Case Study 1: Replicated File Management

Source: S. Jajodia & D. Mutchler,
“Dynamic voting algorithms for maintaining
the consistency of a replicated database”
*ACM Trans, Database Systems*, Vol. 15, No. 2,

one copy: ← if failed, then it is not accessible

Availability

\[
\begin{align*}
A(\infty) & \equiv \text{Availability} = \frac{\mu}{\lambda+\mu} \\
A(t) & \equiv P_1(t) = \frac{\mu}{\mu+\lambda} + \frac{\lambda}{\lambda+\mu} e^{-(\mu+\lambda)t} \\
1 - A(t) & \equiv P_0(t) = \frac{\lambda}{\mu+\lambda} + \frac{\lambda}{\lambda+\mu} e^{-(\mu+\lambda)t}
\end{align*}
\]
Can we use replicated copies to improve availability?
consider only the update operations: suppose we have 7 copies

Cannot update just one copy and leave the others unchanged → will create inconsistency problems
Must maintain one-copy illusion to the user
Consistency algorithms for replicated data:

Static: \( n \) copies
(simple voting) *can do update if a majority of \( n \) copies can be reached & updated

Communication failure

This partition can do update

This partition cannot do update

A write quorum

Another write quorum

No partition can do any update

This partition can still do update
Dynamic voting:

- can do update if a majority of current (up-to-date) copies (since the last update) can be found and updated. These majority copies are called in the “major partition”.

Each copy is associated with a set of local variables:
1) version number (VN): to tell if the local copy is current
2) site cardinality (SC): to tell how many copies are current, e.g., if in the last update, 5 copies were updated, then SC = 5

- Communication failure
- No failure
- This partition can do update because 4 is a majority
** All copies within the major partition are updated & the new SC is set to the # of copies in the major partition.

This partition can do update because 2 is a majority of SC=3

No partition can do update. System halts & must wait for repairs to occur.
Reunion Scenarios

Still not a major partition because the # of copies with the highest version # (i.e. 4) is 1 which is not a majority of 2 (the SC associated with the current copy)

A major partition now

No major partition exists

Repair of network partitioning and node failure

Repair of network partitioning
Availability modeling:

**Site-failure only model: there is only one partition**

system models:

1) failure rate of each site is $\lambda$
2) repair rate of each site is $\mu$
3) updates are frequent and there is always an update immediately following a failure/repair.

Static voting: system is available as long as $k$ out of $n$ are available, so the “site availability” is given by:

$$A_{k=\infty} = \sum_{\left[\frac{n}{2}\right]+1}^{n} \frac{k}{n} \binom{n}{k} \left(\frac{\mu}{\lambda + \mu}\right)^k \left(1 - \frac{\mu}{\lambda + \mu}\right)^{n-k}$$

$$\frac{k}{n} = \text{prob}\left\{\text{an update request arrives at one of } k \text{ sites in the major partition}\right\}$$
Dynamic voting: no simple probability expression exists

Resort to Markov modeling

Petri net modeling

state representation

\((X, Y, Z)\)

X of Y current copies are alive

\(:= Y-X\) of Y current copies are down

\(Y = \text{current site cardinality (SC)}\) or

\# of current copies

Z of the n-Y other sites are alive but out-of-date

n: \# of initial copies (e.g., n=7)
Site Availability: \[ A(\infty) = \sum_{i=2}^{n} \left\{ \text{prob}\{(i, i, 0)\} \times \frac{i}{n} \right\} \]

* in state (1,2,0): no update can be performed because 1 is not a majority of 2.

Q: What would be the Markov model for a “distinguished site id” design?
Site availability comparison results:

* **static voting** is better than **dynamic voting** when \( n=3 \)

  - update is permitted in static voting
  - but not permitted in dynamic voting

* when \( n>3 \) **dynamic voting** is better