1. Appendix A. Improvements for the initial scheme. Research directions

Here is the problem we face: we do not know how to combine recalculations of bound via big and average jobs. Thus we can either process big and average jobs independently, or to recalculate them together (but guarantee that at least one of them would be recalculated).Therefore, we have two directions of improvement.

Scheme with three job categories and two bin categories

Consider the following system and any solution satisfying it:

$$\begin{cases} (B-1)\cdot s\cdot k+(1-\varphi)\cdot m_1\cdot B\geq s\cdot k+m_1+m_2 \\ k+m_1+m_2=m \\ (1-\varphi)\cdot B\leq(\psi\cdot B)^R\cdot\min\{B-s,B\cdot(1-l\cdot\psi)\} \\ m_2=R\cdot(x+(z-1)\cdot k) \\ w=[\frac{x\cdot l}{(z-1)\cdot k}] \\ (\varphi\cdot B)^w\geq\psi\cdot B \\ \varphi\leq\psi \\ z-1<s\leq z \\ \varphi,\psi\in[0,1] \\ l,R,m_1,m_2\geq 0 \\ l,m_1,m_2,R,z\in\mathbb{Z} \end{cases} \tag{A-1}$$

Sketch of idea. Jobs with processing times less or equal to $\varphi B\cdot LB_j$ are called small, jobs with processing times within $(\varphi B\cdot LB_j,\psi\cdot LB_j)$ are called average, and jobs with processing times at least $\psi B\cdot LB_j$ are called big. Processors can be Fast, Normal and Reserved as before. Processor is in the class Fast if it has processing speed s. Some m_1 processors are in the class Normal, and the rest m_2 processors are in the class Reserved. Reserved processors are divided into groups $G_1,\ldots,G_r$. Each group consists of $(z-1)\cdot k+x-1$ processors.

Suppose we have a solution for the scheme above. We would show how to create an algorithm with worst-case performance B from it. Change the previous algorithm. If a job fits some Fast processor or one of m_1 normal processors, put it there as before. Else we put it into some Reserved processor. We consider two selected Reserved processors: one for big jobs, and another for average ones. If a big job comes, we assign it to the selected processor for big jobs, make change with minimal loaded from Normal as in the previous algorithm, and select another processor for big jobs instead of this (from the current group, or the next group if the current group have finished). If we have an average job, assign it to the selected processor for average jobs. After we have l average jobs assigned to the selected processor for average jobs, we make change with minimal loaded from Normal and select another processor for average jobs.

Note that after we get x closed processors each having l average jobs, we get the same recalculation as if we get $(z-1)\cdot k$ processors filled by one big jobs each.

In each group of $(z-1)\cdot k+x-1$ we have either recalculation with big jobs, or recalculation with average jobs. Thus we get recalculation for the lower bound as in the previous algorithm.

Scheme with three job categories with three bin categories

Consider the following system:

$$\begin{cases} m=k+m_1+m_2+m_3 \\ m_2=R_1\cdot(z-1)\cdot k \\ m_3=R_2\cdot(z-1)\cdot k \\ (B-1)\cdot s\cdot k+(1-\varphi)\cdot m_1\cdot B\geq s\cdot k+m_1+m_2+m_3 \\ (1-\varphi)\cdot B\leq(\psi\cdot B)^{R_1}\cdot(B-s) \\ (1-\varphi)\cdot B\leq(\phi\cdot B)^{l\cdot R_2}\cdot(B-l\cdot\psi\cdot B) \\ \varphi\leq\psi \\ z-1<s<z \\ \varphi,\psi\in[0,1] \\ R_i,m_i\geq 0;l>0 \\ l,m_i,R_i,z\in\mathbb{Z} \end{cases} \tag{A-2}$$

Jobs can be small, average or big as mentioned above. The idea is to make separate groups for average-loaded processors, and load there only average jobs. There are R_1 groups of Reserved processors for big jobs (let us call theset processors Big-Reserved), and R_2 groups of Reserved processors for average jobs (let us call these processors Average-Reserved). Each group has size $(z-1)\cdot k$.

If a big or small job comes, we proceed with iniital algorithm using Big-Reserved processors. If an average job comes, we try to put it on one of the Normal or Fast processors if it fits some. In the case it does not, we try to put this job to the current processor m_i from Average-Reserved. If after loading last average job we have l average jobs on m_i, exchange m_i with Normal processor of the smallest load, and start using the next Average-Reserved processor for the upcoming average jobs (that do not fit into Normal or Fast).

To make it all work, we impose conditions stating that if average job does not fit to one of the Normal or Fast processors, it will fit into the current Average-Reserved processor. Note that initial scheme is a particular case of this one with $\varphi = \psi, l = 1, R_2 = 0$.

Further research directions

1. Create similar scheme for for three or more different speeds. It seems that one can build similar scheme for more different speeds, and run computation to calculate the optimal bound.
2. Recalculation during other count of assignemnts, not all z. Then the following two inequalities change:

$$\begin{cases} (1-\varphi)\cdot B \le \left(\min\left\{\varphi\cdot B, \frac{\varphi\cdot B\cdot \text{count}}{s}\right\}\right)^R \cdot (B-s) \\ m_2 = R\cdot k\cdot(\text{count}-1) \end{cases}$$

3. Analyze values of φ that give us the best worst-case performance.