# Toward Parameterized Verification of Synchronous Distributed Applications

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Sagar Chaki, James Edmondson
July 21, 2014

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.



DM-0001493

# Motivation

Distributed algorithms have always been important

- File Systems, Resource Allocation, Internet, …

Increasingly becoming safety-critical

- Robotic, transportation, energy, medical

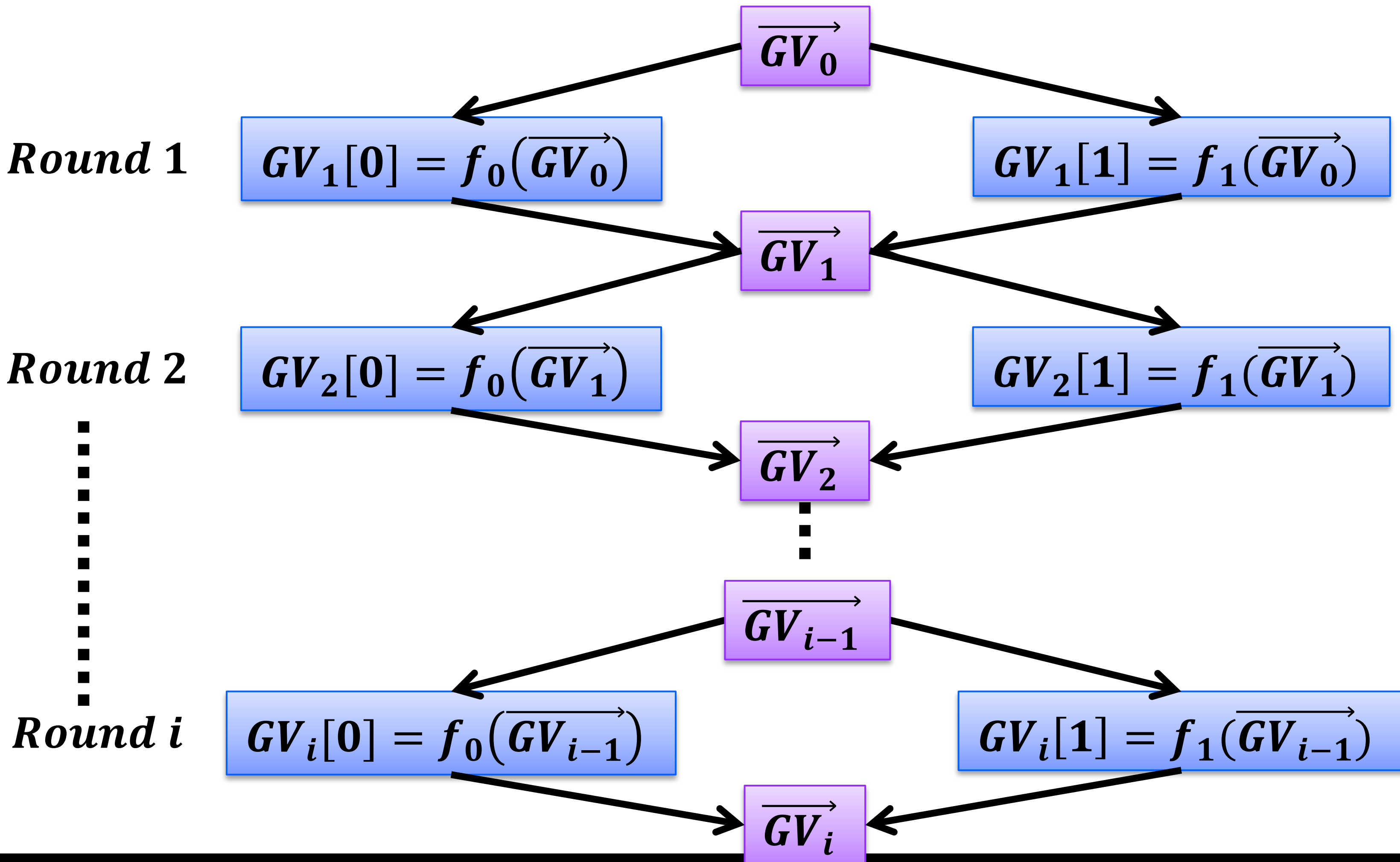
Prove correctness of distributed algorithm implementations

- Pseudo-code is verified manually (semantic gap)
- Implementations are heavily tested (low coverage)


**Model Checking Distributed Applications**
**http://mcda.googlecode.com**

# Synchronous Distributed Algorithm (SDA)

**Node 0** $= f_0()$ **Shared Variables:** $\overrightarrow{GV} = GV[0], GV[1]$ **Node 1** $= f_1()$



# SDA Syntax

Program with $n$ nodes : $P(n)$

- Each node has a distinct $id \in [1, n]$
- Array $GV$ has $n$ elements, $GV[i]$ writable only by node with id $i$

Each element of $GV$ is a bit-vector of width $W \in \mathbb{N}$

- Of those, the first $Z \in [0, W]$ bits are initialized non-deterministically
- The remaining $W - Z$ bits are initialized to $\bot$

In each round, node with id $id$ executes function $\rho$ whose body is a statement

$$
\begin{aligned}
stmt &\coloneqq skip \mid lval = exp && (assignment) \\
&\mid ITE(exp, stmt, stmt) && (if, then, else) \\
&\mid ALL(IV, stmt) && (iterate\ over\ nodes : use\ to\ check\ existence) \\
&\mid \langle stmt^{+} \rangle && (iteration\ of\ statements) \\
lval &\coloneqq GV[id][w] && (lvalues) \\
exp &\coloneqq \top \mid \bot \mid lval \mid GV[iv][w] \mid id \mid IV \mid \diamond(exp^{+}) && (expressions)
\end{aligned}
$$

# SDA Semantics and Verification

States are possible values of $GV$ : denoted $A$

Initial states : $I \subseteq A = \{ a \mid \forall i \in [1, n]. \forall x \in [Z+1, W]. a[i][x] = \bot \}$

Transition Relation : $R \subseteq A \times A = \{ (a, a') \mid \forall i \in [1, n]. a'[i] = \rho(a) \}$

Specification (1-index property) $\phi \coloneqq \forall i. \Psi(i)$

- $\Psi(i)$ is an expression with $i$ as only free variable
- $a \vDash \phi$ defined in a natural manner

Model Checking: $P(n) \vDash \phi \Leftrightarrow \forall a \in A. \forall a_I \in I. (a_I, a) \in R^* \Rightarrow a \vDash \phi$

Parameterized Model Checking: $PARMODCK(P, \phi) \equiv \forall n \in \mathbb{N}. P(n) \vDash \phi$

# Key Results

*Theoretical*

1. $PARMODCK(P, n)\ is\ undecidable$
   - By reducing Post's Correspondence Problem to it

2. $PARMODCK(P, n)\ is\ undecidable\ even\ if\ Z = 1$
   - Each node has just one bit of non-determinism available
   - Reduce SDA with $Z \geq 1$ to a SDA with $Z = 1$

3. $Even\ if\ Z = 0, PARMODCK(P, n)\ has\ not\ cutoff$

*Empirical*

1. $Solving\ PARMODCK(P, n)\ by\ reduction\ to\ array - based\ systems$
   - Experimental results with MCMT and CUBICLE

# Post's Correspondence Problem (PCP)

Input : Two sequences of strings $U = \langle u_1, \dots, u_m \rangle$ and $V = \langle v_1, \dots, v_m \rangle$

Solution : sequence of indices $I = \langle i_1, \dots, i_p \rangle$ with each $i_x \in [1, m]$ s.t.

- $u_{i_1} \cdot \cdots \cdot u_{i_p} = v_{i_1} \cdot \cdots \cdot v_{i_p}$

Question: Does a solution exist?

Example 1 : $U = \langle a, ab, bba \rangle$ $V = \langle baa, aa, bb \rangle$

- Solution = ⟨3,2,3,1⟩ : $bba \cdot ab \cdot bba \cdot a = bbaabbbaa = bb \cdot aa \cdot bb \cdot baa$

Example 2 : $U = \langle aa, aab, baaa \rangle$ $V = \langle a, bb, abb \rangle$

- No solution : each $u_i$ longer than corresponding $v_i$

Known to be undecidable in general

- E. L. Post. A variant of a recursively unsolvable problem, 1946

# Result 1: Reducing PCP to PARMODCK (1)

Use nodes to construct a solution

Each node guesses four numbers : $idu, posu, idv, posv$

- Logically, it represents $posu^{th}$ letter of $u_{idu}$ and $posv^{th}$ letter of $v_{idv}$
- Check if this is a legal solution

Example: $U = \langle a, ab, bba \rangle$ $V = \langle baa, aa, bb \rangle$ Solution = $\langle 3,2,3,1 \rangle$

| $id$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | $b$ | $b$ | $a$ | $a$ | $b$ | $b$ | $b$ | $a$ | $a$ |
| $idu$ | – | 3 | 3 | 3 | 2 | 2 | 3 | 3 | 3 | 1 |
| $posu$ | – | 1 | 2 | 3 | 1 | 2 | 1 | 2 | 3 | 1 |
| $idv$ | – | 3 | 3 | 2 | 2 | 3 | 3 | 1 | 1 | 1 |
| $posv$ | – | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 3 |

Node 0 is special. Does the checking.

Solution String

# Result 1: Reducing PCP to PARMODCK (2)

Example: $U = \langle a, ab, bba \rangle$ $V = \langle baa, aa, bb \rangle$ Solution = $\langle 3,2,3,1 \rangle$

| $id$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | $b$ | $b$ | $a$ | $a$ | $b$ | $b$ | $b$ | $a$ | $a$ |
| $idu$ | – | 3 | 3 | 3 | 2 | 2 | 3 | 3 | 3 | 1 |
| $posu$ | – | 1 | 2 | 3 | 1 | 2 | 1 | 2 | 3 | 1 |
| $idv$ | – | 3 | 3 | 2 | 2 | 3 | 3 | 1 | 1 | 1 |
| $posv$ | – | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 3 |

Checks:

$(Round\ 1)\ id \neq 1 \Rightarrow 1 \leq idu \leq m \wedge 1 \leq posu \leq |u_{idu}|$

$(Round\ 1)\ id \neq 1 \Rightarrow 1 \leq idv \leq m \wedge 1 \leq posv \leq |v_{idv}|$

$(Round\ 1)\ id \neq 1 \Rightarrow u_{idu}[posu] = v_{idv}[posv]$

$(Round\ 2)\ \ id = 2 \Rightarrow (posu = 1 \wedge posv = 1)$

$(Round\ 3)\ id > 2 \Rightarrow (if\ I\ start\ a\ string, then\ previous\ node\ ends\ a\ string,$
$else\ previous\ node\ is\ the\ previous\ letter\ in\ my\ string)$

$(Unbounded\ Rounds)\ Sequence\ of\ idu's = Sequence\ of\ idv's$

- $Protocol\ using\ a\ token\ that\ is\ passed\ from\ left\ to\ right$
- $Succeeds\ iff\ the\ two\ sequences\ match$

# Result 2: Undecidability with $Z = 1$

Possible to simulate a $P(n)$ with $Z > 1$ with a $\tilde{P}(Zn)$ with $Z = 1$

Consider the set of nodes of $\tilde{P}$ with id $1,\ Z+1, 2Z+1, \ldots$

- Denote this set of nodes by $\widetilde{N}$

In the first round, every node in $\widetilde{N}$ copies the single non-deterministic bit from the $Z - 1$ nodes following it

- Essentially gives every node in $\widetilde{N}$ access to $Z$ non-deterministic bits

Subsequently every node in $\widetilde{N}$ simulates the corresponding node of $P$

- Other nodes of $\tilde{P}$ stutter

For any specification $\phi$, $PARMODCK(P, \phi) \Leftrightarrow PARMODCK\left(\tilde{P}, \phi\right)$

# Result 3: No Cutoff even with $Z = 0$

**Theorem:** For every $K \in \mathbb{N}$ there exists a specification $\phi$ and a program $P$ with $Z = 0$ such that $P(K) \vDash \phi \wedge P(K+1) \nvDash \phi$.

**Proof:** Consider $P$ where each element of $GV$ is initialized to 0 (completely deterministic) and $\rho$ is:

$$if\,(id > K)\; GV[id] = 2; else\; GV[id] = 1;$$

Consider specification $\phi \coloneqq \forall i. GV[i] \neq 2$. Clearly, $P(n) \vDash \phi \Leftrightarrow n \leq K$.

■

**Open Problem:** Is $PARMODCK(P, \phi)$ decidable when $Z = 0$?

# Empirical Result

Can reduce each $P$ to an array-based system (ABS)

- ABS = ⟨array of arbitrary size, set of guarded commands⟩
- Each step: enabled command selected non-deterministically and applied
  - Command updates one array element
  - Challenge: how to implement a round
    - all elements must be updated

Solution : based on two phase commit protocol

- Implement a “barrier” using “universal guards”
- Implement Two-Phase-Commit using barrier
- Each transaction is a round
- Experimental results (preliminary, more work needed) in paper

# QUESTIONS?

# Contact Information Slide Format

**Sagar Chaki**
Senior Member of Tech Staff
SSD/CSC
Telephone: +1 412-268-1436
Email: chaki@sei.cmu.edu

**U.S. Mail**
Software Engineering Institute
Customer Relations
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

**Web**
www.sei.cmu.edu
www.sei.cmu.edu/contact.cfm

**Customer Relations**
Email: info@sei.cmu.edu
Telephone: +1 412-268-5800
SEI Phone: +1 412-268-5800
SEI Fax: +1 412-268-6257