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
- $\forall t \in \{0, \ldots, T\} : \sum_{i=1}^{m} \lambda_{i,t} = \lambda_t$
- $\lambda_0 = \lambda_T = 0$
- $\mathcal{X}(0) = \mathcal{X}(T) = 0$, 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 λ, the best method is to assign each server a load of λ/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 tells us, that having x_t active servers, it is always the best method to equally share λ_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:

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

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 λ with x active servers:

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

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 $d(i, 0, \lambda_T) = 0$.

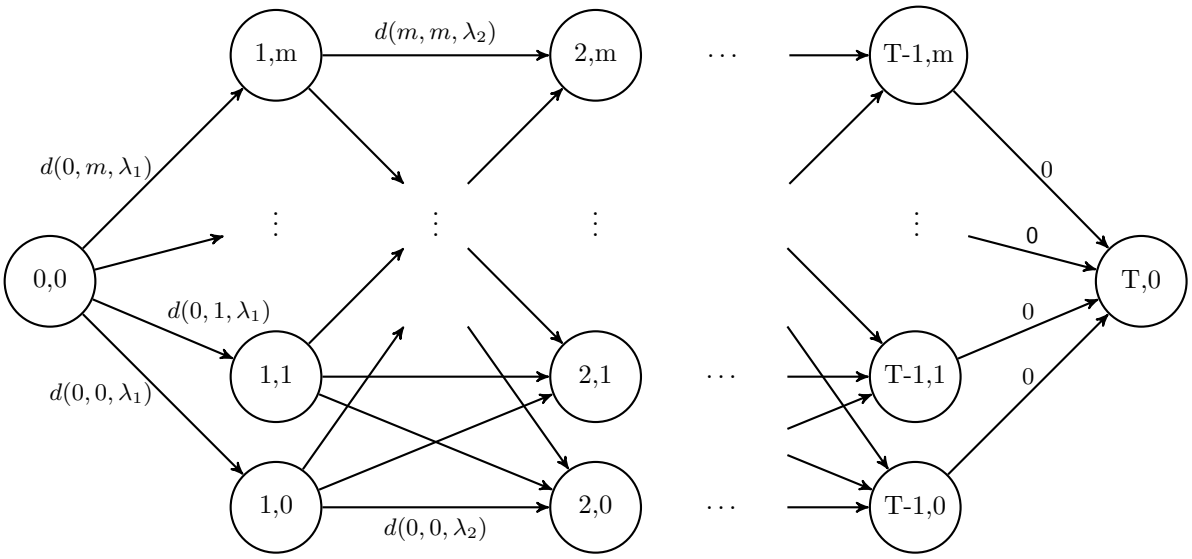

Figure 1: Graph for optimal schedule algorithm.

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:
$\forall t \in [T]$ set $e_t \coloneqq \Big(\big(t-1, \mathcal{X}(t-1)\big), \big(t, \mathcal{X}(t)\big)\Big)$. Then set $P \coloneqq (e_1, \ldots, e_T)$.
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 λ_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 $P = \big((0, 0) = v_0, \ldots, v_T = (T, 0)\big)$ with $v_t \in \big\{(t, i) \mid 0 \leq i \leq m\big\}$ be a shortest path of the graph.
We can construct an optimal schedule from P by setting $\mathcal{X} \coloneqq \big(v_0(1), \ldots, v_T(1)\big)$
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. It's optimality directly follows from the definition of the edges' weights and so does the equality $costs(\mathcal{X}) = costs(P)$.

□

2.2 A 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 n 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

Schedule: Loop 5,8 and 10 run $m+1$ times, loop 7 runs $T-2$ times
Extract: Loop 5 runs $T-2$ times, argmin 4 takes time $m+1$.
For $T, n \to \infty$ it holds:

$$\mathcal{O}(m+1+(T-2)(m+1)^2+T-2+m+1) = \mathcal{O}(2m+T+(T-2)(m+1)^2) = \mathcal{O}(Tm^2) \quad (3)$$

As we need $\log_2(m)$ bits to encode m, the algorithm is exponential in the number of servers.

2.2.2 A memory optimized algorithm

TODO

3 A 2-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 2-approximation algorithm.

3.1 Graph for a 2-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.

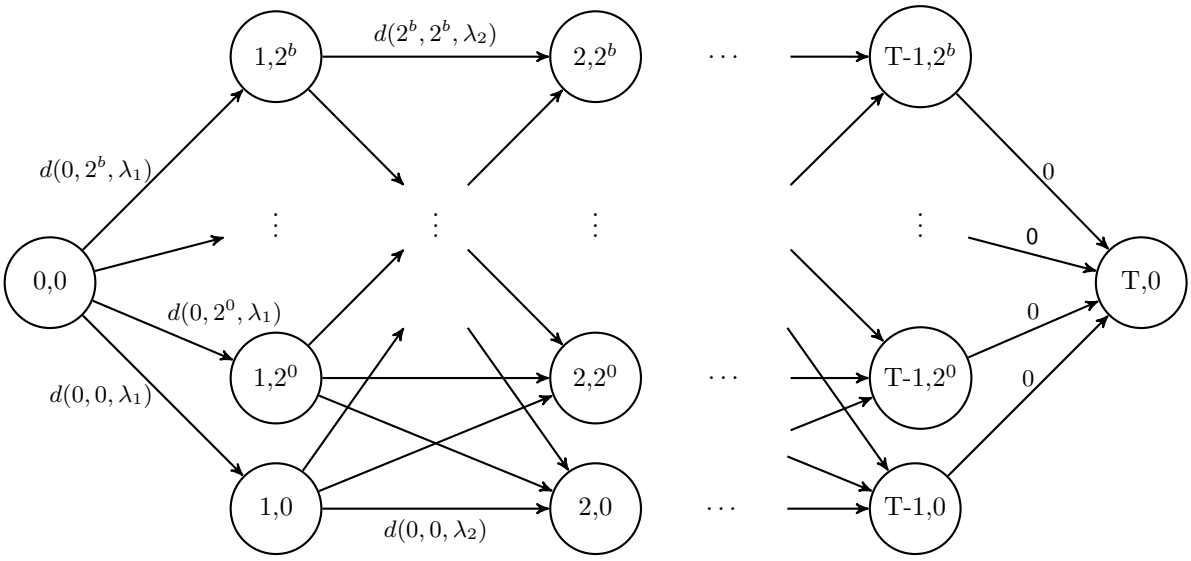

Figure 2: Graph for 2-approximation algorithm

Proposition 3.1.

1. *Any given optimal schedule $\mathcal{X}$ can be transformed to a 2-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 2-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 $0 \leq t \leq T$ we set:

$$x'_t := \begin{cases} min\{2^{\lfloor \log_2(2x_t) \rfloor}, 2^b\}, & \text{if } x_t > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

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

$\forall t \in [T]$ set $e_t \coloneqq \Big(\big(t-1, \mathcal{X}'(t-1)\big), \big(t, \mathcal{X}'(t)\big)\Big)$. Then set $P \coloneqq (e_1, \ldots, e_T)$. By the definition of the edges' weights it follows $costs(\mathcal{X}') = costs(P)$.
Next, lets consider the difference of costs between $\mathcal{X}'$ and $\mathcal{X}$ at every time step t to $t+1$ for $0 \leq t \leq T-1$: The costs incurred by $\mathcal{X}'$ are given by

$$\begin{aligned}
&d\big(\mathcal{X}'(t), \mathcal{X}'(t+1), \lambda_{t+1}\big) && = \beta \max\{0, x'_{t+1} - x'_t\} + c(x'_{t+1}, \lambda_{t+1}) \\
&(\mathcal{X}' \text{ is feasible}) && = \beta \max\{0, x'_{t+1} - x'_t\} + x'_{t+1} f(\lambda_{t+1}/x'_{t+1}) \\
&(4) && \leq \beta \max\{0, x'_{t+1} - x'_t\} + 2x_{t+1} f(\lambda_{t+1}/x'_{t+1}) \\
&(\text{f monotonically increasing}) && \leq \beta \max\{0, x'_{t+1} - x'_t\} + 2x_{t+1} f(\lambda_{t+1}/x_{t+1}) \qquad (5)
\end{aligned}$$

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

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

Therefore, we can estimate the difference:

$$\begin{aligned}
& && d(\mathcal{X}'(t), \mathcal{X}'(t+1), \lambda_{t+1}) - d(\mathcal{X}(t), \mathcal{X}(t+1), \lambda_{t+1}) \\
&(5),(6) && \leq \beta \max\{0, x'_{t+1} - x'_t\} - \beta \max\{0, x_{t+1} - x_t\} + x_{t+1} f(\lambda_{t+1}/x_{t+1}) \\
& && = \beta\big(\max\{0, x'_{t+1} - x'_t\} - \max\{0, x_{t+1} - x_t\}\big) + x_{t+1} f(\lambda_{t+1}/x_{t+1}) \qquad (7)
\end{aligned}$$

(i) $\underline{x_{t+1} \leq x_t}$: From (4) it follows that $x'_{t+1} \leq x'_t$. We continue to simplify (7):

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

$\Rightarrow costs(\mathcal{X}') = 2costs(\mathcal{X})$

(ii) $\underline{x_{t+1} > x_t}$: We simplify (7) and obtain

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

Here we fail.

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

□