Optimal Closest Policy
with QoS and Bandwidth Constraints
for Placing Replicas in Tree Networks

Veronika Rehn-Sonigo

GRAAL team, LIP
École Normale Supérieure de Lyon
France

August 2007
Introduction and motivation

- Replica placement in tree networks
- Set of clients (tree leaves): requests with QoS constraints, known in advance
- Internal nodes may be provided with a replica; in this case they become servers and process requests (up to their capacity limit)

How many replicas required?
Which locations?
Total replica cost?
Introduction and motivation

- Replica placement in tree networks
- Set of clients (tree leaves): requests with QoS constraints, known in advance
- Internal nodes may be provided with a replica; in this case they become servers and process requests (up to their capacity limit)

How many replicas required?
Which locations?
Total replica cost?
Introduction and motivation

- Replica placement in tree networks
- Set of clients (tree leaves): requests with QoS constraints, known in advance
- Internal nodes may be provided with a replica; in this case they become servers and process requests (up to their capacity limit)

How many replicas required?
Which locations?
Total replica cost?
Replica placement in tree networks

Set of clients (tree leaves): requests with QoS constraints, known in advance

Internal nodes may be provided with a replica; in this case they become servers and process requests (up to their capacity limit)

How many replicas required? Which locations? Total replica cost?
Rule of the game

- Handle all client requests, and minimize cost of replicas
  - \( \Rightarrow \text{Replica Placement problem} \)
  - Several policies to assign replicas

\[ W = 10 \]
Rule of the game

- Handle all client requests, and minimize cost of replicas
- → Replica Placement problem
- Several policies to assign replicas

$W = 10$

Diagram:

```
       5
      /|
     / 2
   4/    3
   /      \
  1/       2
  / 3
```
Rule of the game

- Handle all client requests, and minimize cost of replicas
- $\rightarrow$ Replica Placement problem
- Several policies to assign replicas
Rule of the game

- Handle all client requests, and minimize cost of replicas
- \textbf{Replica Placement} problem
- Several policies to assign replicas

\begin{itemize}
\item \textit{Closest}
\end{itemize}
Rule of the game

- Handle all client requests, and minimize cost of replicas
- \( \rightarrow \) Replica Placement problem
- Several policies to assign replicas
Rule of the game

- Handle all client requests, and minimize cost of replicas
- \( \rightarrow \text{Replica Placement problem} \)
- Several policies to assign replicas

\( W = 10 \)

Multiple
Outline

1. Framework

2. Complexity results

3. Optimal Replica Placement Algorithm

4. Conclusion
Outline

1. Framework
2. Complexity results
3. Optimal Replica Placement Algorithm
4. Conclusion
Definitions and notations

- **Distribution tree** $\mathcal{T}$, clients $\mathcal{C}$ (leaf nodes), internal nodes $\mathcal{N}$

- **Client** $v \in \mathcal{C}$:
  - Sends $r(v)$ requests per time unit (number of accesses to a single object database)
  - Quality of service $q(v)$ (response time)

- **Node** $j \in \mathcal{N}$:
  - Can contain the object database replica (server) or not
  - Processing capacity $W$
  - Storage cost $sc_j$

- **Tree edge**: $l \in \mathcal{L}$ (communication link between nodes)
  - Communication time $\text{comm}_l$
  - Bandwidth limit $\text{BW}_l$
Definitions and notations

- Distribution tree $\mathcal{T}$, clients $\mathcal{C}$ (leaf nodes), internal nodes $\mathcal{N}$
- **Client** $v \in \mathcal{C}$:
  - Sends $r(v)$ requests per time unit (number of accesses to a single object database)
  - Quality of service $q(v)$ (response time)
- **Node** $j \in \mathcal{N}$:
  - Can contain the object database replica (server) or not
  - Processing capacity $W$
  - Storage cost $sc_j$
- **Tree edge**: $l \in \mathcal{L}$ (communication link between nodes)
  - Communication time $\text{comm}_l$
  - Bandwidth limit $\text{BW}_l$
Definitions and notations

- Distribution tree $\mathcal{T}$, clients $\mathcal{C}$ (leaf nodes), internal nodes $\mathcal{N}$
- **Client** $v \in \mathcal{C}$:
  - Sends $r(v)$ requests per time unit (number of accesses to a single object database)
  - Quality of service $q(v)$ (response time)
- **Node** $j \in \mathcal{N}$:
  - Can contain the object database replica (server) or not
  - Processing capacity $W$
  - Storage cost $sc_j$
- **Tree edge**: $l \in \mathcal{L}$ (communication link between nodes)
  - Communication time $\text{comm}_l$
  - Bandwidth limit $\text{BW}_l$
Definitions and notations

- Distribution tree $\mathcal{T}$, clients $\mathcal{C}$ (leaf nodes), internal nodes $\mathcal{N}$

- **Client** $v \in \mathcal{C}$:
  - Sends $r(v)$ requests per time unit (number of accesses to a single object database)
  - Quality of service $q(v)$ (response time)

- **Node** $j \in \mathcal{N}$:
  - Can contain the object database replica (server) or not
  - Processing capacity $W$
  - Storage cost $sc_j$

- **Tree edge**: $l \in \mathcal{L}$ (communication link between nodes)
  - Communication time $\text{comm}_l$
  - Bandwidth limit $\text{BW}_l$
Problem instances (1/2)

- Goal: place replicas to process client requests
- Client \( i \in C \): \( \text{Servers}(i) \subseteq \mathcal{N} \) set of servers responsible for processing its requests
- \( r_{i,s} \): number of requests from client \( i \) processed by server \( s \)
  \[ (\sum_{s \in \text{Servers}(i)} r_{i,s} = r_i) \]
- \( R = \{ s \in \mathcal{N} | \exists i \in C, s \in \text{Servers}(i) \} \): set of replicas
Problem instances (2/2)

- Minimize $\sum_{s \in R} sc_s$ under the constraints:
  
  - **Server capacity:** $\forall s \in R, \sum_{i \in C | s \in \text{Servers}(i)} r_{i,s} \leq W_s$
  
  - **QoS:** $\forall i \in C, \forall s \in \text{Servers}(i), \sum_{l \in \text{path}[i \to s]} \text{comm}_l \leq q_i$
  
  - **Link capacity:** $\forall l \in L \sum_{i \in C, s \in \text{Servers}(i) | l \in \text{path}[i \to s]} r_{i,s} \leq BW_l$
  
- Restrict to case where $sc_s = W$:
  
  **Replica Counting** problem on homogeneous platforms.
Problem instances (2/2)

- Minimize $\sum_{s \in R} sc_s$ under the constraints:
  - **Server capacity:** $\forall s \in R, \sum_{i \in C} |s \in \text{Servers}(i)\ r_{i,s} \leq W_s$
  - **QoS:** $\forall i \in C, \forall s \in \text{Servers}(i), \sum_{l \in \text{path}[i\rightarrow s]} \text{comm}_l \leq q_i$
  - **Link capacity:** $\forall l \in L \sum_{i \in C, s \in \text{Servers}(i)} |l \in \text{path}[i\rightarrow s]\ r_{i,s} \leq BW_l$

- Restrict to case where $sc_s = W$:
  **Replica Counting** problem on homogeneous platforms.

Veronika.Rehn@ens-lyon.fr
August 2007
Problem instances (2/2)

- Minimize \( \sum_{s \in R} sc_s \) under the constraints:

  - **Server capacity:** \( \forall s \in R, \sum_{i \in C} |s \in \text{Servers}(i) | r_{i,s} \leq W_s \)
  
  - **QoS:** \( \forall i \in C, \forall s \in \text{Servers}(i), \sum_{l \in \text{path}[i \rightarrow s]} \text{comm}_l \leq q_i \).
  
  - **Link capacity:** \( \forall l \in L, \sum_{i \in C} |s \in \text{Servers}(i) | l \in \text{path}[i \rightarrow s] | r_{i,s} \leq \text{BW}_l \)

- Restrict to case where \( sc_s = W \):
  
  **Replica Counting** problem on homogeneous platforms.
Outline

1. Framework
2. Complexity results
3. Optimal Replica Placement Algorithm
4. Conclusion
Complexity results

**Homogeneous platform**: Replica Counting problem, no bandwidth constraints

<table>
<thead>
<tr>
<th></th>
<th>No QoS</th>
<th>With QoS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Closest</strong></td>
<td>polynomial [Cidon02,Liu06]</td>
<td>polynomial [Liu06]</td>
</tr>
<tr>
<td><strong>Upwards</strong></td>
<td>NP-complete [Be06]</td>
<td>NP-complete [Be06]</td>
</tr>
<tr>
<td><strong>Multiple</strong></td>
<td>polynomial [Be06]</td>
<td>NP-complete [Be07]</td>
</tr>
</tbody>
</table>

**Heterogeneous platforms**: all problems are NP-complete

**New result**: Homogeneous platforms with bandwidth and QoS constraints: Closest remains polynomial [Re07]
Complexity results

**Homogeneous platform:** Replica Counting problem, no bandwidth constraints

<table>
<thead>
<tr>
<th></th>
<th>No QoS</th>
<th>With QoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Closest</td>
<td>polynomial [Cidon02,Liu06]</td>
<td>polynomial [Liu06]</td>
</tr>
<tr>
<td>Upwards</td>
<td>NP-complete [Be06]</td>
<td>NP-complete [Be06]</td>
</tr>
<tr>
<td>Multiple</td>
<td>polynomial [Be06]</td>
<td>NP-complete [Be07]</td>
</tr>
</tbody>
</table>

**Heterogeneous platforms:** all problems are NP-complete

**New result:**
Homogeneous platforms with bandwidth and QoS constraints: Closest remains polynomial [Re07]
**Homogeneous platform:** Replica Counting problem, no bandwidth constraints

<table>
<thead>
<tr>
<th></th>
<th>No QoS</th>
<th>With QoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Closest</td>
<td>polynomial [Cidon02,Liu06]</td>
<td>polynomial [Liu06]</td>
</tr>
<tr>
<td>Upwards</td>
<td>NP-complete [Be06]</td>
<td>NP-complete [Be06]</td>
</tr>
<tr>
<td>Multiple</td>
<td>polynomial [Be06]</td>
<td>NP-complete [Be07]</td>
</tr>
</tbody>
</table>

**Heterogeneous platforms:** all problems are NP-complete

**New result:**
**Homogeneous platforms with bandwidth and QoS constraints:** Closest remains polynomial [Re07]
Outline

1. Framework
2. Complexity results
3. Optimal Replica Placement Algorithm
4. Conclusion
Dealing with Bandwidth Constraints

Optimal *Closest* policy

- Homogeneous platform
- QoS constraints
- Bandwidth constraints

**Base:** optimal algorithm of Liu et al. for homogeneous data grids with QoS constraints

Provided extensions:

- Bandwidth constraints
- \( \mathcal{C} \cap \mathcal{N} = \emptyset \)
Dealing with Bandwidth Constraints

Optimal *Closest* policy

- Homogeneous platform
- QoS constraints
- Bandwidth constraints

**Base:** optimal algorithm of Liu et al. for homogeneous data grids with QoS constraints

Provided extensions:

- Bandwidth constraints
- $C \cap \mathcal{N} = \emptyset$
Basic idea:
computation of the minimal necessary number of replicas in a subtree
Basic idea:
computation of the minimal necessary number of replicas in a subtree

Case 1: too many requests

\[ W = 10 \]
**Basic idea:**
computation of the minimal necessary number of replicas in a subtree

Case 1: too many requests

\[
\sum_i r(i) = 12
\]

\[
W = 10
\]
Basic idea:
computation of the minimal necessary number of replicas in a subtree

Case 1: too many requests

\[ \sum_i r(i) = 12 \]

\[ W = 10 \]

1 replica
**Basic idea:**
computation of the minimal necessary number of replicas in a subtree

**Case 2: QoS constraints**

\[
\begin{align*}
W &= 10 \\
\begin{array}{c}
\text{r} \quad 3 & 5 & 4 \\
\text{q} \quad 1 & 3 & 2
\end{array}
\end{align*}
\]
Basic idea:
computation of the minimal necessary number of replicas in a subtree

Case 2: QoS constraints

- **$W = 10$**
- **$q(i) < hops$**
- **1 replica**
Basic idea:
computation of the minimal necessary number of replicas in a subtree

Case 3: bandwidth constraints

![Diagram of a tree with nodes labeled $b: 4, 5, 2$ and $r: 3, 5, 4$]
Basic idea:
computation of the minimal necessary number of replicas in a subtree

Case 3: bandwidth constraints

\[ W = 10 \]
\[ b(l) < r(i) \]
1 replica
Basic idea:
computation of the minimal necessary number of replicas in a subtree
**Preparation**  Tree transformation

**Step 1**  Bottom up computation of the contribution of client requests

\[ C(v, i) : \text{the contribution of node } v \text{ on its } i\text{-th ancestor} \]

\[ e(v, i) : \text{children of } v \text{ that have to be equipped with a replica to minimize the contribution on the } i\text{-th ancestor of } v \text{ (respecting some additional constraints).} \]
**ORP - Optimal Replica Placement Algorithm**

**Preparation**  
Tree transformation

**Step 1**  
Bottom up computation of the contribution of client requests

**Step 2**  
Top down replica placement

**procedure** `Place-replica (v, i)`

```plaintext
if v ∈ C then
    return;
end
place a replica at each node of e(v, i);
forall c ∈ children(v) do
    if c ∈ e(v, i) then
        Place-replica(c,0);
    else
        Place-replica(c,i+1);
end
end
```
Complexity and Optimality

Theorem

(i) Algorithm ORP runs in polynomial time.

(ii) Algorithm ORP returns an optimal solution to the Replica Placement problem with fixed $W$, QoS and bandwidth constraints, if there exists a solution.
Conclusion

- **Replica Placement** optimization problem with QoS and bandwidth constraints.
- Restriction to *Closest*/Homogeneous instances
- Polynomial runtime
- Optimality
- Interplay of different-nature constraints

Completion of the study on complexity of *Closest*/Homogeneous.

Future work
- Consider the problem with several object types
- Extension with more complex objective functions
**Conclusion**

- Replica Placement optimization problem with QoS and bandwidth constraints.
- Restriction to Closest/Homogeneous instances
- Polynomial runtime
- Optimality
- Interplay of different-nature constraints

Completion of the study on complexity of Closest/Homogeneous.

**Future work**

- Consider the problem with several object types
- Extension with more complex objective functions