Online Dynamic Multiple-choice Vector Bin Packing

A system of K servers runs a sequence of N virtual machines (VM) over some period of time $[0, T]$.

Bins

Each server k is comprised of P NUMA nodes.

Each NUMA node p at server k is characterized by capacity c_{kpj} for each resource $j = 1...R$.

Therefore each server k can be represented as $P \times R$ matrix C_k with elements c_{kpj}.

The capacities of NUMA nodes at the same server are usually (but not always!) equal, then all C_k rows are identical.

Items

Each VM i characterized by

- arrival time a_i ($\min a_i = 0$)
- departure time d_i ($d_i > a_i$, $\max d_i = T$)
- resource requirements r_{ij} for each resource $j = 1...R$
- number of required NUMA nodes n_i
    - Currently it is assumed that $n_i \leq 2$
    - $n_i = 1$ corresponds to *small VM*
    - $n_i = 2$ corresponds to *large VM*

VM resource requirements are evenly spread among used NUMA nodes.

- Therefore VM requirements per-NUMA can be represented as $n_i \times R$ matrix V_i with elements $v_{ij} = \frac{r_{ij}}{n_i}$.
- If $u_i = 1$ (small VM) then V_i is a vector

Decision Variables

Each VM i should be assigned to some server k immediately upon arrival, i.e. at time a_i.

The VM can be placed into any n_i NUMA nodes within the chosen server.

- There are $M_i = \binom{P}{n_i}$ possible placement patterns or item *incarnations* (term used in multiple-choice bin packing).
    - For small VM there are P possible incarnations
    - For large VM and $P = 2$ there is one possible incarnation
    - For large VM and $P = 4$ there are 6 possible incarnations
    - For 4-node ARM server and large VM patterns are restricted to {1,2} and {3,4}, but this case can be reduced to ordinary problem by considering each ARM server as two independent 2-node servers
- Each incarnation m describes the VM resource usage across all server NUMA nodes and is represented by $P \times R$ matrix U_{im}.
- U_{im} includes n_i rows from V_i in arbitrary positions, all other rows are filled with zeros (TODO: describe more formally).

For each VM a server and an incarnation should be selected, which is modeled by two decision variables:

- $x_{ik} \in \{0,1\},\ x_{ik} = 1$ if VM i is assigned to server k
- $z_{im} \in \{0,1\},\ z_{im} = 1$ if incarnation m is used for VM i

Other Notations

Each VM i is running by consuming resources of assigned server during its lifetime $[a_i, d_i]$.

A set of *running VMs* at time t is denoted as $W(t)$:

- $i \in W(t) \iff a_i \le t \le d_i$

A set of *active servers* (with at least one running VM) at time t is denoted as $A(t)$:

- $k \in A(t) \iff \sum_{i \in W(t)} x_{ik} > 0$

Constraints

1. Each VM is assigned to exactly one server:

   $\sum_{k=1}^{K} x_{ik} = 1,\ \ i = 1...N$
2. Exactly one incarnation is used for each VM:

   $\sum_{m=1}^{M_i} z_{im} = 1,\ \ i = 1...N$
3. Allocated resources should not exceed server capacities:

   $\forall t \in T, k = 1...K, j = 1...R : \sum_{i \in W(t)} \sum_{m \in M_i} x_{ik} z_{im} U_{im} \le C_k$

Objectives

- Minimize the maximum number of active servers: $\max_{t \in [0,T]} A(t) \to \min_{x_{ik}, z_{im}}$
- Minimize the total active server time: $\int_0^T A(t) dt \to \min_{x_{ik}, z_{im}}$