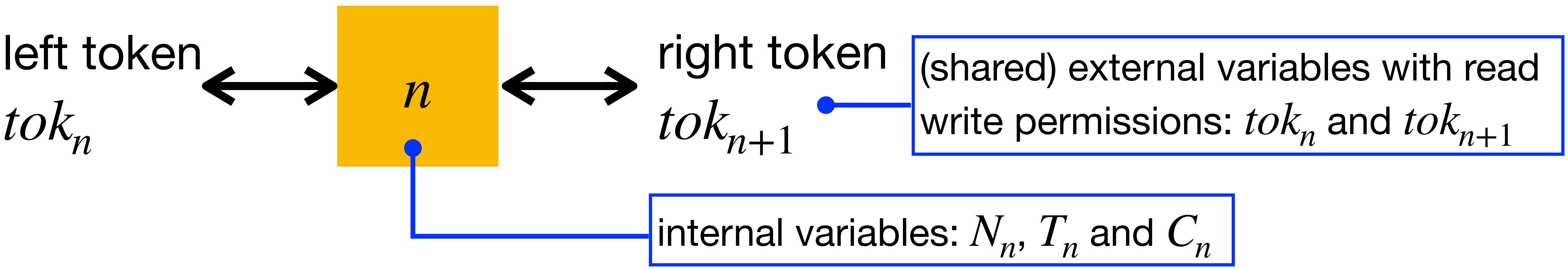
Example: Mutual Exclusion

- Goal: prevent simultaneous access to a critical resource
- [Initialization] A single token
- A process uses the token to access the resource and passes the token on completion
- [Interference] Token is passed clockwise between neighboring processes

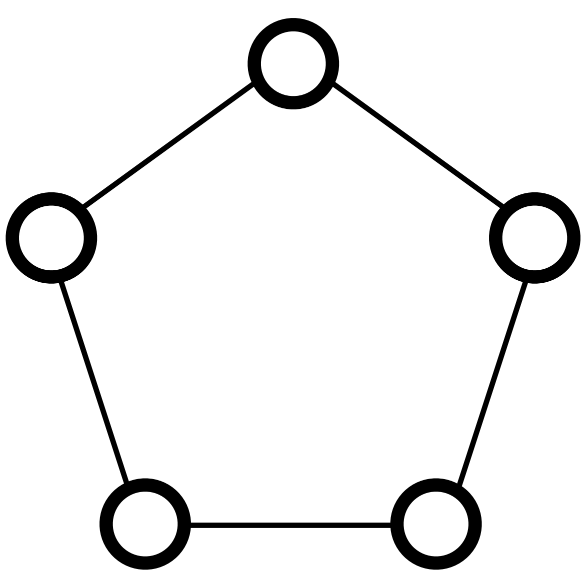
Outline

- Overview
- **Model of Computation**
- Parameterized Synthesis Problem
- Tableau Approach
- Application
- Conclusion and Future Work

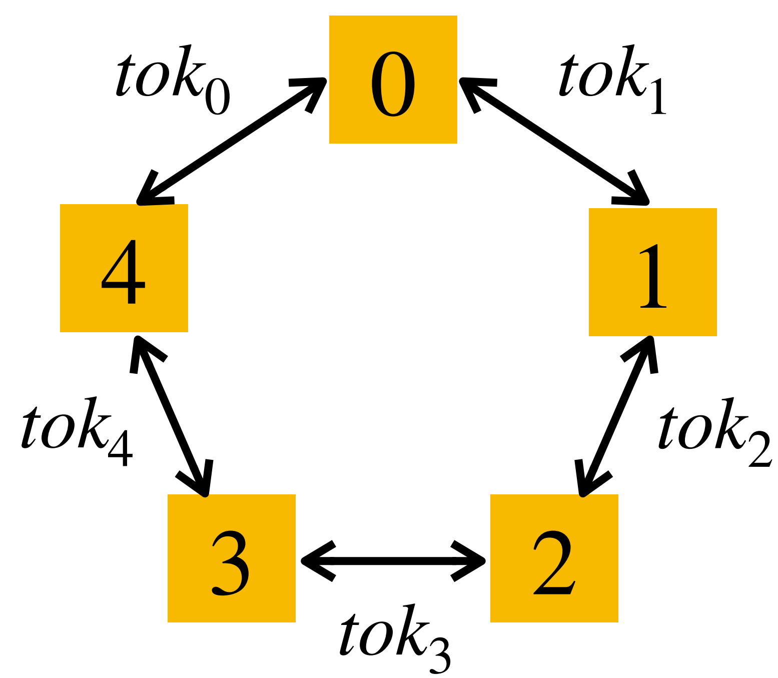
Uniform Ring



The **tile** of a ring-based token-passing mutual exclusion protocol



A ring **architecture** of size 5



An **instance** constructed from the tile

Representative Process

- $P_n = (S_n, S_n^0, T_n, \lambda_n)$
  - A state $s_n \in S_n$ is labeled by internal and external variables
  - For any neighbor m, a joint state is a pair (s_n, t_m)
    - s_n and t_m have the same value for all shared variables
    - e.g., $\left(tok_n, T_n, \neg tok_{n+1} \;,\; \neg tok_{n+1}, N_{n+1}, \neg tok_{n+2} \right)$

Local State Space

- H_n^* is the local state transition system of P_n
- H_n^* has two types of transitions:
  - From s_n to s'_n by n
    - e.g., $tok_n, T_n, \neg tok_{n+1} \xrightarrow{n} tok_n, C_n, \neg tok_{n+1}$
  - [**Interference transitions**] From (s_n, t_m) to (s'_n, t'_m) by neighbors m
    - e.g., $tok_{n-1}, C_{n-1}, \neg tok_n \xrightarrow{n-1} \neg tok_{n-1}, N_{n-1}, tok_n$

Global State Space

- $G = (S, S^0, T, \lambda)$

- Concurrency: nondeterministic interleaving of transitions of processes

- G_n: project out the labels of transitions other than those of n.

Outline

- Overview
- Model of Computation
- **Parameterized Synthesis Problem**
- Tableau Approach
- Application
- Conclusion and Future Work

Local Property

- The local property φ_n describes the behavior of P_n in its neighborhood

- Use fair computation tree logic (Fair CTL) to represent φ_n

- Assumption: unconditionally fair scheduling
$\Phi = F^\infty ex_n \wedge \bigwedge_m F^\infty ex_m$

- $\mathsf{A}_\Phi \mathsf{Y}_n$, $\mathsf{E}_\Phi \mathsf{X}_n$, $\mathsf{A}_\Phi \mathsf{G}$, $\mathsf{E}_\Phi \mathsf{G}$, $\mathsf{A}_\Phi \mathsf{F}$, etc

- e.g., a trying process eventually enters into critical
$\mathsf{A}_\Phi \mathsf{G}(T_n \rightarrow \mathsf{A}_\Phi \mathsf{F} C_n)$

Parameterized Synthesis

Theorem II.2. *Let φ_n be a local FairCTL specification. Let P_n be a process such that its derived H_n^* satisfies φ_n. Every instance of the parameterized system constructed from isomorphic copies of P_n satisfies the global property $\bigwedge_i \varphi_i$.*

Procedure:

Write φ_n by hand

↓

Construct H_n^* automatically

↓

Extract P_n from H_n^* automatically

Bisimulation Relation

Theorem II.1. *([NT18]) H_i^* stuttering-simulates G_i for every i. Moreover, if H_i^* satisfies an 'outward-facing' restriction, then H_i^* and G_i are stuttering-bisimular.*

- Assumption: Outward-facing
  - Interference transitions are independent of the internal state of the interfering process

Outline

- Overview
- Model of Computation
- Parameterized Synthesis Problem
- **Tableau Approach**
- Application
- Conclusion and Future Work

Tableau Method c.f. [EL86], [EC82]

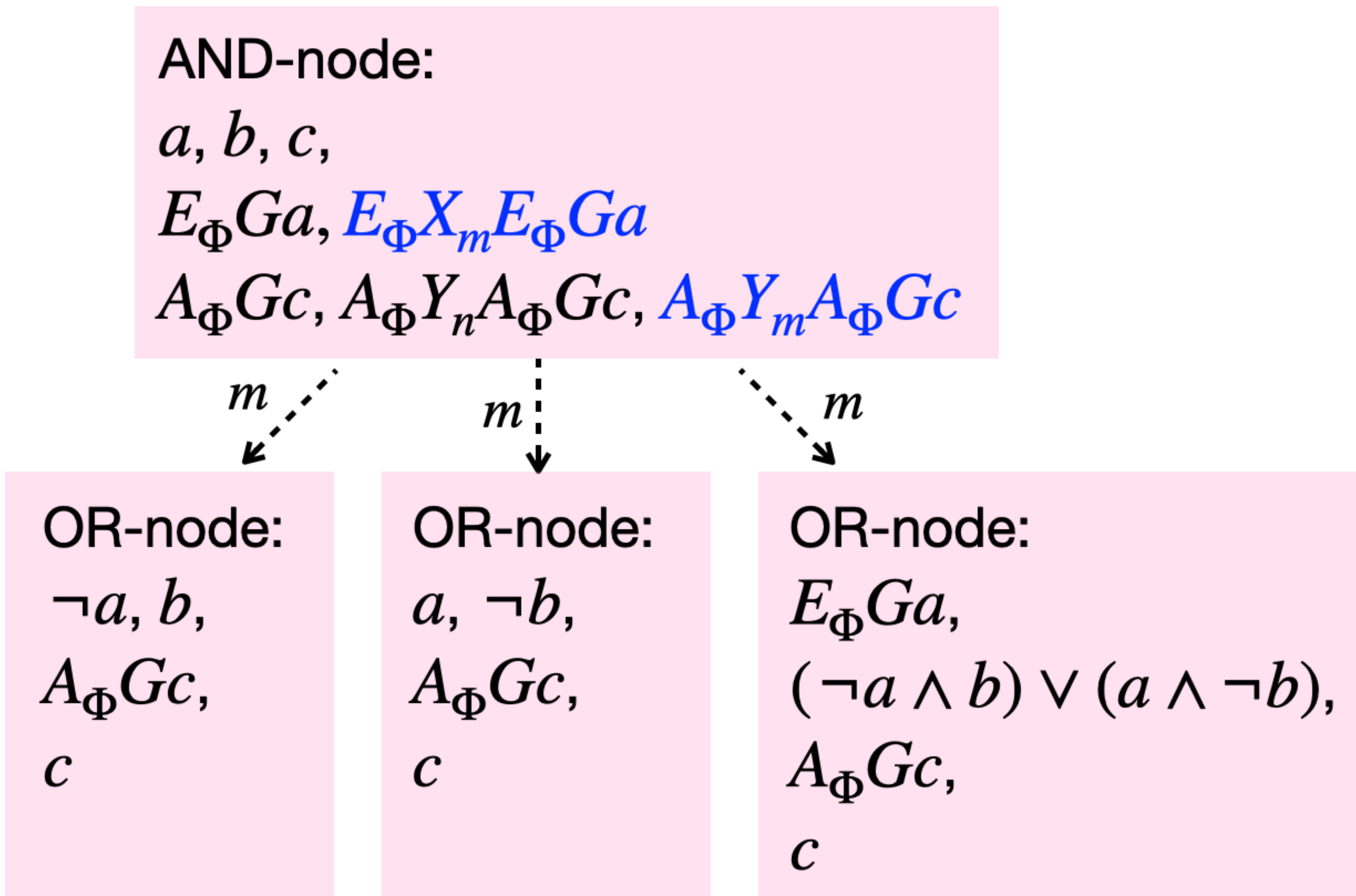
1. Construct the initial tableau $\mathscr{T}_n^0$ from φ_n
2. Construct $\mathscr{T}_n^{i+1}$ from $\mathscr{T}_n^i$
   2.1. Summarize changes by neighbors in $\mathscr{T}_n^i$
   2.2. Add interference transitions
   2.3. Add successors to leaf nodes

repeat, until the tableau reaches its fixpoint

3. Apply deletion rules
4. Extract a model

Interference Transition

Suppose m changes a, b to $\neg a, b$ and $a, \neg b$:



An example of adding interference transitions to an applicable node

Complexity

- The size of the tableau is bounded by $exp(|\varphi_n|)$
  - Local specification with respect to n
- Complexity of synthesizing P_n: polynomial in the size of the tableau c.f. [AAE04]
- Deployment cost: linear in the number of nodes
  - Avoid exponential growth in instance size

Outline

- Overview
- Model of Computation
- Parameterized Synthesis Problem
- Tableau Approach
- **Application**
- Conclusion and Future Work

Application

- Mutual exclusion
- Leader election
- Dining philosophers

Mutual Exclusion

Specification:

1. [Initial condition] Exactly one process has the token
2. [Exclusion] No process has multiple tokens

3. [Non-critical] A process moves from non-critical to trying (while keeping the token) or remains in non-critical (while passing the token)
4. [Trying] A process moves from trying to critical with the token
5. [Critical] A process moves from critical to non-critical while passing the token

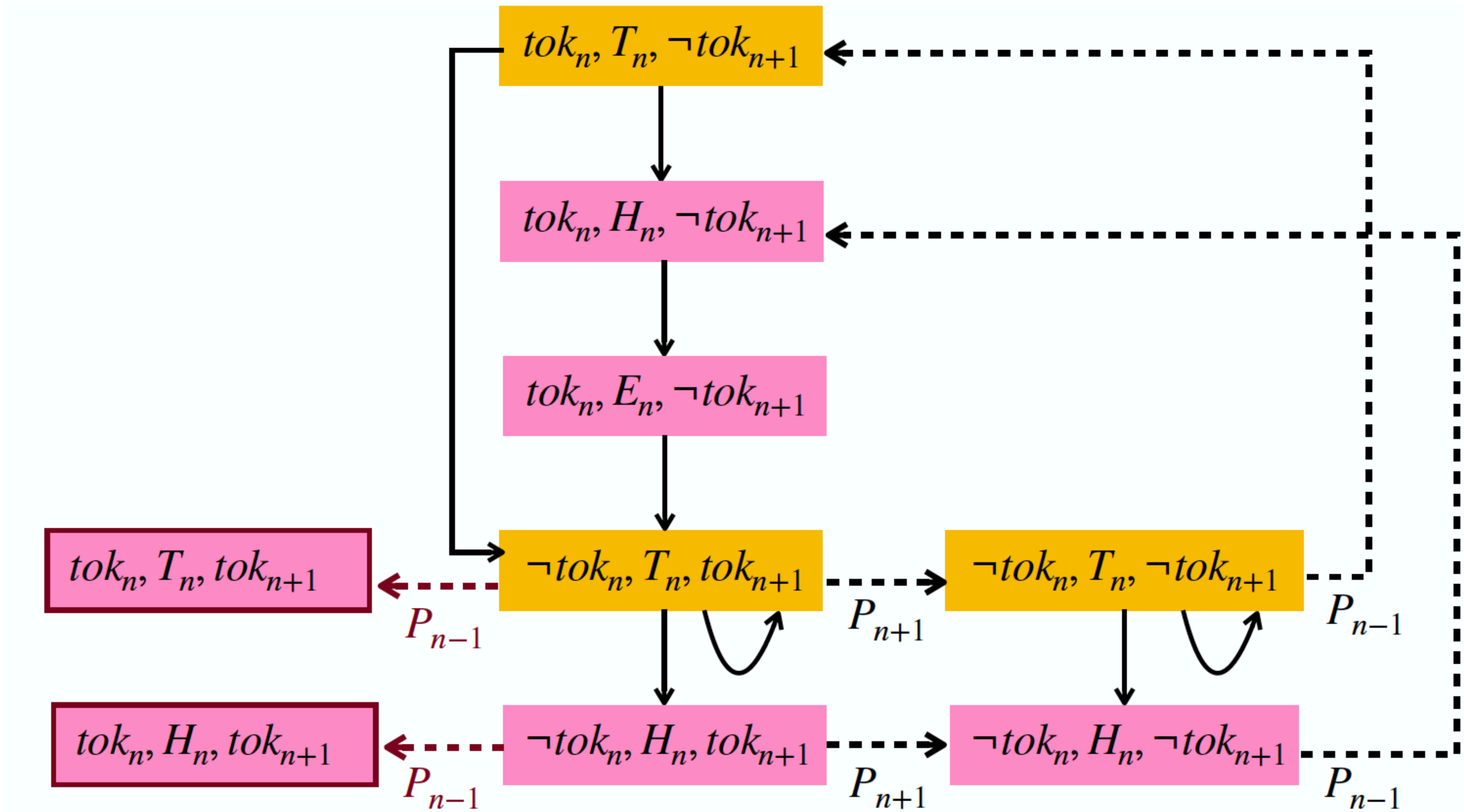
6. [Liveness] A trying process eventually enters into critical
7. [One at a time] A process is in exactly one of the three internal states

Mutual Exclusion

Specification:

1. [Initial condition] $N_n \wedge tok_n \wedge \neg tok_{n+1}$, $N_n \wedge \neg tok_n \wedge tok_{n+1}$, and $N_n \wedge \neg tok_n \wedge \neg tok_{n+1}$
2. [Exclusion] $\mathsf{A}_\Phi \mathsf{G}(\neg tok_n \vee \neg tok_{n+1})$
3. [Non-critical]
   $\mathsf{A}_\Phi \mathsf{G}((N_n \wedge \neg tok_n) \rightarrow (\mathsf{E}_\Phi \mathsf{X}_n(N_n \wedge \neg tok_n) \wedge \mathsf{E}_\Phi \mathsf{X}_n(T_n \wedge \neg tok_n)))$,
   $\mathsf{A}_\Phi \mathsf{G}((N_n \wedge tok_n) \rightarrow (\mathsf{E}_\Phi \mathsf{X}_n(N_n \wedge \neg tok_n \wedge tok_{n+1}) \wedge \mathsf{E}_\Phi \mathsf{X}_n(T_n \wedge tok_n)))$
4. [Trying] $\mathsf{A}_\Phi \mathsf{G}((T_n \wedge tok_n) \rightarrow \mathsf{A}_\Phi \mathsf{Y}_n(C_n \wedge tok_n))$
5. [Critical] $\mathsf{A}_\Phi \mathsf{G}(C_n \rightarrow \mathsf{A}_\Phi \mathsf{Y}_n(N_n \wedge \neg tok_n \wedge tok_{n+1}))$
6. [Liveness] $\mathsf{A}_\Phi \mathsf{G}(T_n \rightarrow \mathsf{A}_\Phi \mathsf{F} C_n)$
7. [One at a time] $\mathsf{A}_\Phi \mathsf{G}(N_n \vee T_n \vee C_n)$, $\mathsf{A}_\Phi \mathsf{G}(N_n \rightarrow (\neg T_n \wedge \neg C_n))$, $\mathsf{A}_\Phi \mathsf{G}(T_n \rightarrow (\neg N_n \wedge \neg C_n))$, and $\mathsf{A}_\Phi \mathsf{G}(C_n \rightarrow (\neg N_n \wedge \neg T_n))$

Mutual Exclusion



The model of φ_n for the mutual exclusion protocol

Outline

- Overview
- Model of Computation
- Parameterized Synthesis Problem
- Tableau Approach
- Application
- **Conclusion and Future Work**

Conclusion

- Reduce the global synthesis problem to local
    - Write φ_n by hand
    - From φ_n a representative process P_n is synthesized
    - Representative process can be deployed in instances of arbitrary size

Future Work

- Explore the reduction from global specifications to local
- Apply our approach to other architectures and applications
  - e.g., tori, wrap-around mesh, and other network patterns constructed from tiles
  - e.g., red-black rings, committee coordination, and termination detection

REFERENCES

[PR90] A. Pnueli and R. Rosner, “Distributed reactive systems are hard to synthesize,” in *Proceedings 31st Annual Symposium on Foundations of Computer Science*, 1990, pp. 746–757 vol.2.

[EL86] E. A. Emerson and C.-L. Lei, “Temporal reasoning under generalized fairness constraints,” in *STACS 86*, B. Monien and G. Vidal-Naquet, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1986, pp. 21–36.

[EC82] E. A. Emerson and E. M. Clarke, “Using branching time temporal logic to synthesize synchronization skeletons,” *Science of Computer Programming*, vol. 2, no. 3, pp. 241–266, 1982.

[NT18] K. S. Namjoshi and R. J. Trefler, “Symmetry reduction for the local mu-calculus,” in *Tools and Algorithms for the Construction and Analysis of Systems*, D. Beyer and M. Huisman, Eds. Cham: Springer International Publishing, 2018, pp. 379–395.

[AAE04] P. C. Attie, A. Arora, and E. A. Emerson, “Synthesis of fault-tolerant concurrent programs,” *ACM Trans. Program. Lang. Syst.*, vol. 26, no. 1, pp. 125–185, Jan. 2004.