Stability of Distributed Algorithms in the face of Incessant Random Faults

Sayan Mitra
R. E. Lee DeVille
University of Illinois at Urbana-Champaign

International Conference on Safety, Security, and Stability of Distributed Systems
Lyon, France November 2009
Given algorithm $A = \langle X, \rightarrow \rangle$ and $L \subseteq X$, $A$ is **self-stabilizing** if:
for every $x_0 \in X$, there exists $n(x_0)$, for every $k > n(x_0)$, for every execution $x_0 \rightarrow x_1 \rightarrow x_2 \ldots x_k$, $x_k \in L$.

**Stabilization time** $ST(A) = \max_{x \in X} n(x)$

If $A$ starts in $L$ then fault-free executions remain in $L$.

Transient failures may corrupt the state and jump outside $L$, but if there are **no further failures** then stabilization guarantees that $L$ is reached within $ST(A)$. 

---

**Diagram:***
- **$L$** and **$L^c$**
- Arrows indicate transitions: **fault** and **recovery**
Stabilization in large systems

This talk is concerned with large distributed systems

• Typically ST ∼ poly(N), where N is # of processes
• MTBF: Mean time between failures
• Large distributed systems ST >> MTBF

MTBF
ST O(N²)
ST O(N)
MTBF

Number of processes (N) →
Stability Analysis under Incessant Faults

- All illegal \((L^c)\) states are not equally bad
  - Graph coloring: more conflicts are worse than fewer conflicts
  - Routing: more nodes with optimal routes is better
  - ...
- **Define a metric on state space** \(Q: X \rightarrow R\)
  - \(Q(x) = 0\) iff \(x \in L\)
  - Higher \(Q(x)\) means “worse” \(x\)

- **Failure rates for process** \(p_i\):
  - \(e_1\): probability that \(p_i\) fails when \(p_i\) is supposed to change its state
  - \(e_2\): probability that \(p_i\) fails when \(p_i\) is not supposed to change

- Algorithm A defines a stochastic process \(A'(e_1, e_2)\)
  - Analyze the behavior of \(A'\) w.r.t. \(Q\)
Outline

Introduction

• Fault model, example
• Analysis with incessant faults
• An observation and an improvement
• Conclusion
Synchronous Shared Memory Algorithms

- Algorithm $p_i = \langle X_i, U_i, \rightarrow \rangle$
  - $X_i$: set of **state variables**, $U_i$: set of **input variables**
  - $\rightarrow_i \subseteq \text{val}(U_i) \times \text{val}(X_i) \times \emptyset(\text{val}(X_i))$: set of **(probabilistic) transitions**

- Example: Dijkstra’s unidirectional token ring
  - $N$ processes $0, 1, \ldots, N-1$
  - $X_i = \{x_i\}$ where $x_i$ is of type $\{0, 1, \ldots, K\}$; $U_i = \{x_{i-1}\}$
  - For $i \neq 0$, if $x_i \neq x_{i-1}$ then $x_i = x_{i-1}$ else $x_i = x_i$
  - For $i=0$, if $x_0 = x_{N-1}$ then $x_0 = x_0 + 1 \mod K$ else $x_0 = x_0$

- Complete system $S = p_1 \parallel p_2 \parallel \ldots \parallel p_{N-1}$
  - a state of $S$: $\mathbf{x}$
  - $\text{token}(\mathbf{x}, i)$: true iff $p_i$ has token in $\mathbf{x}$
  - $\text{hastoken}(\mathbf{x}) = \{ i \mid \text{token}(\mathbf{x}, i) \}$
  - $L(\mathbf{x}) = (|\text{hastoken}(\mathbf{x})| == 1)$
Fault models

• Algorithm $p_i = \langle X_i, U_i, \rightarrow \rangle$

• **Update faults** $e_1 > 0$, $e_2 = 0$, for $i \neq 0$
  • if $x_i \neq x_{i-1}$ then
    - $x_i = x_{i-1}$ with probability $(1-e_1)$
    - $x_i = k$ with probability $e_1/K$, where $k \in \{0,1,\ldots,K\}$
  • else $x_i = x_i$

• **Sleep-Update faults** $e_1 = e_2 > 0$, for $i \neq 0$
  • if $x_i \neq x_{i-1}$ then
    - $x_i = x_{i-1}$ with probability $(1-e_1)$
    - $x_i = k$ with probability $e_1/K$, where $k \in \{0,1,\ldots,K\}$
  • else
    - $x_i = x_i$ with probability $(1-e_1)$
    - $x_i = k$ with probability $e_1/K$, where $k \in \{0,1,\ldots,K\}$
Properties of interest

- Steady State (SS) distribution $\pi$
  $$\pi(\{x \mid \text{hastokens}(x) = a\})$$

- Maximum Expected Recovery Time (MERT)
  $$\max_x \text{ERT}(x), \text{ where } \text{ERT}(x) = \text{expected time to reach } L \text{ from } x$$

- Maximum Expected Holding Time (MEHT)
  maximum expected time before a token disappears from a process
Two kinds of analysis

• Model checking (for small $N$, < 8)
  – Using PRISM model checker

• Asymptotic bounds (for large $N$)
Steady state probabilities with only update faults
Maximum Expected Recovery Time
Update and Sleep-update faults
Large $N$, **update**

**faults** $e_1 = \varepsilon$, $e_2 = 0$

$|\text{hastokens()}|$ grows to $O(N \varepsilon)$ and then drops to 1
Analysis

- $T_t$: Number of tokens at round $t$
- $P_t$: Process ids for processes with tokens at round $t$

- When $p_i$ changes value it goes to a correct value with probability $(1-\varepsilon) + \varepsilon/K$ and an incorrect value with probability $\varepsilon/K$ ($=d$)

- Case $T_t = 1$, $L_t = \{i\}$, $i \neq 0$
  - $P(T_{t+1} = 1) = 1 - (K-2)d$
  - $P(T_{t+1} = 2) = (K-2)d$

- Case $T_t = 1$, $L_t = \{0\}$
  - $P(T_{t+1} = 1) = 1$
Analysis 2

- Case $T_t = q$, $L_t = \{i, i+1, i+2, \ldots, i+q-1\}$, $0 \notin L_t$

- $x_{i-1} \ x_{i} \ x_{i+1} \ x_{i+2} \ x_{i+3} \ \ldots \ x_{i+q-1} \ x_{i+q}$
  - $P(T_{t+1} = q+1) = d(K-2) + O(d^2)$
  - $P(T_{t+1} = q-1) \leq d(2q-1) + O(d^2)$

- Case $T_t = q$ in $g$ separate groups
  - $P(T_{t+1} = q+1) = \epsilon g(K-2)/(K-1) + O(d^2)$
  - $P(T_{t+1} = q-1) \leq \epsilon(2q-1)/(K-1) + O(d^2)$
Analysis 3.

- Define stochastic (birth-death) process $Y_0, Y_1, Y_2, \ldots$ as
- $Y_0 = 1$
- $P(Y_{t+1} = Y_t + 1) = \frac{\varepsilon(K-2)}{(K-1)}$
- $P(Y_{t+1} = Y_t - 1) = \frac{\varepsilon(2q-1)}{(K-1)}$
- $P(Y_{t+1} = Y_t) = 1 - \frac{\varepsilon(K + 2q - 3)}{(K-1)}$

- By Chernoff bound
  - $P(Y_t < T_t) \sim e^{rt}$ for some constant $r < 0$.
  - $Y_N = \frac{N(1-e^{-2\varepsilon})}{2} + o(N) \sim \varepsilon N$

- Starting from a single token at $P_0$, the system generates a token sequence of length $\varepsilon N$ in $N$ steps, with probability exponentially close to 1
Remark on approximate analysis

• The preceding analysis was based on approximating the (complex) stochastic process \( \{T_t\} \) with a much simpler process \( \{Y_t\} \) by conservatively ignoring low probability \( O(d^2) \) transitions.
Modified token ring

state of $p_j x_j \in \{0, \ldots, K-1\}, \ K > N$

$p_0 : \text{if } x_0 = x_{N-1} \text{ then } x_0 := x_0 + 1 \mod K$

$p_j : \text{if } x_j \neq x_{j-1} \text{ and } x_{j-1} = x_{j-2} \text{ then } x_j := x_{j-1}, \text{ for } j > 1$

$O(\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

$O(1-\varepsilon)$

Update errors

mean(tokens)

TR

TR2

$\varepsilon$

$0$ $0.05$ $0.1$ $0.15$ $0.2$ $0.25$ $0.3$
Comparing steady state distributions for original and modified token ring algorithms

Model checking results with PRISM for 5 processes
Graph coloring under U and SU faults
Summary

- **Incessant random fault models** and **metrics** for algorithms seems relevant for stabilization in **large systems**

- **Interaction of algorithm and faults** leads to phenomena and modified algorithms

- Analysis using probabilistic model checking and asymptotic analysis

Future directions

- Comparison of existing algorithms specific robustness metrics in the face of incessant faults

- Lower bounds (w.r.t. robustness to incessant faults) and design of new algorithms

- Verification of algorithms under incessant faults (MCs and MDPs)
Related Work

• Random link failures

• Locally bounded failures
  – Nasterenko and Arora: Local tolerance to unbounded byzantine faults in IEEE SRDS 2002.