Concurrent Self-Adjusting Distributed Tree Networks

Bruna Peres
Olga Goussevskaia
Stefan Schmid
Chen Avin

UFMG
UNIVERSIDADE FEDERAL DE MINAS GERAIS

AALBORG UNIVERSITY
DENMARK
Motivation

• New technologies allow communication networks to be increasingly flexible and reconfigurable

• Traditional networks designs are still optimized toward static metrics
• ProjecToR: Agile Reconfigurable Data Center Interconnect. Ghobadi et al., SIGCOMM'16
Self-Adjusting Data Structures

• Self-adjusting networks ↔ self-adjusting data structures

• Splay Trees
S. Schmid, et al., SplayNet: Towards Locally Self-Adjusting Networks
*IEEE/ACM Transactions on Networking, 2016.*
SplayNets

• Distributed tree network

• Improves the communication cost between two nodes in a self-adjusting manner

• Nodes communicating more frequently become topologically closer to each other over time

• *Lowest common ancestor* $LCA(u,v)$: locality is preserved
Our Contributions

• While SplayNets are inherently intended to distributed applications, so far, only sequential algorithms are known to maintain SplayNets

• We present DiSplayNets, the first distributed and concurrent implementation of SplayNets
• Network model:
  • Binary tree $T$ comprised of a set of $n$ communication nodes

• Sequence of communication requests $\sigma = (\sigma_1, \sigma_2, ..., \sigma_m)$:
  • $\sigma_i = (s, d)$
  • $t_b(\sigma_i)$ and $t_e(\sigma_i)$

• Given $\sigma_i(s, d)$, s and d rotate in parallel towards the LCA(s,d)
  • LCA might change over time
DiSplayNet

- State machine executed by each node in parallel
State machine executed by each node in parallel
DiSplayNet

- State machine executed by each node in parallel
Local reconfigurations

Concurrent Self-Adjusting Distributed Tree Networks

DISC 2017

12
Local reconfigurations

\[ \beta(u) \]

Diagram showing a zig-zig reconfiguration in a distributed tree network.
In order to ensure deadlock and starvation freedom, concurrent operations are performed according to a priority

\[ t_b(\sigma_i(s_i, d_i)) < t_b(\sigma_j(s_j, d_j)) \]
The algorithm is executed in rounds

\[ t_b(\sigma_i(s_i, d_i)) < t_b(\sigma_j(s_j, d_j)) \]
Algorithm

- The algorithm is executed in rounds

\[ t_b(\sigma_i(s, d)) < t_b(\sigma_j(s, d)) \]
The algorithm is executed in rounds

\[ t_b(\sigma_i(s_i, d_i)) < t_b(\sigma_j(s_j, d_j)) \]
Phase 4

- The algorithm is executed in rounds

\[ t_b(\sigma_i(s_i, d_i)) < t_b(\sigma_j(s_j, d_j)) \]
• The algorithm is executed in rounds

\[ t_b(\sigma_i(s_i, d_i)) < t_b(\sigma_j(s_j, d_j)) \]
• Self-adjust to the communication pattern in a fully-decentralized manner

  • Starvation free

  • Deadlock free
Future Work

• Analyze the efficiency

• Work cost: \( W(\text{DiSplayNet}, T_0, \sigma) = \sum_{\sigma_i \in \sigma} w(\sigma_i) \)

• Time cost:
  • Request delay: \( t_d(\sigma_i) = t_e(\sigma_i) - t_b(\sigma_i) \)
  • Makespan: \( T(T_0, \sigma) = \max_{\sigma_i \in \sigma} t_e(\sigma_i) - \min_{\sigma_i \in \sigma} t_b(\sigma_i) \)
Progress Matrix

\[ t_i \]

\[ t_{i+1} \]

\[ t_{i+2} \]

\[ t_{i+3} \]

\[ t_{i+4} \]

\[
\begin{array}{cccccccccccc}
\text{ } & t_1 & t_2 & \ldots & t_i & t_{i+1} & t_{i+2} & t_{i+3} & t_{i+4} & t_{i+5} & \ldots & t_j & \ldots & t_k \\
s_1 & \checkmark & \checkmark & \ldots & \checkmark & \checkmark & \checkmark & \checkmark & \checkmark & \checkmark & \ldots & - & \ldots & - \\
d_1 & \checkmark & \checkmark & \ldots & \checkmark & \checkmark & \checkmark & \checkmark & \checkmark & \checkmark & \ldots & - & \ldots & - \\
s_2 & \checkmark & \checkmark & \ldots & \checkmark & \checkmark & \checkmark & \checkmark & - & - & \ldots & - & \ldots & - \\
d_2 & \checkmark & \checkmark & \ldots & \checkmark & \checkmark & X & \checkmark & - & - & \ldots & - & \ldots & - \\
s_3 & \checkmark & \checkmark & \ldots & X & X & X & X & \checkmark & \checkmark & \ldots & \checkmark & \ldots & - \\
d_3 & X & X & \ldots & X & X & X & X & X & \checkmark & \ldots & \checkmark & \ldots & - \\
\end{array}
\]
## Progress Matrix

<table>
<thead>
<tr>
<th></th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$\ldots$</th>
<th>$t_i$</th>
<th>$t_{i+1}$</th>
<th>$t_{i+2}$</th>
<th>$t_{i+3}$</th>
<th>$t_{i+4}$</th>
<th>$t_{i+5}$</th>
<th>$\ldots$</th>
<th>$t_j$</th>
<th>$\ldots$</th>
<th>$t_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>$-$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$d_1$</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>$-$</td>
<td>$\ldots$</td>
<td>$-$</td>
</tr>
<tr>
<td>$s_2$</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>$-$</td>
<td>$\ldots$</td>
<td>$-$</td>
</tr>
<tr>
<td>$d_2$</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>$-$</td>
<td>$-$</td>
<td>$\ldots$</td>
<td>$-$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$s_3$</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>✓</td>
<td>✓</td>
<td>$\ldots$</td>
<td>✓</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$d_3$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\ldots$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\ldots$</td>
<td>✓</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

### Makespan

The makespan is the duration of the longest path through the matrix, representing the time it takes for all tasks to be completed.

### Work

The work is the total amount of tasks assigned to each time slot, indicating the load on the system at each time point.
Future Work

- Our simulations show first promising results
  - ProjecToR data
    - 128 node randomly selected from 2 production clusters (running a mix of workloads, including MapReduce-type jobs, index builders, and database and storage systems)
  - 1000 requests
  - Poisson process
Future Work

- Our simulations show first promising results
- Individual work CDF
Our simulations show first promising results

Total work

<table>
<thead>
<tr>
<th>Work</th>
<th>DisPlayNet</th>
<th>SplayNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\mu=5)</td>
<td>2254</td>
<td>2358</td>
</tr>
<tr>
<td>(\mu=7)</td>
<td>2229</td>
<td>2424</td>
</tr>
<tr>
<td>(\mu=10)</td>
<td>2380</td>
<td>2317</td>
</tr>
<tr>
<td>(\mu=15)</td>
<td>2345</td>
<td>2470</td>
</tr>
<tr>
<td>(\mu=20)</td>
<td>2258</td>
<td>2213</td>
</tr>
</tbody>
</table>
Future Work

• Our simulations show first promising results
  • Request delay
Future Work

- Our simulations show first promising results
  - Makespan

![Makespan Chart]

DISC 2017
Concurrent Self-Adjusting Distributed Tree Networks
Thank you

Bruna Peres
bperes@dcc.ufmg.br