Computing the Fault-Containment Time of Self-Stabilizing Algorithms using Markov Chains and Lumping

Volker Turau

19th Int. Symposium on Stabilization, Safety, and Security of Distributed Systems
November 7th, 2017

Institute of Telematics
Hamburg University of Technology

TUHH

Self-Stabilizing Systems

- Self-stabilizing systems provide non-masking fault tolerance
- Critical issue:
  Length of time span and extend of disruption until full recovery
- Surprisingly complexity analysis is usually confined to worst case stabilization time starting from an **arbitrary** configuration
- Considering that these systems are intended to provide fault tolerance in the long run this is not the most relevant metric
- Practical point of view:
  Single fault case is more important than arbitrary configurations!

Metrics for Self-Stabilization

- A configuration is called *k-faulty*, if in a legitimate configuration exactly k nodes are hit by a fault
- Consider a 1-faulty configuration. A self-stabilizing algorithm $\mathcal{A}$ has
  - *contamination radius r*
    if only nodes within r-hop neighborhood of faulty node change state during recovery
  - *containment time t*
    if recovery is completed in at most t rounds

Self-Stabilizing Systems

- Why are contamination radius and containment time rarely considered?
- Lack of techniques?
  - General techniques can also applied
- Contributions
  - Markov chains for computing upper bounds for expected containment time and its variance
  - Application of lumping to reduce complexity of Markov chains

1

Examples

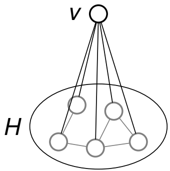
Self-Stabilizing MIS

Algorithm 1: Self-stabilizing algorithm $\mathcal{A}_1$ to compute a MIS

if *state* $=$ *IN* $\wedge \exists w \in N(v)$ *s.t.* $w.state = IN$ **then**
  state $:=$ *OUT*

if *state* $=$ *OUT* $\wedge \forall w \in N(v)$ $w.state = OUT$ **then**
  if *random bit from 0,1 = 1* **then**
    state $:=$ *IN*

- $\mathcal{A}_1$ has contamination radius 2, containment time wcst$(\Delta(G))$



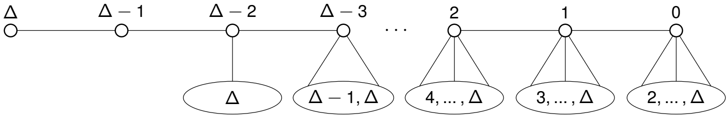
- 1-faulty configurations of $\mathcal{A}_1$ caused by a memory corruption at v changing from *IN* to *OUT*. Containment time is equal to worst case stabilization time for H

Self-Stabilizing $\Delta + 1$-coloring

Algorithm 2: Self-stabilizing $\Delta + 1$-coloring algorithm $\mathcal{A}_2$ [Gra00]

if $c \neq \max\left(\{0, \ldots, \Delta\} \setminus \{w.c \mid w \in N(v)\}\right)$ **then**
  if *random bit from 0,1 = 1* **then**
    $c := \max\left(\{0, \ldots, \Delta\} \setminus \{w.c \mid w \in N(v)\}\right)$

- $\mathcal{A}_2$ has contamination radius and containment time at least $\Delta(G)$



If left-most node is hit by a fault and changes its color to $\Delta - 1$, then all nodes on horizontal line may change color

Self-Stabilizing $\Delta + 1$-coloring

Algorithm 3: Self-stabilizing $\Delta + 1$-coloring algorithm $\mathcal{A}_3$.

if $\exists w \in N(v)$ *s.t.* $c = w.c$ **then**
  if *random bit from 0,1 = 1* **then**
    $c :=$ choose $\{0, \ldots, \Delta\} \setminus \{w.c \mid w \in N(v)\}$

- $\mathcal{A}_3$ has contamination radius 1
- What is the expected containment time?

2

Self-Stabilizing Algorithms and Markov Chains

Markov Chains

- Let $\mathcal{A}$ be a self-stabilizing algorithm, Σ the set of configurations
- $\mathcal{A}$ can be regarded as a Markov chain $\mathcal{C}_\mathcal{A}$ with states Σ, where transition probability from c_i to c_j is equal to $Prob(\mathcal{A}(c_i) = c_j)$
- If $\mathcal{L} \subset \Sigma$ is the set of legitimate configurations of $\mathcal{A}$ then $\mathcal{L}$ is the set of absorbing states of $\mathcal{C}_\mathcal{A}$

Observation

An absorbing state of $\mathcal{C}_\mathcal{A}$ is reached in expected B steps if and only if $\mathcal{A}$ stabilizes in expected B rounds.

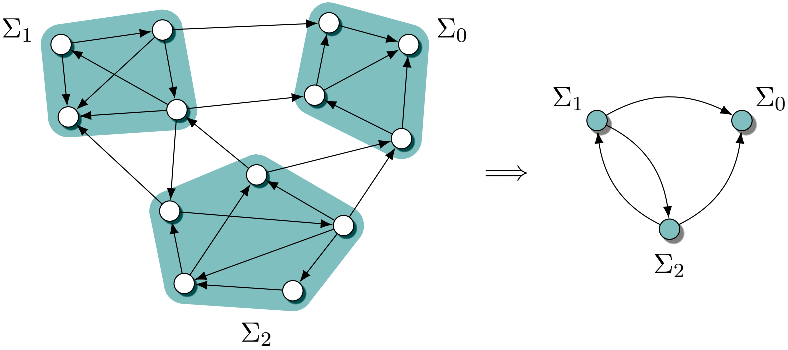
Markov Chains

- Challenges
  - Complexity of Markov chain
  - How to determine $\mathit{Prob}(\mathcal{A}(c_i) = c_j)$?
  - How to compute the expected number of steps?

Containment Time

- Complexity of Markov chain
  - R_v: Subgraph of G induced by nodes engaged in recovery process from a 1-faulty configuration triggered by a fault at v
  - Containment time of $\mathcal{A}$ is equal to stabilization time of $\mathcal{A}$ on R_v
  - Often R_v is much smaller and has a simpler structure than G
- Reduction of complexity of Markov Chain
  - Use lumping to reduce number of states
- How to compute the expected number of steps?
  - Compute fundamental matrix

Lumpable Markov Chains



- What is $Prob(\Sigma_i \to \Sigma_j)$?

Lumpable Markov Chains

- A Markov chain is *lumpable* with respect to partition $P = \{\Sigma_0, \ldots, \Sigma_l\}$ of Σ if for any $\Sigma_i, \Sigma_j \in P$ and any $c_1, c_2 \in \Sigma_i$

$$\sum_{c \in \Sigma_j} p(c_1, c) = \sum_{c \in \Sigma_j} p(c_2, c)$$

- Given a lumpable Markov $\mathcal{C}$ chain define a new Markov chain $\mathcal{C}^P$ with states $\Sigma_0, \ldots, \Sigma_l$ and transition probabilities

$$p(\Sigma_i, \Sigma_j) = \sum_{c \in \Sigma_j} p(c_i, c)$$

Observation

Expected times to reach an absorbing state in $\mathcal{C}$ and $\mathcal{C}^P$ are equal.

Relation of $\mathcal{A}$ and ${\mathcal{C}_{\mathcal{A}}}^{P}$

- Let $\mathcal{A}$ be a self-stabilizing algorithm, Σ the set of configurations
- Let $P = \{\Sigma_0, \ldots, \Sigma_l\}$ be a partition of Σ with $\Sigma_0 = \mathcal{L}$ such that $\mathcal{C}_{\mathcal{A}}$ is lumpable with respect to P

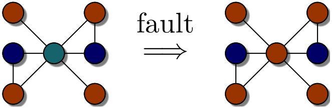
Observation

${\mathcal{C}_{\mathcal{A}}}^{P}$ can be used to calculate the expected containment time of $\mathcal{A}$.

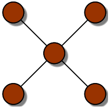
Algorithm $\mathcal{A}_3$ Revisited

if $\exists w \in N(v)$ *s.t.* $c = w.c$ **then**
  if *random bit from 0,1 = 1* **then**
    $c :=$ choose $\{0, \ldots, \Delta\} \setminus \{w.c \mid w \in N(v)\}$

- Consider 1-faulty configuration c_0 where node v has changed its color to c_f causing a conflict



- $R_v = \{w \in N[v] \mid w.c = c_f\}$

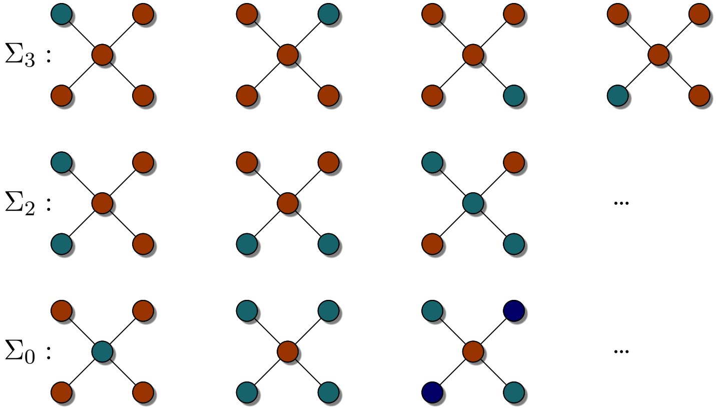


  - R_v is a star graph
  - Only nodes of R_v change their color
  - Nodes choosing a color different from v it become passive

Algorithm $\mathcal{A}_3$ Revisited

- Let Σ_j the configurations reachable from c_0 where exactly j neighbors of v are in conflict with v
- Then $\Sigma_{|R_v|-1} = \{c_0\}$ and $\Sigma_0 \subseteq \mathcal{L}$
- If $c \in \Sigma_i$ then $\mathcal{A}_3(c) \in \Sigma_j$ for some $j \leq i$
- This partitioning is not lumpable
  - Nodes in R_v have different degree in G

States of $\mathcal{C}_{\mathcal{A}_3}$



Algorithm $\mathcal{A}_3$ Revisited

- But, we can make it lumpable!
  - For $j < i$ let $p_{ij} \geq 0$ be a constant with $Prob(\mathcal{A}_3(c) \in \Sigma_j) \geq p_{ij}$ for all $c \in \Sigma_i$
  - Let $p_{ij} = 0$ for $j > i$ and $p_{ii} = 1 - \sum_{j=0}^{i-1} p_{ij}$
  - $P' = (p_{ij})$ describes a new Markov chain ${\mathcal{C}_{\mathcal{A}}}^{P'}$

Observation

Expected number of steps of ${\mathcal{C}_{\mathcal{A}}}^{P'}$ before being absorbed is an upper bound for the expected containment time of $\mathcal{A}_3$.

3

Application

$(\Delta+1)$-Coloring $\mathcal{A}_{col}$

- Conversion of an $(\Delta+1)$-coloring of Barenboim et al. into a self-stabilizing algorithm
- Synchronous $\mathcal{CONGEST}$ model
- Variables
  - c: color of node or $\perp$
  - *final*: is choice of color final

$(\Delta + 1)$-Coloring $\mathcal{A}_{col}$

Algorithm 4: Algorithm $\mathcal{A}_{col}$ as executed by a node v

```
Set&lt;Color&gt; tabu := ∅, occupied := ∅;
broadcast(c, final) to all neighbors w ∈ N(v);
for all neighbors w ∈ N(v) do
    receive(c_w, final_w) from node w;
    if c_w ≠ ⊥ then
        occupied := occupied ∪ {c_w};
        if final_w then  tabu := tabu ∪ {c_w} ;
if c = ⊥ ∨ c > δ(v) then
    final := false;
else
    if final then
        if c ∈ tabu then  final := false ;
    else
        if c ∉ occupied then  final := true ;
if final = false then  c := randomColorOrNull(v, tabu) ;
```

$(\Delta + 1)$-Coloring $\mathcal{A}_{col}$

- $\mathcal{A}_{col}$ is a self-stabilizing $(\Delta + 1)$-coloring algorithm stabilizing in $O(\log n)$ rounds whp in the synchronous model
- With respect to memory and message corruption it has
  - contamination radius 1
  - expected containment time at most $\frac{1}{\ln 2} H_{\Delta_i} + 11/2$ with variance less than 7.5
- Algorithm $\mathcal{A}_{col}$ has expected containment time $O(1)$ for bounded-independence graphs
  - For unit disc graphs this time is at most 8.8

4

Conclusion

Conclusion

- Analysis of self-stabilizing algorithms is often confined to stabilization time starting from arbitrary configurations
- In practice recovery time from 1-faulty configurations more relevant
- Computation of containment time based on Markov chains
- Reduction of complexity with lumping
- Technique applied to a $\Delta + 1$-coloring algorithm yields surprising low bounds

Computing the Fault-Containment Time of Self-Stabilizing Algorithms using Markov Chains and Lumping

19th Int. Symposium on Stabili ystems

Volker Turau

Professor

Phone +49 / (0)40 428 78 3530

e-Mail turau@tuhh.de

http://www.ti5.tu-harburg.de/staff/turau

Institute of Telematics
Hamburg University of Technology

TUHH