Advanced Network Security
2017

Self-stabilisation
Self-stabilisation

- not process failures — but memory failures

these could happen many times

transient errors

error changes the memory content

permanent

network

RAM

data

code

ROM

node
System model

- $n$ nodes
  - uniform: all nodes have the same state & program code
  - non-uniform
  - whether nodes know their identity → if so, it is stored in Root

- communicate through shared memory
  - $G = (V, E)$ $(u, v) \in E$ then these nodes can communicate
  - state reading: $(u, v) \in E$ then $v$ can read $u$'s state (completely)
  - link-register model: $(u, v) \in E$ then $u$ writes a register that $v$ can read

- Configuration $C \in G$: cartesian product of all node states (and link registers if present)
- System \((G, F)\)
  - a node \(i\) has \(f_i \in F\)
  \[
  e' = f_i(c) \leftarrow \text{new step current state}
  \]

- uniformity: \(f_i = f_j\) for \(i, j\)
- known identities: \(f_i(c)\) may depend on \(i\)

- Scheduling
  - central daemon
    - scheduler fairly selects one node \(i\) to take a step: \(C \rightarrow c'\)
  - distributed daemon
    - \(\ldots\) a set of nodes \(I\) to take a step: \(I \rightarrow c'\)
    - all nodes first read their "incoming" states / registers
    - compute the new state
    - and then write the new state \(C\) and the registers
Self-stabilising

L is a global property

but nodes only see local information

ring

Self-stabilising system cannot terminate

a node doesn't know whether the configuration is legitimate

L - legitimate states

bad state

converges

closure

every node is in the critical section

typically defined using a predicate
Some toy problems

- We have one node, and some $R$

  Let $x$ be the state of the node

  \[
  \begin{align*}
  &\text{if } x \in R \rightarrow \text{skip} \\
  &\text{if } x \notin R \rightarrow x \in L
  \end{align*}
  \]

- Suppose there are $n$ nodes, and $G$ is the complete graph, and we are in the state reading model

  Here at least a node can see whether the global configuration is in $R$

  \[
  \begin{align*}
  &\text{if } c \in L \rightarrow \text{skip} \\
  &\text{if } c \notin L \rightarrow ? \text{ a local change may not make } c' \in L
  \end{align*}
  \]

  \[\left(\exists c : J \subsetneq \emptyset\right)\]
Last example

- clock-synchronisation
  - local state $c[i]$ of node $i$ is a local clock
  - you want all clock values to be equal

\[ \Downarrow \]

- complete graph?
- risky?
Mutual exclusion on a ring

One node is privileged

Predicate on local state

Mutual exclusion

Self-stabilisation not possible on rings of non-prime size!
\begin{itemize}
  \item \(N+1\) nodes: 0..N
  \item \(K>N\)
  \item Each node has state \(x[i] \in \{0,..,K-1\}\)
  \item Protocol:
    \begin{align*}
      \text{node } \phi: & \quad \text{if } x[N] = x[0] \rightarrow x[0] \leftarrow x[0]+1 \\
      \text{node } i \neq \phi: & \quad \text{if } x[i-1] \neq x[i] \rightarrow x[i] = x[i-1]
    \end{align*}
  \item Privilege: node \(\phi\) is privileged if \(x[N] = x[0]\)
  \item node \(i\): \(x[i-1] \neq x[i]\)
\end{itemize}

General proof strategy:
\begin{enumerate}
  \item Defining the legitimate states \(L\)
  \item Prove closure: \(c \in L\) and \(c \rightarrow c'\) then \(c' \in L\)
  \item Prove convergence: given \(c \in G\) there is a run \(\sigma\) s.t. \(c' \in L\), \(c \stackrel{\sigma}{\rightarrow} c'\)
\end{enumerate}

Legitimate states are those where we have an \(i\), \(0 \leq i \leq N\)
and \(a \in \{0..K-1\}\) s.t. for all \(j\), \(0 \leq j \leq i\): \(x[j] = a+1 \mod K\).

\[P(i,a): \quad \text{for all } j, \quad i < j \leq N: \quad x[j] = a\]
Proof of correctness
- legitimate states correspond to mutual exclusion
- let's assume a central daemon, and \( K \geq N \)

Closure
- assume system is in a legitimate state, i.e. \( P(i,a) \) holds for some \( i \) and \( a \)
- \( K = i + 1 \mod N \) is enabled
  - if \( k \neq 0 \) then after the step \( x[K] = a + 1 \mod K \), and so \( P(i+1, a) \) holds
  - if \( k = 0 \) then after the step \( x[0] = a + 2 \mod K \) while all other nodes have \( x[j] = a + 1 \mod K \) \( \Rightarrow \)
    \( P(0, a + 1 \mod K) \) holds
Convergence

- the first time node $\varphi$ takes a step: colour it blue
- when a node copies a value from a blue node, it becomes blue
- if all nodes are blue then system is legitimate

Node $\varphi$ is only taking a step if node $N$ has the same value as node $\varphi$.

Node $\varphi$ can take at most $N-1$ steps before node $N$ is blue.

If value is at most $N-1$, and because $N \leq K$ we have no wrap around.