.

APPENDIX A PROOF OF LEMMA 1

We prove by contradiction. First, assume that there exists an optimal reward $r^*(s_i)$ which is smaller than $\theta_i\omega(s_i)$, i.e., $r^*(s_i) - \theta_i\omega(s_i) < 0$. This contradicts with the IR constraint. Second, assume that $r^*(s_i)$ is larger than $\theta_i\omega(s_i)$, i.e., $r^*(s_i) - \theta_i\omega(s_i) > 0$. Since the miner's cost increases with reward, the miner can decrease its cost by reducing the reward $r^*(s_i)$ until $r^*(s_i) - \theta_i\omega(s_i) = 0$. This contradicts with the assumption $r^*(s_i) - \theta_i\omega(s_i) > 0$. Therefore, we complete the proof.

APPENDIX B PROOF OF LEMMA 2: USER PREFERENCE CHARACTERIZATION

A. USER PREFERENCE 1 $(\theta_I, \bar{T})$

We first prove that, for any given fixed t_{max}, the miner prefers users with higher θ_i, i.e., the miner only sets positive contract item for users with the highest confidentiality and sets zero contact item for other type users.

Therefore, given a fixed t_{max}, we represent θ_{max} as the highest confidentiality of users who satisfy $t_i \leq t_{max}$. By contradiction, suppose that there exists a set $\mathcal{I}$ of users that each type $i \in \mathcal{I}$ of users have a positive optimal contract item. At least one $i \in \mathcal{I}$ satisfies $\theta_i > \theta_{max}$.

Suppose that all transaction workload is $\Omega = \sum_{i\in\mathcal{I}} \omega(s_i)$, then the miner's utility U_1 is

$$U_1 = \sum_{i\in\mathcal{I}} \{r(s_i) - \theta_i\omega(s_i)\}. \tag{21}$$

However, the miner's utility U_2 for setting only positive contract item for users with the highest confidentiality is

$$U_2 = r_{max}(s_{max}) - \theta_{max}\omega(s_{max}). \tag{22}$$

Since

$$\begin{aligned} \theta_{max}\omega(s_{max}) &= \theta_{max}\Omega \\ &= \sum_{i\in\mathcal{I}} \theta_{max}\omega(s_i) \\ &< \sum_{i\in\mathcal{I}} \theta_i\omega(s_i), \end{aligned} \tag{23}$$

we have $U_1 > U_2$, which contradicts the contract optimality. Therefore, the proof is completed. The miner can get more reward (profit) only when setting positive contract item for users with the highest confidentiality (higher U means more profit).

B. USER PREFERENCE 2 $(\bar{\theta}, T_I)$

In this subsection, we prove that given the same confidentiality $\bar{\theta}$, the miner prefers a higher t_{max} and users with smaller t_i, i.e., the miner will set $t^*_{max} = \max\{\sum_{i\in\mathcal{I}} t_i\}$. Consider $t_i \leq t_{max}$, the miner's profit under a given time $t_{max} \in [\max\{\sum_{i\in\mathcal{I}} t_i\}, +\infty)$ from (2) is given by

$$U_1 = \sum_{i\in\mathcal{I}} \{r(s_i) - \bar{\theta}\omega(s_i)\}. \tag{24}$$

However, suppose $t_i > t_{max}$, the miner's profit can be expressed as

$$U_2 = -\sum_{i\in\mathcal{I}} \bar{\theta}\omega(s_i), \tag{25}$$

which gives a negative utility to the miners if t_{max} is set small. Since $U_1 > U_2$, we have proved that for users with the same confidentiality θ, the miner prefers smaller delay preference t users and for users with the same delay preference, the miner prefers higher confidentiality users. Next, we want to investigate how the miner will select users with higher confidentiality and lower delay preference (θ_H, t_L) and users with low confidentiality and higher delay preference (θ_L, t_H).

C. (θ_H, T_L) AND (θ_H, T_L)

Let us consider the case that $\theta_1 < \theta_2 < \cdots < \theta_I$ and $t_1 < t_2 < \cdots < t_I$. The miner's profit under complete information contract is given as

$$\max_{t_{max},\varphi} \quad U_{\varphi_i} \tag{26a}$$

$$\text{s.t.} \quad W_{\varphi_i} \geq 0, \quad \forall i \in \mathcal{I}. \tag{26b}$$

From (1), we can solve for the optimal reward and transaction workload expressed as $r(s_i) = \frac{\theta_i^2}{4\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)}$ and $\omega(s_i) = \frac{\theta_i^3}{4\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)}$, respectively. Next, we substitute $r(s_i)$ and $\omega(s_i)$ into (2) to obtain $\Pi(\theta_i, t_i)$.

From Section III-A for any given $t_{max} = t_i$, we can deduce that the miner will only set positive contract items for type-i users and the optimal contract under each t_i is: $i \in \mu_i^*$ with $t^*_{max} = t_i, \varphi_i^* = [\frac{\theta_i^2}{4\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)}, \frac{\theta_i^3}{4\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)}]$, and $\varphi_i^* = \mathbf{0}, \forall i \neq j$. Then the maximized objective function under t_i is expressed as

$$\Pi(\theta_i, t_i) \triangleq \frac{\theta_i^3}{4\mathbb{P}(\gamma(\omega(s_i), \mathbf{x}), s_i)} + \frac{\theta_i^2\left(1 + \frac{\theta_i^2}{4\lambda}\right)}{4\lambda\mathbb{P}(\gamma(\omega(s_i), \mathbf{x}), s_i)}. \tag{27}$$

If t_j is the optimal delay tolerance, the following inequalities hold:

$$\Pi(\theta_j, t_j) \leq \Pi(\theta_i, t_i). \tag{28}$$

APPENDIX C PROOF OF THEOREM 1

We consider the case for $\mu_i^* = \{i\}$ and $|\mu_i^*| > 1$. We provide the proof for $\mu_i^* = \{i\}$ in Section III-A and for $|\mu_i^*| > 1$, the miner obtains the same $\Pi(\theta_i, t_i)$. Consequently, selecting any user type from this set is optimal. Suppose that the miner sets positive contract items for at least two types, e.g., one of the types has a higher delay tolerance and lower confidentiality (represented by type-i), and another type has lower delay tolerance and higher confidentiality (represented by type j). Nonetheless, the t_{max} is unique. According to Section III-A, if the miner sets t_{max} to be the lower delay t_j so that both types can participate, it is optimal only to select type-j since it has higher confidentiality (higher confidentiality increases the miner's reward). This scenario contradicts the assumption. Consequently, offering more than one user type a positive contract item under complete information scenario is not optimal.

Therefore, we can deduce that under the complete information scenario, the following conditions hold,

1) if $\mu_i^* = \{i\}$, the miner's optimal contract is $t_{max}^* = t_i, \varphi_i^* = [\frac{\theta_i^2}{4\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)}, \frac{\theta_i^3}{4\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)}]$, and $\varphi_i^* = \mathbf{0}, \forall i \neq j$
2) if $|\mu_i^*| > 1$, the miner's optimal contract is to choose any one type user $i \in \mu_i^*$ with $t_{max}^* = t_i, \varphi_i^* = [\frac{\theta_i^2}{4\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)}, \frac{\theta_i^3}{4\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)}]$, and $\varphi_i^* = \mathbf{0}, \forall i \neq j$
3) if $|\mu_i^*| > 1$, offering only positive contract to one type $i \in \mu_i^*$ results in the same miner's maximal utility.

APPENDIX D PROOF OF LEMMA 3

Suppose the miner only offers a positive contract item for the user types in χ_i, then the miner's objective function can be represented by

$$\max_{t_{max},\varphi} \quad \mathbb{E}\{U_{\varphi_i}\} \tag{29a}$$

$$= \sum_{n_i=0}^{N} \mathbb{1}_{t_i \le t_{max}} \mathbb{P}(n_i)\{r(s_i) - \theta_i \omega(s_i)\}$$

$$\text{s.t.} \quad (12b), \quad \text{and} \quad (12c), \tag{29b}$$

where $\mathbb{P}(n_i) = \binom{N}{n_i} \mathbb{P}_i^{n_i}(1-\mathbb{P}_i)^{N-n_i}$. To comprehend the behavior of the miner under incomplete information scenario, we assume that the miner will offer a contract that provides a zero profit to users, i.e., setting $\theta_i = \theta_{max}$ and $t_{max} = T_{max}$. As a result, we can represent the objective function in (12a) as $\sum_{n_i=0}^{N} \mathbb{1}_{t_i \le t_{max}} \mathbb{P}(n_i)\{\theta_{max}(\beta + \lambda s_i)\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}), s_i) - \theta_{max}\omega(s_i)\}$.

Similarly, we can find the optimal reward and transaction workload expressed as

$$\omega(s_i) = \frac{\theta_{max}^2}{\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}), s_{\chi_i})} \left(\frac{\sum_{n_{\chi_i=1}}^{N} \mathbb{P}(n_{\chi_i}) \frac{1}{\sqrt{1+\lambda}}}{\sum_{n_{\chi_i=1}}^{N} \mathbb{P}(n_{\chi_i}) n_{\chi_i} \theta_{max}} \right),$$
$$r(s_i) = \frac{\theta_{max}^3}{\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}), s_{\chi_i})} \left(\frac{\sum_{n_{\chi_i=1}}^{N} \mathbb{P}(n_{\chi_i}) \frac{1}{\sqrt{1+\lambda}}}{\sum_{n_{\chi_i=1}}^{N} \mathbb{P}(n_{\chi_i}) n_{\chi_i} \theta_{max}} \right). \tag{30}$$

Finally, applying Theorem 1 with the incomplete information contract formulation and (30), we can obtain the following insights,

1) $t_{max}^* = T_{max}$,
2) for all user types in χ_i:

$$\varphi^* = \frac{\theta_{max}^2}{\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}), s_{\chi_i})} \left(\frac{\sum_{n_{\chi_i=1}}^{N} \mathbb{P}(n_{\chi_i}) \frac{1}{\sqrt{1+\lambda}}}{\sum_{n_{\chi_i=1}}^{N} \mathbb{P}(n_{\chi_i}) n_{\chi_i} \theta_{max}} \right),$$
$$\frac{\theta_{max}^3}{\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}), s_{\chi_i})} \left(\frac{\sum_{n_{\chi_i=1}}^{N} \mathbb{P}(n_{\chi_i}) \frac{1}{\sqrt{1+\lambda}}}{\sum_{n_{\chi_i=1}}^{N} \mathbb{P}(n_{\chi_i}) n_{\chi_i} \theta_{max}} \right). \tag{31}$$

3) for any user type $i \notin \chi_i, \phi^* = \mathbf{0}$.

APPENDIX E PROOF OF PROPOSITION 1

Consider two cases in which the miner can maximize its utility by selecting user types with a lower preference than that by selecting types with higher preference. Firstly, consider type a with higher delay tolerance and small confidentiality; type b has lower delay tolerance and high confidentiality; type c and type d have medium delay tolerance and medium confidentiality, respectively. Both types a and b hold a higher preference than types c and d. Nonetheless, the type set χ_i comprising types a and b will have high delay tolerance and confidentiality, while χ_j types c and d will have medium delay tolerance and confidentiality. Therefore, χ_j can provide a higher utility if its existence probability is not very small.

Secondly, suppose the type set χ_H consisting of types with higher preference than χ_L with lower preference. If χ_L has lower preference than χ_H but has higher existence probability, at least when $N = 1$, the miner would like to select χ_L rather than χ_H. Hence, under incomplete information scenario, the type set consisting of types with lower preference may provide a higher miner utility than those with higher preference.

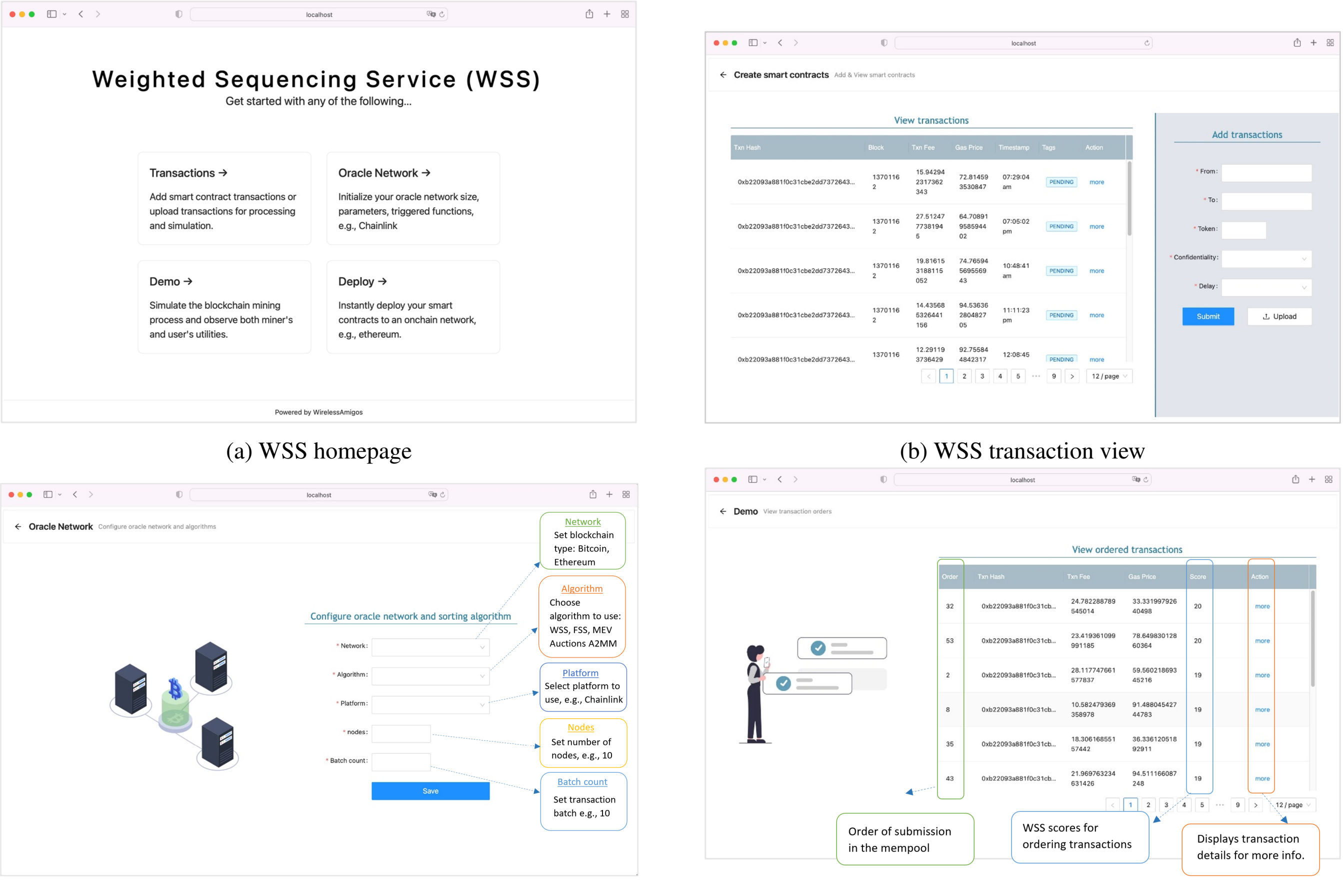
APPENDIX F UNIFORM CONTRACT DESIGN

We consider an optimal uniform contract for our contract analysis as one of the benchmarks. This contract comprises a single uniform contract item for all users. Expressly, $t_{max}^* = t_u, \varphi* = (\frac{1}{\frac{1}{4}\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)\theta_i^2}, \frac{1}{\frac{1}{4}\lambda\mathbb{P}(\gamma(\omega(s_i),\mathbf{x}),s_i)\theta_i^3})$, where t_u is a fixed predetermined delay for all user types.

APPENDIX G WSS IMPLEMENTATION

In this section, we present the software implementation of the WSS system. Fig. 5a shows the software interface of WSS. From Fig. 5a, WSS comprises four main parts: transactions, oracle network (DON), demo, and deploy. WSS transactions provide the platform to add or upload transactions for processing on the blockchain network. The DON provides the functionality that initializes the network size, platform, nodes, triggered functions, and transaction batch count. The demo option simulates the blockchain transaction ordering process for our WSS design. Finally, we present the deploy option to illustrate WSS works on the blockchain when adopted.

Fig. 5b shows the transactions view of our WSS web application. From Fig. 5b, we provide a view transactions and add transactions functionality. The view transactions option shows transactions submitted by users on our platform. Most transactions here will remain pending until ordered for miners to execute. The add transactions can add or upload transactions by interacting with the fields provided. Users will have to specify their transaction confidentiality and delay tolerance when adding transactions. These private information of users are saved and used for our transaction ordering but hidden on the network to avoid frontrunning by users.



localhost

Weighted Sequencing Service (WSS)

Get started with any of the following...

Transactions → Add smart contract transactions or upload transactions for processing and simulation.

Oracle Network → Initialize your oracle network size, parameters, triggered functions, e.g., Chainlink

Demo → Simulate the blockchain mining process and observe both miner's and user's utilities.

Deploy → Instantly deploy your smart contracts to an onchain network, e.g., ethereum.

Powered by WirelessAmigos

(a) WSS homepage

localhost

← **Create smart contracts** Add & View smart contracts

View transactions

Txn Hash	Block	Txn Fee	Gas Price	Timestamp	Tags	Action
0xb22093a881f0c31cbe2dd7372643...	1370116 2	15.94294 2317362 343	72.81459 3530847	07:29:04 am	PENDING	more
0xb22093a881f0c31cbe2dd7372643...	1370116 2	27.51247 7738194 5	64.70891 9585944 02	07:05:02 pm	PENDING	more
0xb22093a881f0c31cbe2dd7372643...	1370116 2	19.81615 3188115 052	74.76594 5695569 43	10:48:41 am	PENDING	more
0xb22093a881f0c31cbe2dd7372643...	1370116 2	14.43568 5326441 156	94.53636 2804827 05	11:11:23 pm	PENDING	more
0xb22093a881f0c31cbe2dd7372643...	1370116	12.29119 3736429	92.75584 4842317	12:08:45	PENDING	more

< 1 2 3 4 5 ··· 9 > 12 / page

Add transactions

* From:

* To:

* Token:

* Confidentiality:

* Delay:

Submit | Upload

(b) WSS transaction view

localhost

← **Oracle Network** Configure oracle network and algorithms

Configure oracle network and sorting algorithm

* Network:

* Algorithm:

* Platform:

* nodes:

* Batch count:

Save

Network: Set blockchain type: Bitcoin, Ethereum

Algorithm: Choose algorithm to use: WSS, FSS, MEV Auctions A2MM

Platform: Select platform to use, e.g., Chainlink

Nodes: Set number of nodes, e.g., 10

Batch count: Set transaction batch e.g., 10

(c) WSS oracle network (DON) setup

localhost

← **Demo** View transaction orders

View ordered transactions

Order	Txn Hash	Txn Fee	Gas Price	Score	Action
32	0xb22093a881f0c31cb...	24.782288789 545014	33.331997926 40498	20	more
53	0xb22093a881f0c31cb...	23.419361099 991185	78.649830128 60364	20	more
2	0xb22093a881f0c31cb...	28.117747661 577837	59.560218693 45216	19	more
8	0xb22093a881f0c31cb...	10.582479369 358978	91.488045427 44783	19	more
35	0xb22093a881f0c31cb...	18.306168551 57442	36.336120518 92911	19	more
43	0xb22093a881f0c31cb...	21.969763234 631426	94.511166087 248	19	more

< 1 2 3 4 5 ··· 9 > 12 / page

Order of submission in the mempool

WSS scores for ordering transactions

Displays transaction details for more info.

(d) WSS demo view

FIGURE 5: Overview of WSS implementation

Fig. 5c illustrates the configuration of our oracle network. We provide the option to choose the network to run the DON on, the algorithm to simulate, platform, the number of nodes, and batch count. In our experiment, we considered ethereum network, WSS algorithm, 10 oracle nodes, and 25 transactions per batch. The number of nodes determines how many oracle nodes are required to run our WSS algorithm, and the transaction batch determines how many transactions that WSS processes in one ordering cycle. This option can also simulate other algorithms such as FSS, MEV auctions, and A2MM.

In Fig. 5d, the demo shows the ordered transactions weights. The weights determine the transaction order based on our proposed WSS algorithm. We show the transaction orders as submitted and their corresponding weights after applying WSS. WSS ensures that miners obtain a maximal utility from the list of transactions in the mempool. The more option comprises the transaction details while obfuscating the users' private information.