Parameterized Verification and Synthesis

Swen Jacobs

February 25\textsuperscript{th} 2019

Scientific Talk in the Habilitation Process
Problem: Correct Design of Parameterized Systems

Concurrent systems are everywhere

Often: parametric number of components

Hard to get correct

Icons made by Freepik from www.flaticon.com
How to Get (Parameterized) Systems Right

Verification Workflow:
1. manual implementation
2. formalize requirements
3. check implementation against specification
4. manual bug-fixing

standard practice for many systems

Very active research on Parameterized Verification

Synthesis Workflow:
1. formalize requirements
2. automatically obtain correct implementation

computationally expensive (or undecidable)

Little previous work on Parameterized Synthesis

Swen Jacobs
Parameterized Verification and Synthesis
I. (Parameterized) Verification and Synthesis: State of the Art
II. Parameterized Synthesis based on Cutoffs
III. Cutoff Results for Verification and Synthesis
PARAMETERIZED VERIFICATION AND SYNTHESIS: STATE OF THE ART
For finite-state systems, we can decide verification problems:

- **Model Checking** [EC80, QS82]

- **Symbolic Model Checking** [B+92, CVWY92]

**state space explosion:** explicit-state model checking fails for large state spaces

**symbolic representations** allow us to handle systems with “$10^{20}$ states and beyond”

$$ (x_1 \land x_4) \lor (x_2 \land x_5) \lor (x_3 \land x_6) $$
For many applications, state space is not fixed, but depends on parameters such as
- the number of users/participants
- the size of data structures

**Expectation:** most errors manifest already in systems of „small“ size

**Counterexample:** cache coherence protocols correct with small number of participants [C+92], but exhibit errors for larger number [K+97]

need **formal argument** why correctness extends to systems of arbitrary size
Parameterized Verification is difficult:

Even if systems can be represented as compositions of finite-state components, simple safety properties can be undecidable \[^{[S88]}\].

This led to research into

- restrictions that yield **decidable cases**
- **decidable approximations**
- **semi-decision procedures**
For finite-state systems, similar situation as in verification:

- finite-state **two-player game** yields implementations [BL69,PR89]
- **symbolic implementations** can solve problems of significant complexity [JB06]

**Major difference:**
Synthesis of distributed systems is **undecidable** in general [PR90,FS05]

**Parameterized synthesis:**
Emerson and Attie synthesize pairs of processes that can be generalized to parameterized systems [EA98] – restrictions on specifications, process implementations & system model make the problem decidable, but limit generality
PARAMETERIZED SYNTHESIS BASED ON CUTOFFS
Industrial synthesis benchmark:
Synthesize bus controller with locked accesses, bursts, and other features from temporal logic specification.

Parameterized in # of masters accessing the bus.

Can we avoid this explosion and solve the general case instead?
Idea: Synthesis of Replicable Building Blocks

Consider distributed arbiter with specification

- **Request-response:** \( \bigwedge_{i \in \{1, \ldots, n\}} G(r_i \rightarrow F g_i) \)
- **Mutual exclusion:** \( \bigwedge_{i \neq j \in \{1, \ldots, n\}} G \neg (g_i \land g_j) \)

To obtain systems that work for any \( n \), we synthesize components that act only on local information and therefore can be replicated.

How to ensure correctness for all sizes?
Synthesis Problems

**Specification Language:**

LTL

- temporal operators $G, F, U$
- propositional variables & connectives

**Parameterized case:** indexed LTL\(\mathcal{X}\)

- indexed variables $p_i$, index quantifiers $\forall i, \exists i$. 

**Implementation:**

given as *labelled transition system* (LTS)

**Architecture:**

- Parameterized architecture: Sequence of architectures (of increasing size).

**Parameterized Synthesis Problem:**

**Given:**
1. indexed LTL Specification $\varphi$
2. parameterized Architecture $A$

**Find:**
Implementation $S$ with $S, A_n \models \varphi$ for all $n$
Existing Approaches:
- Implementations not correct in general, or param. verification does not succeed
- Synthesis does not scale
Cutoffs for Parameterized Systems

Consider

• a class $P$ of parameterized systems, defined by a parameterized architecture $A$ and additional restrictions on the process implementations,

• a class $\Phi$ of specifications, e.g. indexed LTL with fixed # of indices.

Cutoff:

A number $c \in \mathbb{N}$ is a cutoff for $P$ and $\Phi$ if for every specification $\varphi \in \Phi$ and every $S$ from $P$, the following holds:

$$\forall n \geq c: (S, A_c \models \varphi \iff S, A_n \models \varphi)$$
Parameterized Synthesis

Class of Specifications

Class of Parameterized Systems

Cutoff

LTL Specification

Architecture

Communication Primitives Cutoff Conditions

Bounded [SF07] Synthesis + Encoding of System Class

Parameterized Verification

Counterexample

Parameterized Correctness by Construction
Example: Simple Distributed Arbiter

Distributed arbiter with specification

Request-response: $\forall i: G(r_i \rightarrow Fg_i)$

Mutual exclusion: $\forall i \neq j: G\neg(g_i \land g_j)$

What is a suitable class of parameterized systems?
Cutoff Results for Token Rings

**Theorem [EN95]:**

In token rings with interleaving semantics and fair token passing, a given process implementation satisfies a specification $\varphi \in \text{indexed } \text{CTL}^* \setminus X$ in all rings iff it satisfies $\varphi$ in rings of small size.

For $\forall i. \varphi(i)$, cutoff is 2.

For $\forall i, j. \varphi(i, j)$, cutoff is 4.

**Corollary:** For parameterized synthesis, it is sufficient to synthesize a process implementation satisfying $\varphi$ (and Thm. conditions) in a small ring.

*Need to guarantee additional cutoff conditions*
Parameterized Synthesis based on Cutoff Results

Distributed arbiter in token ring of 4 processes with specification

Request-response: \( \forall i: G(r_i \rightarrow Fg_i) \)

Mutual exclusion: \( \forall i \neq j: G \neg(g_i \land g_j) \)

synthesized in \( \sim 10 \) sec.

Cutoff results **guarantee correctness** in rings of arbitrary size.

**Challenges:**
- **Scalability** (in size of specification)
- **Reduction only possible for limited class of systems and specifications**
Parameterized Synthesis of an AMBA Bus Controller

Reduction:
Existing cutoffs for CTL*\X
in token-rings
(interleaving semantics)

Extended in several dimensions, incl.:
• synchronous systems [SYNT14]
• local X operator [VMCAI13]
• global assumptions [VMCAI14,SYNT14]

Bounded Synthesis:
Existing approach for fixed-size systems
Adapted to token-rings [TACAS12]
Optimized (for token-rings) [VMCAI13,SYNT14]

We synthesized a solution for the AMBA Bus Controller with parameterized correctness guarantee

Figure 1: Formal specification of the AMBA AHB [12], in the GR(1) fragment of LTL.
CUTOFF RESULTS FOR VERIFICATION AND SYNTHESIS
Decidability Results for Parameterized Verification

Existing decidability and undecidability results:

• Many separate results
• Many different system models, sometimes with implicit assumptions

Hard to get an overview

Goal: collect, compare and unify

decidability results in parameterized verification
(for systems with uniform finite-state components)
Parameterized Verification Survey

We

• systematically compared existing models and decidability results,
• reviewed proof methods to obtain them, and
• introduced common computational model that captures existing models of systems communicating via different forms of synchronization:

- token-passing
- pairwise rendezvous
- broadcasts
- lossy synchronization
- guarded transitions
Parameterized Verification Survey

**Process Templates:**
- Single

**Synch. Primitives:**
- Single

**System Model:**
- Basic Model, Section 2.2
  - +Directions, Section 4.1.1
  - +Guards, Section 6.2
  - +Lossy Com., Section 7.2

**Specifications:**
- LTL, CTL*
- LTL, (ω-)regular
- LTL, (ω-)regular

**Topologies:**
- Ring, Arbitrary
- Clique

**Chapter References:**
- Token-Passing Systems, Chapter 4
- Rendezvous/Broadcast, Chapter 5
- Guarded Protocols, Chapter 6
- Ad-Hoc Networks, Chapter 7

Swen Jacobs
Parameterized Verification and Synthesis
Decidability Results for Rendezvous and Broadcast

- Decidable
- Undecidable

Diagram showing decidability results for Rendezvous and Broadcast.
Decidability Results for Token-Passing Systems

Large areas not known before
Consider general token-passing systems with

- arbitrary topologies
- branching-time specifications, i.e., fragments of indexed CTL*

Observation: existing cutoff results are limited

- either to non-branching topology (token rings) or to non-branching specifications (fragments of LTL)
- to processes that do not control or observe different directions in branching topologies

Insight: effect of token communication is captured by its movement through the network

New Results:

- undecidable if processes control directions or specifications can branch unboundedly
- cutoffs exist for arbitrary topologies and specifications with bounded branching
Additional Results

Cutoffs for Guarded Protocols:

• added support for **fairness assumptions** [VMCAI16]
• showed how to obtain **smaller cutoffs** depending on additional parameters [VMCAI18]

Synthesis of Fault-Tolerant Parameterized Systems:

• **Self-stabilization** (against transient, global faults) [CAV16, OPODIS18]
• **Byzantine fault-tolerance** (against permanent, local faults) [CAV16]
Recent Related Work

• **Lots** of new contributions to parameterized verification literature

• On the border to adjacent fields:
  - „computational algorithm design“ [D+16]
  - verification of multi-agent systems [KL16]
  - parameterized planning [GMRS16]
  - synthesis of distributed algorithms [LKW17]

• Synthesis with identifiers [ESKG14]

• Synthesis of self-stabilizing rings [EK17]

• Control of parameterized systems [BLS18]
SUMMARY
I. (Parameterized) Verification and Synthesis: State of the Art
Parameterized verification is important and difficult, lots of different approaches
Parameterized synthesis has rarely been considered

II. Parameterized Synthesis based on Cutoffs
First general approach for parameterized synthesis
Scales to long-standing industrial benchmark

III. Cutoff Results for Verification and Synthesis
Surveyed existing decidability & cutoff results
Generalized these results to close gaps and make them useful in synthesis

Thank You!


Bibliography (in order of appearance)