COMP 7720 - Online Algorithms

Paging and $k$-Server Problem

Shahin Kamali

Lecture 12 - Oct. 16, 2017

University of Manitoba
Review & Plan
Today’s objectives

- \( k \)-server problem
  - Offline algorithms
  - Work-function algorithm
$k$-Server Problem
**k-sever problem**

- We have a metric space of size $m$
  - $k < m$ servers in the graph
- A sequence of $n$ requests to the vertices of the graph
  - Each request should be served by a server
- Minimize the total distance moved by servers

\[ \sigma = \langle S \ M \ K \ A \ D \ B \ D \ B \ D \rangle \]
\[ \text{costs} = 2 \ 0 \ 2 \ 1 \ 1 \]
Major Results

- For any metric $G$, no deterministic $k$-server algorithm $\text{Alg}$ can have a competitive ratio smaller than $k$.

- $k$-server conjecture: for any metric space, there is a deterministic algorithm with competitive ratio of $k$.

- Double coverage algorithm (DCA)
  - proves $k$-server conjecture holds for paths, trees, and cases with $k = 2$
  - It is not useful for any other metric (i.e., metrics with a cycle and $k \geq 3$)

- The balancing algorithm (Balance)
  - proves $k$-server conjecture for cases with $m = k + 1$ ($m$ is the size of the metric).
  - is not competitive for general metrics (even when $k = 2$).
Work Function Algorithm

- Sometimes an offline algorithm can be used as a reference for taking online algorithms
  - Look how the optimal offline algorithm would have served the sequence (if it ended right now)
Optimal Offline Algorithm for \( k \)-server

- A **configuration** indicates the placement of \( k \) servers.
- Consider an initial configuration \( C_0 \) and a sequence
  \[ \sigma = \langle x_1, x_2, \ldots, x_t, \ldots, x_n \rangle. \]
- Given a configuration \( X \), the **work function** \( w_t(X) \) is the cost of
  optimal solution for serving \( x_1, \ldots, x_t \) and ending up at
  configuration \( X \).
- Define the distance \( d \) between two configurations as the total
  distance required for servers to move in order to covert one
  configuration to another.
Work Function Examples

- Given a configuration $X$, the work function $w_t(X)$ is the cost of optimal solution for serving $x_1, \ldots, x_t$ and ending up at configuration $X$.

- Assume $\sigma = \langle B, A, B, A, C, D \rangle$
  What is $w_0((B, D))$?

- What is $w_1(B, D)$?
  - Serve the request to $B$ and be at conf. $(B, D)$?
  - $w_1(B, D) = 1$.

- What is $w_1(A, D)$?
  - serve the request to $B$ and be at conf. $(A, D)$
  - move $A$ to $B$ and take it back $\rightarrow w_1(A, D) = 2$

- What is $w_2(A, D)$? $\rightarrow$ serve the requests to $BA$ and be at conf. $(A, D)$
  - move $A$ to $B$ and take it back $\rightarrow w_2(A, D) = 2$
Work Function Examples

- Given a configuration $X$, the work function $w_t(X)$ is the cost of optimal solution for serving $x_1, \ldots, x_t$ and ending up at configuration $X$.

- Assume $\sigma = \langle BABACD \rangle$

- What is $w_3(A, D)$?
  - Serve the requests to $BAB$ and be at conf. $(B, D)$?
  - $w_3(A, D) = 4$.

- What is $w_3(A, B)$?
  - Serve the requests to $BAB$ and be at conf. $(A, B) \rightarrow w_3(A, B) = 2$.
  - $w_3(A, B) < w_3(A, D) \rightarrow$ optimal algorithm prefers to have its servers on $A$ and $B$ rather than $A$ and $D$ after serving $t = 3$ requests
    - Greedy is not optimal!
**Computing Work Function**

- Given a configuration $X$, the **work function** $w_t(X)$ is the cost of optimal solution for serving $x_1, \ldots, x_{t-1}, x_t$ and ending up at configuration $X$.

- How to compute work function $w_t(X)$?

- Let $Y_1$ be the config. of $\text{OPT}$ after serving $x_{t-1}$, $Y_2$ is the config. after serving $x_t$ (so $x_t \in Y_2$, i.e., $Y$ has a server at $x_t$).
  - $\text{OPT}$'s configuration changes from $Y_1$ to $Y_2$ and then to $X$
  - For fixed $Y_1, Y_2$ we have
    $$w_t(X) = w_{t-1}(Y_1) + d(Y_1, Y_2) + d(Y_2, X); \ x_t \in Y_2$$

- $\text{OPT}$ chose the previous configurations so that work function (its cost) is minimized
  - $w_t(X) = \min_{Y_1, Y_2} \{w_{t-1}(Y_1) + d(Y_1, Y_2) + d(Y_2, X)\}$ so that $x_t \in Y_2$
    $$\overset{Z=Y_1=Y_2}{\Rightarrow}$$
  - $$w_t(X) = \min_Z \{w_{t-1}(Z) + d(X, Z)\}$$ so that $x_t \in Z$
  - $w_0(X) = d(X, C_0)$
Computing Work Function

\[ w_t(X) = \min_Z \{ w_{t-1}(Z) + d(X, Z) \} \quad x_t \in Z; \quad w_0(X) = d(X, C_0) \]

Find all values of work function values using dynamic programming!

E.g.,

\[ w_{t-1}(C_1) = 21, \; w_{t-1}(C_2) = 15, \; w_{t-1}(C_3) = 10, \; w_{t-1}(C_4) = 11 \]

\[ d(C_1, C_2) = 3, \; d(C_1, C_3) = 5, \; d(C_1, C_4) = 1. \]

\[ d(C_2, C_3) = 4, \; d(C_2, C_4) = 2, \; d(C_3, C_4) = 2. \]

Assume \( x_t \) is present in all \( C_1, C_2, C_3 \) but not in \( C_4 \).

\[ w_t(C_1) = \min \{ (21 + 0, 15 + 3, 10 + 5) = 15 \} \]

\[ w_t(C_2) = \min \{ (21 + 3, 15 + 0, 10 + 4) = 14 \} \]

\[ w_t(C_3) = \min \{ (21 + 5, 15 + 4, 10 + 0) = 10 \} \]

\[ w_t(C_4) = \min \{ (21 + 1, 15 + 2, 10 + 2) = 12 \} \]
Optimal Offline Algorithm

- Find all values of work-function using dynamic programming
- Find the configuration with minimum work function after serving all sequence, i.e., \( \min_{Z} w_n(Z) \)
- Move backward to find the right moves!
- Can I do this in online manner?
  - We can set work function values online!
  - We cannot do the backward move

---

**confs\input**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>....</th>
<th>n-1</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>21</td>
</tr>
<tr>
<td><strong>C2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td><strong>C3</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td><strong>C4</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11</td>
</tr>
</tbody>
</table>
Work-function algorithm

- Maintain work-function values in an online manner
- Assume we are at configuration $X$ before serving the $t$'th request to $x$
- There are $k$ options for a lazy algorithm to serve the $t$th request
  - Each associated with a configuration $Y$ (so that $x \in Y$)
- work-function algorithm selects the configuration $Y$ so that minimizes $w_t(Y) + d(X, Y)$
Work Function Algorithm Examples

- Assume $\sigma = \langle B \ A \ B \ A \ B \ A \ldots \rangle$

- Current configuration: $(A, D)$

  - Current request: $B$
    - config. $(A, B)$:
      $$w_3(A, B) + d((A, B), (A, D)) = 2 + 2 = 4$$
    - config $(A, D)$:
      $$w_3(A, D) + d((A, D), (A, D)) = 4 + 0 = 4$$

  - Both configurations are the same (if algorithm chooses $(A, D)$, it requires moving server 2 to B and moving it back to D $\rightarrow$ the non-lazy algorithm chooses $(A, B)$)

```
cons\input

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A,B)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>(A,D)</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>
```
General graphs

- Work-function algorithm:
  - has a competitive ratio of $2k - 1$ competitive for general metrics.
  - $k$-competitive for line, star, and graphs with $m \leq k + 2$.
  - Trees and general graphs?

- Work-function algorithm is conjectured to be $k$-competitive for any metric
  - It might answer the $k$-server conjecture in affirmative (but we are not sure)
**Work-function Framework**

- Define a ‘configuration’ as the state of an algorithm
  - locations of servers or state of the linked-list (list update), etc.
- Define the ‘distance’ between two configurations based on the cost model
  - distance moved by servers or number of paid exchanges to change the state of the list from one config. to another
- Define the work function $w_t(X)$ as the cost of $\text{OPT}$ for serving $t$ requests and ending up at config. $X$
  - Maintain the work-function in an online manner.
- Work-function algorithm: assume we are at configuration $C$; switch to a configuration $Y$ that minimizes $w_t(Y) + d(C, Y)$