

Distributed Simulation

Fault Tolerant Simulation with the SimpleSpec Automaton

John P. Daigle

Department of Computer Science
Georgia State University

05.16.06

Outline

- 1 Introduction
 - Problem
 - Core Findings
 - The Model

- 2 Decision Problems, Reducibility and Simulation
 - Introduction
 - Decision Problems
 - Fault Tolerance

Outline

- 1 Introduction
 - Problem
 - Core Findings
 - The Model
- 2 Decision Problems, Reducibility and Simulation
 - Introduction
 - Decision Problems
 - Fault Tolerance

Theory

- Computability (Solvability) is a basic problem in CS
- It is a particular problem in Distributed Computing
 - Influenced by the Number of Processes and Faults
 - ex: The Byzantine Generals Problem
- Simulation is a powerful tool in Computation Theory
- ex: Simulating a multi-tape Turing Machine on a single tape Turing machine proves equivalence between the two models

The BG Distributed Simulation Algorithm

Definition

The BG Distributed Simulation Algorithm allows a set of $f + 1$ processes with up to f failures to wait-free simulate a larger system of n processes, that may also exhibit up to f failures.

wait-free an algorithm in which any non-failing process terminates, regardless of the failure of other processes.

simulation in this case, a program \mathcal{S} is a simulation of a program \mathcal{P} if \mathcal{S} produces the same output as \mathcal{P} for the same input. We are not interested in simulating internal processes.

Application

k-set agreement

The n -process k -set agreement problem is a decision problem in distributed systems. The processes n propose values, and decide on at most k of them. A question about n -process, k -set agreement is whether there is a k -fault-tolerant solution for some n .

- 1 It is already known that no wait-free, k -fault tolerant solution exists for a $k + 1$ -process, k -set agreement problem.
- 2 The BG-simulation algorithm reduces an n process, f failure problem to a $f + 1$ process, f failure problem.
- 3 There is no solution to a k -set agreement, k -failure problem.

This finding is counterintuitive and significant.

Snapshot Shared Memory System

- The algorithms presented are presented in terms of I/O automata.
- The system is assumed to be asynchronous
- The memory model is a *snapshot shared memory*
- Process failure is only by means of a $stop_i$ operation for a process i .

Definition

A snapshot variable is a length n vector (where n is the number of processes), accessed by *update* and *snap* operations.

- 1 *snap* returns the entire vector and can be called by any operation.
- 2 $update(i,r)$ changes the i 'th component of the vector to r , called by process i
- 3 each item i can be a vector of size j , an update can change any j in i

Outline

- 1 Introduction
 - Problem
 - Core Findings
 - The Model
- 2 Decision Problems, Reducibility and Simulation
 - Introduction
 - Decision Problems
 - Fault Tolerance

Decision Problems and Reducibility

A formal statement of the claim:

$$\begin{array}{ccc}
 D & \xrightarrow{\text{reducible}} & D' \\
 \uparrow \text{solves} & & \uparrow \text{solves} \\
 \mathcal{P} & \xrightarrow{\text{simulates}} & \mathcal{P}'
 \end{array}$$

Where D and D' are decision problems, and \mathcal{P} and \mathcal{P}' are systems.

notation

- $R(x)$: There exists y such that $(x, y) \in R$, where R is a relation from X to Y ($X \xrightarrow{R} Y$)
- $R \cdot S$ if $X \xrightarrow{R} Y$ and $Y \xrightarrow{S} Z$, illustrates a relation from X to Z

Formal Definition of an n-port decision problem

Definition

- V an arbitrary set of values
- V^n the set of all vectors w of length $n \mid \forall w[i], i \in V$
- n -port decision problem $D = \langle \mathcal{I}, \mathcal{O}, \Delta \rangle$
 - \mathcal{I} a set of *input vectors*, $\mathcal{I} \subseteq V^n$
 - \mathcal{O} a set of *output vectors*, $\mathcal{O} \subseteq V^n$
 - Δ , a total relation from \mathcal{I} to \mathcal{O}

Example

n -process, k -set agreement problem over a set of values, $V, |V| \geq k + 1$, where V^n is as previously defined

- $\mathcal{I}: V^n$
- \mathcal{O} : Every $v \in V^n \mid v$ contains at most k different values
- $\forall w \in \mathcal{I}, \Delta(w) = v \mid$ each value of $v \in w$

Solving a Decision Problem I

Question

- Let $D = \langle \mathcal{I}, \mathcal{O}, \Delta \rangle$ be an n -port decision problem.
- When does an I/O automaton A solve D ?

Composition of A

- 1 inputs $init(v)_i$, outputs $decide(v)_i$ such that $v \in V \wedge 1 \leq i \leq n$.
- 2 each i is associated to a process of A , $init(v)_i, decide(v)_i$, occur in *port* i .
- 3 A is composed with a user automaton U that submits at most one $init_i$ on each port i . notation: $A \times U$

Solving a Decision Problem II

Definition

Well-formedness A only produces a $decide_i$ if there is a preceding $init_i$, A responds at most once on a given port i

Correct Answers if $init_i$ events occur on all ports, forming $w \in \mathcal{I}$, $decide_i$ completes to $\Delta(w)$

f -failure termination if $init$ events occur on all ports, and $stop$ events occur on at most f ports, $decide$ events occur on all non-failing ports.

If $A \times U$ guarantees well-formedness and correct answers, A solves D . A guarantees f -failure termination provided $A \times U$ satisfies condition 3. A guarantees *wait-free termination* if it guarantees $n - 1$ failure termination.

Fault Tolerant Reducibility I

Concept

Assume n -port decision problem D , n' -port decision problem D' , and an integer f , $0 \leq f \leq n'$. We wish to define:

$$D \xrightarrow{f\text{-reducible}} D'$$

Fault Tolerant Reducibility II

the G Relation

- $G = G(g_1, g_2, \dots, g_n)$ a total relation from \mathcal{I} to \mathcal{I}'
- In the BG-simulation algorithm, a shared memory system \mathcal{P} simulates an f -fault tolerant system \mathcal{P}' that solves D' .
- \mathcal{P} should solve D
- \mathcal{P} obtains $w \in \mathcal{I}$
- each i generates from $w[i][j]$ some vector $g_i(w(i)) \in \mathcal{I}'$
- for each $w \in \mathcal{I}$, there is $G(w) \subseteq \mathcal{I}'$ of possible input vectors for \mathcal{P}'

Fault Tolerant Reducibility III

The F Relation

- a total relation from \mathcal{O}' to $(views_f(\mathcal{O}'))^n$
- notation: if process i fails, $w(i) = \perp$
- for any $w \in V^n$, and $0 \leq f \leq n$ $views_f(w) = V^n \cup \{\top\}$ where at most f components of w are changed to \top
- $F(w) = (views_f(w))^n$

The H Relation

- $H = H(f, h_1, h_2, \dots, h_n)$ a total relation from $(views_f(\mathcal{O}'))^n$ to V^n
- h_i is a function from $views_f(\mathcal{O}')$ to $\mathcal{O}(i)$
- $\forall x \in (views_f(\mathcal{O}'))^n$, $H(x)$ contains exactly the length n vector w , $w(i) = h_i(x(i))$

Fault Tolerant Reducibility IV

Definition

$$\begin{array}{ccccc}
 \mathcal{I} & & \xrightarrow{G} & & \mathcal{I}' \\
 \downarrow \Delta & & & & \downarrow \Delta' \\
 \mathcal{O} & \xleftarrow{H} & F(\mathcal{O}') & \xleftarrow{F} & \mathcal{O}'
 \end{array}$$

D is f -reducible to D' via G, H , written as $D \leq_f^{G,H} D'$, provided that $G \cdot \Delta' \cdot F \cdot H \subseteq \Delta$

Examples

- ① (n, k) -set agreement is f -reducible to (n', k') -set agreement for $k \geq k'$, $f < \min(n, n')$
- ② (n, k) -set agreement is *not* f -reducible to (n', k') -set agreement if $k \leq f < k'$

Fault-tolerant simulation

- \mathcal{P} must simulate \mathcal{P}'
- Simulation can be defined in terms of the *SimpleSpec* Automaton.
- Broadly, if *SimpleSpec* solves D' , \mathcal{P} simulates \mathcal{P}'

SimpleSpec I

Signature

Input:

$init(v)_i, i \in \{1, \dots, n\}$

Output:

$decide(v)_i, i \in \{1, \dots, n\}$

Internal:

$sim - init_j, j \in \{1, \dots, n\}$

$sim - snap_j, j \in \{1, \dots, n'\}$

$sim - update_j, j \in \{1, \dots, n'\}$

$sim - local_j, j \in \{1, \dots, n'\}$

$sim - decide_j, j \in \{1, \dots, n'\}$

SimpleSpec II

States

sim – *mem*, a memory of \mathcal{P}' (an element of $R^{n'}$), initially the initial memory $(r_0)^{n'}$

for each $i \in \{1, \dots, n\}$:

input(i) $\in V \cup \{\perp\}$, initially \perp

reported(i), a Boolean initially *false*

for each $j \in \{1, \dots, n'\}$:

sim – *state*(j), a state of j

sim – *decision*(j) $\in V \cup \{\perp\}$, initially \perp

SimpleSpec III

Transitions I

$init(v)_i$

Effect:

$input(i) := v$

$sim - init_j$

Precondition:

$nextop(sim - state(j)) = \text{"init"}$

for some i

$input(i) \perp$

$v = g_i(input(i))(j)$

Effect:

$sim - state(j) := trans - init(sim - state(j), v)$

SimpleSpec IV

Transitions II

$sim - snap_j$

Precondition:

$$nextop(sim - state(j)) = \text{"snap"}$$

Effect:

$$\begin{aligned} sim - state(j) &:= \\ trans - snap(sim - state(j), sim - mem) \end{aligned}$$

$sim - update_j$

Precondition:

$$nextop(sim - state(j)) = (\text{"update"}, r)$$

Effect:

$$\begin{aligned} sim - state(j) &:= trans(sim - state(j)) \\ sim - mem(j) &:= r \end{aligned}$$

SimpleSpec V

Transitions III

sim – *local*_{*j*}

Precondition:

$$\text{nextop}(\text{sim} - \text{state}(j)) = \text{"local"}$$

Effect:

$$\text{sim} - \text{state}(j) := \text{trans}(\text{sim} - \text{state}(j))$$

sim – *decide*_{*j*}

Precondition:

$$\text{nextop}(\text{sim} - \text{state}(j)) = (\text{"decide"}, v)$$

Effect:

$$\text{sim} - \text{state}(j) := \text{trans}(\text{sim} - \text{state}(j))$$

$$\text{sim} - \text{decision}(j) := v$$

SimpleSpec VI

Transitions IV

decide(v) _{i}

Precondition:

input(i) \top

reported(i) = *false*

w is a “subvector” of *sim* – *decision*

$|w| \geq n' - f$

$v = h_i(w)$

Effect:

reported(i) := *true*

Tasks

Not Used

Fault Tolerance

Suppose \mathcal{P} is an n process shared memory system. \mathcal{P}' is an n process shared memory system, and $0 \leq f \leq n'$. Then \mathcal{P} simulates \mathcal{P}' if:

- 1 \mathcal{P} solves SimpleSpec for \mathcal{P}'
- 2 If \mathcal{P}' guarantees f – *failure* termination, so does \mathcal{P}