



Figure 1: grafy ze kterych jsem vychazel

- Horizontální osa reprezentuje čas (tedy jakýkoliv bod na této ose je  $t_i$ ) a je rozdělena na časové intervaly -  $m_i$
- Vertikální osy jsou následující hodnoty
  - **Solution Cost** - cena za exekuci optimalizačního algoritmu v závislosti na čase - to, kolik se reálně zaplatí za alokované zdroje
  - **Allocated resources** - počet alokovaných zdrojů (tedy číslo) v závislosti na čase, dá se dále dělit na CPU, RAM, DISK
  - **Solution Value** - cena řešení, které vytvořil optimalizační algoritmus v závislosti na čase

## 1 Moment

**M... moments** - definují časové intervaly v systému, nový interval začíná vždy ve chvíli, kdy se změnila alokace zdrojů pro určitý job, vztahuje se tedy k jobu. V podstatě to můžu pojmenovat i jinak, zatím je to pracovní verze.

Délka časového intervalu je doba, po kterou určitý job měl přidělené určité zdroje a jedná se o časovou hodnotu - tedy například 2 minuty.

$$|m_i| = t_{i+1} - t_i$$

Tedy například v předešlém grafu existují 4 momenty:

$$\begin{aligned}|m_0| &= t_1 - 0 \\ |m_1| &= t_2 - t_1 \\ |m_2| &= t_3 - t_2 \\ |m_3| &= T - t_3\end{aligned}$$

## 2 Solution Value

**V... solution value** - cena výsledku optimalizačního algoritmu, musí se jednat o fuknici zdrojů a času, protože je to závislé - čím více resourců a času, tím je cena výsledného plánu/řešení menší (nicméně tohle nebude lineární závislost)

## 2.1 Nápad 1.

Obecně to tedy bude vypadat nějak takhle:

$$v = f_a(t, R, d)$$

Kde:

- $a$  je algoritmus, který exekuuje daný optimalizační algoritmus (např. TASP) - každý bude mít jinou funkci, protože to řeší jinak
- $t$  je délka časového intervalu, při kterém byly dostupné zdroje  $R$  (tedy délka momentu, k tomu se dostanu dále)
- $d$  jsou daná data, která jsou vstupem algoritmu

Pak  $v = f_a$  je definována jako:

Schopnost algoritmu  $a$  dosáhnout na datech  $d$  při alokaci zdrojů  $R$  za čas  $t$  ceny řešení  $v$ .

Pokud tedy použiju *moment* tak je tak hodnota funkce pro každý moment jiná - je závislá na zdrojích, které jsou v každém momentu jiné, ale její význam zůstává stejný.

$$v_{m_i} = f_a(|m_i|, R_{m_i}, d_m)$$

Data jsou také závislá na momentu, protože podle definice funkce  $f_a$  musí být data na vstupu do dalšího momentu modifikována tak, aby odpovídala částečnému řešení optimalizačního problému z předchozího momentu.

Jinými slovy, pokud funkce  $f_a$  vrátí "plán" na konci momentu  $m_i$  musí se tento "plán" vložit do dat pro moment  $m_{i+1}$ . Tedy v každém momentu probíhá zlepšení řešení z předchozího momentu. Technicky by to snad mělo být řešitelné (minimálně TASP tohle úmí - můžu mu předhodit částečný plán a on mi ho bude zlepšovat).

Pak minimální cena celého optimalizačního problému (tedy to, co je konečné řešení vráceného uřivateli) se definuje jako:

$$v_j = \min\{f_a(|m_{j,i}|, R_{j,m_{j,i}}, d_{j,m_{j,i}})\}$$

Kde  $j$  je id jobu a  $i = 0 \dots M$ ,  $M$  je počet realokací zdrojů v průběhu času.

Abych řekl pravdu, tak se mi tady moc nelíbí to min, ale nenapadlo mi, jak bych se toho zbavil

## 2.2 Nápad 2.

Druhý pokus o formalizaci funkce pro solution value.

Funkce  $f_a$  bude nově iterační a bude reprezentovat zlepšení ”nějakého” plánu za jednu iteraci. Tedy obecně:

$$s = f_a(t, R, d)$$

Schopnost algoritmu  $a$  zlepšit řešení  $d$  při alokaci zdrojů  $R$  za čas  $t$  na řešení  $s$ .

A pokud správně zaindexujeme a použijeme zase *moment* jako měření úseků času:

$$\begin{aligned} s_{j,m_{j,i+1}} &= f_a(|m_{j,i}|, R_{j,m_{j,i}}, s_{j,m_i}) \\ s_{j,m_{j,0}} &= \emptyset \end{aligned}$$

Nyní potřebujeme novou funkci, která nám bude říkat hodnotu (solution value) řešení:

$$v = h_a(s_j)$$

Tedy indexované:

$$\begin{aligned} v_{j,m_{j,i}} &= h_a(s_{j,m_{j,i}}) \\ v_{j,m_{j,i+1}} &\leq v_{j,m_{j,i}} \\ v_{j,m_{j,0}} &= \infty \end{aligned}$$

Ve výsledku tedy budu minimalizovat:

$$\min h_a(s_{j,M_j}) = h_a(f_a(|M_j|, R_{j,M_j}, s_{j,M_j-1}))$$

Kdy vlastně moment  $M$  je poslední v časové řadě momentů.

Díky této definici říkám, že ty data (řešení problému) jsou na sobě závislá v čase - nové řešení je vždy ”dále” v čase.

## 3 Solution Cost

Aneb kolik stálo alokování zdrojů - tedy kolik jsem zaplatil např. Amazonu za naplánování nějakých dat na CPU/RAM/DISK.

Obecně

$$p = g_h(t, R)$$

Kde funkce  $g_h$  říká, kolik peněz stojí provozování zdrojů  $R$  za čas  $t$  u providera zdrojů (ie Amazon)  $h$ .

Zase bude závislá pro moment a také pro job.

$$p_{j,m_{j,i}} = g_h(|m_{j,i}|, R_{j,m_{j,i}})$$

Tedy výsledná cena exekuce jednoho jobu bude:

$$p_j = \sum_{i=0}^M g_h(|m_{j,i}|, R_{j,m_{j,i}})$$

## 4 Závěr

Díky tomu budu schopen vyjádřit solution value a solution cost v každém čase, v jakékoliv konfiguraci alokovaných zdrojů. Minimalizuji tedy pro kazdy job  $j$ :

### 4.1 Napad 1

$$\begin{aligned} & \min \{ \min \{ f_a(|m_{j,i}|, R_{j,m_{j,i}}, d_{j,m_{j,i}}) \} \} \quad \text{for } i = 0 \dots M \\ & \min \sum_{i=0}^M g_h(|m_{j,i}|, R_{j,m_{j,i}}) \end{aligned}$$

### 4.2 Napad 2

$$\begin{aligned} & \min h_a(s_{j,m_{j,M}}) \\ & \min \sum_{i=0}^M g_h(|m_{j,i}|, R_{j,m_{j,i}}) \end{aligned}$$

Tedy v tomto pripade hledam idealni koncový moment  $M$  a řadu konfigurací  $R$  takovou, aby výsledek byl co nejmenší.