# 1 Introduction

## 1.1 Input and conventions

TODO: rewrite in nice paragraph
Input:

- $m \in \mathbb{N}$: Number of homogeneous servers
- $T \in \mathbb{N}$: Number of time steps
- $\beta \in \mathbb{R}_{\geq 0}$: Power up costs
- $\lambda_0, \ldots, \lambda_T \in [0, m]$: Arrival rates

Notations:

- Let $\lambda_{i,t}$ be the assigned arrival rate at time t for server i
- Let $x_t$ be the number of active servers at time t
- Let $\mathcal{X} \coloneqq (x_0, \ldots, x_T)$ be the sequence of active servers
- If $A = (a_0, \ldots, a_n)$ is a tuple with $n+1$ entries, we write $A(i)$ for the i-th component of A with $0 \leq i \leq n$

Requirements:

- Convex cost function $f$
- Power down costs are w.l.o.g. equal to 0
- $\forall t \in \{0, \ldots, T\} : \sum_{i=1}^{m} \lambda_{i,t} = \lambda_t$
- $\lambda_0 = \lambda_T = 0$ and $\mathcal{X}(0) = \mathcal{X}(T) = 0$, i.e. all servers are powered down at $t = 0$ and $t = T$

## 1.2 Preliminaries

**Lemma 1.1.** *Given a convex cost function $f$, $x$ active servers and an arrival rate $\lambda$, the best method is to assign each server a load of $\lambda/x$.*

*Proof.* $\forall x \in \mathbb{N}, \mu_i \in [0,1] : \sum_{i=1}^{x} \mu_i = 1 :$

$$\begin{aligned} f\Big(\frac{\lambda}{x}\Big) = f\Big(\sum_{i=1}^{x} \frac{\mu_i \lambda}{x}\Big) &\overset{\text{Jensen's inequality}}{\leq} \sum_{i=1}^{x} \frac{1}{x} f(\mu_i \lambda) \\ \Leftrightarrow x f\Big(\frac{\lambda}{x}\Big) &\leq \sum_{i=1}^{x} f(\mu_i \lambda) \end{aligned}$$

□

Lemma 1.1 shows us, that having $x_t$ active servers, it is always the best method to equally share $\lambda_t$ on all $x_t$ active servers. This allows us to uniquely identify an optimal schedule by the sequence of numbers of active servers $\mathcal{X}$. The costs of a schedule are then given by:

$$costs(\mathcal{X}) \coloneqq \sum_{t=0}^{T} \underbrace{\beta \max\{0, x_t - x_{t-1}\}}_{\text{power up costs}} + x_t f(\lambda_t / t)$$

**Definition 1.2.** We call a schedule $\mathcal{X}$ **feasible** if

$$\forall t \in \{0, \dots, T\} : x_t \geq \lambda_t$$

In particular, every optimal schedule is feasible.

# 2 Optimal scheduling for m homogeneous servers

TODO: introduction text

## 2.1 Graph for an optimal schedule

We construct a directed acyclic graph as follows:
$\forall t \in [T-1]$ and $i, j \in \{0, \dots, m\}$ we add vertices $(t, i)$ modelling the number of active servers at time t. Furthermore, we add vertices $(0, 0)$ and $(T, 0)$ for our initial and final state respectively.
In order to warrant that there are at least $\lceil \lambda_t \rceil$ active servers $\forall t \in [T-1]$, we define an auxiliary function which calculates the costs for handling an arrival rate $\lambda$ with $x$ active servers:

$$c(x, \lambda) \coloneqq \begin{cases} 0 & \text{if } x = 0 \\ xf(\lambda/x), & \text{if } x \neq 0 \wedge \lambda \leq x \\ \infty, & \text{otherwise} \end{cases} \tag{1}$$

Then, $\forall t \in [T-2]$ and $i, j \in \{0, \dots, m\}$ we add edges from $(t, i)$ to $(t+1, j)$ with weight

$$d(i, j, \lambda_{t+1}) \coloneqq \beta \max\{0, j - i\} + c(j, \lambda_{t+1}) \tag{2}$$

Finally, for $0 \leq i \leq m$ we add edges from $(0, 0)$ to $(1, i)$ with weight $d(0, i, \lambda_1)$ and from $(T-1, i)$ to $(T, 0)$ with weight $d(i, 0, \lambda_T) = 0$.

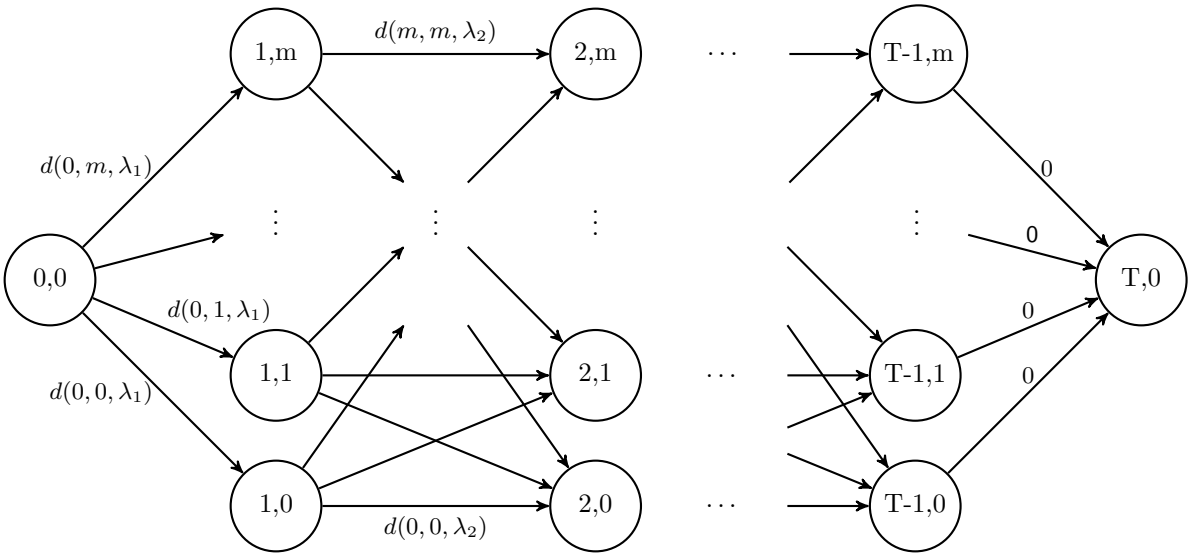


Figure 1: Graph for optimal schedule algorithm.

**Note:** All edges from $(t, i)$ to $(t+1, j)$ have weight $d(i, j, \lambda_{t+1})$

**Proposition 2.1.** *Any given optimal schedule $\mathcal{X}$ corresponds to a shortest path $P$ from $(0,0)$ to $(T,0)$ with $costs(\mathcal{X}) = costs(P)$ and vice versa.*

*Proof.*

"⇒": We construct a feasible path in our graph from $\mathcal{X}$ as follows:

$$\text{First set} \qquad e_t \coloneqq \Big(\big(t, \mathcal{X}(t)\big), \big(t+1, \mathcal{X}(t+1)\big)\Big), \forall t \in \{0, \dots, T-1\}$$
$$\text{then set} \qquad P \coloneqq (e_0, \dots, e_{T-1})$$

As each edge $e_t$ in our graph has weight $d\big(\mathcal{X}(t-1), \mathcal{X}(t), \lambda_t\big)$ and hence corresponds to the costs of switching from $\mathcal{X}(t-1)$ to $\mathcal{X}(t)$ servers and processing $\lambda_t$ with $\mathcal{X}(t)$ active servers, it directly follows that $P$ is a shortest path of the graph with $costs(P) = costs(\mathcal{X})$.

"⇐": Let $P = \big((0,0) = v_0, \dots, v_T = (T,0)\big)$ with $v_t \in \big\{(t,i) \mid 0 \le i \le m\big\}$ be a shortest path of the graph.
We can construct an optimal schedule from $P$ by setting $\mathcal{X} \coloneqq \big(v_0(1), \dots, v_T(1)\big)$
By definition (1) it is guaranteed that $P$ only traverses edges such that there are enough active servers $\forall t \in [T]$. Therefore, the created schedule is feasible. Its optimality directly follows from the definition of the edges' weights and so does the equality $costs(\mathcal{X}) = costs(P)$.

□

## 2.2 A pseudo-polynomial minimum cost algorithm

**Algorithm 1** Calculate costs for $m$ homogeneous servers

**Require:** Convex cost function $f$, $\lambda_0 = \lambda_T = 0$, $\forall t \in [T-1] : \lambda_t \in [0, m]$

```
1:  function SCHEDULE(m, T, β, λ_1, ..., λ_{T-1})
2:      if T < 2 then return
3:      let p[2...T-1, m] and M[1...T-1, m] be new arrays
4:      for j ← 0 to m do
5:          M[1, j] ← d(0, j, λ_1)
6:      for t ← 1 to T-2 do
7:          for j ← 0 to m do
8:              opt ← ∞
9:              for i ← 0 to m do
10:                 M[t+1, j] ← M[t, i] + d(i, j, λ_{t+1})
11:                 if M[t+1, j] < opt then
12:                     opt ← M[t+1, j]
13:                     p[t+1, j] ← i
14:             M[t+1, j] ← opt
15:     return p and M
```

**Algorithm 2** Extract schedule for $m$ homogeneous servers

```
1: function EXTRACT(m, p, M, T)
2:     let x[0...T] be a new array
3:     x[0] ← x[T] ← 0
4:     if T < 2 then return x        ▷ Trivial solution
5:     x[T − 1] ← arg min_{0≤i≤m} {M[T − 1, i]}
6:     for t ← T − 2 to 1 do
7:         x[t] ← p[t + 1, x[t + 1]]
8:     return x
```

#### 2.2.1 Runtime analysis

The algorithm visits every vertex and every edge of the graph exactly once. As the number of vertices is bounded by $\mathcal{O}(Tm)$ and the number of edges is bounded by $\mathcal{O}(Tm^2)$ the running time is given by:

$$\mathcal{O}(Tm + Tm^2) = \mathcal{O}(Tm^2) \tag{3}$$

As we need $\log_2(m)$ bits to encode m, the running time is polynomial in the numeric value of the input but exponential in the length of the input. Hence, the algorithm is pseudo-polynomial.

#### 2.2.2 A memory optimized algorithm

TODO: use only array with size 2m

# 3 A 4-approximation algorithm for monotonically increasing convex f

We consider a modification of the problem discussed in chapter 2. Assuming that f is convex and monotonically increasing, we can modify our algorithm to obtain a polynomial time 4-approximation algorithm.

## 3.1 Graph for a 4-optimal schedule

We modify our graph from chapter 2.1 to the reduce the number of vertices. For this, we stop adding m vertices for each timestep but using vertices that approximate the number of active servers. First, let $b := \lceil \log_2(m) \rceil$. We add vertices $(t, 0)$ and $(t, 2^i), \forall t \in [T-1], 0 \leq i \leq b$. All edges and weights are added analogous to chapter 2.1.

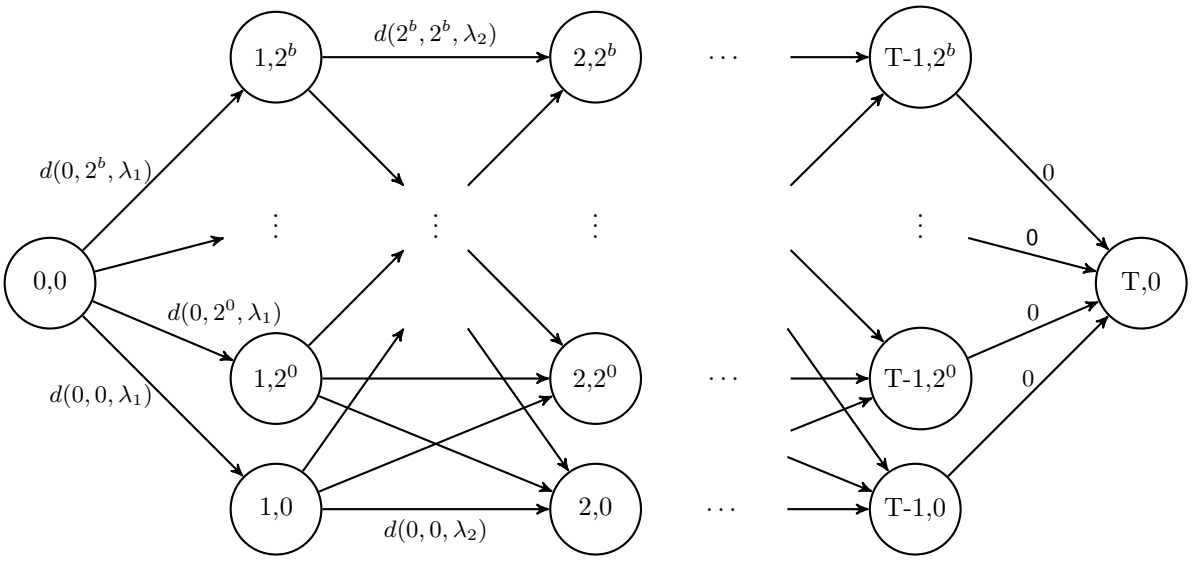


Figure 2: Graph for 2-approximation algorithm

**Definition 3.1.** Let $\mathcal{X} = (x_0, \ldots, x_T)$ be a schedule and $t > 0$.
We say that $\mathcal{X}$ changes its **state** at time t if

$$x_t \neq x_{t-1}$$

and that $\mathcal{X}$ changes its **2-state** at time t if

$$x_t = 0 \quad \text{or} \quad x_t \notin [2^{\lfloor \log_2(x_{t-1}) \rceil}, 2^{\lfloor \log_2(x_{t-1}) \rceil}]$$

**Proposition 3.2.**

1. *Any given optimal schedule $\mathcal{X}$ can be transformed to a 4-optimal schedule $\mathcal{X}'$ which corresponds to a path $P$ from $(0,0)$ to $(T,0)$ with $costs(\mathcal{X}') = costs(P)$.*
2. *Any shortest path $P$ from $(0,0)$ to $(T,0)$ corresponds to a 4-optimal schedule $\mathcal{X}'$ with $costs(P) = costs(\mathcal{X}')$.*

*Proof.*

1. Assume we have an optimal schedule identified by $\mathcal{X} = (x_0, \ldots, x_T)$. For $0 < i < T$ we inductively set:

$$x'_0 \coloneqq 0, \qquad x'_{i+1} \coloneqq \begin{cases} \min\{2^{\lfloor \log_2(2x_{t+1}) \rfloor}, 2^b\}, & \text{if } 0 < x_t \leq x_{t+1} \\ 2^{\lceil \log_2(2x_{t+1}) \rceil}, & \text{if } 0 < x_{t+1} < x_t \text{ and } x'_t \geq 4x_{t+1} \\ x'_t, & \text{if } 0 < x_{t+1} < x_t \text{ and } x'_t < 4x_{t+1} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

   Then let $\mathcal{X}' \coloneqq (x'_0, \ldots, x'_T)$ be the modified sequence of active servers. Notice that $x_t \leq x'_t \leq 4x_t$ holds as $x'_t$ is at most the smallest power of two larger than $2x_t$ which implies that $\mathcal{X}'$ is feasible.
   We can now construct a feasible path in our graph from $\mathcal{X}'$ as follows:

$$\text{First set} \qquad e_t \coloneqq \Big(\big(t, \mathcal{X}'(t)\big), \big(t+1, \mathcal{X}'(t+1)\big)\Big), \forall t \in \{0, \ldots, T-1\}$$
$$\text{then set} \qquad P \coloneqq (e_0, \ldots, e_{T-1})$$

By the definition of the edges' weights it follows that $costs(\mathcal{X}') = costs(P)$.
Next, let $(t_0 = 0, t_1, \ldots, t_n = 0)$ be the sequence of times where the optimal schedule changes its 2-state. Notice that if the new schedule $\mathcal{X}'$ changes its state, then we must be in one of the states $x_{t_i}$.

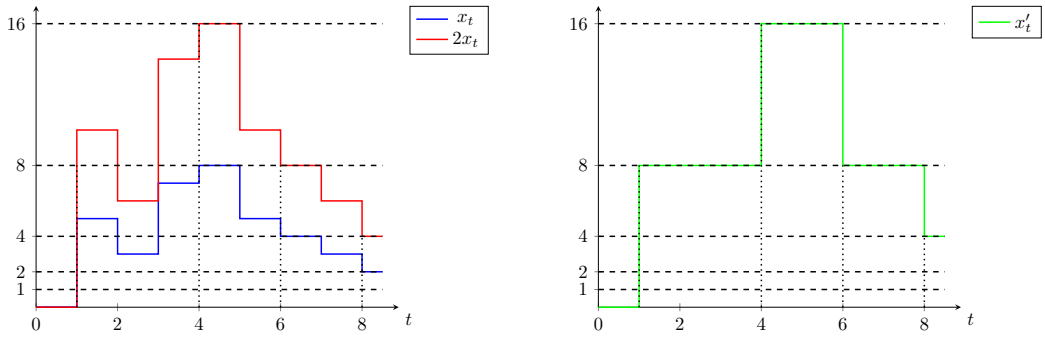


Figure 3: Adaption of an optimal schedule

For this reason, we now only have to consider the fraction of costs between $\mathcal{X}'$ and $\mathcal{X}$ for every time step $t_i$:
The costs incurred by $\mathcal{X}'$ are given by

$$\begin{aligned}
&d\big(\mathcal{X}'(t), \mathcal{X}'(t+1), \lambda_{t+1}\big) && = \beta \max\{0, x'_{t+1} - x'_t\} + c(x'_{t+1}, \lambda_{t+1}) \\
&(\mathcal{X}' \text{ is feasible}) && = \beta \max\{0, x'_{t+1} - x'_t\} + x'_{t+1} f(\lambda_{t+1}/x'_{t+1}) \\
&(4) && \leq \beta \max\{0, x'_{t+1} - x'_t\} + 4x_{t+1} f(\lambda_{t+1}/x'_{t+1}) \\
&(\text{f monotonically increasing}) && \leq \beta \max\{0, x'_{t+1} - x'_t\} + 4x_{t+1} f(\lambda_{t+1}/x_{t+1}) \qquad (5)
\end{aligned}$$

and the costs of $\mathcal{X}$ by

$$\begin{aligned}
d\big(\mathcal{X}(t), \mathcal{X}(t+1), \lambda_{t+1}\big) &= \beta \max\{0, x_{t+1} - x_t\} + c(x_{t+1}, \lambda_{t+1}) \\
(\mathcal{X} \text{ is feasible}) \qquad &= \beta \max\{0, x_{t+1} - x_t\} + x_{t+1} f(\lambda_{t+1}/x_{t+1}) \qquad (6)
\end{aligned}$$

(i) $\underline{x_{t+1} \leq x_t}$: From (4) it follows that $x'_{t+1} \leq x'_t$. We continue to simplify:

$$\begin{aligned}
& && \frac{d\big(\mathcal{X}'(t), \mathcal{X}'(t+1), \lambda_{t+1}\big)}{d\big(\mathcal{X}(t), \mathcal{X}(t+1), \lambda_{t+1}\big)} \\
(5),(6) \quad &\leq && \frac{\beta \max\{0, x'_{t+1} - x'_t\} + 4x_{t+1} f(\lambda_{t+1}/x_{t+1})}{\beta \max\{0, x_{t+1} - x_t\} + x_{t+1} f(\lambda_{t+1}/x_{t+1})} \\
&= && \frac{4x_{t+1} f(\lambda_{t+1}/x_{t+1})}{x_{t+1} f(\lambda_{t+1}/x_{t+1})} \\
&= && 4
\end{aligned}$$

$\Rightarrow costs(\mathcal{X}') \leq 4costs(\mathcal{X})$

(ii) $\underline{x_t < x_{t+1}}$: We simplify (**??**) and obtain

$$\begin{aligned}
& \frac{d\big(\mathcal{X}'(t), \mathcal{X}'(t+1), \lambda_{t+1}\big)}{d\big(\mathcal{X}(t), \mathcal{X}(t+1), \lambda_{t+1}\big)} \\
\leq\ & \frac{\beta \max\{0, x'_{t+1} - x'_t\} + 4x_{t+1} f(\lambda_{t+1}/x_{t+1})}{\beta \max\{0, x_{t+1} - x_t\} + x_{t+1} f(\lambda_{t+1}/x_{t+1})} \\
=\ & \frac{\beta(x'_{t+1} - x'_t) + 4x_{t+1} f(\lambda_{t+1}/x_{t+1})}{\beta(x_{t+1} - x_t) + x_{t+1} f(\lambda_{t+1}/x_{t+1})} \\
(4)\quad =\ & \frac{\beta(\min\{2^{\lfloor \log_2(2x_{t+1}) \rfloor}, 2^b\} - x'_t) + 4x_{t+1} f(\lambda_{t+1}/x_{t+1})}{\beta(x_{t+1} - x_t) + x_{t+1} f(\lambda_{t+1}/x_{t+1})} \\
\leq\ & \frac{\beta(2^{\lfloor \log_2(2x_{t+1}) \rfloor} - x'_t) + 4x_{t+1} f(\lambda_{t+1}/x_{t+1})}{\beta(x_{t+1} - x_t) + x_{t+1} f(\lambda_{t+1}/x_{t+1})} \\
\leq\ & \frac{\beta(2x_{t+1} - x_t) + 4x_{t+1} f(\lambda_{t+1}/x_{t+1})}{\beta(x_{t+1} - x_t) + x_{t+1} f(\lambda_{t+1}/x_{t+1})} \\
\leq\ & 4\frac{\beta(x_{t+1}/2 - x_t/4) + x_{t+1} f(\lambda_{t+1}/x_{t+1})}{\beta(x_{t+1} - x_t) + x_{t+1} f(\lambda_{t+1}/x_{t+1})} \\
\geq\ & 4
\end{aligned}$$

Here we fail.

2. From (1) we obtain that we can construct a 4-optimal path $P'$ from any optimal schedule. Now, let $P$ be a shortest path. We have $costs(P) \leq costs(P')$ and since every feasible path in $P$ corresponds to an feasible schedule $\mathcal{X}$ with $costs(P) = costs(\mathcal{X})$, $P$ must also be at least 4-optimal.

□