Dijkstra’s Self stabilization

CS 451 Lecture 5
August 12, 2003

Prof. R.K. Joshi
Dept of CSE
IIT Bombay
Self Stabilization

- Self correcting systems
- Local nodes may corrupt their local states
- System heals itself/stabilizes after a bounded number of iterations/steps
Connected network of processors, and their states
Privileges

Boolean functions of its own state and of its neighbours’ states

When such a boolean function is true, we say that a privilege is ‘present’

Example

\[\text{If } (S = L) \text{ then } S = S + 1\]
Selecting Privileges for execution

From among many priviledges which may be present in the entire system, only one is selected for execution.

i.e. imagin a central scheduler that picks up any one privilege from among those present (nondeterministically).

After the execution of selected privilege, the system again continues the same way.
Legitimate State

Is the system as a whole in ‘legitimate state’?

Requirements:
- In each legitimate state, one or more privileges are present.
- In each legitimate state, each possible move brings the system back to a legitimate state.
- Each privilege must be present at least in one legitimate state.
- For any pair of legitimate states, there exists a sequence of moves transferring the system from one into another.
When is a system self stabilizing?

If and only if

Regardless of the privilege selected each time for the next move, at least one privilege is always present

And

The system is a guaranteed to enter legitimate state in a finite number of moves
Do non-trivial Self stabilizing systems exist?

How about all states as legitimate states?

Not obvious whether local moves lead to can assure convergence to global legitimacy

Edsger Dijkstra presented 3 machines in his paper (CACM November 1974)
Machine description

• N+1 machines (0..N)

• L: State of left neighbour
  \[(i-1) \mod (N+1)\]

• R: State of right neighbour
  \[(i+1) \mod (N+1)\]

• S: State of itself
Connected network of processors, and their states

For machine 5, S is 3, L is 4, R is 2
Privilege description

\textbf{If \underline{privilege}}  then \underline{corresponding move}
K State machine with exactly one privilege per machine

**Bottom (machine 0)**
If \( L=S \) then \( S = S + 1 \mod K \)

**Others**
If \( L \neq S \) then \( S = L \)
K State machine

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

Bottom

if (S=L) S=L mod K

Others

if (S≠L) S=L
Who has privileges?

Machine 0

Machine 1

Machine 3

Machine 4

Machine 5

Bottom
if \( S = L \) \( S = L + 1 \mod K \)

Others
if \( S \neq L \) \( S = L \)
Who has privileges?

Is this correct?

Bottom
\[ \text{if}\ (S = L)\ S = L + 1\ \text{mod}\ K\ \text{fi} \]

Others
\[ \text{if}\ (S \neq L)\ S = L\ \text{fi} \]
Who has privileges?

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

\[ N=4 \]

\[ K=5 \]

Bottom

\[ \text{if } (S=L) \text{ then } S=L+1 \mod K \text{ fi} \]

Others

\[ \text{if } (S \neq L) \text{ then } S=L \text{ fi} \]
Let's select one of the privileges nondeterministically through a central daemon
Selecting machine 2's move...

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

N=4
K=5

Bottom
if (S=L) S=L+1 mod K
Others
if (S!=L) S=L
fi
After machine 2’s move

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

Bottom
if \( S = L \) \( S = L + 1 \mod K \)

Others
if \( S \neq L \) \( S = L \)
Selecting machine 5’s move...

Bottom
if \(S = L\) \(S = L + 1 \mod K\) 
Others
if \(S \neq L\) \(S = L\)

\[
\begin{array}{c}
\text{Machine 0} \\
\text{Machine 1} \\
\text{Machine 2} \\
\text{Machine 3} \\
\text{Machine 4} \\
\text{Machine 5}
\end{array}
\]
After machine 5’s move...

Example code for scheduling:

```plaintext
Bottom
if (S=L) S=L+1 mod K fi
Others
if (S≠L) S=L fi
```
Selecting machine 1's move..

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

\[ N = 4 \]
\[ K = 5 \]

\[
\text{Bottom if } (S = L) \quad S = L + 1 \mod K \quad \text{fi}
\]
\[
\text{Others if } (S \neq L) \quad S = L \quad \text{fi}
\]
After machine 1’s move...

```
Bottom: if (S=L) S=L+1 mod K fi
      if (S≠L) S=L fi
```

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5
Selecting machine 4’s move..

Machine 3

Machine 2

Machine 1

Machine 0

Machine 4

Machine 5

N = 4
K = 5

\[
\text{Bottom: } \begin{cases} 
(S \leq L) & S = L + 1 \mod K \\
(S > L) & S = L
\end{cases}
\]

\text{Others: } if \ (S \leq L) \ then \ S = L \ else \ S = L + 1 \mod K
After machine 4’s move...

Machine 0

Machine 2

Machine 3

Machine 4

Machine 5

\( N = 4 \)

\( K = 5 \)

\[ \text{Bottom if } (S = L) \ S = L + 1 \text{ mod } K \ \text{fi} \]

\[ \text{Others if } (S \neq L) \ S = L \ \text{fi} \]
Selecting machine 2’s move...

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

\[
\begin{align*}
\text{Bottom} & \quad \text{if } (S=L) \quad S = L+1 \mod K \\
\text{Others} & \quad \text{if } (S \neq L) \quad S = L
\end{align*}
\]
After machine 2's move...

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

Bottom
if (S=L) S=L mod K
Others
if (S≠L) S=L
Selecting machine 3’s move...

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

N=4
K=5

Bottom:
\[
\text{if } (S=L) \quad S = L + 1 \mod K
\]

Others:
\[
\text{if } (S \neq L) \quad S = L
\]
After machine 3’s move...

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

Bottom

if (S=L) S=L+1 mod K

Others

if (S≠L) S=L

N=4
K=5
After machine 5’s move..

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

N=4
K=5

Bottom
if (S=L) S=L+1 mod K

Others
if (S!=L) S=L
After machine 4’s move...

```
Bottom
if (S=L) S=L+1 mod K
Others
   if (S≠L) S=L
```

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

N=4
K=5
After machine 5's move...

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

Bottom if (S=L) S=L+1 mod K fi

Others if (S≠L) S=L fi
After machine 0’s move...

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

Bottom
if (S=L) S=L+1 mod K fi
Others
if (S≠L) S=L fi
After machine 1's move...

Machine 0

Machine 1

Machine 2

Machine 3

Machine 4

Machine 5

N=4
K=5

Bottom
if (S=L) S=L+1 mod K fi
Others
if (S≠L) S=L fi
The system is trapped in legitimate state

- In this machine the legitimate state was defined as exactly one privilege in the whole system.
- You can observe that this has been reached.
- The privilege will now make rounds in the ring.
- Application- The legitimacy criterion is that of mutual exclusion (only 1 privilege at a time). The algorithm is token ring.
- But notice that the system is self-stabilizing i.e. it recovers from any error state (non-legitimate state) on its own in finite number of steps.
Why does the idea work?

i.e. what’s the intuition behind this machine?

- discussed in the class

Exercise problem: Lift. Is it possible?
Reference Reading for other 2 machines