A PROOF OF THEOREM 3.6

Theorem 3.6 *The SMPO-A problem is NP-hard.*

Proof. An instance $\mathcal{I}$ of the load-balancing problem is: given a set of jobs $J\,(|J| = n)$, $\forall j_i \in J$ its processing time is t_i, a set of identical machines $M\,(|M| = m)$. the goal is to obtain an assignment $f : J \to M$ to assign each job to one machine, while minimize the maximum total processing time of one machine computed by $\max_{m_j \in M} \sum_{\forall f(j_i)=m_j} t_i$. By given the instance $\mathcal{I}$, we can construct an instance O of SMPO-A problem by performing the following mapping: we map the J in $\mathcal{I}$ to E^* in O and $|J|$ in $\mathcal{I}$ to k in O. At the same time, we map the t_i of each job in J to the proof frequency of each element in E^*. Then, we map M in $\mathcal{I}$ to S^* in O. We force the set E in O to fulfill $|E| = k|S|$ and for all $e_i \in E$ we set $p_i = 0$. By doing so, we can obtain a pair of valid δ_s and $|S|$ such that $|S^*| = \lceil \delta_s |E \bigcup E^*| \rceil - |S|$ can hold. After creating E we then randomly assign $e_i \in E$ into $|S|$ sets with the cardinality of k.

Since $\forall e_i \in E$, $p_i = 0$, the optimal solution to O is to assign elements in E^* to S^* in a load-balancing way. It implies that if there is an algorithm that can solve the SMPO-A problem in polynomial time, it can also obtain the result of the load-balancing problem. Thus, the SMPO-A problem is also NP-hard. □

B PROOF OF THEOREM 3.8

Theorem 3.8 *The SMPO-R problem is NP-hard.*

Proof. Given an instance $\mathcal{A}$ of the SMPO-A problem defined in Def. 3.5, we construct an instance $\mathcal{R}$ of SMPO-R problem by performing the following mapping: we map E in $\mathcal{A}$ to the $E - E^r$ in $\mathcal{R}$, S in $\mathcal{A}$ to the $S - S^r$ in $\mathcal{R}$ with the corresponding usage frequency of each set. Meanwhile, we map E^* in $\mathcal{A}$ to the E^r in $\mathcal{R}$ and let $S(e)$ to be 1 $\forall e \in E^r$. Finally, we map δ_s, k and S^* in $\mathcal{A}$ to the δ_s, k and S^r in $\mathcal{R}$ respectively. Notice that, for E in $\mathcal{R}$, it equals to $E \bigcup E^*$ in $\mathcal{A}$. For S in $\mathcal{R}$, it equals to $S \bigcup S^*$ in $\mathcal{A}$. Also, we set E^d in $\mathcal{R}$ to be $\emptyset$ and let $|S^d| = |S^r|$.

In this case, we can map any instance of the SMPO-A problem to the SMPO-R problem. Since the SMPO-A has been proved to be NP-hard, the SMPO-R problem is NP-hard as well. □

C PROOF OF THEOREM 5.3

Theorem 5.3 *The expected approximation ratio of Algo. 4 is 2 when $\delta_s \le \frac{1}{\sqrt[3]{|E^r|}}$ and $\sqrt[3]{|E^r|}$ when $\frac{1}{\sqrt[3]{|E^r|}} < \delta_s \le 1$.*

Proof. **State 1: $S_{max} \in S^r$ after phase 1**. Each set in S^r contains $\frac{\sum_{\forall e_i \in E^r} \mathcal{I}(e_i)}{|S^r|}$ elements on average. In *Lemma 2*, let $n = |S^r|$, $m = \frac{\sum_{\forall e_i \in E^r} \mathcal{I}(e_i)}{|S^r|}$, $\epsilon = \mu$, $\alpha = 2$. When $\delta_s \le \frac{1}{\sqrt[3]{|E^r|}}$, since $\sum_{\forall e_i \in E^r} \mathcal{I}(e_i) \ge |E^r|$, $n \le \lceil \delta_s |E^r| \rceil$, we have $\frac{m}{n^{\frac{1}{2}}} = \frac{\sum_{\forall e_i \in E^r} \mathcal{I}(e_i)}{|S^r|^{\frac{3}{2}}} \ge \frac{|E^r|}{\lceil \delta_s |E^r| \rceil^{\frac{3}{2}}} \ge 1 \Rightarrow m \ge n^{\frac{1}{2}}$. Thus, $\mathbb{E}(u_{max}) = \mathbb{E}(\max_{\forall S_i \in S^r} S_i(m)) \le \frac{2\mu}{\delta_s} = \frac{2\sum_{\forall S_i \in S} u_i}{\delta_s |E|} = 2L_3^r = 2OPT$. When $\frac{1}{\sqrt[3]{|E^r|}} < \delta_s \le 1$, $\mathbb{E}(u_{max}) \le \frac{L_1^r}{\delta_s} \le \sqrt[3]{|E^r|}OPT$.

State 2: $S_{max} \notin S^r$ after phase 1. Due to line 8 in Algo. 4, if $\delta_s \le \frac{1}{\sqrt[3]{|E^r|}}$, $\forall S_i \in S^r : u_i = 2\sum_{\forall e_i \in S_i} \frac{\sum_{\forall S_l \in S} u_l}{|E|(|\mathcal{A}(e_i)| + \mathcal{I}(e_i))} \le 2L_3^r$. Meanwhile we also always have $\forall S_i \in S^r : u_i \le 2L_2^r$. Thus, the rest proof is similar to State 2 in Theorem 4.4 □