Network Partitions

Subsets of nodes may be isolated or nodes may be slow in responding
Network Partitions

Causes

Wired network disruptions
  E.g., due to natural disaster

Wireless network obstacles

(Semi-)autonomous nodes

Single node failures undistinguishable from partitions
  E.g., network card fails
Partitions without Replication

If some data is unavailable then stuck
Even if all data is available, must cope with partition during commit protocol
Quorums

\( C_1 = \{ \{ a, b, c \}, \{ a, b, d \}, \{ a, c, d \}, \{ b, c, d \} \} \)
\( A_1 = \{ \{ a, b \}, \{ a, c \}, \{ a, d \}, \{ b, c \}, \{ b, d \}, \{ c, d \} \} \)

Important property
\( X \in C \implies \forall Y \in A: X \cap Y \neq \emptyset \)
\( Y \in A \implies \forall X \in C: X \cap Y \neq \emptyset \)
Quorums

Vote assignments
One way to implement quorums

\[ V_C + V_A > V_T \]

- \( V_C \) votes to commit
- \( V_A \) votes to abort
- \( V_T \) total votes

To commit: \( V_C \geq 3 \)
To abort: \( V_A \geq 2 \)
Quorums

Commit protocols must enforce quorum

If node knows transaction could have committed (aborted), it cannot abort (commit) even if abort (commit) quorum available

All commit protocols are *blocking* (with network partitions)
Quorums

3PC with quorum
To make commit decision: commit quorum
To make abort decision: abort quorum

Example 1
Votes for commit $V_C = 3$
Votes for abort $V_A = 3$

old coordinator ?

new coordinator

$\text{old coordinator} \times \text{new coordinator}$
Quorums

Example 1

Coordinator could not have committed
Have abort quorum
→ Try to abort
Quorums

Example 2

1. PC
2. W

\[ V_C = 3 \]
\[ V_A = 3 \]

old coordinator

new coordinator

?
Quorums

Example 2

Coordinator could not have aborted
Have commit quorum
→ Try to commit
Quorums

Example 3

old coordinator

? ②

① PC

① W

new coordinator

① W

V_C = 3
V_A = 3
Quorums

Example 3

Insufficient votes
→ Block

\[ V_C = 3 \]
\[ V_A = 3 \]
Quorums

Problematic state

① PC
① PA
① W

Possible?
Could the transaction have aborted?
Could the transaction have committed?
What to do next?
Quorums

Problematic state

1. PC
2. PA
3. W

Possible? Yes
Could the transaction have aborted? Yes
Could the transaction have committed? Yes
What to do next? Block
Quorums

Scenario 1
{PC, PA, W} after commit

```plaintext
<table>
<thead>
<tr>
<th>time</th>
<th>1</th>
<th>W</th>
<th>PC</th>
<th>?</th>
<th>?</th>
<th>PC</th>
<th>PC</th>
<th>C</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>PC</td>
<td>C</td>
<td>?</td>
</tr>
<tr>
<td>3</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>PC</td>
<td>PC</td>
<td>PC</td>
</tr>
<tr>
<td>4</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>PA</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>PA</td>
</tr>
</tbody>
</table>
```

___ = coordinator
Quorums

Scenario 2

\{ PC, PA, W \} after abort

<table>
<thead>
<tr>
<th>time</th>
<th>①</th>
<th>①</th>
<th>①</th>
<th>①</th>
<th>①</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>PA</td>
<td>A</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>PA</td>
<td>A</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>PC</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>PC</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>PA</td>
<td>PA</td>
<td>PA</td>
<td>PA</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>W</td>
</tr>
</tbody>
</table>
Quorums

Not all quorums can be implemented via votes

\[ C_2 = \{ \{ a, b \}, \{ c, d \} \} \]
\[ A_2 = \{ \{ a, c \}, \{ a, d \}, \{ b, c \}, \{ b, d \} \} \]
Partitions with Replication

Options

1. All copies are required for updates
2. Groups may update, but at most one at a time
3. Any group may update
Partitions with Replication

At most one operational group

Coteries

Update propagation

Multiple operational groups
Coteries

Key idea
When operational groups change, at least one node should be shared across previous group and current group, so that it can carry over state
Coteries

\[ \begin{align*}
\bullet & a \\
\bullet & b \\
\bullet & c \\
\bullet & d
\end{align*} \]

\[ C_1 = \{ \{ a, b, c \}, \{ a, b, d \}, \{ a, c, d \}, \{ b, c, d \} \} \]
\[ C_2 = \{ \{ a, b \}, \{ a, c \}, \{ a, d \}, \{ b, c, d \} \} \]

**Important property**

\[ X \in C \Rightarrow \forall Y \in C: X \cap Y \neq \emptyset \]

\[ \{ \{ a, b \}, \{ c, d \} \} \text{ not valid} \]
Coteries

Reading replicated data
Can relax coterie requirement

\[ C_1 = \{ \{ a, b, c \}, \{ a, b, d \}, \{ a, c, d \}, \{ b, c, d \} \} \]
\[ R_1 = \{ \{ a, b \}, \{ a, c \}, \{ a, d \}, \{ b, c \}, \{ b, d \}, \{ c, d \} \} \]
\[ C_2 = R_2 = \{ \{ a, b \}, \{ a, c \}, \{ a, d \}, \{ b, c, d \} \} \]
Coteries

Votes

$2 \ V_W > V_T, \ V_W + V_R > V_T$

To write get 3 votes ( $V_W$ )
To read get 2 votes ( $V_R$ )

$C_1 = \{ \{ a, b, c \}, \{ a, b, d \}, \{ a, c, d \}, \{ b, c, d \} \}$
$R_1 = \{ \{ a, b \}, \{ a, c \}, \{ a, d \}, \{ b, c \}, \{ b, d \}, \{ c, d \} \}$
Coteries

Votes

\[ 2 \ V_W > V_T, \ V_W + V_R > V_T \]

To write get 3 votes ( \( V_W \) )
To read get 2 votes ( \( V_R \) )

\[ C_2 = R_2 = \{ \{ a, b \}, \{ a, c \}, \{ a, d \}, \{ b, c, d \} \} \]
Coteries

Which one is better?

\[ C_1 = \{ \{ a, b, c \}, \{ a, b, d \}, \{ a, c, d \}, \{ b, c, d \} \} \]
\[ R_1 = \{ \{ a, b \}, \{ a, c \}, \{ a, d \}, \{ b, c \}, \{ b, d \}, \{ c, d \} \} \]

\[ C_2 = R_2 = \{ \{ a, b \}, \{ a, c \}, \{ a, d \}, \{ b, c, d \} \} \]
Coteries

Not all coteries have vote assignments
Coteries

Not all coteries have vote assignments

Example

Nodes \{ a, b, c, d, e, f \}

\[ C = \{ \{ a, b \}, \{ a, c, d \}, \{ a, c, e \}, \{ a, d, f \}, \{ a, e, f \}, \{ b, c, f \}, \{ b, d, e \} \} \]
Update Propagation

Problem

Example

Now

T₁ is committed at a, b

T₁

b

c
Update Propagation

Problem

Example

Later

\[ \begin{array}{c}
\bullet b \\
T_1 \cdot a \\
\bullet c
\end{array} \]

\[ T_2 \text{ reads at } c \text{ (not seeing } T_1) \]

Then writes and commits at a, c
Update Propagation

Solution

Each node keeps list of committed transactions

Compare list at read site with those at write sites

Update sites that missed transactions
Update Propagation

Solution

Example revisited

\[ T_0, T_1 \cdot a \]

\[ T_0, T_1 \cdot b \]

\[ \cdot c \]

\[ T_0 \]
Update Propagation

Solution
Example revisited

- $T_0, T_1 \bullet a$
- $T_0, T_1 \bullet b$
- $\bullet c$
- $T_0$

$T_2$ coordinator

List = $T_0$

read
Update Propagation

Solution

Example revisited

\[
\begin{align*}
T_0, T_1 & \longrightarrow a \\
T_0, T_1 & \longrightarrow b \\
T_0 & \longrightarrow c \\
\end{align*}
\]

exec (list = \(T_0\))

exec (list = \(T_0\))
Update Propagation

Solution

Example revisited

Get new data ($T_1$) from a
Update Propagation

Each node must keep updates for transactions until all nodes have seen them
Multiple Operational Groups

\[ \text{DB}_0 \rightarrow \text{DB}_1 \rightarrow \text{DB}_3 \rightarrow \text{DB}_4 \]

\[ \text{DB}_0 \rightarrow \text{DB}_1 \rightarrow \text{DB}_2 \rightarrow \text{DB}_4 \]
Multiple Operational Groups

Integration options
1. Compensate transactions to make schedules match
2. Data patch (i.e., apply semantic fix)
Multiple Operational Groups

Compensation

Example

```
T₀, T₁, T₂

DB₁

T₀, T₃, T₄

DB₂

DB₃
```

T₀
Multiple Operational Groups

Compensation

Assume $T_1$ commutes with $T_3$ and $T_4$

E.g., no conflicting operations

$DB_2$

Schedule $T_0, T_3, T_4, T_1$ is equivalent to $T_0, T_1, T_3, T_4$

$DB_3$

Schedule $T_0, T_1, T_2, T_2^{-1}, T_3, T_4$ is equivalent to $T_0, T_1, T_3, T_4$
Multiple Operational Groups

Compensation

\[ \text{T}_0, \text{T}_1, \text{T}_2 \]

\[ \text{DB}_1 \rightarrow \text{DB}_3 \]

\[ \text{T}_0, \text{T}_3, \text{T}_4 \]

\[ \text{DB}_2 \rightarrow \text{DB}_4 \]

\[ \text{T}_0, \text{T}_1, \text{T}_3, \text{T}_4 \]
Multiple Operational Groups

Compensation

In general, based on characteristics of transactions, schedules can be *merged*.
Multiple Operational Groups

Data patch

Forget about schedules
Integrate differing values via *rules*

\[
\begin{array}{c|c|c|c}
\hline
& x & y & z \\
\hline
x & a & & \\
y & b & ts=10 & \\
z & & & 5 \\
\hline
\end{array}
\]

both copies

\[
\begin{array}{c|c}
\hline
& x \\
\hline
y & d \mid ts=11 \\
z & 6 \\
\hline
\end{array}
\]

site 1

\[
\begin{array}{c|c}
\hline
& x \\
\hline
y & e \mid ts=12 \\
z & 6 \\
\hline
\end{array}
\]

site 2
Multiple Operational Groups

Data patch

E.g., simple rules

For x site 1 wins
For y latest timestamp wins
For z add increments
Multiple Operational Groups

Data patch
Summary

**Partitions without replication**
Abort, commit quorums
3PC with quorums

**Partitions with replication**
At most one operational group
  - Coteries
  - Update propagation
Multiple operational groups