# 1 Numerical Results in Section 3

We first provide the numerical results in Section 3.

## 1.1 Lower Bound

Let $n = 16$ and $\{p_i\}_{i\in[n]} = \{0.0, 0.1, 0.19, 0.27, 0.315, 0.355, 0.395, 0.44, 0.485, 0.535, 0.595, 0.665, 0.74, 0.875, 1.195, 1000.0\}$, we need to show that the optimal value of following optimization problem has a lower bound of 0.72.

$$
\begin{aligned}
\min_{\substack{s_1,s_2\cdots,s_n\\ b_1,b_2,\cdots,b_n,r}} \quad & r \\
\text{s.t.} \quad & s_i, b_i \ge 0 && \forall i \in [n] \\
& \sum_{i=1}^{n} s_i \ge 1 \quad \text{and} \quad \sum_{i=1}^{n} b_i \ge 1 \\
& \sum_{i=1}^{n} s_i \le 1 + \frac{1}{p_n} \quad \text{and} \quad \sum_{i=1}^{n} b_i \le 1 + \frac{1}{p_n} \\
& \sum_{i=1}^{n}\sum_{j=1}^{n} s_i b_j \max(p_i, p_j) \ge 1 \\
& \sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1}\sum_{j=t+1}^{n} s_i b_j (p_j - p_i) \le r && \forall t \in [n]
\end{aligned}
$$

We could see that the optimum is at least 0.72 is equivalent to the non-existence of feasible point $(s_i, b_i, r)$ in the following region:

$$
\begin{aligned}
& r \le 0.72 \\
& s_i, b_i \ge 0 && \forall i \in [n] \\
& \sum_{i=1}^{n} s_i \ge 1 \quad \text{and} \quad \sum_{i=1}^{n} b_i \ge 1 \\
& \sum_{i=1}^{n} s_i \le 1 + \frac{1}{p_n} \quad \text{and} \quad \sum_{i=1}^{n} b_i \le 1 + \frac{1}{p_n} \\
& \sum_{i=1}^{n}\sum_{j=1}^{n} s_i b_j \max(p_i, p_j) \ge 1 \\
& \sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1}\sum_{j=t+1}^{n} s_i b_j (p_j - p_i) \le r && \forall t \in [n]
\end{aligned}
$$

What's more, if $(s_i, b_i, r)$ is a feasible point, so is $(s_i, b_i, 0.72)$. Therefore, it suffices to check whether the optimum of following optimization problem is greater than 1 or not.

$$
\begin{aligned}
\max_{\substack{s_1,s_2\cdots,s_n\\ b_1,b_2,\cdots,b_n,r}} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} s_i b_j \max(p_i,p_j) \\
\text{s.t.} \quad & s_i, b_i \geq 0 && \forall i \in [n] \\
& \sum_{i=1}^{n} s_i \geq 1 \quad \text{and} \quad \sum_{i=1}^{n} b_i \geq 1 \\
& \sum_{i=1}^{n} s_i \leq 1 + \frac{1}{p_n} \quad \text{and} \quad \sum_{i=1}^{n} b_i \leq 1 + \frac{1}{p_n} \\
& \sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1}\sum_{j=t+1}^{n} s_i b_j (p_j - p_i) \leq 0.72 && \forall t \in [n]
\end{aligned}
$$

If the optimum above is less that 1, this means that the lower bound is at least 0.72. We implement such optimization problem via Gurobi and the code could be found at FullInfo – LowerBound.py. We ran it on a compute node with 40 cores and 200GB memory and it took 24 hours to finish the computation with a provable upper bound lower than 1.

### 1.2 Upper Bound

In this setting, we define $n = 80$. We define $p_1 = 0, p_n = 1000$, and $p_i = (i+4)/100.0$ for $1 < i < n$. The instance is in example.txt, and we use calc.cpp to calculate the approximation ratio of the optimal mechanism on such instance.

## 2 Numerical Results in Section 4

### 2.1 Knowing $\mathbb{E}[S]$

We choose $n = 50$ and $p_n = 1000$. Besides, $p_i = (i-1)/20$ for $i \in [n-1]$. We choose $\{w_i\}_{i\in[n]}$ according to some heuristics. Please check $\mathsf{ES}_1$.py for the details that how we choose $\{w_i\}_{i\in[n]}$.

We need to give a lower bound of the following optimization problem $\mathcal{O}$:

$$
\begin{aligned}
\min_{\substack{s_1,s_2\cdots,s_n\\ b_1,b_2,\cdots,b_n}} \quad & \frac{\sum_{t=1}^{n} w_t \left(\sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1}\sum_{j=t+1}^{n} s_i b_j (p_j - p_i)\right)}{\sum_{i=1}^{n}\sum_{j=1}^{n} s_i b_j \max(p_i,p_j)} \\
\text{s.t.} \quad & s_i, b_i \geq 0 && \forall i \in [n] \\
& \sum_{i=1}^{n} s_i \geq 1 \quad \text{and} \quad \sum_{i=1}^{n} s_i \leq 1 + \frac{1}{p_n} \quad \text{and} \quad \sum_{i=1}^{n-1} s_i \leq 1 \\
& \sum_{i=1}^{n} b_i \geq 1 \quad \text{and} \quad \sum_{i=1}^{n-1} b_i \leq 1 \\
& \sum_{i=1}^{n} s_i \cdot p_i = 1
\end{aligned}
$$

To verify the optimum of the optimization problem above has a lower bound of at least 0.65, it suffices to show that the optimal value of the following problem is non-negative:

$$
\begin{aligned}
\min_{\substack{s_1,s_2\cdots,s_n\\ b_1,b_2,\cdots,b_n}} \quad & \sum_{t=1}^{n} w_t \left( \sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1} \sum_{j=t+1}^{n} s_i b_j (p_j - p_i) \right) - 0.65 \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} s_i b_j \max(p_i, p_j) \\
\text{s.t.} \quad & s_i, b_i \geq 0 && \forall i \in [n] \\
& \sum_{i=1}^{n} s_i \geq 1 \quad \text{and} \quad \sum_{i=1}^{n} s_i \leq 1 + \frac{1}{p_n} \quad \text{and} \quad \sum_{i=1}^{n-1} s_i \leq 1 \\
& \sum_{i=1}^{n} b_i \geq 1 \quad \text{and} \quad \sum_{i=1}^{n-1} b_i \leq 1 \\
& \sum_{i=1}^{n} s_i \cdot p_i = 1
\end{aligned}
$$

However, such optimization problem is hard to solve directly since we do not have a constraint towards $b_n$. Therefore, we separate it into two case where $b_n \leq 10$ and $b_n > 10$. When $b_n \leq 10$, we just simply add it into the constraints.

$$
\begin{aligned}
\min_{\substack{s_1,s_2\cdots,s_n\\ b_1,b_2,\cdots,b_n}} \quad & \sum_{t=1}^{n} w_t \left( \sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1} \sum_{j=t+1}^{n} s_i b_j (p_j - p_i) \right) - 0.65 \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} s_i b_j \max(p_i, p_j) \\
\text{s.t.} \quad & s_i, b_i \geq 0 && \forall i \in [n] \\
& \sum_{i=1}^{n} s_i \geq 1 \quad \text{and} \quad \sum_{i=1}^{n} s_i \leq 1 + \frac{1}{p_n} \quad \text{and} \quad \sum_{i=1}^{n-1} s_i \leq 1 \\
& \sum_{i=1}^{n} b_i \geq 1 \quad \text{and} \quad \sum_{i=1}^{n-1} b_i \leq 1 \quad \text{and} \quad b_n \leq 10 \\
& \sum_{i=1}^{n} s_i \cdot p_i = 1
\end{aligned}
$$

We again implement such optimization problem via Gurobi and the code could be found at $\mathsf{ES}_1$.py. We ran it on a compute node with 40 cores and 200GB memory and it took 1 minute to finish the computation with a provable upper bound greater than 0.

When $b_n > 10$, we could see that most of the welfare is generated by $b_n$. Therefore, we only need to consider the welfare gain from $b_n$.

$$
\begin{aligned}
\min_{s_1,s_2\cdots,s_n} \quad & \sum_{t=1}^{n-1} w_t \sum_{i=1}^{t-1} s_i \cdot (p_n - p_i) \\
\text{s.t.} \quad & s_i \geq 0 && \forall i \in [n] \\
& \sum_{i=1}^{n} s_i \geq 1 \quad \text{and} \quad \sum_{i=1}^{n} s_i \leq 1 + \frac{1}{p_n} \quad \text{and} \quad \sum_{i=1}^{n-1} s_i \leq 1 \\
& \sum_{i=1}^{n} s_i \cdot p_i = 1
\end{aligned}
$$

This is a LP and thus is easy to solve. We solve it in $\mathsf{ES}_2$.py and get that optimal solution is 651.27. Thus, for any feasible solution $(s_i, b_i)$ in the first optimization program $\mathcal{O}$ satisfying $b_n > 10$, we know that $(s_i)_{i\in[n]}$ is also a feasible solution for the optimization problem above. Therefore

$$
\begin{aligned}
&\frac{\sum_{t=1}^{n} w_t \left(\sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1}\sum_{j=t+1}^{n} s_i b_j (p_j - p_i)\right)}{\sum_{i=1}^{n}\sum_{j=1}^{n} s_i b_j \max(p_i, p_j)} \\
\geq &\frac{1 + \sum_{t=1}^{n-1}\sum_{i=1}^{t-1} w_t s_i b_n (p_n - p_i)}{\sum_{i=1}^{n} s_i b_n p_n + \sum_{i=1}^{n}\sum_{j=1}^{n-1} s_i b_j \max(p_i, p_j)} \\
\geq &\frac{1 + 651.27 b_m}{1000 \cdot b_n \cdot (1 + 10^{-3}) + \mathbb{E}[S] + p_{n-1} \cdot 1.001} \\
\geq &\frac{651.27 \cdot b_n + 1}{1001 \cdot b_n + 1 + 2.5 \cdot 1.001}
\end{aligned}
$$

Since $b_n > 10$, the minimum is attained at $b_n = 10$ and it is $0.6505 \geq 0.65$. Therefore, we show that the approximation ratio is at least 0.65 for this carefully chosen set $\{p_i, w_i\}$.

### 2.2 Knowing $\mathbb{E}[B]$

Similarly, we also choose $n = 120$ and $p_n = 1000$. We define $p_i = (i-1)/40$ for $i \in [110]$ and $p_i = (n-10)/40 + (i-(n-10)-1)/5$ for $i$ in $[111, 119]$. It's easy to verify that $p_i$ are increasing. We choose $\{w_i\}_{i\in[n]}$ according to some heuristics. Please check "EB.py" for the details that how we choose $\{w_i\}_{i\in[n]}$.

Similarly, we need to prove the following optimization problem has a lower bound of 0.65.

$$
\begin{aligned}
\min_{\substack{s_1,s_2\cdots,s_n\\ b_1,b_2,\cdots,b_n}} \quad & \frac{\sum_{t=1}^{n} w_t \left(\sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1}\sum_{j=t+1}^{n} s_i b_j (p_j - p_i)\right)}{\sum_{i=1}^{n}\sum_{j=1}^{n} s_i b_j \max(p_i, p_j)} \\
\text{s.t.} \quad & s_i, b_i \geq 0 \qquad\qquad \forall i \in [n] \\
& \sum_{i=1}^{n} b_i \geq 1 \text{ and } \sum_{i=1}^{n} b_i \leq 1 + \frac{1}{p_n} \text{ and } \sum_{i=1}^{n-1} b_i \leq 1 \\
& \sum_{i=1}^{n} s_i \geq 1 \text{ and } \sum_{i=1}^{n-1} s_i \leq 1 \\
& \sum_{i=1}^{n} b_i \cdot p_i = 1
\end{aligned}
$$

Similarly, since we don't have any constraints towards $s_n$, so we again divide it into two case that $s_n \leq 10$ and $s_n > 10$.

When $s_n \leq 10$, we simply add it to constraints and check whether the following optimization problem has a optimum of at least 0.

$$
\begin{aligned}
\min_{\substack{s_1,s_2\cdots,s_n\\ b_1,b_2,\cdots,b_n}} \quad & \sum_{t=1}^{n} w_t \left(\sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1}\sum_{j=t+1}^{n} s_i b_j (p_j - p_i)\right) - 0.65 \sum_{i=1}^{n}\sum_{j=1}^{n} s_i b_j \max(p_i, p_j) \\
\text{s.t.} \quad & s_i, b_i \geq 0 \qquad\qquad \forall i \in [n] \\
& \sum_{i=1}^{n} b_i \geq 1 \text{ and } \sum_{i=1}^{n} b_i \leq 1 + \frac{1}{p_n} \text{ and } \sum_{i=1}^{n-1} b_i \leq 1 \\
& \sum_{i=1}^{n} s_i \geq 1 \text{ and } \sum_{i=1}^{n-1} s_i \leq 1 \text{ and } s_n \leq 10 \\
& \sum_{i=1}^{n} b_i \cdot p_i = 1
\end{aligned}
$$

We implement such optimization problem via Gurobi and the code could be found at EB.py. We ran it on a compute node with 40 cores and 200GB memory and it took 3 minute to finish the computation with a provable upper bound greater than 0.

When $s_n > 10$, we could see that

$$\begin{aligned}
&\frac{\sum_{t=1}^{n} w_t \left(\sum_{i=1}^{n} s_i p_i + \sum_{i=1}^{t-1}\sum_{j=t+1}^{n} s_i b_j (p_j - p_i)\right)}{\sum_{i=1}^{n}\sum_{j=1}^{n} s_i b_j \max(p_i, p_j)} \\
\geq &\frac{s_n p_n}{\sum_{i=1}^{n} s_n b_i p_n + \sum_{i=1}^{n-1}\sum_{j=1}^{n} s_i b_j \max(p_i, p_j)} \\
&\geq \frac{1000 \cdot s_n}{1000 s_n (1 + 10^{-3}) + \mathbb{E}[B] + p_{n-1} \cdot 1.001} \\
&\geq \frac{1000 \cdot s_n}{1001 \cdot s_n + 1 + 2.5 \cdot 1.001} \geq 0.65
\end{aligned}$$

when $s_n > 10$.

These finishes our proof.

## 3 Numerical Results in Section 5

We provide two programs, i.e. symmetric.py and general.py that respectively compute the approximation ratio in the symmetric and general setting for the order statistics choose the $i$th order statistics w.p. $w_i$. Call the function Ratio($\mathbf{w}$) and we would get the corresponding ratio.