Mechanism for Robust Procurements

Yingqian Zhang[1] and Sicco Verwer[2]

[1] Erasmus University Rotterdam, The Netherlands
yqzhang@ese.eur.nl
[2] Radboud University Nijmegen, The Netherlands
S.Verwer@cs.ru.nl

Abstract. We model robust procurement as an optimization problem. We show that its decision version is NP-complete, and propose a backtracking algorithm with cuts that reduce the search-space to find the optimal solution. We then develop a mechanism that motivates agents to truthfully report their private information (i.e., truthful in dominant strategies), maximizes the social welfare (efficient), and ensures non-negative utilities of the participating agents even after execution (post-execution individually rational). In the experiments, we compare our mechanism with an iterated greedy first-price mechanism that represents the current practice in public procurements, in terms of the expected social welfare and the expected payments of the auctioneer. The results show that in terms of social welfare, our mechanism outperforms the greedy approach in all cases except when there exist cheap and reliable agents who can finish the job in time. In terms of payments, our mechanism outperforms the current practice when there are many potential contractors and the optimization constraints are tight.

1 Introduction

Auctions have been used to allocate goods and services in business-to-business, business-to-customer markets, and in public procurements. The increasing popularity of adopting auctions is largely due to its efficiency of allocating goods and its transparency. In the domain of public procurements where the buyers are public sector organizations, it has been specified by EU that any large procurement needs to be conducted through some type of tender/auction [1]. Despite its advantages, there are many challenges in designing auctions, and they often end up with an outcome that is far from optimal, as is shown for example by the recent news about an European public tender for the purchase of a new gun for the Dutch Police [3]:

The manufacturer of the pistol that initially won the contract could not deliver the requisite quality in a mass production setting. Hence, the manufacturer that finished second in the tender was asked to provide the service. However, another two contractors appealed to court and insisted that a new procurement should be conducted. The court decided that a new tender process should be organized in order to choose another product if the original winner should be rejected [4]. This will delay the introduction of a new police pistol by 1 to 1.5 years [5].

This delay and the resulting additional costs are highly undesirable. In this case, only one winner was selected from an auction based on some criteria. Unfortunately, in the face of uncertainties on services, this winner determination solution is not robust enough to ensure a reliable outcome. Furthermore, the contractors are self-interested and they often appear to have a biased optimism about the risks in order to win a contract [8]. Similar situations where uncertain service quality and self-interested agents degrade the benefit of the society/business are not uncommon in public procurement domain or in business-to-business markets [6]. In this paper, we aim to design a more robust auction by introducing redundancy into the selected solution. More specifically, we consider each contractor's reliability of executing the contract in winner determination and we use backup contractors to recover from possible failures. As a result, using auctions, we determine not only one winning agent but (possibly) several backup agents. Including multiple winners in the procurement solution has several potential advantages: to quickly recover from failures, to avoid disputes and legal procedures (as shown in the gun tender case of the Dutch Police), and to optimize the expected performance in the presence of uncertainties. However, it is notably difficult to develop the optimal robust plan (or effective implementation) if the contractors misreport their true information. Therefore, this paper also studies how to design mechanisms [10] in order to prevent misreporting. Research into mechanism design is about creating incentives for self-interested agents to report the correct information, i.e., to be truthful. In many cases payments are used to create these incentives. A mechanism then chooses a (preferably socially optimal) outcome and specifies such payments, both based on the reported information.

We are not the first who study how to design mechanisms in the presence of uncertainties. In the public procurement domain, Chen et al. [6] study how to model *quality* levels for the procured items from the suppliers. Every supplier can make several bids including a cost and a quality level. This way, a public sector organization can select an optimal combination of services by considering both cost and quality. There is a body of work on designing mechanisms when *execution of outcome* is uncertain [9,11]. When the execution of the chosen outcome is realized by agents themselves, an agent's *actual valuation* of the result depends not on the selected outcome, but on the *actual outcome*. Thus, a two-stage mechanism is required to obtain truthfulness for such settings [9]. The idea behind such a two-stage mechanism is that the payments to agents are computed based on the actual outcome. These mechanisms, however, do not handle recovering from failures. Examples of mechanisms that consider failure handling are proposed by [13]. Stein et al. [13] propose mechanisms for procurement in service-oriented systems, where a customer procures a task with a deadline. Service providers are uncertain about their execution times. These are taken into account in a robust procurement strategy that may invoke a provider at any time before the deadline in order to increase the probability of successful execution of the task. Furthermore, the customer may have several providers executing the same given task concurrently. Various mechanisms have been introduced to tackle the two situations when the only private information is the cost of performing task, and when additionally, the uncertainty of execution duration of each provider is also private.

Our work is similar to [13] in the way of introducing redundancy to tackle the uncertain execution and to enhance the expected performance. However, the work of [13] cannot be directly applied in our problem, as our model is built with the following different assumptions: (1) Multiple contractors cannot perform the same service at the same time. For instance, it is not realistic that two construction companies build a road concurrently. (2) The past performance of a contractor is public knowledge. It is often true that in real-world procurements, organizations and firms maintain scores for their suppliers and contractors, and use these scores to determine the outcomes [7]. Such scores can be obtained from a scorecard system (e.g., Oracle's Peoplesoft Manufacturing Scorecard [2]), or from trust/reputation systems [12] that have been investigated extensively within the agent community.

In this paper, we construct an algorithm and a mechanism for incentivizing truth-telling in public procurement problems with uncertainties. Our contributions are the development of a framework for studying such procurement problems, proving that minimizing cost in this framework is NP-complete, developing a quick algorithm that minimizes this cost (Section 3), and providing a novel multi-stage mechanism that has desirable properties such as efficient, truthful in dominant strategy, and post-execution individually rational (Section 4). We show experimentally (Section 5) that our approach significantly outperforms the current practice (starting a new auction after the first winner has failed) in settings where there are many potential contractors and the constraints in the optimization problem are tight. We stress here that although our motivating domain is in public procurements, the proposed mechanism can also be applied to other business-to-business or business-to-customer domain.

2 Preliminaries

In our model, there is one auctioneer (procurer) and a set A of bidders (contractors). The procurer announces a job with a deadline D and a minimal completion probability γ. Each bidder $i \in A$ submits a bid that consists of the following information: $< c_i, d_i, \beta_i >$, where c_i is the cost of executing the job, d_i is the duration of completing the job, and β_i specifies the reservation fee. The reservation fee occurs when an agent i is selected as a backup agent in the procurement schedule. In addition, we assume the auctioneer has the information about the reliability $r_i \in [0, 1]$ of each participant bidder i. We assume that all the reliabilities are independent.Given the set of bids and the reliability of agents, the procurer determines a set of winners $S = (A_{i_1}, ..., A_{i_m}) \subseteq A$ as the outcome ϕ. S is a total ordered set, that is, A_{i_1} is the first winner who will be given the contract immediately, and $A_{i_2}, ..., A_{i_m}$ are backup agents. A_{i_2} will be invoked at time d_{i_1} to execute the job if A_{i_1} fails to deliver as promised by its deadline d_{i_1}, and A_{i_3} will be invoked at time $d_{i_1} + d_{i_2}$ once A_{i_2} fails to complete the job at $d_{i_1} + d_{i_2}$, etc[1]. The probability that an ordered set of contractors $S = (A_{i_1}, ..., A_{i_m})$ will finish the project within the deadline $\sum_{k=1}^{m} d_{i_k}$ is equal to $1 - \prod_{k=1}^{m}(1 - r_{i_k})$. The expected cost incurred by S then becomes:

$$E[Cost(S)] = \sum_{k=1}^{m}(c_{i_k} + \beta_{i_{k+1}}) \prod_{l=0}^{k-1}(1 - r_{i_l}) \tag{1}$$

[1] For simplicity, we assume a new provider starts immediately after the previous provider should have completed the job. Our model can be easily extended to include delays in starting times.

where $r_{i_0} = 0$ and $\beta_{i_{m+1}} = 0$. Denote by $\mathbb{S}$ the possible ordered sets of contractors that may finish the project within deadline D with probability at least γ, that is:

$$\mathbb{S} = \{(A_{i_1}, ..., A_{i_m}) : \sum_{k=1}^{S} d_{i_k} \leq D \textit{ and } \prod_{k=1}^{m}(1 - r_{i_k}) \leq 1 - \gamma\} \tag{2}$$

The robust procurement problem (RPP) can be now defined as the following constrained optimization problem: $\min_{S \in \mathbb{S}} E[Cost(S)]$.

Example 1. Suppose a job is announced with the deadline $D = 100$ and the required completion probability $\gamma = 0.85$. There are 3 contractors $\{A_1, A_2, A_3\}$, with reliabilities $r_1 = 0.7$, $r_2 = 0.8$, $r_3 = 0.6$ respectively. Their bids are: A_1: $< c_1 = 20, d_1 = 40, \beta_1 = 5 >$, A_2: $< c_2 = 20, d_2 = 30, \beta_2 = 7 >$, and A_3: $< c_3 = 15, d_3 = 40, \beta_3 = 7 >$. Given the bids, the optimal schedule is: $S = (A_3, A_1)$. A_3 will be awarded to execute the job immediately, and A_1 will receive a reservation fee. In case of failure upon A_3's deadline d_3, contractor A_1 will take up the job at time d_3. It is easy to verify that S satisfies the constraints, i.e., the total durations of S is $d_3 + d_1 = 80 < D$, and the completion probability is $1 - (1 - 0.6)(1 - 0.7) = 0.88 > \gamma$. The total cost is computed by Equation 1: $c_3 + \beta_1 + c_1(1 - r_3) = 28$, which is optimal. ■

In this paper, we consider a setting where agents are self-interested and their declarations, i.e. $< c_i, d_i, \beta_i >$, are private information. Let this so-called *type* of each agent i be denoted by θ_i. A *type profile* θ is a vector $(\theta_1, \ldots, \theta_n)$ associating each agent with a type. We use $\theta_{-i} = (\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_n)$ to denote the type profile without the type of agent i. Given a type profile, a direct-revelation mechanism selects an *outcome* $\phi = f(\theta)$ using an algorithm f from the set of possible outcomes, and a payment $\bar{p}_i(\phi, \theta)$ for each agent that together define the utility of an agent $\bar{u}_i(\phi, \theta) = \bar{v}_i(\phi, \theta_i) - \bar{p}_i(\phi, \theta)$. $\bar{v}_i(\phi, \theta_i)$ specifies the expected valuation of agent i on the outcome ϕ. Let $S_\phi = (A_{a_1}, ..., A_{a_m})$ denote the ordered set given the outcome ϕ. The expected valuation of agent i (i.e. A_{a_i}) prior to execution is computed as:

$$\bar{v}_i(\phi, \theta_i) = -c_i \prod_{l=0}^{i-1}(1 - r_l) - \beta_i \prod_{l=0}^{i-2}(1 - r_l) \tag{3}$$

Note the realized valuation of a contractor A_{a_i} after execution of the procurement schedule depends on the actual execution, and it is defined as follows.

$$v_i(\phi, \theta_i) = \begin{cases} -c_i & \text{if } A_{a_i} = A_{a_1}; \\ -c_i - \beta_i & \text{if } A_{a_i} \neq A_{a_1}, \text{ and } A_{a_i} \text{ is invoked;} \\ -\beta_i & \text{if } A_{a_{i-1}} \text{ is invoked and } A_{a_i} \text{ is not;} \\ 0 & \text{otherwise.} \end{cases}$$

The procurer's job and the bidders' reliabilities are public information in the system. The expected *social welfare* is defined as the sum of the valuations of the contractor agents, i.e., $\bar{w}(\phi) = \sum_{i=1}^{m} \bar{v}_i(\phi, \theta_i)$. The main aim of mechanism design is to construct mechanisms that select outcomes where the expected social welfare is highest. These mechanisms are called *efficient*. A major step in being able to obtain the efficient outcome is to give incentives to agents to reveal their true type. We say a mechanism is incentive compatible if truthful reporting is an equilibrium. We are interested in those mechanisms that are truthful in dominant strategies. We call such mechanisms *truthful* mechanisms. Since we have probabilistic information about whether the outcome will be executed successfully, our definitions are ex-ante (see e.g. [11]).

Definition 1. *A mechanism is* truthful*, if for every agent i, its true type θ_i, and every other type $\hat{\theta}_i$, and for all other agents' true types θ_{-i} and declared types $\hat{\theta}_{-i}$, it holds that $\bar{u}_i\left(\phi, (\theta_i, \hat{\theta}_{-i})\right) \geq \bar{u}_i\left(\phi', (\hat{\theta}_i, \hat{\theta}_{-i})\right)$, where $\phi = f(\theta_i, \hat{\theta}_{-i})$ and $\phi' = f(\hat{\theta}_i, \hat{\theta}_{-i})$.*

Thus, with a truthful mechanism, an agent maximizes its expected utility by declaring its type truthfully, no matter what costs and durations other agents declare.

A mechanism is *individually rational* (IR) if for every agent i, for every $\hat{\theta} = (\theta_i, \hat{\theta}_{-i})$ where agent i is truthful, the utility of agent i is non-negative, i.e., $\bar{u}_i(f(\theta_i, \hat{\theta}), (\theta_i, \hat{\theta})) \geq 0$. Following [13], we are interested in mechanisms that are *post-execution* IR, as it is important that a truthful agent will not receive a negative utility no matter what the actual execution outcome will be. More formally, a mechanism is post-execution IR if a truthful agent i's utility after execution $u_i(f(\hat{\theta})) \geq 0$, regardless of θ_{-i}.

3 Complexity and Algorithm for RPP

We show that the robust procurement problem RPP is computationally hard by showing that the subproblem of finding a suitable set of contractors is NP-complete.

ROBUST PROCUREMENT WITHOUT COSTS

Instance: Given a set of bids $< c_i, d_i, \beta_i >$ from agents A, the reliability of agents r_i for each $i \in A$, a deadline D, and a predefined reliability threshold γ.

Question: Is S empty, i.e., can the job be completed on time with sufficient probability?

Theorem 1. *The robust procurement without costs problem is* NP*-complete.*

Proof. The proof is by reduction from a well known NP-complete problem:

SUBSETSUM

Instance: Given a set of positive integers $I = \{v_1, \ldots, v_m\}$ and a desired value K.

Question: Does there exists a subset of these integers $S \subseteq I$, such that they sum to K: $\sum_{v_i \in S} v_i = K$?

We map a subsetsum instance as follows.

- $< c_i, d_i, \beta_i > = < 0, v_i, 0 >$, for $1 \leq i \leq m$
- $r_i = 1 - 2^{-v_i}$; $D = K$; $\gamma = 1 - 2^{-K}$.

($\Rightarrow$) Suppose $S = \{v_{i_1}, \ldots, v_{i_m}\}$ is a solution for the subsetsum problem, then since $\sum_{k=1}^{m} v_i = K$, we have:

1. $\sum_{k=1}^{m} d_{i_k} = K \leq D$, and
2. $\prod_{k=1}^{m} 2^{-v_{i_k}} = \prod_{k=1}^{m}(1 - r_{i_k}) = 2^{-K} \leq 1 - \gamma$.

Thus the corresponding robust procurement without costs problem is satisfied.
($\Leftarrow$) Suppose there exists a solution $S = (A_{i_1}, ..., A_{i_m})$ to the mapped robust procurement without costs problem instance, then:

1. $\sum_{k=1}^{m} d_{i_k} = \sum_{k=1}^{m} v_{i_k} \leq D = K$, and
2. $\prod_{k=1}^{m}(1 - r_{i_k}) = \prod_{k=1}^{m} 2^{-v_{i_k}} \leq 1 - \gamma \ = 2^{-K}$, therefore $2^{\sum_{k=1}^{m} -v_i} \leq 2^{-K}$, $\sum_{k=1}^{m} -v_{i_k} \leq -K$, and thus $\sum_{k=1}^{m} v_{i_k} \geq K$.

Since $\sum_{k=1}^{m} v_{i_k} \geq K$ and $\sum_{k=1}^{m} v_{i_k} \leq K$, we have $\sum_{k=1}^{m} v_{i_k} = K$. Thus, there exists a solution for the original subsetsum problem.

The mapping only requires the computation of a polynomial number of multiplications. The verification of a solution to the robust procurement problem without costs can be done in polynomial time by computing sums, products, and checking bounds. Therefore, it is in NP. □

The robust procurement problem is thus a computationally hard problem. Although this means that it is difficult to optimize in theory, in practice an optimal solution can often be found efficiently because the number of contractors with a total duration less than the job deadline is typically small. Furthermore, the number of contractors required to reach the desired reliability is usually limited. The total search space is therefore typically small, and even a brute-force search should often be able to find the optimal solution within reasonable time.

In underconstrained settings with many contractors, considering all possible orderings will take very long. We therefore propose a backtracking algorithm that iteratively appends one contractor to a partially constructed solution (Algorithm 1). The algorithm uses bounds to further reduce the search space based on the following observations: (1) Adding a contractor to the end of an ordering always increases the expected cost of the ordering. (2) Adding a contractor always increases the total duration of an ordering. The algorithm can therefore safely cut searches on orderings that: i) have a duration greater than the deadline, ii) have sub-orderings that do not satisfy the reliability constraint, and iii) have a total cost greater than the total cost of an already visited valid ordering. With these three cuts in place, the algorithm is very fast in practice.

4 Mechanism for RPP

We now study how to design a mechanism in order to incentivize agents to declare their true private information. One possible truthful mechanism is the Vickrey-Clarke-Groves (VCG) mechanism. However, it has been shown in [13] that the VCG mechanism is individual rational in expectation, but not post-execution individually rational. Therefore, an agent may get a negative (realized) utility after the execution of the procurement outcome. The main reason is that the payment was computed based on an agent's expected marginal contribution to the system, and hence, in some execution outcome, it cannot compensate the agent's actual costs. The authors of [13] proposed two types of mechanisms to ensure post-execution IR, however, these two mechanisms are not

Algorithm 1. Backtracking recursion for robust procurement

```
Require: a set of bids $< c_i, d_i, \beta_i >$ from agents $A$, the reliability of each agent $pr_i$, a deadline $D$, a reliability threshold $\gamma$, an ordered subset of contractors $S = (A_{i_1}, ..., A_{i_m})$, and a globally declared MINCOST variable
Ensure: returns an outcome in $\mathcal{S}$ with smallest expected cost
  if $\sum_{k=1}^{m} d_{i_k} > D$ or $E[Cost(S)] >$ MINCOST then
    return $\emptyset$
  end if
  if $\prod_{k=1}^{m}(1 - r_{i_k}) < 1 - \gamma$ then
    MINCOST := $\min(E[Cost(S)]$,MINCOST$)$
    return $S$
  end if
  VALUE := $\infty$, $S' := \emptyset$
  for all agents $a \in A \setminus S$ do
    $S'$ = RECURSE($S$.APPEND($a$), ...)
    if $S' \neq \emptyset$ then VALUE := $\min(E[Cost(S')]$,VALUE$)$
  end for
  return $S'$ that resulted in VALUE
```

efficient, i.e., they produce sub-optimal social welfare. In this paper, we borrow the idea of defining payments on the realization of the computed outcome [9] and define a multi-stage Grove mechanism that is efficient, truthful and post-execution individually rational. We assume that the outcome, i.e., whether or not an agent successfully completes the given job, can be observed by the auctioneer. The payment of an agent then thus comes to depend upon this observed outcome. We now introduce the multi-stage Groves mechanism, called RobustProcurement, for the robust procurement problem.

Definition 2. *(*RobustProcurement *mechanism)*
The auctioneer announces a job with deadline and completion probability threshold γ.

1. *The contractor agents declare their types $\theta = (\theta_i, \theta_{-i})$ to the auctioneer.*
2. *The auctioneer then finds an optimal schedule ϕ using Algorithm 1, and informs each agent of the outcome ϕ.*
3. *Every agent i in the ordered set $S_\phi = (A_{a_1}, ..., A_{a_m})$ receives its marginal contribution as payment:*

$$p_i(\phi, \theta_i) = -\bar{w}(\phi, \theta) + \bar{w}_{-i}(\phi', \theta_{-i}), \tag{4}$$

   where $\bar{w}(\phi, \theta)$ is the expected social welfare, $\phi' = f(\theta_{-i})$ is the efficient outcome computed by Algorithm 1 without agent i's participation, and $\bar{w}_{-i}(\phi', \theta)$ is the social welfare on ϕ'.
4. *The first winner A_{a_1} receives an additional payment:*

$$p'_i(\phi, \theta_i) = v_i(\phi, \theta_i) \tag{5}$$

   Note the term in the equation is agent's realized valuation, and therefore $v_1(\phi, \theta_1) = -c_{A_{a_1}}$. A_{a_1} starts to execute the job. The second winner A_{a_2} is notified to be standby. The reservation cost is incurred.

5. *At the deadline of agent $A_{a_{i-1}}$ for $2 \leq i \leq |S| - 1$, do*
   - *Transfer the additional payment to agent i (i.e. agent A_{a_i}) computed by Equation 5, where agent A_{a_i}'s realized valuation $v_i(\phi, \theta_i)$ is computed based on the realization of the schedule ϕ, i.e.,*

$$v_i(\phi, \theta_i) = \begin{cases} -c_i - \beta_i & \textit{if } A_{a_{i-1}} \textit{ does not complete the job by its deadline;} \\ -\beta_i & \textit{otherwise.} \end{cases}$$

   - *If $A_{a_{i-1}}$ completes the job, inform the remaining agents $j \in S$, and terminate the mechanism. Otherwise, agent A_{a_i} starts to execute the job, and agent $A_{a_{i+1}}$ is notified to be stand-by.*
   - *Loop step 5 until the mechanism is terminated.*

The multi-stage mechanism RobustProcurement combines the traditional Grove mechanism where payments are based on the computed outcome and the two-stage Grove mechanism that payments are computed according to the realized outcome. In RobustProcurement mechanism, before execution, the auctioneer transfers to every agent i in the schedule S_ϕ the amount defined in Equation 4. In additional, some agents may receive additional payments during execution as shown in Equation 5. The advantages of such a payment scheme is that the mechanism is guaranteed to sufficiently compensate the agents' actual costs, and thus it is post-execution IR. In addition, the RobustProcurement is efficient and truthful thanks to the Grove mechanism. Before we prove these properties, we illustrate our mechanism by the following example.

Example 2. Given the problem instance in Example 1, the mechanism finds the optimal schedule ϕ with the ordered set $S_\phi = (A_3, A_1)$. Let $\bar{w}(\phi)$ denote the social welfare on ϕ. Note that without A_3's participation, the optimal schedule will be $S_{\phi_{-3}} = (A_2, A_1)$. Without A_1's participation, the optimal schedule becomes $S_{\phi_{-1}} = (A_3, A_2)$. The mechanism computes the following payments (Equation 4) for two agents A_3 and A_1, prior to execution: $p_3 = -\bar{w}(\phi) + \bar{w}_{-3}(\phi_{-3}) = (c_3 + \beta_1 + c_1(1 - r_3)) - (c_2 + \beta_1 + c_1(1 - r_2)) = 28 - 29 = -1$, and $p_1 = -\bar{w}(\phi) + \bar{w}_{-1}(\phi_{-1}) = (c_3 + \beta_1 + c_1(1 - r_3)) - (c_3 + \beta_2 + c_2(1 - r_3)) = 28 - 30 = -2$. Note that A_3 also receives the additional payment: $p'_3 = v_3 = -c_3 = -15$.

After receiving payments, agent A_3 starts to execute the job. By its deadline $d_3 = 40$, the mechanism checks whether A_3 completes the job. There are two cases:

Case 1: A_3 completes the job, the second winner A_1's additional payment is computed by Equation 5 as follows: $p'_1 = v_1 = -\beta_1 = -5$, and the mechanism terminates.

Case 2: If A_3 did not complete the job, A_1 receives the additional payment: $p''_1 = v_1 = -(\beta_1 + c_1) = -25$, and A_1 starts to execute the job.

Notice that the realized utility (after execution) of agent A_3 is: $u_3 = v_3 - (p_3 + p'_3) = -c_3 - (-1 - 15) = 1$. The realized utility for A_1 in Case 1: $u_1 = v_1 - (p_1 + p'_1) = 2$, and in case 2 $u_1 = v_1 - (p_1 + p''_1) = -(\beta_1 + c_1) - (p_1 + p''_1) = 2$. ■

Theorem 2. *The* RobustProcurement *mechanism is efficient.*

Proof. Given an outcome ϕ with an ordered set S, the expected social welfare $\bar{w}(\phi)$ is defined as the sum of the expected valuations of all agents that are computed by Equation 3:

$$\bar{w}(\phi) = \sum_{i=1}^{m} \bar{v}_i(\phi, \theta_i) = \sum_{a_i \in S} -c_{a_i} \prod_{l=0}^{i-1}(1 - r_{a_l}) - \beta_{a_i} \prod_{l=0}^{i-2}(1 - r_{a_l}) = -E[Cost(S)].$$

Thus, the optimal solution for the constrained optimization problem maximizes the expected social welfare, and gives an efficient mechanism. □

Theorem 3. *The* RobustProcurement *mechanism is truthful in dominant strategies.*

Proof. Let θ_i be the true type of agent i and $\hat{\theta}_i$ be any other type. Given all other agents' declared types $\hat{\theta}_{-i}$ and θ_i, let $\phi^* = f(\theta_i, \hat{\theta}_{-i})$ be the optimal outcome that maximizes the expected social welfare $\bar{w}(\phi^*, \theta_i, \hat{\theta}_{-i})$. Denote the resulting outcome if agent i reporting $\hat{\theta}_i$ by $\phi = f(\hat{\theta}_i, \hat{\theta}_{-i})$. Agent i's utility by declaring $\hat{\theta}_i$ is computed as: $u_i\left(\phi, (\hat{\theta}_i, \hat{\theta}_{-i})\right) = v_i(\phi, \theta_i) - p_i(\phi, \hat{\theta}_i)$. Due to the linearity of expectation, its expected utility is then: $\bar{u}_i\left(\phi, (\hat{\theta}_i, \hat{\theta}_{-i})\right) = \bar{v}_i(\phi, \theta_i) - \bar{p}_i(\phi, \hat{\theta}_i)$. The expected payment of i is computed as the sum of the payment in Equation 4 and the expected additional payment in Equation 5. Hence, $\bar{p}_i(\phi, \hat{\theta}_i) = -\bar{w}(\phi, \hat{\theta}) + \bar{w}_{-i}(\phi', \hat{\theta}_{-i}) + \bar{v}_i(\phi, \hat{\theta}_i)$.

Therefore, the expected utility of i is: $\bar{u}_i\left(\phi, (\hat{\theta}_i, \hat{\theta}_{-i})\right) = \bar{v}_i(\phi, \theta_i) - \bar{p}_i(\phi, \hat{\theta}_i) = \bar{w}(\phi, \hat{\theta}_i, \hat{\theta}_{-i}) - \bar{w}_{-i}(\phi', \hat{\theta}_{-i})$. Agent i cannot influence the value of the second term, but i can maximize the first term by declaring $\hat{\theta}_i = \theta_i$ as $\phi^* = f(\theta_i, \hat{\theta}_{-i})$ maximizes $\bar{w}$. □

Theorem 4. *The* RobustProcurement *mechanism is individually rational and post-execution individually rational.*

Proof. Let $\phi = f(\theta_i, \hat{\theta}_{-i})$ be the outcome when agent i declares truthfully, and $\phi' = f(\hat{\theta}_{-i})$. If agent $i \notin S_\phi$, then $\bar{u}_i = 0$. If $i \in S_\phi$, then its expected utility is computed as that shown in the proof for truthfulness: $\bar{u}_i\left(\phi, (\theta_i, \hat{\theta}_{-i})\right) = \bar{w}(\phi, \theta_i, \hat{\theta}_{-i}) - \bar{w}_{-i}(\phi', \hat{\theta}_{-i})$. As the mechanism is efficient, $\bar{w}(\phi, \theta_i, \hat{\theta}_{-i}) \geq \bar{w}_{-i}(\phi', \theta_{-i})$. Thus, a truthful agent will always receive a non-negative expected utility.

We now show the mechanism is also post-execution IR. When agent $i \notin S_\phi$, it receives 0 as payment, and its cost is 0, hence its realized utility $u_i = 0$. For agent $i \in S_\phi$, we distinguish three cases:

- Agent i is the first winner, i.e. A_{a_1}. A_{a_1} receives the payment that is computed by Equation 4 and the additional payment (Equation 5). Since its expected cost is its declared cost, its realized utility is equal to the expected utility. Therefore, A_{a_1} will not receive a negative utility after execution.
- When agent $i \neq A_{a_1}$ and it is invoked for executing the job or notified to be on standby: in this case, besides the payment computed in Equation 4, it receives the addition payment based on Equation 5, and thus its realized utility is:

$$\begin{aligned} u_i\left(\phi, (\theta_i, \hat{\theta}_{-i})\right) &= v_i(\phi, \theta_i) - (p_i(\phi, \theta_i, \hat{\theta}_{-i}) + p'_i(\phi, \theta_i, \hat{\theta}_{-i})) \\ &= v_i(\phi, \theta_i) - (-\bar{w}(\phi, \theta_i, \hat{\theta}_{-i}) + \bar{w}_{-i}(\phi', \hat{\theta}_{-i}) + v_i(\phi, \theta_i)) \\ &= \bar{w}(\phi, \theta_i, \hat{\theta}_{-i}) - \bar{w}_{-i}(\phi', \theta_{-i}) \geq 0 \end{aligned}$$

– If agent i is not even notified to be on standby, its payment is the one computed by Equation 4. Since its actual task cost and reservation cost are 0, $v_i(\phi, \theta_i) = 0$. Its realized utility is hence: $u_i\left(\phi, (\theta_i, \hat{\theta}_{-i})\right) = v_i(\phi, \theta_i) - (-\bar{w}(\phi, \theta_i, \hat{\theta}_{-i}) + \bar{w}_{-i}(\phi', \hat{\theta}_{-i}))$. Since $\bar{w}(\phi, \theta_i, \hat{\theta}_{-i}) - \bar{w}_{-i}(\phi', \hat{\theta}_{-i})) \geq 0$ and $v_i(\phi, \theta_i) = 0$, we have $u_i\left(\phi, (\theta_i, \hat{\theta}_{-i})\right) \geq 0$.

Therefore, agent i always receive non-negative utility after execution. □

5 Experiments

In order to evaluate our algorithm and mechanism, we investigate their performance and compare it to a greedy mechanism that represents the current practice in procurement: create a first-price auction, select a winning contractor, and create a new auction if the contractor fails. The mechanism iteratively selects a contractor:

Definition 3. *(Iterated greedy mechanism)*

1. *The contractors $A_i \in A$ declare their types $\theta_i =< c_i, d_i >$ to the auctioneer.*
2. *The auctioneer then finds a winner A' using the following greedy algorithm: Create for every agent A_i an ordered multiset $S = (A_i, A_i, \ldots, A_i)$ containing the minimum amount of copies of A_i such that $\prod_{A_i \in S}(1 - r_i) \leq 1 - \gamma\}$. If in addition it holds that $\sum_{A_i \in S} d_i \leq D$, compute $E[Cost(S)]$ excluding reservation costs and use this value as a heuristic, use ∞ otherwise.*
3. *The auctioneer pays to the winner A' the amount equal to its declared cost c_i.*
4. *The contractor A' executes the job. If it fails by its deadline $d_{A'}$, the mechanism starts another iteration to select another winner without A'.*

The intuition behind the greedy mechanism is that in procurement practice, one wants to select a contractor that minimizes the expected cost of finishing job before the deadline with sufficient probability. Since the greedy method makes no reservations, these costs are set to zero. Instead, the greedy method creates an entire new auction for selecting the second winner. Since there will be costs associated with this in procurement practice, we include a constant cost value for all greedy iterations after the first. For comparison purposes, we link this value to the average reservation costs of all contractors in our experiments. We compare our RobustProcurement mechanism with the greedy mechanism in terms of: (1) *social welfare*, i.e., the expected total cost by the contractor agents, and (2) *the expected total payment* that the auctioneer pays to the contractors. The social welfare measures the quality of the schedules returned by the two mechanisms. Since the iterated greedy mechanism uses a first-price auction, the mechanism is typically not truthful, i.e., agents may misreport their private information. Therefore, the difference of the total payments induced by two mechanisms indicates the price of truthfulness, i.e., the additional cost that the auctioneer needs to pay in order to incentivize the truthful bidding of the contractors.

Setup. Every robust procurement instance contains a set of bids $< c_i, d_i, \beta_i >$ from agents A, the reliability of agents r_i, a deadline of the job D, and a predefined completion probability threshold γ. A random agent is generated by first sampling the cost c_i and duration d_i uniformly at random from $[50, 150]$. The reliability values r_i are then drawn depending on the sampled cost and duration:

$c_i + d_i \in$	$[100, 140)$	$[140, 180)$	$[180, 220)$	$[220, 260)$	$[260, 300]$
$r_i \in$	$[0.2, 0.4]$	$[0.3, 0.5]$	$[0.4, 0.6]$	$[0.5, 0.7]$	$[0.6, 0.8]$

An agent's reliability is thus sampled uniformly at random from a different range, depending on the sum of its cost and duration. This ensures that an agent is cheap and fast if and only if it is unreliable. Similarly, slow and expensive agents are reliable, and average agents have an average reliability. This scheme also ensures that no agent is generated that "dominates" the others, i.e., an agent that has the highest reliability and smallest costs and execution durations. Such an agent would make the optimization problem a lot easier and very unrealistic. Finally, the reservation cost β_i of every agent is drawn uniformly at random from $[5, 15]$.

Using this generator, we create many different sets of agents for a given combination of D, γ, and the number of agents m. We use 300, 400, and 500 as possible deadline values. The threshold values are either 0.9, 0.95, and 0.975. The number of agents are set to 10, 20, or 50. For every combination, we generate 50 different problem instances. Every instance is solved using our robust procurement algorithm and compared with the result of running the aforementioned greedy procedure using three different costs for every iteration after the first. We compute a greedy solution first for 0 re-iteration costs (i.e., zero overhead in the figures), then one for 1 time the average reservation costs of the agents (i.e., once overhead), and finally 2 times the average reservation costs (i.e., twice overhead). Since in practice, restarting an entire auction can be very costly and cause severe additional delays [3], we believe this last setting to be the closest to the real-world cases. Furthermore, although the iterated greedy mechanism uses first-price and thus does not guarantee agents declare their true costs and durations, we make the assumption that all agents bid truthfully for comparison purposes. Note that this assumption is often not true in practice [8], and it is in the favor of the greedy mechanism in terms of the total payment.

Results of 0.9 reliability threshold *(Figure 1).* In general, the total cost incurred using robust procurement algorithm is less than the total cost of the greedy method. Sometimes, however, the cost can be higher even in the case of the third greedy version. This is because our method always pays the reservation cost to every redundant (standby) agent in the schedule, while the greedy method only pays an additional cost if the first winning agent failed. Thus, we expect to spend about 10 more than greedy version two in the same solution (depending on the agents and their reservation costs). Since the difference between our total cost and that of the second greedy version is consistently below 10 and often below 0, our algorithm thus often finds better solutions than greedy.

The difference between our payments and the greedy algorithm, unfortunately, are typically larger than 0. This means it is costly to motivate agents to be truthful. This holds especially for the problem instances with a high deadline (500). Interestingly, in these cases the greedy method also performs very well. We believe both are due to a few

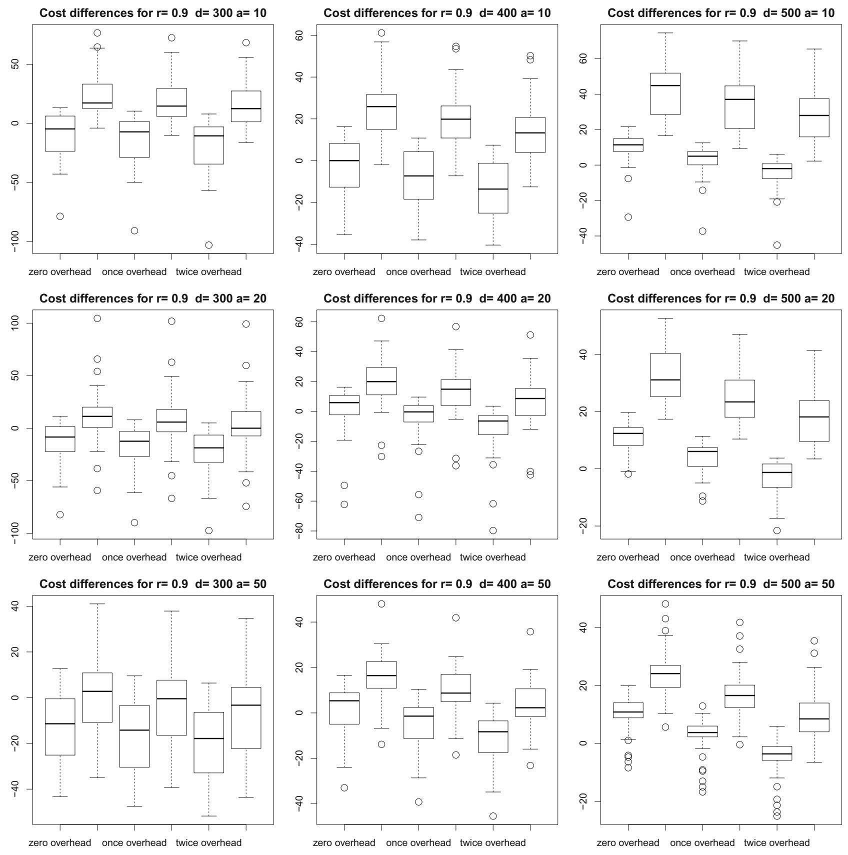

Fig. 1. Boxplots of the total cost (first plot) and payment (second plot) obtained using our exact method minus the cost of the 3 greedy versions (3 times 2 plots) on the instances (50 per plot) that require a reliability of 0.9 ($r := \gamma, a := m, d := D$).

slow but cheap agents that together can finish the task with sufficient probability. The greedy rule will select these agents. Furthermore, their marginal contribution is large since they are both cheap and reliable. In the smaller deadline cases, such agents cannot be used since they will fail to reach sufficient reliability before the deadline.

Overall, our robust procurement mechanism only outperforms (using payment) the greedy method on the instances with the smallest deadline 300 and the largest number of agents 50. With such a tight deadline, optimization is difficult and greedy performs bad. Furthermore, with many agents, the marginal contribution of each of these agents is smaller, leading to small payments.

Results of 0.95 reliability threshold *(Figure 2).* These instances are more constrained than those in Figure 1. This has a positive effect on the performance of our method compared to greedy. In terms of social welfare, the difference is now often below 0, even in the case of zero overhead. With a high deadline, their performance is now comparable.

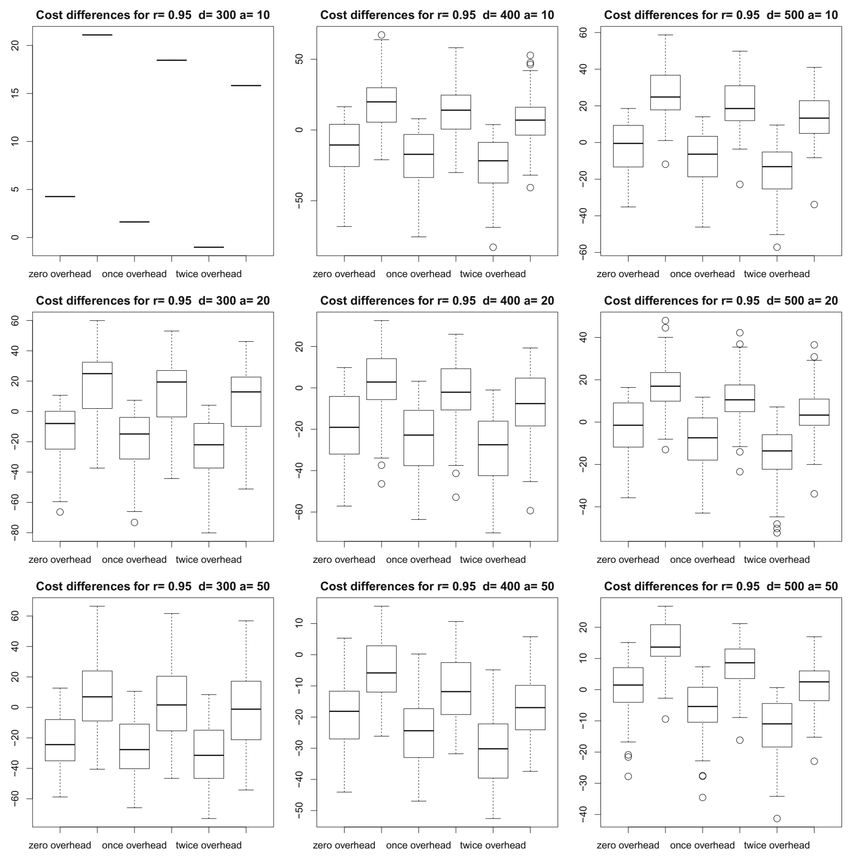

Fig. 2. The same plots as Figure 1 but for the instances that require a reliability of 0.95. There is no plot for 10 agents with a deadline of 300 since this setting resulted in to few solvable instances.

In terms of payments, our method still pays more than greedy. This is not very surprising since greedy does not include any payments that make the agents bid truthfully. It is therefore reasonable to claim that our mechanism outperforms the greedy approach if the difference between payment and cost is located around the 0 mark, instead of strictly below it. In this view, our mechanism outperforms greedy in many cases: all except those with a high deadline (500), however, even in those cases, it is better to use our mechanism when there are many agents (50) and the overhead is more than twice the average reservation cost. An interesting phenomenon is that although our mechanism in general performs well in tightly constrained problems (because finding a good solution is hard), if these constraints are too tight (deadline 300), it can still be better to use a greedy approach because the payments can be very high due to the existence of critical agents who are important for obtaining a good solution quality in these cases.

Results of 0.975 reliability threshold *(Figure 3).* These instances are very tightly constrained, resulting in almost no solution for the 300 deadline cases. Our mechanism

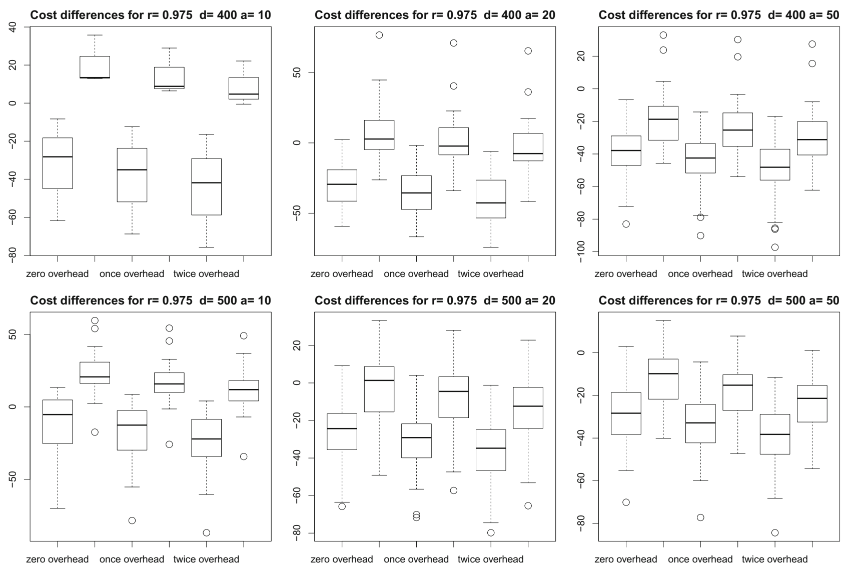

Fig. 3. The same plots as Figure 1 for the instances that require a reliability of 0.975. The plots for a deadline of 300 are not shown because very few instances resulted in a solution.

clearly outperforms the greedy approach. Only in the case of very few (10) agents, the payment required to make them bid truthfully can be very high because no good replacement exists. In all other very tightly constrained instances, it is nearly always beneficial to use our mechanism instead of the greedy approach that represents current practice, even with the added payments needed to make the agents truthful.

6 Conclusion

We model robust procurement as an optimization problem. We show that its decision version is NP-complete, and propose a backtracking algorithm with cuts that reduce the search-space to find the optimal solution. If the number of contractors required to make a sufficiently robust procurement is small (about 8 in our experiments), the algorithm typically finishes within minutes. We then develop a mechanism that determines a sequence of winners, each of which is invoked only if the previous agent fails upon execution, and every winner receives payments. The mechanism motivates agents to truthfully report their private information (costs and deadlines), maximizes the social welfare, and ensures non-negative utilities of the participants even after execution.

In the experiments, we compare our mechanism with an iterated greedy first-price mechanism that represents the current practice in public procurements. The results show that in terms of social welfare, our mechanism outperforms the greedy approach in all cases except when there exist cheap and reliable agents who can finish the job in time. In terms of payments, our mechanism outperforms the current practice when there are many potential contractors and constraints in the optimization problem are tight. The results are promising especially considering that in the experiments, the potential cost increase due to misreporting of the agents is disregarded in the greedy mechanism. Our mechanism offers a potential solution to the serious problem of additional costs and delays in public procurement problems due to uncertainty on execution and strategic misreporting by contractors.

In the public procurement problems, even if a contractor is trying its best to execute the given task, it can still fail to complete by the deadline due to external uncertainties. This leads to the difficulty of distinguishing between intentional failures and failures caused by external uncertainties. Due to this difficulty we do not address the issue of execution incentives (see [13,9]) in this paper. However, it is certainly important to investigate how other techniques such as verification and contracting can be combined with mechanism design to avoid intentional failures.

Acknowledgements. We thank Adriana Gabor for the insightful discussions.

References

1. Commission Regulation (EC) No 1177/2009, `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2009:314:0064:0065:EN:PDF`
2. PeopleSoft manufacturing scorecard, `http://www.oracle.com/us/products/applications/peoplesoft-enterprise/financial-management/064743.html`
3. Aanbesteding politiewapen moet over (January 23, 2012), `www.nos.nl` (Dutch), `http://nos.nl/artikel/333629-aanbesteding-politiewapen-moet-over.html`
4. Court decision ljn: Bv1636, rechtbank 's-gravenhage, 408727/kg za 11-1438 (January 24, 2012), `http://www.rechtspraak.nl` (Dutch)
5. Vertraging nieuw politiepistool 1 tot 1,5 jaar (January 24, 2012), `http://www.volkskrant.nl` (Dutch)
6. Chen, J., Huang, H., Kauffman, R.J.: A public procurement combinatorial auction mechanism with quality assignment. Decis. Support Syst. 51(3), 480–492 (2011)
7. Federgruen, A., Yang, N.: Optimal supply diversification under general supply risks. Oper. Res. 57(6), 1451–1468 (2009)
8. Flyvbjerg, B., Holm, M.K.S., Buhl, S.L.: Underestimating costs in public works projects: Error or lie? Journal of the American Planning Association 68(3), 279–295 (2002)
9. Mezzetti, C.: Mechanism design with interdependent valuations: Efficiency. Econometrica 72(5), 1617–1626 (2004)
10. Nisan, N., Ronen, A.: Algorithmic mechanism design. Games and Economic Behavior 35, 166–196 (2001)
11. Porter, R., Ronen, A., Shoham, Y., Tennenholtz, M.: Fault tolerant mechanism design. Artif. Intell. 172(15), 1783–1799 (2008)
12. Ramchurn, S.D., Huynh, D., Jennings, N.R.: Trust in multi-agent systems. Knowl. Eng. Rev. 19(1), 1–25 (2004)
13. Stein, S., Gerding, E.H., Rogers, A., Larson, K., Jennings, N.R.: Algorithms and mechanisms for procuring services with uncertain durations using redundancy. Artif. Intell. 175(14-15), 2021–2060 (2011)