Distributed Transactional Memory for General Networks

Gokarna Sharma
Costas Busch
Srivathsan Srinivasagopalan

Louisiana State University

May 24, 2012
Distributed Transactional Memory

• Transactions run on network nodes

• They ask for shared objects distributed over the network for either read or write

• They appear to execute *atomically*

• The reads and writes on shared objects are supported through three operations:
  - Publish
  - Lookup
  - Move
Suppose the object $\xi$ is at node $\bullet$ and $\bullet$ is a requesting node.

Suppose transactions are immobile and the objects are mobile.
Lookup operation

Replicates the object to the requesting node
Lookup operation

Replicates the object to the requesting nodes
Move operation

Relocates the object explicitly to the requesting node
Move operation

Relocates the object explicitly to the requesting node
Need a **distributed directory protocol**

- To provide objects to the requesting nodes efficiently implementing Publish, Lookup, and Move operations
- To maintain consistency among the shared object copies
## Existing Approaches

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Stretch</th>
<th>Network Kind</th>
<th>Runs on</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow [DISC'98]</td>
<td>$O(S_{ST})=O(D)$</td>
<td>General</td>
<td>Spanning tree</td>
</tr>
<tr>
<td>Relay [OPODIS'09]</td>
<td>$O(S_{ST})=O(D)$</td>
<td>General</td>
<td>Spanning tree</td>
</tr>
<tr>
<td>Combine [SSS'10]</td>
<td>$O(S_{OT})=O(D)$</td>
<td>General</td>
<td>Overlay tree</td>
</tr>
<tr>
<td>Ballistic [DISC'05]</td>
<td>$O(\log D)$</td>
<td>Constant-doubling dimension</td>
<td>Hierarchical directory with independent sets</td>
</tr>
</tbody>
</table>

- D is the diameter of the network kind
- $S_*$ is the stretch of the tree used
**Scalability Issues/Race Conditions**

**Locking is required**

- Level \( k \) probes left to right
- Level \( k+1 \)

Ballistic configuration at time \( t \)

- From root
- Lookup from \( C \) is probing parent(\( B \)) at \( t \)
- Move parent(\( A \))
- Lookup parent(\( A \))
- Parent(\( A \))
- Object
**Spiral** directory protocol for general networks with $O(\log^2 n \cdot \log D)$ stretch avoiding race conditions
In The Remaining...

- Model
- Hierarchical Directory Construction
- How \textit{Spiral} Supports \textit{Publish}, \textit{Lookup}, and \textit{Move}
- Analogy to a Distributed Queue
- \textit{Spiral} Hierarchy Parameters and Analysis
  - Lookup Stretch
  - Move Stretch
- Discussion
Model

- General network $G = (V,E)$ of $n$ reliable nodes with diameter $D$

- One shared object

- Nodes receive-compute-send atomically

- Nodes are uniquely identified

- Node $u$ can send to node $v$ if it knows $v$

- One node executes one request at a time
Spiral Approach: Hierarchical clustering

Network graph
Spiral Approach: Hierarchical clustering

Alternative representation as a hierarchy tree with leader nodes
At the lowest level (level 0) every node is a cluster.

Directories at each level cluster, downward pointer if object locality known.
A Publish operation

➢ Assume that ξ is the creator of ξ which invokes the Publish operation
➢ Nodes know their parent in the hierarchy
Send request to the leader
Continue up phase

Sets downward pointer while going up
Continue up phase

Sets downward pointer while going up
Root node found, stop up phase
A successful Publish operation

Predecessor node $\xi$
Supporting a Move operation

- Initially, nodes point downward to object owner (predecessor node) due to Publish operation
- Nodes know their parent in the hierarchy
Send request to leader node of the cluster upward in hierarchy
Continue up phase until downward pointer found

Sets downward path while going up
Continue up phase

Sets downward path while going up
Continue up phase

Sets downward path while going up
Downward pointer found, start down phase

Discards path while going down
Continue down phase

Discards path while going down
Continue down phase

Discards path while going down
Predecessor reached, object is moved from node ● to node ●

Lookup is similar without change in the directory structure and only a read-only copy of the object is sent
Distributed Queue

root

head
tail
Distributed Queue

root

v

u

head

tail

u v
Distributed Queue

root

u v w

head

tail
Distributed Queue
Distributed Queue
Spiral is Starvation Free

All requests terminate.

- There is always a path of downward pointers from the root node to a leaf node.
- No set of finite number of requests whose successor links form a cycle.
- All the requests terminate in a bounded amount of time.
Spiral avoids Race condition

- Do not need to lock simultaneously multiple parent nodes in the same label.
- Label all the parents in each level and visit them in the order of the labels.
Spiral Hierarchy

- \((O(\log n), O(\log n))\)-labeled sparse cover hierarchy constructed from \(O(\log n)\) hierarchical partitions

- Level 0, each node belongs to exactly one cluster

- Level \(h\), all the nodes belong to one cluster with root \(r\)

- Level \(0 < i < h\), each node belongs to exactly \(O(\log n)\) clusters which are labeled different
Spiral Hierarchy

- How to find a predecessor node?
  - Via spiral paths for each leaf node $u$ by visiting leaders of all the clusters that contain $u$ from level 0 to the root level.

The hierarchy guarantees:

1. For any two nodes $u, v$, their spiral paths $p(u)$ and $p(v)$ meet at level $\min\{h, \log(\text{dist}(u,v)) + 2\}$

2. $\text{length}(p_i(u))$ is at most $O(2^i \log^2 n)$
p(v) is a (canonical) downward path
If there is no Move, a Lookup $r$ from $w$ finds downward path to $v$ in level $\log \text{dist}(u,v)) + 2 = O(i)$.

When there are Moves, it can be shown that $r$ finds downward path to $v$ in level $k = O(i + \log \log^2 n)$.

$$C(r)/C^*(r) = O(2^k \log^2 n) + O(2^k \log n) + O(2^i \log^2 n) / 2^{i-1} = O(\log^4 n)$$
Analysis: move Stretch

Assume a sequential execution $R$ of $l+1$ Move requests, where $r_0$ is an initial Publish request.

\[
C^*(R) \geq \max_{1 \leq k \leq h} (S_k - 1) \cdot 2^{k-1}
\]

\[
C(R) \geq \sum_{k=1}^{h} (S_k - 1) \cdot O(2^k \log^2 n)
\]

Thus,

\[
C(R)/C^*(R) = \frac{\sum_{k=1}^{h} (S_k - 1) \cdot O(2^k \log^2 n)}{\max_{1 \leq k \leq h} (S_k - 1) \cdot 2^{k-1}}
\]

\[
= O(\log^2 n \cdot h) \cdot \frac{\max_{1 \leq k \leq h} (S_k - 1) \cdot 2^{k-1}}{\max_{1 \leq k \leq h} (S_k - 1) \cdot 2^{k-1}}
\]

\[
= O(\log^2 n \cdot \log D)
\]
Summary

- A distributed directory protocol **Spiral** for general networks that
  - Has poly-logarithmic stretch
  - Is starvation free
  - Avoids race conditions
  - Factors in the stretch are mainly due to the parameters of the hierarchical clustering
Thank you!!!