Review & Plan

Review & Plan
Today’s objectives

- $k$-server problem
  - Offline algorithms
  - Work-function algorithm
- Techniques for advice lower bounds
k-Server Problem
Introduction

\textit{k}-sever problem

We have a metric space of size \( m \) with \( k \)-servers in the graph. A sequence of \( n \) requests to the vertices of the graph is to be served by a server. Minimize the total distance moved by the servers.

\[ \sigma = < S, M, K, A, D, B, D, B, D > \]

\[ \text{costs} = 2, 0, 2, 1, 1, 1, 1, 1, 1 \]
**Introduction**

**$k$-sever problem**

- We have a metric space of size $m$
  - $k < m$ servers in the graph

**Diagram:**

```
σ = < S M K A D B D B D >
costs = 2 0 2 1 1 1 1 1 1
```
Introduction

**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

$\sigma = <S, M, K, A, D, B, D, B, D>$

Costs: $2, 0, 2, 1, 1, 1, 1, 1, 1, 1$
**Introduction**

**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

\[ \sigma = < S, M, K, A, D, B, D, B, D > \]
\[ \text{costs} = 2 \ 0 \ 2 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \]
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
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

\[
\sigma = < S \ M \ K \ A \ D \ B \ D \ B \ D > \\
\text{costs} = 2 \ 0 \ 2 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 
\]
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
**Introduction**

**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

\[ \sigma = < S \ M \ K \ A \ D \ B \ D \ B \ D > \]
\[ \text{costs} = 2 \ 0 \ 2 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \]
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

\[
\sigma = < S, M, K, A, D, B, D, B, D >
\]
\[
\text{costs} = 2, 0, 2, 1, 1, 1, 1, 1, 1
\]
For any metric $G$, no deterministic $k$-server algorithm 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 $k$.  

Major Results

- For any metric $G$, no deterministic $k$-server algorithm 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 $k \geq 3$)
Introduction

**Major Results**

- For any metric $G$, no deterministic $k$-server algorithm 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 wen $k = 2$).
Sometimes an offline algorithm can used as a reference for taking online algorithms

- Look how the optimal offline algorithm would have served the sequence (if it ended right now)
A configuration indicates the placement of $k$ servers.
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.
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 $X$.

$C_0 = (A, D), X = (A, B), d(C_0, X) = 2$
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 $w_0((B, D))$?

$C_0 = (B, D), Y = (A, C), d(C_0, Y) = 1$
Introduction

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 $\sigma = \langle BABA\text{CD} \rangle$
What is $w_0((B, D))$? It is 1!

$C_0 = (B, D), \ Y = (A, C), \ d(C_0, Y) = 1$
Introduction

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 $w_0((B, D))$? It is 1!

What is $w_1(B, D)$?
- Serve the request to $B$ and be at conf. $(B, D)$?
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 \( w_0((B, D)) \)? it is 1!

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

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 $w_0((B, D))$? It is 1!

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

What is $w_1(A, D)$?
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 $w_0((B,D))$? It is 1!

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 $\rightarrow w_1(A,D) = 2$
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 $\sigma = \langle BABA CD \rangle$
What is $w_0((B, D))$? It is 1!

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. $(A, D)$
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 $\sigma = \langle BABA CD \rangle$
  - What is $w_0((B, D))$? It is 1!

- 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 $\rightarrow w_2(A, D) = 2$
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 $w_3(A, D)$?
**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. $(B, D)$?
  - $w_3(A, D) = 4$.
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 $w_3(A, B)$?
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 $\sigma = \langle BABA, CD \rangle$

- 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. $(A, B) \rightarrow w_3(A, B) = 2$. 

Greedy is not optimal!
Introduction

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

- What is $w_3(A, B)$?
  - Serve the requests to $B A B$ 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 $OPT$ 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 $x_t$).

- $OPT$'s configuration changes from $Y_1$ to $Y_2$ and then to $X$
- For fixed $Y_1, Y_2$ we have

$$w_t(X) = w_{t-1}(Y_1) + d(Y_1, Y_2) + d(Y_2, X); x_t \in Y_2$$
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 $x_t \in Y_2$

\[ \frac{Z=Y_1=Y_2}{w_t(X) = \min_{Z} \{w_{t-1}(Z) + d(X, Z)\}} \] so that $x_t \in Z$
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 $OPT$ 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 $x_t$).

- $OPT$'s configuration changes from $Y_1$ to $Y_2$ and then to $X$.
- For fixed $Y_1, Y_2$ we have
  \[ w_t(X) = w_{t-1}(Y_1) + d(Y_1, Y_2) + d(Y_2, X); x_t \in Y_2 \]

$OPT$ chose the previous configurations so that work function (its cost) is minimized

\[ w_t(X) = \min_{Y_1, Y_2} \{w_{t-1}(Y_1) + d(Y_1, Y_2) + d(Y_2, X)\} \text{ so that } x_t \in Y_2 \]

\[ Z = Y_1 = Y_2 \]

\[ w_t(X) = \min_Z \{w_{t-1}(Z) + d(X, Z)\} \text{ so that } x_t \in Z \]

\[ w_0(X) = d(X, C_0) \]
Computing Work Function

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

Find all values of work function values using dynamic programming!
Introduction

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 \( C_4 \).

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

If \( \text{OPT} \) wants to get to configuration \( X \), it moves from \( C_3 \) to \( C_1 \)!
Introduction

Computing Work Function

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

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 \( C_4 \).

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

If \( \text{OPT} \) wants to get to configuration \( X \), it moves from \( C_3 \) to \( C_1 \)!

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

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 \( C_4 \).
\( w_t(C_1) = \min\{(21 + 0, 15 + 3, 10 + 5) = 15\} \)

If \( \text{OPT} \) wants to get to configuration \( X \), it moves from \( C_3 \) to \( C_1 \)!

\( w_t(C_2) = \min\{(21 + 3, 15 + 0, 10 + 4) = 14\} \)
\( w_t(C_3) = \min\{(21 + 5, 15 + 4, 10 + 0) = 10\} \)
Computing Work Function

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

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 \( C_4 \).

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

If \( \text{OPT} \) wants to get to configuration \( X \), it moves from \( C_3 \) to \( C_1 \)!

\[ w_t(C_2) = \min \{ (21 + 3, 15 + 0, 10 + 4) = 14 \} \]
\[ w_t(C_3) = \min \{ (21 + 5, 15 + 4, 10 + 0) = 10 \} \]
\[ w_t(C_4) = \min \{ (21 + 1, 15 + 2, 10 + 2) = 12 \} \]
Introduction

Optimal Offline Algorithm

- Find all values of work-function using dynamic programming

```
conf\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>C1</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C3</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
Introduction

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., \( \min_Z w_n(Z) \)

```
confs\input

\begin{array}{cccccccc}
& 0 & 1 & 2 & \ldots & n-1 & n \\
C1 & \ast & \ast & & & & \\
C2 & \ast & \ast & & & & \\
C3 & \ast & \ast & & & & \\
C4 & \ast & \ast & & & & \\
\end{array}
```
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., \( \min_Z w_n(Z) \)

\[ \text{confs\ input} \]

<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>C1</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C3</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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., \( \min_Z w_n(Z) \)

<table>
<thead>
<tr>
<th>cons</th>
<th>input</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>...</th>
<th>n-1</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td></td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>* * * * * * * * * * * * * *</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td></td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>* * * * * * * * * * * * * *</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>C3</td>
<td></td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>* * * * * * * * * * * * * *</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>C4</td>
<td></td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>* * * * * * * * * * * * * *</td>
<td>11</td>
<td></td>
</tr>
</tbody>
</table>
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., \( \min_Z w_n(Z) \)

```
conf\textbf{s/input}

\begin{array}{cccccc}
 & 0 & 1 & 2 & \ldots & n-1 & n \\
C1 & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & 21 & 15 \\
C2 & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & 15 & 14 \\
C3 & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & 10 & 14 \\
C4 & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & 11 & 16 \\
\end{array}
```
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., \( \min_{Z} w_n(Z) \)

```
confs\input

\begin{tabular}{c|cccccc}
\hline
& 0 & 1 & 2 & \cdots & n-1 & n \\
\hline
C1 & * & * & * & * & * & * & * & * & * & * & * & * & * & * & 21 & 15 \\
C2 & * & * & * & * & * & * & * & * & * & * & * & * & * & * & 15 & 14 \\
C3 & * & * & * & * & * & * & * & * & * & * & * & * & * & * & 10 & 14 \\
C4 & * & * & * & * & * & * & * & * & * & * & * & * & * & * & 11 & 16 \\
\hline
\end{tabular}
```
## Introduction

### 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!

### 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>( C1 )</td>
<td>* * * * * * * * * * * * * * * * * *</td>
<td>21</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( C2 )</td>
<td>* * * * * * * * * * * * * * * * * *</td>
<td>15</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( C3 )</td>
<td>* * * * * * * * * * * * * * * * * *</td>
<td>10</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( C4 )</td>
<td>* * * * * * * * * * * * * * * * * *</td>
<td>11</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Introduction

**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., \( \min_w w_n(Z) \)
- Move backward to find the right moves!
- Can I do this in online manner?

```markdown
<table>
<thead>
<tr>
<th>confs\input</th>
<th>1</th>
<th>2</th>
<th>....</th>
<th>n-1</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>**</td>
</tr>
<tr>
<td>C2</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>**</td>
</tr>
<tr>
<td>C3</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>**</td>
</tr>
<tr>
<td>C4</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>**</td>
</tr>
</tbody>
</table>
```

COMP 7720 - Online Algorithms  Paging and k-Server Problem
Introduction

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!

```
conf\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>C1</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>C2</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>C3</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>C4</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>
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>C1</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*...</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>C2</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*...</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>C3</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*...</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>C4</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*...</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

> 21 15
> 15 14
> 10 14
> 11 16
Introduction

**Work-function algorithm**

- Maintain work-function values in an online manner
- Assume we are at configuration $X$ before serving the $t$’th request to $x$
Introduction

**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

- 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

- 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 $w_t(Y) + d(X, Y)$
Assume $\sigma = \langle B \ A \ B \ A \ B \ A \ldots \rangle$

Current configuration: $(A, D)$
Work Function Algorithm Examples

- Assume $\sigma = \langle B \ A \ B \ A \ B \ A \ldots \rangle$
- Current configuration: $(A, D)$

<table>
<thead>
<tr>
<th>configs\input</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A,B)</td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>(A,D)</td>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

COMP 7720 - Online Algorithms  Paging and $k$-Server Problem
Introduction

Work Function Algorithm Examples

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

- Current configuration: $(A, D)$

- Current request: $B$

  config. $(A, B)$:
  \[ w_1(A, B) + d((A, B), (A, D)) = 2 + 2 = 4 \]

  config $(A, D)$:
  \[ w_1(A, D) + d((A, D), (A, D)) = 2 + 0 = 2 \]

  $\rightarrow$ config. $(A, D)$ is preferred!

\begin{tabular}{c|c|c}
  & 0 & 1 & 2 \\
\hline
$(A,B)$ & 2 & 2 & \\
$(A,D)$ & 0 & 2 & \\
\end{tabular}
Assume $\sigma = \langle B\ A\ B\ A\ B\ A\ \ldots \rangle$

Current configuration: $(A, D)$

Current request: $A$

config. $(A, B)$:

$w_2(A, B) + d((A, B), (A, D)) = 2 + 2 = 4$

config $(A, D)$:

$w_2(A, D) + d((A, D), (A, D)) = 2 + 0 = 2$

$\rightarrow$ config. $(A, D)$ is preferred!

\begin{tabular}{c|ccc}
  & 0 & 1 & 2 \\
\hline
$(A, B)$ & 2 & 2 & 2 \\
$(A, D)$ & 0 & 2 & 2 \\
\hline
\end{tabular}
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 (assume algorithm chooses $(A, D)$)

<table>
<thead>
<tr>
<th>confs\input</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(A,B)$</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$(A,D)$</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>
Work Function Algorithm Examples

- Assume $\sigma = \langle B \ A \ B \ A \ B \ A \ \ldots \rangle$
- Current configuration: $(A, D)$
  - Current request: $A$
    - config. $(A, B)$:
      $w_4(A, B) + d((A, B), (A, D)) = 2 + 2 = 4$
    - config $(A, D)$:
      $w_4(A, D) + d((A, D), (A, D)) = 4 + 0 = 4$
  - Both configurations are the same (assume algorithm chooses $(A, D)$)

```
conf\input

\begin{array}{c|c|c|c|c}
0 & 1 & 2 & 3 & 4 \\
\hline
(A, B) & 2 & 2 & 2 & 2 \\
(A, D) & 0 & 2 & 2 & 4 & 4 \\
\end{array}
```
Assume $\sigma = \langle B \ A \ B \ A \ B \ A \ldots \rangle$

Current configuration: $(A, D)$

Current request: $A$
config. $(A, B)$:
$w_5(A, B) + d((A, B), (A, D)) = 2 + 2 = 4$
config $(A, D)$:
$w_5(A, D) + d((A, D), (A, D)) = 6 + 0 = 6$
Now $(A, B)$ is preferred $\rightarrow \text{move server 2 instead of 1!}$
Assume $\sigma = \langle B \ A \ B \ A \ B \ A \ldots \rangle$

Current configuration: $(A, D)$

The worse-case sequences for greedy do not cause problem for work function algorithm!

<table>
<thead>
<tr>
<th>confs\input</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>
Introduction

General graphs

- Work-function algorithm:
General graphs

- Work-function algorithm:
  - has a competitive ratio of $2k - 1$ competitive for general metrics.
Introduction

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?
Introduction

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
Introduction

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 configuration to another.
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
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 OPT for serving $t$ requests and ending up at config. $X$
  - Maintain the work-function in an online manner.
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 $w_t(Y) + d(C, Y)$