GAME-THEORETIC MODELING OF MULTI-AGENT SYSTEMS: APPLICATIONS IN SYSTEMS ENGINEERING AND ACQUISITION PROCESSES

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Salar Safarkhani

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. Ilias Bilionis, Chair

School of Mechanical Engineering

Dr. Jitesh Panchal

School of Mechanical Engineering

Dr. Thanh Nguyen

Krannert School of Management

Dr. Karthik Kannan

Krannert School of Management

Approved by:

Dr. Nicole Key

Head of the Graduate Program

To the ones I love.

ACKNOWLEDGMENTS

First, I would like to thank my adviser, professor Ilias Bilionis, for his consistent support, guidance, and motivation during my Ph.D. I appreciate the opportunity he gave me to participate in this exciting interdisciplinary research. Ilias is smart, enthusiastic, knowledgeable, and funny. He was not only my adviser but also my friend who always thought of me and helped me to the most.

I also thank my Ph.D. committee members, professor Jitesh Panchal, professor Thanh Nguyen, and professor Karthik Kannan for their advice and help. I enjoyed and learned from all the research meetings that we had together.

I gratefully acknowledge the funding for my Ph.D., received from Purdue University and the National Science Foundation (NSF), under Grant NO. 1728165.

I also thank my friends for all the fun and good times we had together during my years at Purdue University.

Special thanks go to my family, my parents, and my siblings for their love, patience, and moral support for all these years. Last but not least, I am thankful to Lydia, for her encouragement and unconditional love.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Safarkhani, Salar Ph.D., Purdue University, August 2020. Game-Theoretic Modeling of Multi-Agent Systems: Applications in Systems Engineering and Acquisition Processes. Major Professor: Ilias Bilionis.

The process of acquiring the large-scale complex systems is usually characterized with cost and schedule overruns. To investigate the causes of this problem, we may view the acquisition of a complex system in several different time scales. At finer time scales, one may study different stages of the acquisition process from the intricate details of the entire systems engineering process to communication between design teams to how individual designers solve problems. At the largest time scale one may consider the acquisition process as series of actions which are, request for bids, bidding and auctioning, contracting, and finally building and deploying the system, without resolving the fine details that occur within each step. In this work, we study the acquisition processes in multiple scales. First, we develop a game-theoretic model for engineering of the systems in the building and deploying stage. We model the interactions among the systems and subsystem engineers as a principal-agent problem. We develop a one-shot shallow systems engineering process and obtain the optimum transfer functions that best incentivize the subsystem engineers to maximize the expected system-level utility. The core of the principal-agent model is the quality function which maps the effort of the agent to the performance (quality) of the system. Therefore, we build the stochastic quality function by modeling the design process as a sequential decision-making problem. Second, we develop and evaluate a model of the acquisition process that accounts for the strategic behavior of different parties. We cast our model in terms of government-funded projects and assume the following steps. First, the government publishes a request for bids. Then, private

firms offer their proposals in a bidding process and the winner bidder enters in a contract with the government. The contract describes the system requirements and the corresponding monetary transfers for meeting them. The winner firm devotes effort to deliver a system that fulfills the requirements. This can be assumed as a game that the government plays with the bidder firms. We study how different parameters in the acquisition procedure affect the bidders' behaviors and therefore, the utility of the government. Using reinforcement learning, we seek to learn the optimal policies of involved actors in this game. In particular, we study how the requirements, contract types such as cost-plus and incentive-based contracts, number of bidders, problem complexity, etc., affect the acquisition procedure. Furthermore, we study the bidding strategy of the private firms and how the contract types affect their strategic behavior.

1. INTRODUCTION

The acquisition process of large-scale complex systems is usually suffered from cost and schedule overruns. To study the causes of such problems, we may consider the system acquisition procedure in several different time scales. First, one may focus on the cognitive causes and investigate how a designer or any decision maker in the design process evaluates and builds a certain part of the system. This is helpful to recognize any pattern that certain type of individuals follow to make progress in the design process. In a larger scale, we may study the engineering of the systems from the management perspective. In this scale, one may investigate the effect of incentives, team communication, and systems engineering process on the successful building and deploying the system. Even in a larger scale, we may study the whole acquisition process from the very first to the very last stages. For instance, an abstract model of such a study includes, request for proposals and bids, auctioning, contracting, and investing and deploying the systems. In this work, we study the acquisition procedure of the systems from multiple perspectives.

First, let us focus on the engineering and management of the systems. Systems engineering process (SEP) coordinates the efforts of many individuals to design a complex system and it includes, requirements engineering, functional decomposition and allocation, and sub-contracting processes. These processes are well documented in systems engineering handbooks such as [1, 2]. The success of these processes rests on two implicit but fundamental assumptions: (a) incentive alignment, i.e., participants are only driven by the incentive structures set up by these processes (either requirements or value functions), and (b) information availability, i.e., information is freely available to those who need it. The implication of these assumptions is that participants truthfully reveal and utilize information towards the achievement of system-level objectives. However, phenomena such as biased information passing [3]

and strategic misrepresentation [4] clearly show that these assumptions are not valid. Instead, people exhibit strategic behavior where the information is incomplete and dispersed. These deviations potentially contribute to cost and schedule overruns that plague the majority of large systems engineering projects across multiple industries such as transportation [5], power [6], defense [7], and space [8]. Currently, there is a lack of theoretical foundations to help in understanding the effects of these deviations on the performance (expected outcomes) of SEP. Traditional systems engineering approaches are based on the principles of hierarchical decomposition, with the following assumptions (a): the system-level objective can be achieved by decomposing the higher-level requirements into sub-system and component-level requirements, and (b): subsystems designed to perform these lower-level requirements can be composed into higher-level systems. Requirements are used for coordination of activities both within an organization and for external contracting. However, Collopy [9] has shown that using the requirements within the SEP creates design trade conflicts among different subsystems, resulting in dead losses within the system, and an inferior outcome. To address the limitations of requirements-driven design, Collopy et al. [10] propose value-driven design (VDD) as a better alternative. VDD is a systems design approach that starts with the identification of a system-level value function and guides the systems engineer (SE) to construct subsystem value functions that are aligned with the system goals. Therefore, each subsystem objective results in a self-contained design problem that can be assigned to a team and achieved independently. According to VDD, the subsystem engineers (sSE) and contractors should maximize the objective functions passed down by the SE instead of trying to meet requirements. By guiding the subsystem and component designers, as opposed to hard requirements, VDD promises reduced cost growth and performance erosion. One of the challenges in the application of VDD within systems engineering is the identification of individual objective functions from the system-level objective function. Collopy [11] shows an approach to deriving component-level objectives from system-level objectives based on the assumption of linearity. The implicit assumption in this approach is that

individual teams would faithfully follow the objective functions assigned to them. However, this would only be true if the personal value functions of the individual stakeholders are aligned with the objectives assigned to them. Identification of the lower-level objective functions is significantly challenging when there is misalignment between the objective functions derived from the system-level and the stakeholders' actual objectives. Therefore, VDD make the latent assumption that the goals of the human agents involved in the SEP are aligned with the SE goals. However, this assumption ignores the possibility that the design agents, like all humans, may have personal agendas that not necessarily aligned with the system-level goals. The problem is further complicated because of information asymmetry. The SE does not have complete information about the capabilities and costs of the agents. This information is usually privately held by the sSE's who may behave strategically to maximize their objectives based on their private information. An example of such strategic behavior in systems engineering is strategic misrepresentation, which consists of purposeful low-balling of initial cost estimates to increase the likelihood of project approval [4]. Contrary to RE and VDD, it is more plausible that the design agents seek to maximize a personal expected utility. Indeed, there is experimental evidence that the quality of the outcome of a design task is strongly affected by the reward and effort cost anticipated by the agent [12]. In other words, the agent decides how much effort and resources to devote to a design task after taking into account the potential reward and personal dis-utilities. In the field, the reward could be explicitly implemented as an annual performance-based bonus, or, as it is the case most often, it could be implicitly encoded in expectations about job security, promotion, professional reputation, etc.

1.1 Game-theoretic Modeling of the Systems Engineering Process

Game theory is the field of mathematical modeling of strategic behavior among some logical decision-makers. Therefore, to capture the human aspect in SEP, one may follow a game-theoretic approach [13, 14]. Most generally, the SEP should be

modeled as a dynamical hierarchical network game with incomplete information. Each layer of the hierarchy represents interactions among the SE and some sSEs, or the sSEs and other engineers or contractors. We will model the interactions in this hierarchy as a principal-agent problem. With the term "principal", we refer to any individual delegating a task, while we reserve the term "agent" for the individual carrying out the task. Note that an agent may simultaneously be the principal in a set of interactions down the network. For example, the sSE is the agent when considering their interaction with the SE (the principal), but the principal when considering their interaction with a contractor (the agent). In Fig. 1.1, we show such a representation

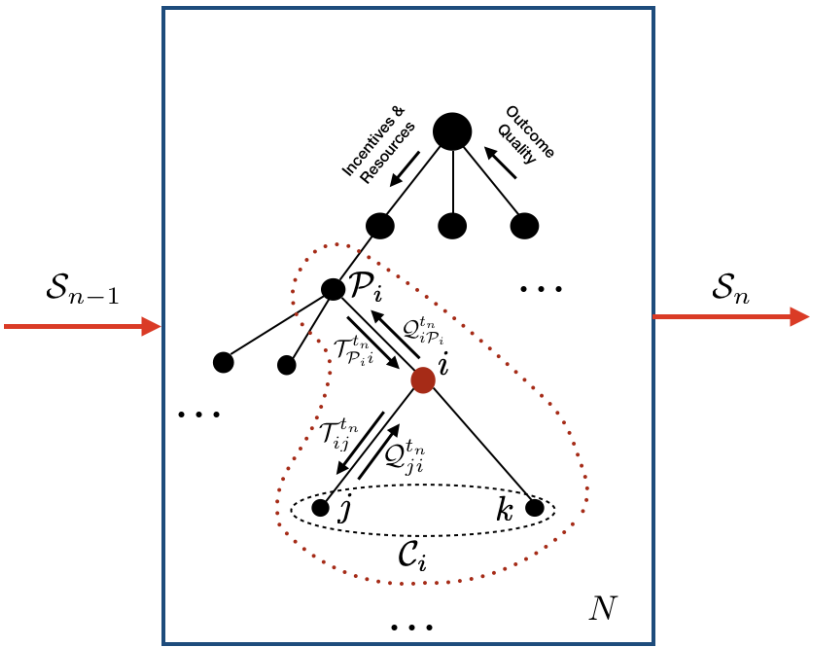


Fig. 1.1. The schematic representation of hierarchical systems engineering process using plate notation. $\mathcal{P}_i$ is the parent of node i and $\mathcal{C}_i$ denotes the children of the node i. $\mathcal{T}_{ij}^{t_n}$ is the incentives and resources that flow down from a principal (in this case i) to it's agents (in this case j) at time step t_n. $\mathcal{Q}_{ji}^{t_n}$ is the quality of the outcome of the effort of the agent (in this case j) that is delivered to the principal (in this case i), at time step t_n. $\mathcal{S}_{n-1}$ is the state at iteration $n-1$ where the information flows down to the iteration n. Similarly, the information from state $\mathcal{S}_n$ passes to the next iterations.

of the systems engineering hierarchy. Each node represents a systems or subsystem engineer in this hierarchy. Each layer of the hierarchy captures the interactions of a *principal*, e.g., the manager, the SE, or a sSE, with several *agents*, e.g., the SE, sSE's, component engineers, or contractors. The input and output of each time step such as t_n, are the information from states $\mathcal{S}_{n-1}$ and $\mathcal{S}_n$, respectively. Let us consider the node i in this hierarchy. We show the parent and children of the node i by $\mathcal{P}_i$ and $\mathcal{C}_i$, respectively. At each iteration, the game proceeds as follows. The principals delegate tasks to the agents alongside incentives which are performance-based contracts ($\mathcal{T}_{ij}^{t_n}$). The agents devote a certain amount of effort by maximizing their expected utility and deliver their task outcome back to the principal with a certain quality ($\mathcal{Q}_{ji}$). The principals may not be able to monitor the effort of the agents which causes moral hazard [15]. Moreover, the principal may not have complete information about the task-difficulty or problem-solving skills of the agents. The goal of the agents, is to maximize their expected utility given the incentives provided by the principal, and the principal selects the incentive structure that maximizes the expected utility of the system. Some practical examples of using expected utility in the decision making can be found in [16, 17]. In the economics literature, this is known as the *mechanism design* problem [18].

The problems of designing incentive structures under incomplete information are at the heart of the mechanism design theory [19]. Research on mechanism design, grounded in game theory, has resulted in the vast literature on auctions [20], contests [21], and contracts [22]. The theory of mechanism design initiated within economics, but has received significant recent attention in other fields such as computer science [23]. Agency theory [24], in particular, addresses this issue of incentive design as it pertains to systems engineering. It addresses the situation where there is a conflict between the desires of the principal and the agent, and it is impractical for the principal to observe the actions and the cost structure of the agents. Assuming that the involved engineers in different levels of systems engineering hierarchy are self-interested individuals, the SEP can be modeled as a principal-agent problem where

the systems engineer is the principal and the subsystem engineer is the agent. In this work we demonstrate how the generalized principal-agent model can be adapted to model the situation of a SE delegating work to sSE's as a one-shot game. The SEP is "one-shot" in the sense that decisions are made in *one* iteration and they are *final*. We derive the incentive-based contracts that the principal must offer to the agents in order to maximize the system utility. We assume that the systems engineer maximizes the expected utility of the system, while the subsystem engineers seek to maximize their expected utilities. Furthermore, the systems engineer is unable to monitor the effort of the subsystem engineers and may not have complete information about their types or the complexity of the design task. However, the systems engineer can incentivize the subsystem engineers by proposing specific contracts. To obtain an optimal incentive, we pose and solve numerically a bi-level optimization problem.

One key component of such a principal-agent model, is the quality function that accounts for the principal's beliefs about the task outcome. We define the quality function as the stochastic map between an agent's effort and the quality of the product they deliver. We will study how the principal's belief about the task-difficulty and problem-solving skills of the agents can induce a mathematical model for the quality function. To this end, we build a stochastic model of the assigned task, and then we develop a reduced order representation to use in a game-theoretic model.

1.2 Game-theoretic Modeling of the System Acquisition Process

In an other prospective, we model the whole acquisition process of the large complex systems. We develop and evaluate a model of the acquisition process that accounts for the strategic behavior of different parties. Specifically, we cast our model in terms of government-funded projects and assume the following steps. First, the government publishes a request for bids. Then, private firms offer their proposals in a bidding process and the winner bidder enters in a contract with the government. The contract describes the system requirements and the corresponding monetary trans-

fers for meeting them. The winner company devotes effort to deliver a system that fulfills the requirements. This can be assumed as a game that the government plays with the bidder companies. The objective is to study how different parameters in the acquisition procedure affect the agents' behaviors and therefore the quality and cost of the deliverable system. Using reinforcement learning [25], we seek to learn the optimal policies of involved actors in this game. We model each party as a reinforcement learning agent that participates in the acquisition process. The goal of these agents is to learn the optimal actions in each step that maximize their expected return. These reinforcement learning agents play the acquisition process game and they learn how to bid, and how much effort to devote on building and deploying the systems. Therefore, we discover how the requirements, contract types such as cost-plus and incentive-based contracts, number of bidders, problem complexity, etc., affect the system acquisition process. This analysis can potentially inform the government about the possibility of cost overruns later in the process of building a complex system. Furthermore, we study the bidding strategy of the private firms and how the contact types affect their optimal behavior.

The outline of the dissertation is as follows. In chapter 2, we develop the quality function to understand the effect of task-difficulty and problem solving skills on the design performance of the agents. In chapter 3, we develop a principal-agent model of the one-shot shallow SEP and we derive the incentive-based contracts under incomplete information. In chapter 4, we use deep reinforcement learning to model the acquisition process game. We investigate the optimal policy of the bidder companies under different scenarios.

2. QUALITY FUNCTION

2.1 Introduction

Systems engineering processes (SEP) require the coordination of a large number of individuals, e.g., managers, systems engineers (SE), subsystem engineers (sSE), and contractors, to establish the design, deployment, operation, and retirement of complex systems [1, 26]. Naturally, these individuals are self-interested, i.e., they have their personal agendas which are not necessarily aligned with the system-level objectives. It has been postulated that this discrepancy between organizational and personal goals may be one of the leading factors behind the increasing cost overruns and delays in modern systems engineering [27]. However, to date, there is no comprehensive SEP theory which models the effects of human behavior.

To account for human behavior, we may model SEPs within a game-theoretic framework [13, 28–30]. In particular, a SEP can be viewed as a dynamic hierarchical network game. Each layer of the hierarchy captures the interactions of a *principal*, e.g., the manager, the SE, or a sSE, with several *agents*, e.g., the SE, sSEs, component engineers, or contractors. At each iteration, the principal assigns tasks to the agents encoded via performance-based contracts. The agents select how much effort to devote to their tasks by maximizing their expected utility, a common assumption known as individual rationality [18]. Deviations from individual rationality can be modeled using elements from behavioral economics [31], e.g., by replacing maximum utility theory with cumulative prospect theory [32]. Finally, the agents report the product of their efforts back to the principal. Note that in this hierarchy, the agent of a layer may be the principal of a subsequent layer. The goal of the principal is to select the contracts that maximize the expected system-level utility. In the economics literature, this is known as the *mechanism design* problem [18].

A critical component of such a principal-agent model is the *quality function*, defined as the stochastic map between an agent's effort and the quality of the product they deliver. To be more precise, the quality function models the beliefs of the principal about the effort-to-quality map, i.e., it models what the principal *thinks* they can get if the agent decides to spend a given amount of effort. Note that, quality is a system functionality measure and, therefore, it is a context-dependent variable. For instance, the quality can be defined as the Mach number at the exit of a nuzzle, the power of the propulsion system in satellite design, etc. For analytical convenience, the quality function is usually taken to be a linear function of effort with some additive Gaussian noise [33]. This simplistic assumption is not sufficient for capturing the beliefs about the outcome of an assigned task within the context of a SEP. Focusing on the early design stages of a SEP, the outcome of design tasks depends predominately on beliefs about the difficulty of the underlying problem and the problem-solving skills of the engaged agent.

The objective of this chapter is to mathematically model the dependence of the quality function on a principal's beliefs about the task-difficulty and the problem-solving skills of an agent, within the context of the early design stages of a SEP. In other words, the objective is to demonstrate how the principal's beliefs about task-difficulty and the problem-solving skills of the agent induce a specific mathematical form for the quality function. We achieve this in two steps: (1) constructing a generative stochastic model of the delegated task, and (2) developing a reduced order representation suitable for use in an extensive game-theoretic framework. The generative model is essentially a random process labeled by effort. Each sample from this random process is a plausible effort-to-quality map. The reduced-order model is a mathematically convenient approximation of this random process constructed from multiple samples of effort-to-quality maps.

The details of the generative model are as follows. The design task assigned to an agent is modeled as a scalar function maximization problem. An agent's effort is measured in function evaluations used to solve this maximization problem. We

assume that the principal encodes their beliefs about the difficulty of the problem using a Gaussian process (GP) prior [34] over the space of possible scalar functions, e.g., by selecting a suitable mean and covariance function. We identify two agent types: "naïve" and "skillful", based on their problem-solving skill, i.e., how they use the output of each function evaluation to update their belief (state-of-knowledge) about the function being maximized. A naïve agent solves the maximization problem using random search, i.e., the agent does not use the information obtained in the previous function evaluation. A skillful agent solves the maximization problem using Bayesian global optimization [35–37]. That is, the naïve agent does not learn from past experience whereas the skillful agent does learn in an optimal way. We selected these two extreme agent types because they correspond to the worst and the best expected agent performance and, thus, they yield a lower and upper bound to the quality function of intermediate types. The naïve approach has an alternative interpretation as a parallel search representative of a scenario where there is a team of engineers, all developing different ideas concurrently, and the team gets together and decides on the best solution at the end of the process. Note that there is extensive literature on the differences between novices and experts, particularly in terms of the amount of knowledge, how their domain knowledge is structured, their problem-solving strategies, and how these differences affect the outcomes of problem-solving and decision-making tasks [38–42]. The problem-solving skill considered here, is one of the many important factors that play a role in an agent's expertise. Other factors such as the amount and the structure of the domain-specific knowledge are out of the scope of this chapter.

To obtain a plausible realization of the effort-to-quality map, we sample a scalar function from the GP prior and we simulate the behavior of the agent. Using extensive sampling datasets, we constructed a reduced order model by employing the Karhunen-Loève theorem [43].

The outline of this chapter is as follows. In Sec. 2.2, we present our methodology starting with the definition of the quality function. In Sec. 2.2.1, we model a

design task as a function maximization problem. In Sec. 2.2.2, we model the state of knowledge of the principal about the function that the agent is maximizing and we define the concept of task-difficulty. Sec. 2.2.3 discusses the modeling of the problem-solving skills of an agent. The reduced order model is presented in Sec. 2.2.4. In Sec. 4.3, we present our numerical experiments and study the effect of task-difficulty and problem-solving skills on the quality function. We conclude in Sec. 2.4.

2.2 Methodology

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, where Ω is a sample space, $\mathcal{F}$ is a σ-algebra of subsets of Ω, and $\mathbb{P}$ is a probability measure. Elements of Ω are denoted by $\omega \in \Omega$. An $\omega \in \Omega$ corresponds to a specific realization of everything that is random in our models.

Consider the early design phase of a SEP and focus on an agent that resides at the leaves of the hierarchy, i.e., of a component engineer or a subcontractor. The behavior of these leaf agents depends only on inputs from the immediately higher level of the hierarchy and, thus, it is a natural starting point. The behavior of agents that lie at intermediate points of the SEP hierarchy should be defined recursively given inputs from even higher levels and assuming that subordinate agents act optimally. Such recursive constructions are the subject of ongoing research and will not be discussed here. We will construct a stochastic model of the quality function of this leaf agent based on some generic beliefs about the difficulty of the task that is assigned to them as well as their problem-solving skills. We assume that there is only one task that is assigned to the agent and, therefore, the agent does not need to deal with prioritizing over multiple tasks. To this end, let $E = \{0, 1, 2, \dots\}$ be the space of possible effort levels that the agent may choose, denoting a specific effort level by $e \in E$. Similarly, let $\Theta = \{\text{“naïve”}, \text{“skillful”}\}$ be the set of possible agent types, denoting a specific type by $\theta \in \Theta$. For simplicity, let us measure the quality characterizing the outcome

of a design task with a single real number, e.g., the utility/value of that design to the principal allocating the task.

Using these definitions, the *quality function* can be thought as a random process

$$Q : E \times \Theta \times \Omega \to \mathbb{R},$$

i.e., as a collection of Borel measurable functions $\{Q(e, \theta, \cdot)\}_{e \in E, \theta \in \Theta}$. Of course, not all such random processes yield reasonable quality functions. Any explicit model we construct should satisfy a minimum set of mathematical properties:

1. $Q(e, \theta, \cdot)$ must have increasing paths in e, i.e., for all $e_1, e_2 \in E$ with $e_1 < e_2$, we must have:

   $$Q(e_1, \theta, \omega) \leq Q(e_2, \theta, \omega),$$

   for all $\theta \in \Theta$, and for almost all $\omega \in \Omega$. This ensures that the more effort an agent spends the better the quality of the product of the task. However, note that this property of the quality function is only justified when the design task is simple, i.e., when the agent designs a simple system component or a simple subsystem. The property does not necessarily generalize to the system level. To the contrary, there is empirical evidence that increased effort may lead to increased (sub)system complexity and, as a result, to decreased (sub)system reliability [44]. One may argue that the monotonicity of the quality function could be retained even if (i) the effort increments are sufficiently large to allow for proper system verification, and; (ii) the system design is required to revert to a previous state if additional design efforts lead to decreased quality. Of course, such a formal correction does not transfer to real systems as one cannot verify the specifications with absolute certainty or enforce the reversion policy.

2. $Q(e, \theta, \omega)$ must be bounded above in probability, i.e., for all $\epsilon > 0$, there exists an $M > 0$ such that

   $$\mathbb{P}\left[\{\omega : Q(e, \theta, \omega) > M\}\right] < \epsilon,$$

for all $e \in E, \theta \in \Theta$. This ensures that the outcome quality cannot grow without bound no matter how much the effort of the agent is, e.g., because of physical limitations in the design.

We will now construct a generative model of $Q(e, \theta, \omega)$ that satisfies these properties and, furthermore, it makes explicit the dependence on task-difficulty and agent skills.

2.2.1 Modeling the design task as a scalar function maximization

Let us assume that the agent's task is to maximize a scalar function over a set of candidate designs. For clarity, let the set of candidate designs be $X = [0, 1]$ (the ideas can easily be generalized to arbitrary sets.) The principal does not know exactly what the agent's objective function is. Nevertheless, let us assume that the principal believes that the scalar function the agent is tasked with maximizing is a sample from a random process $A : X \times \Omega \to \mathbb{R}$. Let us refer to $A(x, \omega)$ as the *attribute function*. Thus, the principal believes that the agent is solving:

$$\max_{x \in X} A(x, \omega), \tag{2.1}$$

but they do not know exactly what $A(x, \omega)$ is.

2.2.2 Modeling the beliefs of the principal about the attribute function

To proceed, let us assume that the principal models the attribute function $A(x, \omega)$ as a GP, i.e.,

$$A \sim (m, k)\,, \tag{2.2}$$

where $m : X \to \mathbb{R}$ and $k : X \times X \to \mathbb{R}$ are the mean and the covariance function, respectively. The choices of GP priors is motivated by their successful application to human function learning by Griffiths et al. [35]. The beliefs of the principal about the plausible $A(x, \omega)$ are encoded in their choice of mean and covariance functions. Of course, any particular choice is context-dependent and the principal should make

every effort to use any available data to estimate them. However, to advance our study, let us assume that the principal's beliefs are reflected by the choice:

$$\begin{aligned} m(x) &= c \\ k(x,x') &= \sigma_s^2 \exp\left\{-\tfrac{(x-x')^2}{2l^2}\right\}, \end{aligned} \tag{2.3}$$

with signal strength σ_s and length-scale l. A constant mean encodes the principal's lack of knowledge about any particular trend of the attribute function. The regularity of the covariance function determines the regularity of sampled attribute functions, see [45]. Here, our choice of the *squared exponential* covariance function guarantees that the sampled attribute functions are infinitely differentiable with respect to x. The choice of the signal strength controls the principal's beliefs about the possible variations of the attribute function about its mean. Without loss of generality, we may set $c = 0$ and $\sigma_s = 1$, after a suitable affine transformation of the attribute function. Therefore, the only remaining parameter is the length-scale l. The length-scale of the covariance function is a measure of the task-difficulty. On one hand, decreasing the length-scale, the fluctuations of the attribute function increase, making the task of the agent more difficult. On the other hand, increasing the length-scales yields smoother attribute functions thus making the underlying task easier. Of course, there are other aspects of task-difficulty such as the number of dimensions, possible discontinuities, discrete choices, etc. Those are not considered in this work.

2.2.3 Modeling the problem-solving skills of the agent

The agent solves the problem of Eq. (2.1) by repeatedly evaluating the attribute function at design points of their choice. Each function evaluation counts as one unit of effort. Let $X_i : \Theta \times \Omega \to X$ be the random variables corresponding to the agent's query of the attribute function at effort level $i = 1, 2, \ldots$, and $A_i : \Theta \times \Omega \to \mathbb{R}$ be the corresponding attributes they observe, i.e.,

$$A_i(\theta, \omega) := A(X_i(\theta, \omega), \omega), \tag{2.4}$$

for $\theta \in \Theta$.

The random variables X_i are not necessarily independent since at effort level $i+1$, the agent may use all observations $(X_1, A_1), \ldots, (X_i, A_i)$ before they decide on X_{i+1}. Mathematically, this statement implies that the random variable X_{i+1} must be measurable with respect to the σ-algebra $\mathcal{F}_i$ generated by $(X_1, A_1), \ldots, (X_i, A_i)$ [46]. In other words, the random process X_i must be *adapted* to the *filtration* $\{\mathcal{F}_i\}_{i \in E}$. The exact nature of this process depends on the beliefs of the principal about the skills of the agent, see Secs. 2.2.3 and 2.2.3 for specific choices corresponding to a naïve and a skillful agent.

In any case, we are now in the position to define mathematically the quality function $Q(e, \theta, \omega)$. It is:

$$Q(e, \theta, \omega) = \max_{1 \leq i \leq e} A_i(\theta, \omega). \tag{2.5}$$

Note that this definition does satisfy the two requirements for the quality function that we posed at the beginning of Sec. 2.2, namely that $Q(e, \theta, \omega)$ is an increasing function of e and that it is bounded above in probability. Furthermore, we are here operating under the assumption that the agent returns the best attribute they have found, i.e., that they are *honest*. Dishonest behavior, e.g., putting a design in the back-pocket for later use, is not modeled in this work.

Note that the goal is not to build a descriptive (predictive) model of problem-solving behavior of real designers. Rather, we are investigating the mathematical implications that different information acquisition strategies (problem-solving skills) have in the form of quality function. However, assuming that real designers do maximize a well-defined mathematical function, we can conclude that the quality function corresponding to a naïve/skillful agent provides a lower/upper bound to the quality that should be expected by a real person lacking any domain-specific knowledge. While the specific quality functions are not applicable when the agents have domain-specific knowledge, the methodology can still be used by modeling the agent's knowledge as a prior belief on the mapping between the candidate designs and the attribute function, which can be used as an input to the GP. That scenario is not considered here, because such a mapping depends on the specific domain under consideration. In

all other cases, the model is only a crude approximation of real agent behavior which may, nevertheless, be a good starting point for posing and solving the mechanism design problem.

A naïve agent

The case $\theta =$ "naïve" corresponds to an agent that ignores past experience and simply chooses function evaluations at random. Mathematically,

$$X_i(\theta = \text{“naïve”}) \sim \mathcal{U}(X), \tag{2.6}$$

for all $i = 1, 2, \ldots$, where $\mathcal{U}(X)$ is the uniform distribution over the space of feasible designs X.

A skillful agent

The case $\theta =$ "skillful" corresponds to an agent that learns from past experience and queries the function trying to exploit what they have learned. The problem of how individuals acquire new knowledge is known as *human function learning*. Griffiths et al. [35] model human function learning using a GP. Here we follow their approach. In particular, we assume that the agent's prior knowledge about the attribute function $A(x, \omega)$ is captured by the GP prior of Eq. (2.2). In economic terms, we assume that $A(x, \omega)$ is *common knowledge* for the principal and the agent. It is also possible to model the case in which the agent has *private knowledge*, but this is beyond the scope of this work.

Now, let $i \in E$ and assume that the agent has already selected i designs,

$$X_{1:i} = (X_1, \ldots, X_i), \tag{2.7}$$

and they have observed the corresponding i attributes:

$$A_{1:i} = (A_1, \ldots, A_i). \tag{2.8}$$

We assume that the agent updates their state of knowledge about $A(x,\omega)$ by using Bayes rule to condition the prior GP of Eq. (2.2) on the observed data $X_{1:i}$ and $A_{1:i}$. The result is the *posterior GP*:

$$A|X_{1:i}, A_{1:i} \sim (m_i, k_i)\,, \tag{2.9}$$

where the posterior mean and covariance functions are given by:

$$m_i(x) = m(x) + k(x, X_{1:i})k^{-1}(X_{1:i}, X_{1:i})\left(A_{1:i} - m(X_{1:i})\right), \tag{2.10}$$

and

$$k_i(x,x') = k(x,x') - k(x, X_{1:i})k^{-1}(X_{1:i}, X_{1:i})k(X_{1:i}, x'), \tag{2.11}$$

respectively (see Ch. 2 of [34]). In these formulas, we have extended the definition of the mean and the covariance functions so that for any $X^1_{1:i^1}$ and $X^2_{1:i^2}$, $m(X^1_{1:i^1}) = \left(m\left(X^1_1\right), \ldots, m\left(X^1_{i^1}\right)\right)$, and $k\left(X^1_{1:i^1}, X^2_{1:i^2}\right)$ is the $\mathbb{R}^{i^1 \times i^2}$ matrix with (s,t)-element $k(X^1_s, X^2_t)$.

Following the experimental results in [36] and [47], we assume that a "skillful" agent selects X_{i+1} by maximizing the expected improvement in the attribute function. Suppose that the agent made a hypothetical query at $x \in X$ and they observed the attribute value $a \in \mathbb{R}$. The *improvement* they would have gotten over the observed attributes $A_{1:i}$ is

$$I_i(x,a) = \max\left\{0, a - \max_{1\le j\le i} A_j\right\}. \tag{2.12}$$

The *expected improvement* is obtained by taking the expectation of $I_i(x,a)$ over the agent's beliefs about a as captured by the posterior GP of Eq. (2.9), i.e.,

$$\mathrm{EI}_i(x) = \int I_i(x,a)\mathcal{N}(a|m_i(x), \sigma_i^2(x))da, \tag{2.13}$$

where $\sigma_i^2(x) = k_i(x,x)$, and $N(\cdot|\mu,\sigma^2)$ is the probability density function of a Gaussian random variable with mean μ and variance σ^2. It is actually possible to carry out the integration analytically [48] yielding:

$$\mathrm{EI}_i(x) = \left(m_i\left(x\right) - \max_{1\le j\le i} A_j\right)\Phi\left(Z_i(x)\right) + \sigma_i\left(x\right)\phi\left(Z_i(x)\right), \tag{2.14}$$

where

$$Z_i(x) = \frac{m_i(x) - \max_{1\leq j\leq i} A_j}{\sigma_i(x)},$$

and ϕ and Φ are the probability density function (PDF) and the cumulative distribution function (CDF) of standard normal, respectively. Therefore, the information acquisition strategy followed by the agent is assumed to be:

$$X_{i+1}(\theta = \text{“skillful”}) = \arg\max_{x\in X} \mathrm{EI}_i(x). \tag{2.15}$$

One should notice that the skillfulness and task-difficulty, are not uncorrelated, e.g., a very difficult task for a junior engineer with a limited level of experience may seem an easy task for a senior engineer. Therefore, the following two interpretations are equivalent: 1) For a given skillful agent, adjust the lengthscale of the covariance function to represent the proper task-difficulty that suits the agent's skill; 2) For a given task, adjust the lengthscale of the covariance function to represent the skillfulness of the agent that suit the task. Here, we use the first interpretation.

2.2.4 Constructing a reduced order model

The quality function random process $Q(e, \theta, \omega)$ is not analytically available. Unfortunately, this makes its use in a game-theoretic context, e.g., for the study of Nash equilibria and optimal mechanisms of SEPs, extremely difficult. To remedy the situation, we propose to use samples from $Q(e, \theta, \omega)$ to construct a computationally efficient reduced order model. We outline this proposal below.

Let us consider a particular type of agent, $\theta \in \Theta$. According to the Karhunen-Loève theorem [43], if $Q(e, \theta, \omega)$ is square integrable for all $e \in E$, i.e., if $\mathbb{E}\left[Q^2(e, \theta, \omega)\right] < \infty$, then it admits the following representation:

$$Q(e, \theta, \omega) = Q^0(e, \theta) + \sum_{k=1}^{\infty} \sqrt{\lambda_k(\theta)}\xi_k(\omega)\phi_k(e, \theta), \tag{2.16}$$

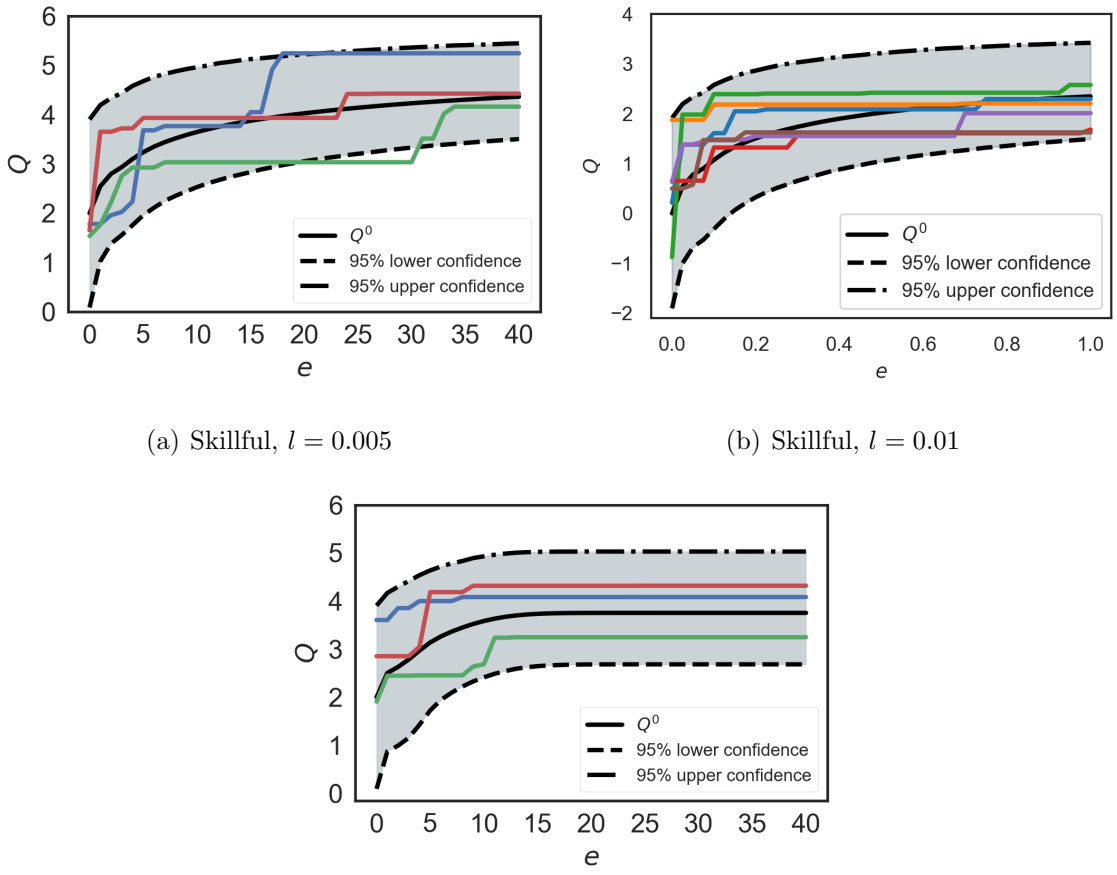
where $Q^0(e, \theta)$ is the mean of the random field, $\lambda_k(\theta)$ and $\phi_k(e; \theta)$ are the eigenvalues and eigenvectors of its covariance function, respectively, and the ξ_k are uncorrelated zero mean and unit variance random variables.

The idea is to estimate all these quantities involved in Eq. (2.16) samples of the stochastic process $Q(e, \theta, \omega)$. This is achieved through the following algorithm: (i) sample a plausible attribute function from the GP specifying the beliefs of the principal (see Sec. 2.2); (ii) using the sampled attribute function as the underlying truth, simulate the behavior of an agent attempting to maximize it (see Sec. 2.3); and (iii) return the resulting realization of the effort vs quality function (see Eq. (5)). To get a practical model, we truncate Eq. (2.16) at M terms so that we capture at least 90% of the spectral energy of the random field. Since the effort levels are discrete, KLE is equivalent to principal component analysis (PCA), which we carry out using the Python package scikit-learn [49]. We approximate the probability density function of each ξ_k, i.e., pdf (ξ_k), using Gaussian mixture model [50].

2.3 Numerical Results

We run exhaustive numerical simulations to understand the quality function dependence on task-difficulty and agent skill. In all simulations, the GP prior, which represents the principal's state of knowledge about the attribute function (Sec. 2.2.2), has a mean function $m(x) = 2$ and a signal strength $\sigma_s = 1$. We consider four levels of different task-difficulty as captured by the covariance length-scale choices $l = 0.005, 0.01, 0.05$, and 0.1, along with the two agent skill levels (naïve and skillful). That is, the total number of cases we simulate is 4 (complexity levels) $\times$ 2 (skill levels) = 8 (cases). For each case, we take $50,000$ Monte Carlo (MC) samples from the corresponding random field $\{Q(e, \theta, \omega)\}_{1 \leq e \leq 40}$. Using these $50,000$ samples, we construct the reduced order model of Sec. 2.2.4 which we proceed to compare systematically with them.

In Figs 2.1 and 2.2, we depict three random quality function samples (solid colored lines), the mean of all $50,000$ sampled quality functions (solid black line), and the 95% confidence levels (gray shaded area between the black dashed and dashed-dotted lines) for $\theta =$ skillful and $\theta =$ naïve, respectively. The Figs. 2.1(a), 2.1(b), and 2.1(c),



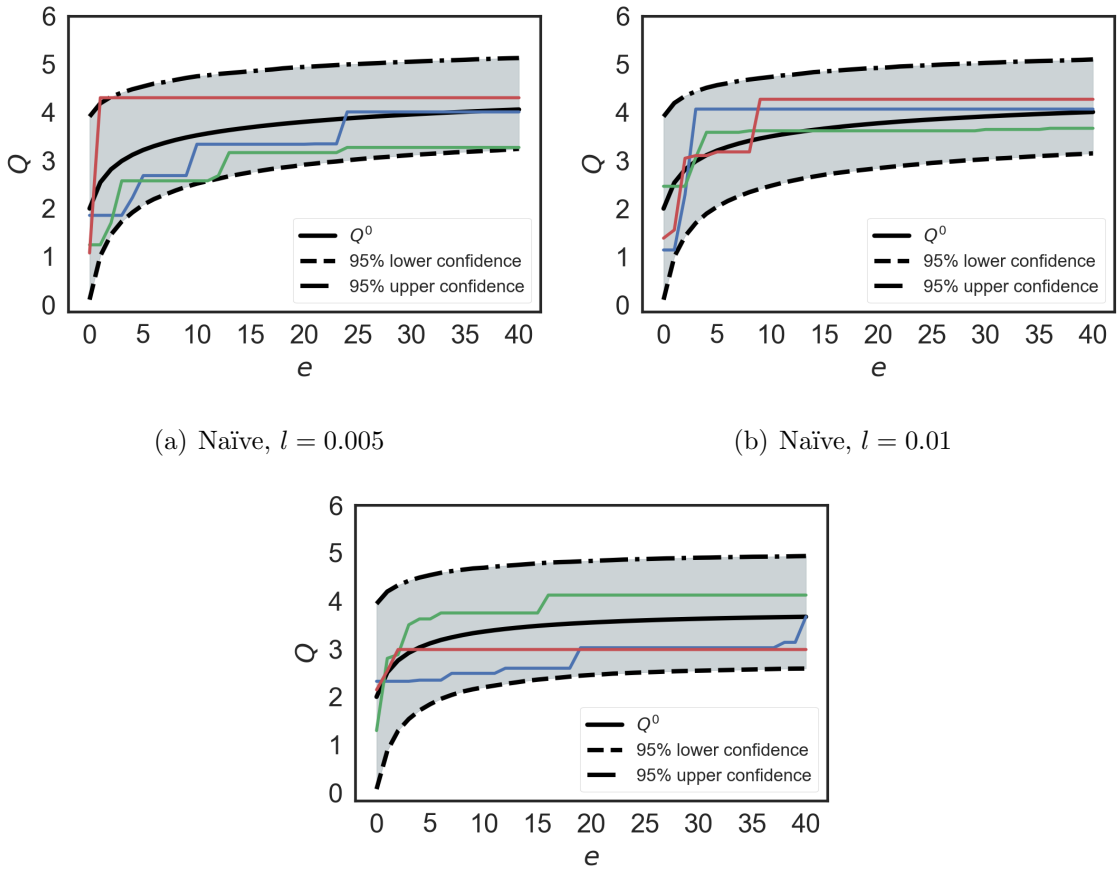
(a) Skillful, $l = 0.005$

(b) Skillful, $l = 0.01$

(c) Skillful, $l = 0.05$

Fig. 2.1. Three random quality function samples (solid colored lines), the mean of all $50,000$ sampled quality functions (solid black line), and the 95% confidence levels (gray shaded area between the black dashed and dashed-dotted lines) for $\theta =$ skillful.

and Figs 2.2(a), 2.2(b), and 2.2(c) are associated with decreasing task-difficulty since they correspond to length-scale choices of $l = 0.005, 0.01$, and 0.05, respectively. We observe the following. First, all samples across every case are increasing piecewise constant and bounded from above functions of e. Second, the mean quality function $(Q^0(e, \theta))$ is increasing and concave showing a clear dependence on task-difficulty and agent skill which we study in the next paragraph. Third, the uncertainty is greater



(a) Naïve, $l = 0.005$

(b) Naïve, $l = 0.01$

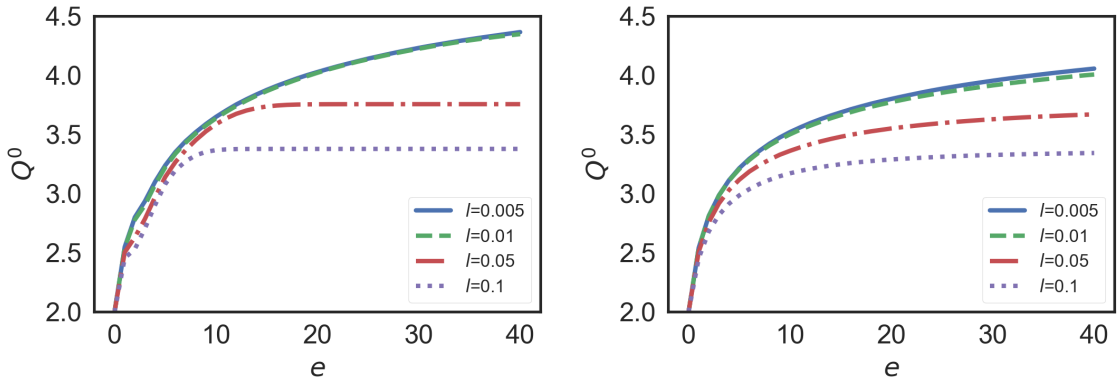
(c) Naïve, $l = 0.05$

Fig. 2.2. Three random quality function samples (solid colored lines), the mean of all 50,000 sampled quality functions (solid black line), and the 95% confidence levels (gray shaded area between the black dashed and dashed-dotted lines) for $\theta =$ naïve.

for small efforts, decreases mildly as the effort level increases, albeit it seems to have a definite non-zero lower bound. The latter is also discussed below.

Fig. 2.3 depicts the mean quality function ($Q^0(e, \theta)$) for all 8 cases. First note that the upper bound to the $Q^0(e, \theta)$ increases with decreasing length-scale. This phenomenon is related to the distribution of the maximum of Gaussian random fields on compact domains [51]. Namely, the expectation of the maximum of a Gaussian random field on a compact domain increases as the length-scale decreases. To under-

stand this phenomenon intuitively, take into account that the samples from GP priors with smaller length-scales have more opportunity to wiggle around the compact domain and reach extreme values. Therefore, comparing the absolute values of $Q^0(e,\theta)$ across different length-scales is nonsensical. What is comparable across complexities is the amount of effort required to exceed a certain percentage of the maximum quality, e.g., the first effort level $e^* = e^*(\epsilon)$ for which $Q^0(e^*(\epsilon),\theta)/\sup_{e\in E} Q^0(e,\theta) > 1-\epsilon$ to become flat for some $\epsilon > 0$. Naturally, in Fig. 2.3 we observe that the maximum is reached later as task-difficulty is increased. However, the mean quality function is directly comparable across agent skill levels. Comparing Fig. 2.3 (a) with (b), we observe that naïve agents require more effort to reach the same quality for the same level of task-difficulty.



(a) Skillful (b) Naïve

Fig. 2.3. The effect of task-difficulty on the mean of quality function ($Q^0(e,\theta)$) for skillful and naïve agents.

Now, we focus on the reduced order model of Sec. 2.2.4. To construct it, we perform PCA on 80% of the MC samples. The remaining 20% of the MC samples are used for validation. We present our results for length-scales $l = 0.01$ and 0.05, which need 5 and 3 terms in the KLE to capture the 90% of spectral energy, respectively.

In Figs 2.4 and 2.5, we show the PDFs of retained ξ_k's for skillful and naïve agents, respectively. In all cases, the first KLE component ξ_1 is almost a perfect standard normal. However, for the higher order terms, we start observing distinct non-Gaussian features. Also, the PDFs of the first three ξ_k's, for both length-scales and skill levels, are identical. It is only after the 4-th KLE component that we start observing differences in the PDFs.

In Fig. 2.6, we show the eigenvalues (λ) and eigenfunctions (ϕ) of the reduced order model for the two length-scales. As expected, the eigenvalues decay faster for decreasing task-difficulty. However, we do not observe any significant differences across skills. The eigenfunctions (especially at lower orders) seem almost independent of skill, but the higher order ones do exhibit a small variation as task-difficulty changes. We outline and interpret intuitively these findings below.

The first eigenfunction for both agent types and all levels of task-difficulty is almost constant. That is, the first eigenfunction just adds a constant to the mean quality function. Therefore, the first eigenfunction captures the uncertainty in the maximum of the underlying attribute function. Furthermore, taking into account that the PDF of ξ_1 is almost a perfect standard normal, we see that the assumption of additive Gaussian noise is valid to first order.

The higher order eigenfunctions are non-constant. However, note that they have a bump at small efforts, but they converge to zero as the level of effort increases. This bump is an indication that these eigenfunctions capture uncertainties associated with the agent's search process. Furthermore, the higher the order of the eigenfunction, the more effort is needed for the bump to appear. That is the high order eigenfunctions capture uncertainties in later stages of the search process. Consistent with this intuitive interpretation, notice that the eigenfunction bumps move to the left as task-difficulty decreases. This is a reflection of the fact that in less difficult tasks, critical discoveries occur earlier.

Finally, in Fig. 2.7, we compare the distribution of the $Q\left(e, \theta, \omega\right)$ at effort levels $e =$ 20 and 40 of reduced order model with those of the 20% MC samples that we set aside.

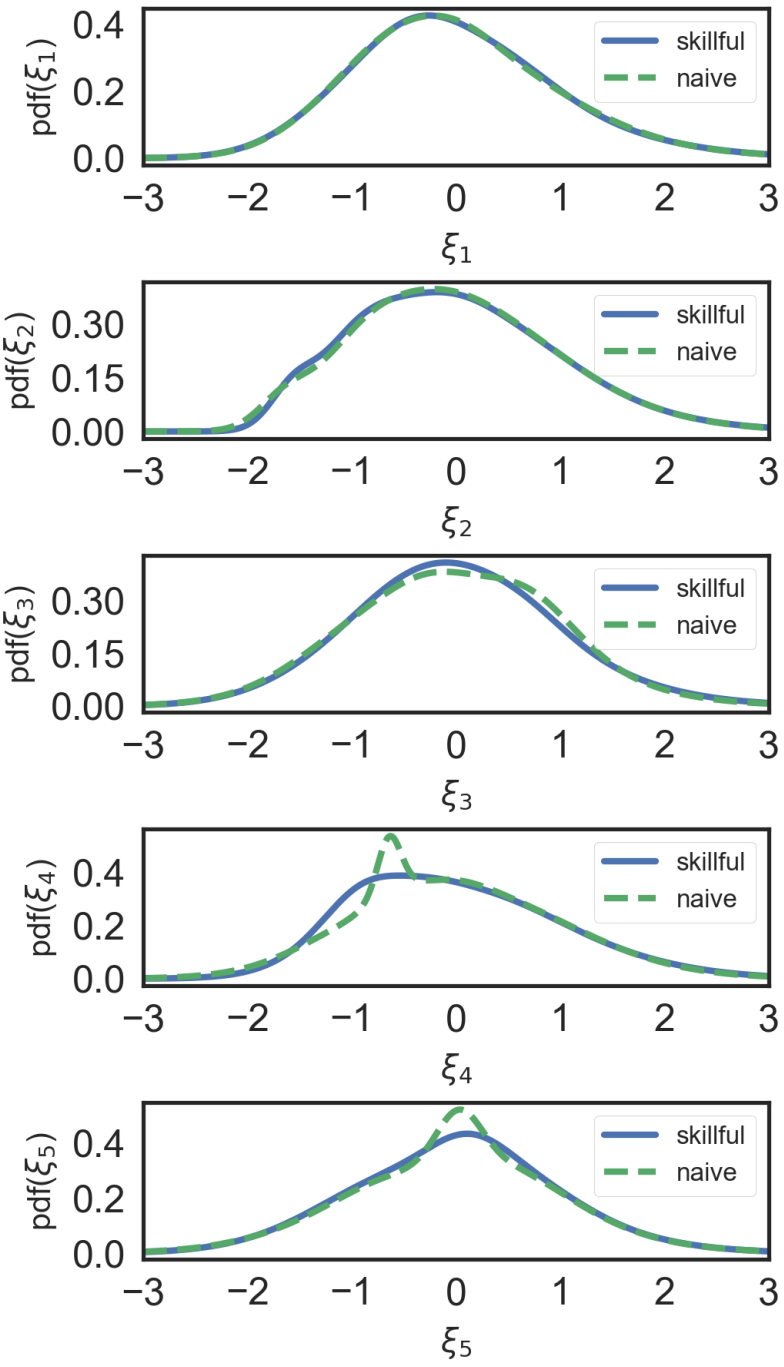


Fig. 2.4. The PDF of random variables $\xi_1, \cdots, \xi_5$ of reduced order model for skillful and naïve agents with $l = 0.01$.

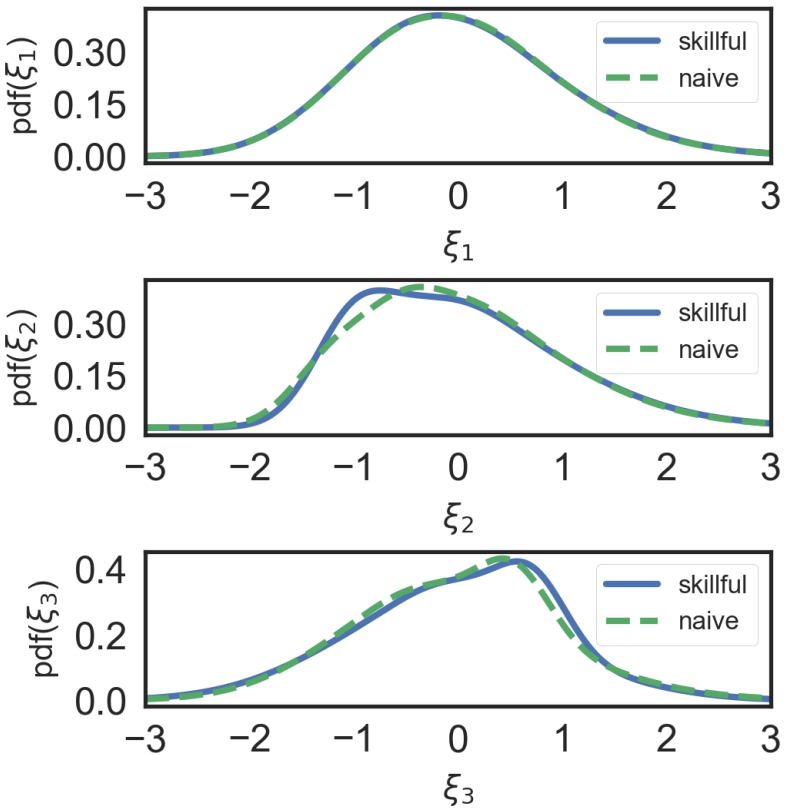
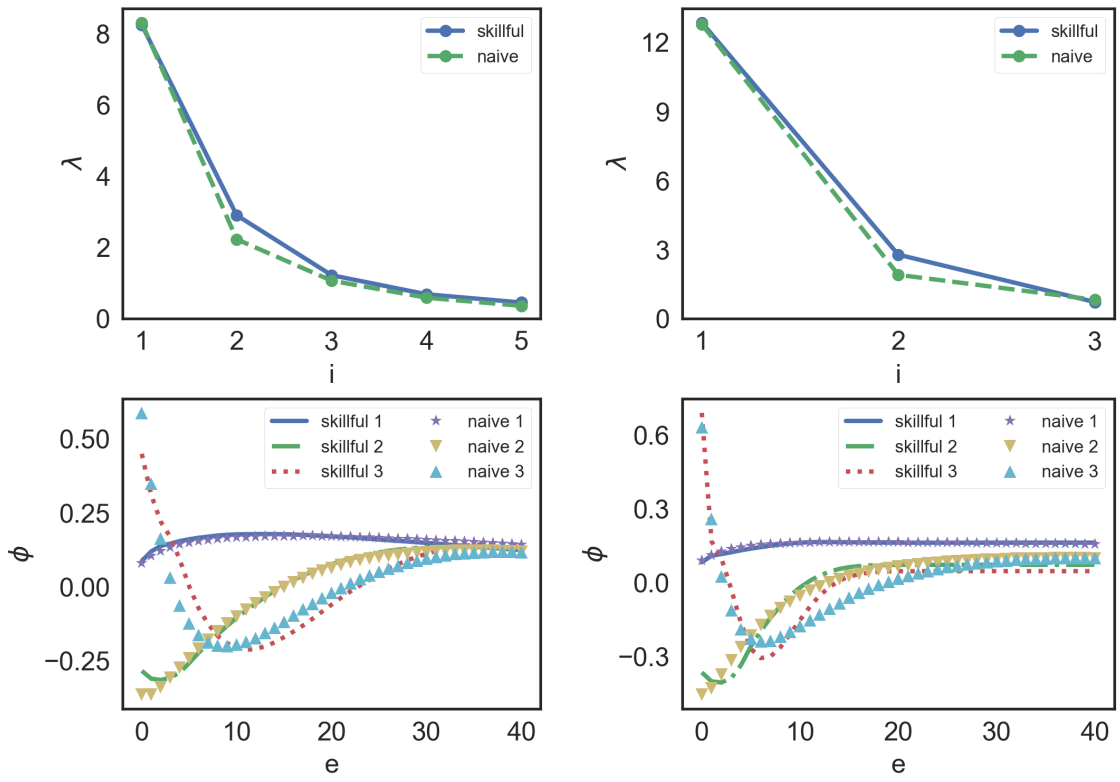


Fig. 2.5. The PDF of random variables $\xi_1, \cdots, \xi_3$ of reduced order model for skillful and naïve agents with $l = 0.05$.

The results obtained with the reduced order model match closely those obtained with the MC samples. As expected, the reduced order model slightly underestimates the variance. In Fig. 2.8, we show the $Q^0(e, \theta)$ and 95% lower and upper confidence levels of $Q(e, \theta, \omega)$ alongside some sample functions with the reduced order random model, with $l = 0.01$ and 0.05. Our results match with those from MC samples shown in Figs 2.1 and 2.2, albeit information about the "high frequency" behavior of the agent has been coarse-grained.

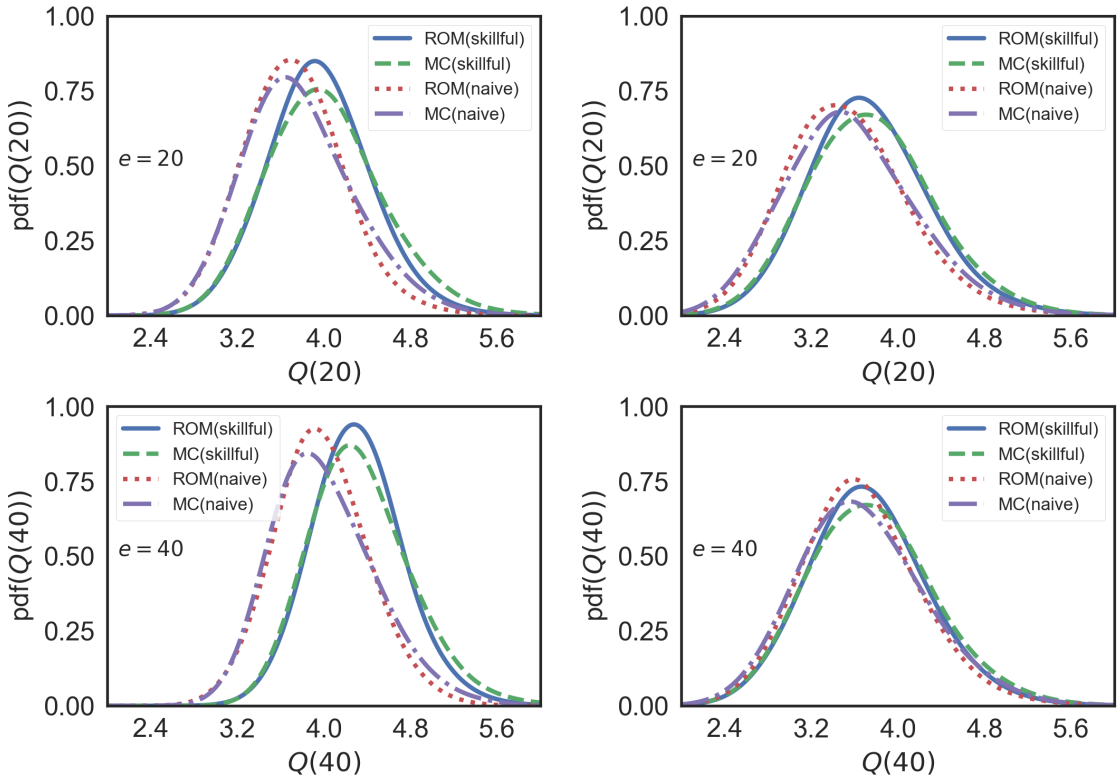


(a) $l = 0.01$ (b) $l = 0.05$

Fig. 2.6. The eigenvalues (λ) and eigenfunction (ϕ) of the reduced order model that capture more than 90% of spectral energy of the random field for skillful and naïve agents with $l = 0.01$ and 0.05. Note, only first 3 eigenfunctions out of 5 are shown for $l = 0.01$.

2.4 Conclusions

In this chapter, we modeled the quality function of a leaf agent in the early design stages of the SEP hierarchy as a stochastic process. Our approach relies on the assumption that the design task assigned to an agent can be modeled as a scalar maximization problem. We explicitly captured the principal's beliefs about the task-difficulty and problem-solving skills of an agent. We studied two types of agents, a skillful agent who follows the Bayesian optimization algorithm in solving the maximization problem, and a naïve agent who searches randomly in the design

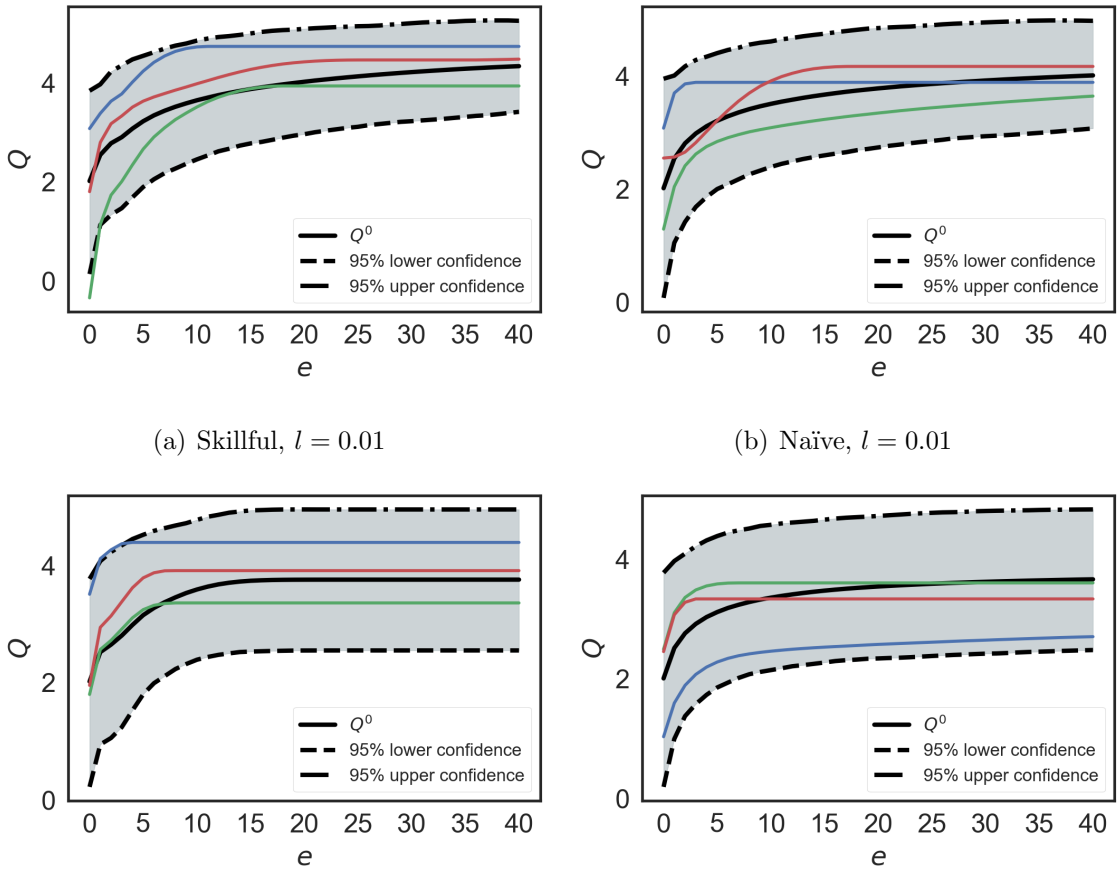


(a) $l = 0.01$ (b) $l = 0.05$

Fig. 2.7. The distribution of the quality at effort levels $e = 20$ and 40 with reduced order model (ROM) and Monte Carlo (MC) samples for skillful and naïve agents with $l = 0.01$ and 0.05.

space to solve the maximization problem. Finally, we constructed a reduced order model based on the KLE of the quality function that can be used in an extensive game-theoretic model of the SEP.

We found that the common assumption that the quality function is linear with additive Gaussian noise is insufficient. We showed that the quality function is an increasing concave function with an almost flat portion after the early stage of the effort. The derivative and curvature of the quality function depend on the task-difficulty and problem-solving skills of the agent. The derivative is large at early stages of the effort, and it becomes smaller as the effort level increases for both skillful



(a) Skillful, $l = 0.01$

(b) Naïve, $l = 0.01$

(c) Skillful, $l = 0.05$

(d) Naïve, $l = 0.05$

Fig. 2.8. Three random quality function samples (solid colored lines), the mean of all 10,000 sampled quality functions from reduced order random model (solid black line), and the 95% confidence levels (gray shaded area between the black dashed and dashed-dotted lines). The first and the second columns correspond to $\theta =$ skillful and $\theta =$ naïve, respectively.

and naïve agents. The derivative at early stages of effort is lower for a naïve agent than that for a skillful agent. We also saw that the eigenfunctions of the reduced order model can be interpreted in the following way. The first eigenfunction is a constant that captures the uncertainty about the maximum possible quality value. The higher order eigenfunctions capture the uncertainty about the search process. We demonstrated that the statistical properties of the reduced order model match

those of the MC samples of the full-blown stochastic process closely, albeit the fine details are coarse-grained. Therefore, we conclude that one may use the reduced order model in a principal-agent representation of the SEP.

There remain several open questions. First, we assumed that the agent starts the maximization problem from scratch. Usually, a designer may already know a lot about the attribute function and this information may or may not be available to the principal [52]. However, we anticipate that our framework is easily adjusted to this case. Second, we assumed that the agent has only one information source, i.e., that there is no alternative way to gain information about the attribute. Such alternative sources of information could be simulations of varying task-difficulty or building prototypes. Third, our account of task-difficulty is quite restrictive as we have not covered all the possibilities such as varying design dimensions, covariance smoothness, and the existence of discrete choices. Fourth, we did not discuss the quality function associated with design tasks requiring the discovery of the Pareto efficient frontier of multi-objective optimization problems. Fifth, we have not discussed how the SE forms their original beliefs about the task-difficulty and the problem-solving skills of the agents nor how they update them when presented with new data. Finally, we have ignored the possibility of iterative communication between the SE and the agents which, among other things, may result to the convergence of beliefs about the attribute function and, consequently, to less uncertain quality functions. All these open questions are the subject of ongoing modeling, experimental, and observational research.

3. ONE-SHOT SHALLOW SYSTEMS ENGINEERING PROCESS

3.1 Introduction

Cost and schedule overruns plague the majority of large systems engineering projects across multiple industry sectors including [5], power [6], defense [7], and space [8]. As design mistakes are more expensive to correct during the production and operation phases, the design phase of the systems engineering process (SEP) has the largest potential impact on cost and schedule overruns. Collopy et al. [27] argued that requirements engineering (RE), which is a fundamental part of the design phase, is a major source of inefficiencies in systems engineering. In response, they developed value-driven design (VDD) [53], a systems design approach that starts with the identification of a system-level value function and guides the systems engineer (SE) to construct subsystem value functions that are aligned with the system goals. According to VDD, the subsystem engineers (sSE) and contractors should maximize the objective functions passed down by the SE instead of trying to meet requirements. Since then, researchers have suggested various generalizations of VDD [54–60], while applying it to many applications [61–65].

RE and VDD make the assumption that the goals of the human agents involved in the SEP are aligned with the SE goals. In particular, RE assumes that, agents attempt to maximize the probability of meeting the requirements, while VDD assumes that they will maximize the objective functions supplied by the SE. However, this assumption ignores the possibility that the design agents, as all humans, may have personal agendas that not necessarily aligned with the system-level goals.

Contrary to RE and VDD, it is more plausible that the design agents seek to maximize their own objectives. Indeed, there is experimental evidence that the quality

of the outcome of a design task is strongly affected by the reward anticipated by the agent [12, 66, 67]. In other words, the agent decides how much effort and resources to devote to a design task after taking into account the potential reward. In the field, the reward could be explicitly implemented as an annual performance-based bonus, or, as it is the case most often, it could be implicitly encoded in expectations about job security, promotion, professional reputation, etc. To capture the human aspect in SEPs, one possible way is to follow a game-theoretic approach [13], [14]. Most generally, the SEP can be modeled as a dynamical hierarchical network game with incomplete information. Each layer of the hierarchy represents interactions among the SE and some sSEs, or the sSEs and other engineers or contractors. With the term "principal," we refer to any individual delegating a task, while we reserve the term "agent" for the individual carrying out the task. Note that an agent may simultaneously be the principal in a set of interactions down the network. For example, the sSE is the agent when considering their interaction with the SE (the principal), but the principal when considering their interaction with a contractor (the agent). At each time step, the principals pass down delegated tasks along with incentives, the agents choose the effort levels that maximize their expected utility, perform the task, and return the outcome to the principals.

The iterative and hierarchical nature of real SEPs makes them extremely difficult to model in their full generality. Given that our aim is to develop a theory of SEPs, we start from the simplest possible version of a SEP which retains, nevertheless, some of the important elements of the real process. Specifically, the objective of this chapter is to develop and analyze a principal-agent model of a one-shot, shallow SEP. The SEP is "one-shot" in the sense that decisions are made in one iteration and they are final. The term "shallow" refers to a one-layer-deep SEP hierarchy, i.e., only the SE (principal) and the sSEs (agents) are involved. The agents maximize their expected utility given the incentives provided by the principal, and the principal selects the incentive structure that maximizes the expected utility of the system. We pose this

mechanism design problem [68] as a bi-level optimization problem and we solve it numerically.

A key component of our SEP model is the quality function of an agent. The quality function is a stochastic process that models the principal's beliefs about the outcome of the delegated design task given that the agent devotes a certain amount of effort. The quality function is affected by what the principal believes about the task complexity and the problem solving skills of the agent. Following our work [69], we model the design task as a maximization problem where the agent seeks the optimal solution. The principal expresses their prior beliefs about the task complexity by modeling the objective function as a random draw from a Gaussian process prior with a suitably selected covariance function.

As we showed in [69], conditioned on knowing the task complexity and the agent type, the quality function is well approximated by an increasing, concave function of effort with additive Gaussian noise. However, we will use a linear approximation for the quality function.

We study numerically two different scenarios. The first scenario assumes that the SE knows the agent types and the task complexity, but they do not observe the agent's effort. This situation is known in game theory as a *moral hazard* problem [70]. The most common way to solve a moral hazard problem is to use the first order approach (FOA) [71]. In the FOA, the incentive compatibility constraint of the agent is replaced by its first order necessary condition. However, the FOA depends on the convexity of the distribution function in effort which is not valid in our case. There have been several attempts to solve the principal-agent model where the requirements of the FOA may fail, nonetheless they must still satisfy the monotone likelihood ratio property [72].

In the second scenario, we study the case of moral hazard with simultaneous *adverse selection* [73], i.e., the SE observes neither the effort nor the type of agents nor the task complexity. This is a Bayesian game with incomplete information. [74]. In this case, the SE experiences additional loss in their expected utility, because the

sSEs' can pretend to have different types. The revelation principle [75] guarantees that it suffices to search for the optimal mechanism within the set of incentive compatible mechanisms, i.e., within the set of mechanisms in which the sSEs are telling the truth about their types and technology maturity. Here, we solve the optimization problem in the principal-agent model, numerically with making no assumptions about the quality function.

This chapter is organized as follows. In section 3.2 we will derive the mathematical model of the SEP and we will study the type-independent and type-dependent optimal contracts. We will also introduce the value and utility functions. In section 3.3, we perform an exhaustive numerical study and show the solutions for several case studies. Finally, we conclude in section 4.4.

3.2 Modeling a One-shot, Shallow Systems Engineering Process

3.2.1 Basic definitions and notation

As mentioned in the introduction, we develop a model of a one-shot (the game evolves in one iteration and the decisions are final), shallow (one-layer-deep hierarchy) SEP. The SE has decomposed the system into N subsystems and assigned a sSE to each one of them. We use $i = 1, \ldots, N$ to label each subsystem. From now on, we refer to the SE as the principal and the sSEs as the agents. The principal delegates tasks to the agents along with incentives. The agents choose how much effort to devote on their task by maximizing their expected utility. The principal, anticipates this reaction and selects the incentives that maximize the system-level expected utility.

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space where, Ω is the sample space, $\mathcal{F}$ is a σ-algebra, and $\mathbb{P}$ is the probability measure. With $\omega \in \Omega$ we refer to the random state of nature. We use upper case letters for random variables (r.v.), bold upper case letters for their range, and lower case letters for their possible values. For example, the type of agent i is a r.v. Θ_i taking M_i discrete values θ_i in the set $\mathbf{\Theta}_i \equiv \Theta_i(\Omega) = \{1, \ldots, M_i\}$. Collectively, we denote all types with the N-dimensional tuple $\Theta = (\Theta_1, \ldots, \Theta_N)$ and

we reserve Θ_{-i} to refer to the $(N-1)$-dimensional tuple containing all elements of Θ except Θ_i. This notation carries to any N-dimensional tuple. For example, θ and θ_{-i} are the type values for all agents and all agents except i, respectively. The range of Θ is $\mathbf{\Theta} = \times_{i=1}^{N} \mathbf{\Theta}_i$.

The principal believes that the agents types vary independently, i.e., they assign a probability mass function (p.m.f.) on Θ that factorizes over types as follows:

$$\mathbb{P}[\Theta = \theta] = \prod_{i=1}^{N} \mathbb{P}[\Theta_i = \theta_i] = \prod_{i=1}^{N} p_{i\theta_i}, \tag{3.1}$$

for all θ in $\mathbf{\Theta}$, where $p_{ik} \geq 0$ is the probability that agent i has type k, for k in $\mathbf{\Theta}_i$. Of course, we must have $\sum_{k=1}^{M_j} p_{ik} = 1$, for all $i = 1, \ldots, N$.

Each agent knows their type, but their state of knowledge about all other agents is the same as the principal's. That is, if agent i is of type $\Theta_i = \theta_i$, then their state of knowledge about everyone else is captured by the p.m.f.:

$$\mathbb{P}[\Theta_{-i} = \theta_{-i} | \Theta_i = \theta_i] = \frac{\mathbb{P}[\Theta = \theta]}{\mathbb{P}[\Theta_i = \theta_i]} = \prod_{j \neq i} p_{j\theta_j}. \tag{3.2}$$

Agent i chooses a normalized effort level $e_i \in [0, 1]$ for his delegated task. We assume that this normalized effort is the percentage of an agent's maximum available effort. The units of the normalized effort depend on the nature of the agent's subsystem. If the principal and the agent are both part of same organization then the effort can be the time that the agent dedicates to the delegated task in a particular period of time, e.g., in a fiscal year. On the other hand, if the agent is a contractor, then the effort can be the percentage of the available yearly budget that the contractor spends on the assigned task. We represent the monetary cost of the i-th agent's effort with the random process $C_i(e_i)$. In economic terms, $C_i(e_i)$ is the opportunity cost, i.e., the payoff of the best alternative project in agent could devote their effort. In general, we know that the process $C_i(e_i)$ should be an increasing function of the effort e_i. For simplicity, we assume that the cost of effort of the agents is quadratic,

$$C_i(e_i) := c_{i\Theta_i} e_i^2, \tag{3.3}$$

with a type-dependent coefficient $c_{ik} > 0$ for all k in $\mathbf{\Theta}_i$.

The quality function of the i-th agent is a real valued random $Q_i(e_i) := Q_i(e_i)$ process paremeterized by the effort e_i. The quality function models everybody's beliefs about the design capabilities of agent i. The interpretation of the quality function is as follows. If agent i devotes to the task an effort of level e_i, then they produce a random outcome of quality $Q_i(e_i)$. In our previous work [69], we created a stochastic model for the quality function of a designer where we explicitly captured its dependence on the problem-solving skills of the designer and on the task complexity. In that work, we showed that $Q_i(e_i)$ has increasing and concave sample paths, that its mean function is increasing concave, and the standard deviation is decreasing with effort, albeit mildly, it is independent of the problem-solving skills of the designer, and it only increases mildly with increasing task complexity. Examining the spectral decomposition of the process for various cases, we observed that it can be well-approximated by:

$$Q_i(e_i) = q^0_{i\Theta_i}(e_i) + \sigma_{i\Theta_i}\Xi_i, \tag{3.4}$$

where, for k in $\mathbf{\Theta}_i$, $q^0_{ik}(e_i)$ is an increasing, concave, type-dependent mean quality function, $\sigma_{ik} > 0$ is a type-dependent standard deviation parameter capturing the aleatory uncertainty of the design process, and Ξ_i is a standard normal r.v. If we further assume that the time window for design is relatively small, then the $q^0_{ik}(e_i)$ term can be approximated as a linear function. Therefore, we will assume that the quality function is:

$$Q_i(e_i) = \kappa_{i\Theta_i} e_i + \sigma_{i\Theta_i}\Xi_i, \tag{3.5}$$

where, κ is inversely proportional to the complexity of the problem. For instance, a large κ corresponds to a low-complexity task while a small κ corresponds to a high-complexity task. The standard deviation parameter σ captures the inherent uncertainty of the design process and depends on the maturity of the underlying technology. In summary, an agent's type is characterized by the triplet cost-complexity-uncertainty.

From the perspective of the principal, the r.v.'s Ξ_i are independent of the agents' types Θ_i as they represent the uncertain state of nature. A stronger assumption that we employ is that the Ξ_i's are also independent to each other. This assumption is strong because it essentially means that the qualities of the various subsystems are decoupled. Under these independence assumptions, the state of knowledge of the principal is captured by the following probability measure:

$$\mathbb{P}\left[\Theta = \theta, \Xi \in \times_{i=1}^{N} \mathbf{B}_i\right] = \prod_{i=1}^{N} \left[p_{i\theta_i} \int_{\mathbf{B}_i} \phi(\xi_i) d\xi_i \right], \tag{3.6}$$

for all $\theta \in \mathbf{\Theta}$ and all Borel-measurable $\mathbf{B}_i \subset \mathbb{R}$. Assuming that all these are common knowledge, the state of knowledge of agent i after they observe their type θ_i (but before they observe Ξ_i) is

$$\begin{aligned} &\mathbb{P}\left[\Theta_{-i} = \theta_{-i}, \xi \in \times_{i=1}^{N} \mathbf{B}_i \middle| \Theta_i = \theta_i\right] \\ &\quad = \frac{\mathbb{P}\left[\Theta = \theta, \xi \in \times_{i=1}^{N} \mathbf{B}_i\right]}{\mathbb{P}[\Theta_i = \theta_i]} \\ &= \mathbb{P}[\Theta_{-i} = \theta_{-i} | \Theta_i = \theta_i] \prod_{i=1}^{N} \left[\int_{\mathbf{B}_i} \phi(\xi_i) d\xi_i \right] \end{aligned} \tag{3.7}$$

Finally, we use $\mathbb{E}[\cdot]$ to denote the expectation of any quantity over the state of knowledge of the principal as characterized by the probability measure of Eq. (3.6). That is, the expectation of any function $f(\Theta, \Xi)$ of the agent types Θ and the state of nature Ξ is

$$\mathbb{E}[f(\Theta, \Xi)] = \sum_{\theta \in \mathbf{\Theta}} \int_{\mathbb{R}^N} f(\theta, \xi) \prod_{i=1}^{N} \left[p_{i\theta_i} \phi(\xi_i) \right] d\xi. \tag{3.8}$$

Similarly, we use the notation $\mathbb{E}_{i\theta_i}[\cdot]$ to denote the conditional expectation over the state of knowledge of an agent i who knows that their type is $\Theta_i = \theta_i$. This is the expectation $\mathbb{E}[\cdot | \Theta_i = \theta_i]$ with respect to the probability measure of Eq. (3.2) and we have:

$$\mathbb{E}_{ik}[f(\Theta, \Xi)] = \sum_{\theta_{-i} \in \mathbf{\Theta}_{-i}} \int_{\mathbb{R}^N} f(\theta_i, \theta_{-i}, \xi) \frac{\prod_{j=1}^{N} \left[p_{j\theta_j} \phi(\xi_j) \right]}{p_{i\theta_i}} d\xi. \tag{3.9}$$

3.2.2 Type-independent optimal contracts

We start by considering the case where the principal offers a single take-it-or-leave-it contract independent of the agent type. This is the situation usually encountered in

contractual relationships between the SE and the sSEs within the same organization. The principal offers the contract and the agent decides whether or not to accept it. If the agent accepts, then they select their level of effort by maximizing their expected utility, they work on their design task, they return the outcome quality back to the principal, and they receive their reward. We show a schematic view of this type of contracts in Fig. 3.1(a). A contract is a monetary *transfer function* $t_i : \mathbb{R} \to \mathbb{R}$ that specifies the agent's *compensation* $t_i(q_i)$ contingent on the quality level q_i. Therefore, the payoff of the i-th agent is the random process:

$$\Pi_i(e_i) = t_i\left(Q_i\left(e_i\right)\right) - C_i(e_i). \tag{3.10}$$

We assume that the agent knows their type, but they choose the optimal effort level ex-ante, i.e., they choose the effort level before seeing the state of the nature Ξ_i. Denoting their monetary utility function by $U_i(\pi_i) = u_{i\Theta_i}(\pi_i)$, the i-th agent selects an effort level by solving:

$$e^*_{i\theta_i} = \underset{e_i \in [0,1]}{\arg\max}\ \mathbb{E}_{i\theta_i}\left[U_i\left(\Pi_i(e_i)\right)\right]. \tag{3.11}$$

Let Q^*_i be the r.v. representing the quality function that the principal should expect from agent i if they act optimally, i.e.,

$$Q^*_i = Q_i(e^*_{i_{\Theta_i}}). \tag{3.12}$$

Then the system level value is a r.v. of the form

$$V = v(Q^*), \tag{3.13}$$

where $v : \mathbb{R}^N \to \mathbb{R}$ is a function of the subsystem outcomes Q^*. We introduce the form of the value function, $v(q)$, in Sec. 3.3. Note that, even though in this work the r.v. V is assumed to be just a function of Q^*, in reality it may also depend on the random state of nature, e.g., future prices, demand for the system services. Consideration of the latter is problem-dependent and beyond the scope of this work.

Given the system value V and taking into account the transfers to the agents, the system-level payoff is the r.v.

$$\Pi_0 = V - \sum_{i=1}^{N} t_i(Q_i^*). \tag{3.14}$$

If the monetary utility of the principal is $u_0(\pi_0)$, then they should select the transfer functions $t(\cdot) = (t_1(\cdot), \ldots, t_N(\cdot))$ by solving:

$$t^*(\cdot) = \arg\max_{t(\cdot)} \mathbb{E}\left[u_0\left(\Pi_0\right)\right]. \tag{3.15}$$

However, guarantee that they want to participate in the SEP, the expected utility of the sSEs must be greater than the expected utility they would enjoy if they participated in another project. Therefore, the SE must solve Eq. (3.15) subject to the *participation constraints*:

$$\mathbb{E}_{i\theta_i}\left[U_i\left(\Pi_i\right)\right] \geq \bar{u}_{i\theta_i}, \tag{3.16}$$

for all possible values of θ_i, and all $i = 1, \ldots, N$, where $\bar{u}_{i\theta_i}$ is known as the *reservation utility* of agent i.

3.2.3 Type-dependent optimal contracts

By offering a single transfer function, the principal is unable to differentiate between the various agent types when adverse selection is an issue. That is, all agent types, independently of their cost, complexity, and uncertainty attributes, are offered exactly the same transfer function. In other words, with a single transfer function the principal is actually targeting the average agent. This necessarily leads to inefficiencies stemming from problems such as paying an agent involved in a low-complexity task more than a same cost and uncertainty agent involved in a high-complexity task.

The principal can gain in efficiency by offering different transfer functions (if any exist) that target specific agent types. For example, the principal could offer a transfer function that is suitable for cost-efficient, low-complexity, low-uncertainty agents, and one for cost-inefficient agents, low-complexity, low-uncertainty, etc., for

any other combination that is supported by the principal's prior knowledge about the types of the agent population. To implement this strategy the principal can employ the following extension to the mechanism of Sec. 3.2.2. Prior to initiating work, the agents announce their types to the principal and they receive a contract that matches the announced type. In Fig. 3.1(b), we show how this type of contract evolves in time. Let us formulate this idea mathematically. The i-th agent announces a type θ_i' in $\mathbf{\Theta}_i$ (not necessarily the same as their true type θ_i), and they receive the associated, type-specific, transfer function $t_{i\theta_i'}(\cdot)$. The payoff to agent i is now:

$$\Pi_i(e_i, \theta_i') = t_{i\theta_i'}(Q_i(e_i)) - C_i(e_i), \tag{3.17}$$

where all other quantities are like before. Given the announcement of a type θ_i', the rational thing to do for agent i is to select a level of $e_i^*(\theta_i, \theta_i')$ by maximizing their expected utility, i.e., by solving:

$$e^*_{i\theta_i\theta_i'} = \arg\max_{e_i\in[0,1]} \mathbb{E}_{i\theta_i}[U_i(\Pi_i(e_i, \theta_i'))]. \tag{3.18}$$

Of course, the announcement of θ_i' is also a matter of choice and a rational agent should select also by maximizing their expected utility. The obvious issue here is that agents can lie about their type. For example, a cost-efficient agent (agent with low cost of effort) may pretend to be a cost-inefficient agent (agent with high cost of effort). Fortunately, the revelation principle [75] comes to the rescue and simplifies the situation. It guarantees that, among the optimal mechanisms, there is one that is incentive compatible. Thus it will be sufficient if the principal constraints their contracts to over truth-telling mechanisms. Mathematically, to enforce truth-telling, the SE must satisfy the *incentive compatibility* constraints:

$$\mathbb{E}_{i\theta_i}[U_i(\Pi_i(e^*_{i\Theta_i\theta_i}, \theta_i))] \geq \mathbb{E}_{i\theta_i}[U_i(\Pi_i(e^*_{i\Theta_i\theta_i'}, \theta_i'))], \tag{3.19}$$

for all $\theta_i \neq \theta_i'$ in $\mathbf{\Theta}_i$. Eq. (3.19) expresses mathematically that "the expected payoff of agent i when they are telling the truth is always greater than or equal to the expected payoff they would enjoy if they lied."

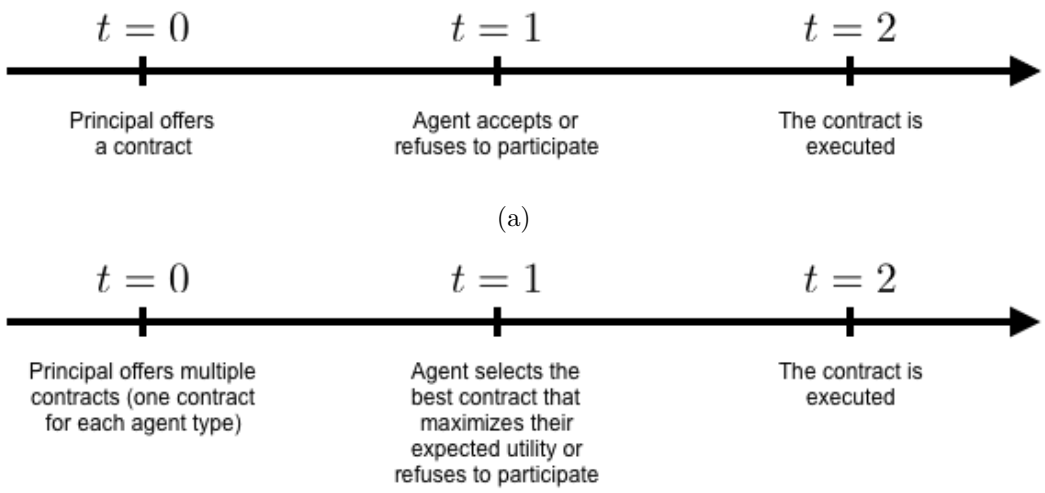


Fig. 3.1. (a): Timing of the contract for type-independent contracts. (b): Timing of the contract for type-dependent contracts.

Similar to the developments of Sec. 3.2.2, the quality that the SE expects to receive is:

$$Q_i^* = Q_i(e^*_{i\Theta_i\Theta_i}), \tag{3.20}$$

where we use the fact that the mechanism is incentive-compatible. The payoff of the SE becomes:

$$\Pi_0 = V - \sum_{i=1}^{N} t_{i\Theta_i}(Q_i^*). \tag{3.21}$$

Therefore, to select the optimal transfer functions, the SE must solve:

$$\max_{t(\cdot,\cdot)} \mathbb{E}\left[u_0(\Pi_0)\right], \tag{3.22}$$

subject to the incentive compatibility constraints of Eq. (3.19), and the participation constraints:

$$\mathbb{E}_{i\theta_i}[U_i(\Pi_i(e^*_{i\Theta_i\Theta_i}, \Theta_i))] \geq \bar{u}_{i\theta_i}, \tag{3.23}$$

for all $\theta_i \in \mathbf{\Theta}_i$, where we also assume that the incentive compatibility constrains hold.

3.2.4 Parameterization of the transfer functions

Transfer functions must be practically implementable. That is, they must be easily understood by the agent when expressed in the form of a contract. To be easily implementable, transfer functions should be easy to convey in the form of a table. To achieve this, we restrict our attention to functions that are made out of constants, step functions, linear functions, or combinations of these.

Despite the fact that including such functions would likely enhance the principal's payoff, we exclude transfer functions that encode penalties for poor agent performance, i.e., transfer functions that can take negative values. First, contracts with penalties may not be implementable if the principal and the agent reside within the same organization. Second, even when the agent is an external contractor penalties are not commonly encountered in practice. In particular, if the SE is a sensitive government office, e.g., the department of defense, national security may dictate that the contractors should be protected from bankruptcy. Third, we do not expect our theory to be empirically valid when penalties are included since, according to prospect theory [76], humans perceive losses differently. They are risk-seeking when the reference point starts at a loss and risk-averse when the reference point starts at a gain.

To overcome these issues we restrict our attention to transfer functions that include three simple additive terms: a constant term representing a participation payment, i.e., a payment received for accepting to be part of the project; a constant payment that is activated when a requirement is met; and a linear increasing part activated after meeting the requirement. The role of the latter two part is to incentivize the agent to meet and exceed the requirements.

We now describe this parameterization mathematically. The transfer function associated with type k in $\boldsymbol{\Theta_i}$ of agent i is parameterized by:

$$\begin{aligned} t_{ik}\left(q_i\right) \quad = \quad & a_{ik,0} + a_{ik,1}\,\mathrm{H}\left(q_i - a_{ik,2}\right) \\ & + a_{ik,3}\left(q_i - a_{ik,2}\right)\mathrm{H}\left(q_i - a_{ik,2}\right), \end{aligned} \tag{3.24}$$

where H is the Heaviside function ($H(x) = 1$ if $x \geq 0$ and 0 otherwise), and all the parameters $a_{ik,0}, \ldots, a_{ik,3}$ are non-negative. In Eq. (3.24), $a_{ik,0}$ is the participation reward, $a_{ik,1}$ is the award for exceeding the passed-down requirement, $a_{ik,2}$ is the passed-down requirement, and $a_{ik,3}$ the payoff per unit quality exceeding the passed-down requirement. We will call these form of transfer functions the "requirement based plus incentive" (RPI) transfer function. In case the $a_{ik,3} = 0$, we call it the "requirement based" (RB) transfer function. At this point, it is worth mentioning that the passed-down requirement $a_{ik,2}$ is not necessarily the same as the true system requirement r_i, see our results in Sec. 3.3. As we have shown in earlier work [14], the optimal passed-down requirement differs from the true system requirement. For notational convenience, we denote by $\mathbf{a}_{ik} \in \mathbb{R}_+^4$ ($\mathbb{R} = \{x \in \mathbb{R} : x \geq 0\}$) the transfer parameters pertaining to agent i of type $k \in \mathbf{\Theta}_i$, i.e.,

$$\mathbf{a}_{ik} = (a_{ik,0}, \ldots, a_{ik,3}) \,. \tag{3.25}$$

Similarly, with $\mathbf{a}_i \in \mathbb{R}_+^{4M_i}$ we denote the transfer parameters pertaining to agent i for all types, i.e.,

$$\mathbf{a}_i = (\mathbf{a}_{i1}, \ldots, \mathbf{a}_{iM_i}) \,, \tag{3.26}$$

and with $\mathbf{a} \in \mathbb{R}_+^{4\sum_{i=1}^N M_i}$ all the transfer parameters collectively, i.e.,

$$\mathbf{a} = (\mathbf{a}_1, \ldots, \mathbf{a}_N) \,. \tag{3.27}$$

3.2.5 Numerical solution of the optimal contract problem

The optimal contract problem is a an intractable bi-level, non-linear programming problem.In particular, the SE's problem is for the case of type-dependent contracts is to maximize the expected system-level utility over the class of implementable contracts, i.e.,

$$\max_{\mathbf{a}} \mathbb{E}\left[u_0(\Pi_0)\right] , \tag{3.28}$$

subject to

1. *contract implementability constraints*:

$$a_{ik,j} \geq 0, \tag{3.29}$$

for all $i = 1, \ldots, N, k = 1, \ldots, M_i, j = 0, \ldots, 3$;

2. *individual rationality constraints*:

$$e^*_{ikl} = \arg\max_{e_i \in [0,1]} \mathbb{E}_{ik}[U_i(\Pi_i(e_i, l))], \tag{3.30}$$

for all $i = 1, \ldots, N, k = 1, \ldots, M_i, l = 1, \ldots, M_i$;

3. *participation constraints*:

$$\mathbb{E}_{ik}[U_i(\Pi_i(e^*_{ikk}, k))] \geq \bar{u}_{ik}, \tag{3.31}$$

for all $i = 1, \ldots, N, k = 1, \ldots, M_i$; and

4. *incentive compatibility constraints*:

$$\mathbb{E}_{ik}[U_i(\Pi_i(e^*_{ikk}, k))] \geq \mathbb{E}_{ik}[U_i(\Pi_i(e^*_{ikl}, l))], \tag{3.32}$$

for all $i = 1, \ldots, N$ and $k \neq l$ in $\{1, \ldots, M_i\}$.

For the case of type-independent contracts, one adds the constraint $\mathbf{a}_{ik} = \mathbf{a}_{il}$ for all $i = 1, \ldots, N$ and $k \neq l$ in $\{1, \ldots, M_i\}$ and the incentive compatibility constraints are removed.

A common approach to solving bi-level programming problems is to replace the internal optimization with the corresponding Karush-Kuhn-Tucker (KKT) condition. This approach is used when the internal problem is concave, i.e., when it has a unique maximum. However, in our case, concavity is not guaranteed, and we resort to nested optimization. We implement everything in Python using the Theano [77] symbolic computation package exploit automatic differentiation. We solve the follower problem using sequential least squares programming (SLSQP) as implemented in the scipy package. [78] We use simulated annealing to find the global optimum point of

the leader problem. We first convert the constraint problem to the unconstrained problem using the penalty method such that:

$$f(\mathbf{a}) = \mathbb{E}\left[u_0(\Pi_0)\right] + \sum_{i=1}^{N_c} \min\left(g_i(\mathbf{a}), 0\right), \tag{3.33}$$

where $g_i(\cdot)$'s are the constraints in Eqs. (3.29-3.32). Maximizing the $f(\mathbf{a})$ in Eq. 3.33, is equivalent to finding the mode of the distribution:

$$\pi_\gamma(\mathbf{a}) \propto \exp\left(\gamma f(\mathbf{a})\right), \tag{3.34}$$

we use Sequential Monte Carlo (SMC) [79] method to sample from this distribution by increasing γ from 0.001 to 50. To perform the SMC, we use the "pysmc" package [80]. To ensure the computational efficiency of our approach, we need to use a numerical quadrature rule to approximate the expectation over Ξ. We need to be able to carry out expectations of the form of Eq. 3.9 a.k.a. Eq. 3.8 and Eq. 3.7. Since, we have at most two possible types in our case studies, the summation over the possible types is trivial. Focusing on expectations over Ξ, we evaluate them using a sparse grid quadrature rule [81]. In particular, any expectation of the form $\mathbb{E}[g(\Xi)]$ is approximated by:

$$\mathbb{E}[g(\Xi)] \approx \sum_{s=1}^{N_s} w^{(s)} g\left(\xi^{(s)}\right), \tag{3.35}$$

where $w^{(s)}$ and $\xi^{(s)}$ are the $N_s = 127$ quadrature points of the level 6 sparse grid quadrature constructed by the Gauss-Hermite 1D quadrature rule.

3.2.6 Value function and risk behavior

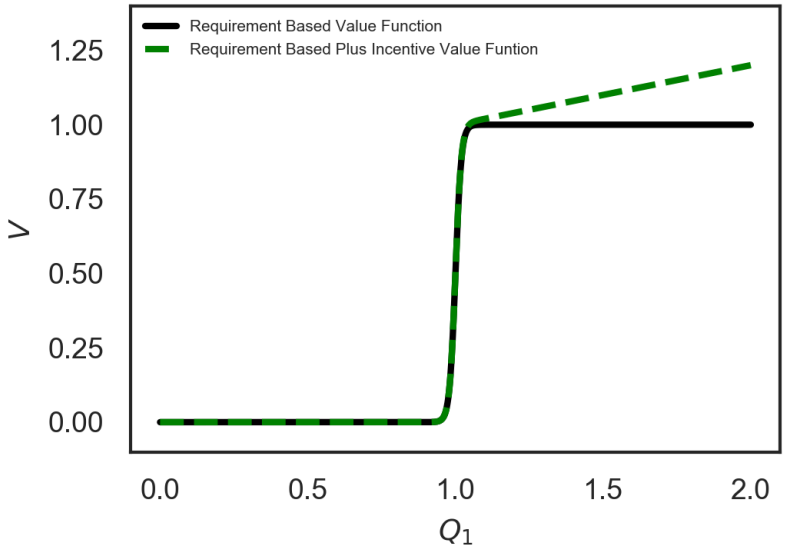
We assume two types of value functions, namely, the requirement based (RB) and requirement based plus incentive (RPI). Mathematically, we define these two value functions as:

$$V_{\text{RB}} := v_0 \prod_{i=1}^{N} \left\{H(Q_i^* - 1)\right\}. \tag{3.36}$$

and,

$$V_{\text{RPI}} := v_0 \prod_{i=1}^{N} \left\{H(Q_i^* - 1)\right\} \left[1 + 0.2(Q_i^* - 1)\right], \tag{3.37}$$

respectively. In Fig. 3.2, we show these two value functions for one subsystem.



(a)

Fig. 3.2. The RB value function (black solid line) and RPI value function (green dashed line).

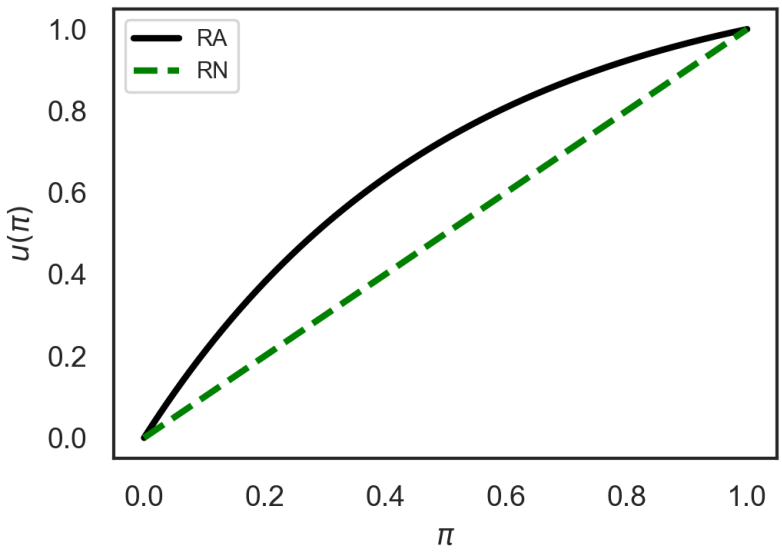
We consider two different risk behaviors for individuals, risk averse (RA) and risk neutral (RN). We use the utility function in Eq. (3.38), for the risk behavior of the agents and principal,

$$u\left(\pi\left(\cdot\right)\right) = \begin{cases} a - be^{-c\pi(\cdot)}, & \text{for RA} \\ \pi\left(\cdot\right), & \text{for RN}, \end{cases} \tag{3.38}$$

where $c = 2$ for a RA agent. The parameters a and b are:

$$a = b = \frac{1}{1 - e^{-c}}.$$

We show these utility functions for the two different risk behaviors in Fig. 3.3.



(a)

Fig. 3.3. The utility functions for risk averse (RA) (black solid line), risk neutral (RN) (green dashed line).

3.3 Numerical Examples

In this section, we start by performing an exhaustive numerical investigation of the effects of task complexity, agent's cost of effort, uncertainty in the quality of the returned task, and adverse selection. In Sec. 3.3.1, we study the "moral hazard only" scenario with the RB transfer and value functions. In Sec. 3.3.1, we study the effect of the RPI transfer and value functions. We study the "moral hazard with adverse selection" in Sec. 3.3.1.

3.3.1 Numerical investigation of the proposed model

In these numerical investigations we consider a single risk neutral principal and a risk averse agent. Each case study corresponds to a choice of task complexity (κ in Eq. (3.5)), cost of effort (c in Eq. (3.3)), and performance uncertainty (σ in Eq.

(3.5)). With regards to task complexity, we select $\kappa = 2.5$ for an easy task and $\kappa = 1.5$ for a hard task. For the cost of effort parameter, we associate $c = 0.1$ and $c = 0.4$ with the low- and high-cost agents, respectively. Finally, low- and high-uncertainty tasks are characterized by $\sigma = 0.1$ and $\sigma = 0.4$, respectively.

Note that, the parameters $\kappa_{i\theta_i}$, $c_{i\theta_i}$, and $\sigma_{i\theta_i}$ have two indices. The first index i is the agent's (subsystem's) number and the second index is the type of the agent. We begin with a series of cases with a single agent with a known type denoted by 1 (moral-hazard-only case studies). In these cases, the parameters corresponding to complexity, cost and uncertainty take the values κ_{11}, c_{11}, and σ_{11}, respectively. We end with a series of cases with a single agent but with an unknown type that can take two discrete, equally probable values 1 and 2 (moral-hazard-and-adverse-selection case studies). Consequently, κ_{11} denotes the effort coefficient of a type-1 agent 1, κ_{12} the same for a type-2 agent, and so on for all the other parameters.

To avoid numerical difficulties and singularities, we replace all Heaviside functions with a sigmoids, i.e.,

$$\hat{H}_\alpha(x) = \frac{1}{1 + e^{-\alpha x}}, \tag{3.39}$$

where the parameter α controls the slope. We choose $\alpha = 50$ for the transfer functions and $\alpha = 100$ for the value function. We consider two types of value functions, RB and RPI value functions, see Sec. 3.2.6. For the RB value function we use the transfer function of Eq. (3.24) constrained so $a_{ik}, 3 = 0$ (RB transfer function). In other words, the agent is paid a constant amount if they achieve the requirement and there is no payment per quality exceeding the requirement. For the case of RPI value function, we remove this constraint.

Moral hazard with RB transfer and value functions

Consider the case of a single risk-averse agent of known type and a risk-neutral principal with an RB value function. In Fig. 3.4(a), we show the transfer functions for several agent types covering all possible combinations of low/high complexity,

low/high cost, and low/high task uncertainty. Fig. 3.4(b) depicts the probability that the principal's expected utility exceeds a given threshold for all these combinations. We refer to this curve as the *exceedance curve*. Finally, in tables 3.1 and 3.2, we report the expected utility of the principal for the low and high cost agents, respectively. We make the following observations:

1. For the same level of task complexity and uncertainty, but with increasing cost of effort:

   (a) the optimal passed-down requirement decreases;

   (b) the optimal payment for achieving the requirement increases;

   (c) the principal's expected utility decreases; and

   (d) the exceedance curve shifts to the left.

   Intuitively, as the agent's cost of effort increases, the principal must make the contract more attractive to ensure that the participation constraints are satisfied. As a consequence, the probability that the principal's expected utility exceeds a given threshold decreases.

2. For the same level of task uncertainty and cost of effort, but with increasing complexity:

   (a) the optimal passed-down requirement decreases;

   (b) the optimal payment for achieving the requirement increases;

   (c) the principal's expected utility decreases; and

   (d) the exceedance curve shifts to the left.

   Thus, we see that an increase in task complexity has a similar effect as an increase in the agent's cost of effort. As in the previous case, to make sure that the agent wants to participate, the principal has to make the contract more attractive as task complexity increases.

3. For the same level of task complexity and cost of effort, but with increasing uncertainty:

   (a) the optimal passed-down requirement increases;

   (b) the optimal payment for achieving the requirement increases;

   (c) the principals expected utility decreases;

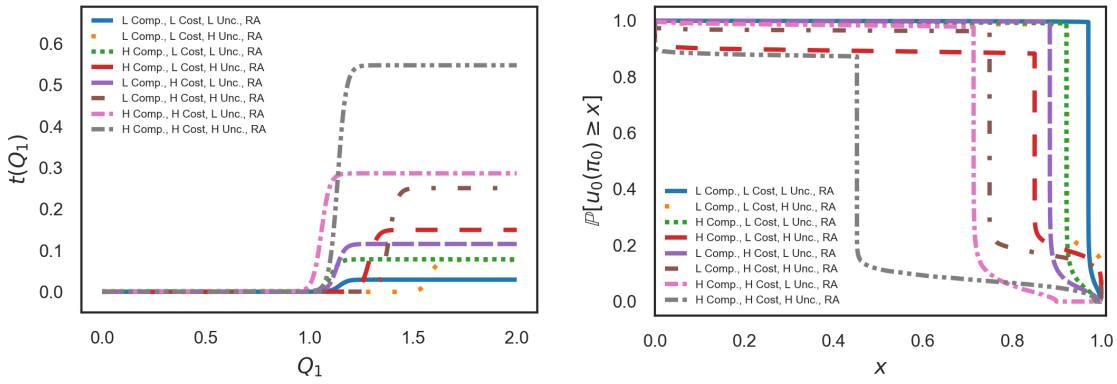
   (d) the exceedance curve shifts towards the bottom right.

   This case is the most interesting. Here as the uncertainty of the task increases, the principal must increase the passed-down requirement to ensure that they are hedged against failure. At the same time, however, they must also increase the payment to ensure that the agent still has an incentive to participate.

4. For all cases considered, the optimal passed down requirement is greater than the true requirement (which is set to one). Note, however, this is not universally true. Our study does not examine all possible combinations of cost, quality, and utility functions that could have been considered. Indeed, as we showed in our previous work [14], there are situations in which a smaller-than-the-true requirement can be optimal.

Table 3.1.
The expected utility of the principal for low cost agent with RB value function.

	Low Uncertainty	High Uncertainty
Low Complexity	0.97	0.93
High Complexity	0.92	0.79



(a) Transfer functions using RB value function.

(b) Exceedance curve.

Fig. 3.4. L and H stand for low and high, respectively, Comp. and Unc. stand for complexity and uncertainty, respectively. The low and high complexity denote the $\kappa_{11} = 2.5$ and $\kappa_{11} = 1.5$, respectively, low and high cost denote $c_{11} = 0.1$ and $c_{11} = 0.4$, respectively, low and high uncertainty denote $\sigma_{11} = 0.1$ and $\sigma_{11} = 0.4$, respectively, RA denotes the risk averse agent. (a): The RB transfer functions for several different agent types with respect to outcome of the subsystem (Q_1) for moral hazard scenario. (b): The exceedance for the moral hazard scenario using the RB transfer function.

Table 3.2.
The expected utility of the principal for high cost agent with RB value function.

	Low Uncertainty	High Uncertainty
Low Complexity	0.89	0.77
High Complexity	0.72	0.45

Moral hazard with RPI transfer and value functions

This case is identical to Sec. 3.3.1, albeit we use the RPI value function, see Sec. 3.2.6, and the RPI transfer function, see Eq. (3.24). Fig 3.5(a), depicts the transfer functions for all combinations of agent types and task complexities. In Fig. 3.5(b),

we show the exceedance curve using the RPI value and transfer functions. Finally, in tables 3.3 and 3.4, we report the expected utility of the principal using the RPI transfer and value functions for the low and high cost agents, respectively. The results are qualitative similar to Sec. 3.3.1, with the additional observations:

1. For the same level of task complexity, uncertainty and agent cost, the optimal reward for achieving the requirement decreases compared to the same cases in Sec. 3.3.1. Intuitively, as the principal has the option to reward the agent based on the quality exceeding the requirement, they prefer to pay less for fulfilling the requirement. Instead, the principal incentivizes the agent to improve the quality beyond the optimal passed-down requirement.

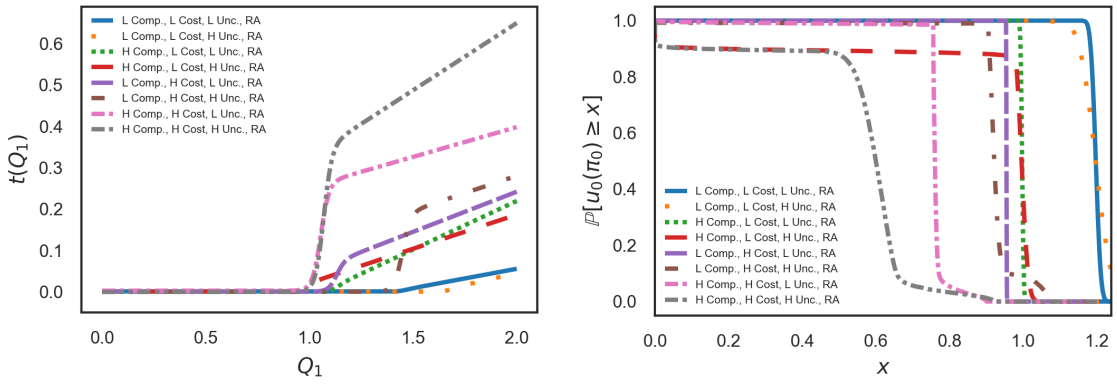
2. The slope of the transfer function beyond the passed-down requirement is almost identical to the slope of the value function.

Table 3.3.
The expected utility of the principal for low cost agent with RPI value function.

	Low Uncertainty	High Uncertainty
Low Complexity	1.2	1.2
High Complexity	1.0	0.89

Table 3.4.
The expected utility of the principal for high cost agent with RPI value function.

	Low Uncertainty	High Uncertainty
Low Complexity	0.95	0.93
High Complexity	0.76	0.56



(a) Transfer functions using RPI value function.

(b) Exceedance curve.

Fig. 3.5. L and H stand for low and high, respectively, Comp. and Unc. stand for complexity and uncertainty, respectively. The low and high complexity denote the $\kappa_{11} = 2.5$ and $\kappa_{11} = 1.5$, respectively, low and high cost denote $c_{11} = 0.1$ and $c_{11} = 0.4$, respectively, low and high uncertainty denote $\sigma_{11} = 0.1$ and $\sigma_{11} = 0.4$, respectively, RA denotes the risk averse agent. (a): The RPI transfer functions for several different agent types with respect to outcome of the subsystem (Q_1) for moral hazard scenario. (b): The exceedance curve for the moral hazard scenario using the RPI transfer function.

In table 3.5, we summarize our observations for the results in Sec. 3.3.1 and 3.3.1. In this table, we show how the passed-down requirement and payment change when we fix two parameters of the model (we denote it by "fix" in the table) and vary the third parameter. We denote increase by ↑ and decrease by ↓.

Table 3.5.
Summary of the observations.

complexity	agent cost	uncertainty	requirement	payment
↑	fix	fix	↓	↑
fix	↑	fix	↓	↑
fix	fix	↑	↑	↑

Moral hazard with adverse selection

Consider the case of a single risk-averse agent of unknown type which takes two possible values, and a risk-neutral principal with a RB value function. We consider two possibilities for the unknown type:

1. *Unknown cost of effort.* Here, we set $\kappa_{11} = \kappa_{12} = 1.5$ ($p(\kappa = \kappa_{11} = 1.5) = 1$), $\sigma_{11} = \sigma_{12} = 0.1$ ($p(\sigma = \sigma_{11} = 0.1) = 1$), and $p(c = c_{11} = 0.1) = 0.5$ and $p(c = c_{12} = 0.4) = 0.5$

2. *Unknown task complexity.* For the unknown quality we assume that $p(\kappa = \kappa_{11} = 2.5) = 0.5$ and $p(\kappa = \kappa_{12} = 1.5) = 0.5$, $\sigma_{11} = \sigma_{12} = 0.4$ ($p(\sigma = \sigma_{11} = 0.4) = 1$), and $c_{11} = c_{12} = 0.4$ ($p(c = c_{11} = 0.4) = 1$).

In this scenario, we maximize the expected utility of the principal subject to constraints in Eqs. (3.29-3.32). The incentive compatibility constraint, Eq. (3.32), guarantees that the agent will choose the contract that is suitable for their true type. In other words, as there are two agent types' possibilities, the principal must offer two contracts, see Fig. 3.1(b). These two contracts must be designed in a way that there is no benefit for the agent to deviate from their true type, i.e., the contracts enforce the agent to be truth telling.

Solving the constraint optimization problem yields:

$$\mathbf{a}_{11} = \mathbf{a}_{12} = (0, 0.29, 1.06),$$

i.e., the two contracts collapse into one. Note that the resulting contract is the same as the pure moral hazard case, Sec. 3.3.1, for an agent with type $\kappa_{11} = 1.5$, $\sigma_{11} = 0.1$, and $c_{11} = 0.4$. In other words, the principal must behave as if there was only a high-cost agent. That is, there are no contacts that can differentiate between a low- and a high-cost agent in this case.

A similar outcome occurs for unknown task complexity. The solution of the constraint optimization problem for this scenario is:

$$\mathbf{a}_{11} = \mathbf{a}_{12} = (0, 0.08, 1.11),$$

which is the same as the optimum contract that is offered for the pure moral hazard case, Sec. 3.3.1, for an agent with type $\kappa_{11} = 1.5$, $\sigma_{11} = 0.4$, and $c_{11} = 0.4$. Therefore, in this case the principal must behave as if there the task is of high complexity.

Note that in both cases above, the collapse of the two contracts to one contract is not a generalizable property of our model. In particular, it may not happen if more flexible transfer functions are allowed, e.g., ones that allow performance penalties.

In Fig. 3.6, we show the transfer functions for the adverse selection scenarios with unknown cost and unknown quality. In tables 3.6 and 3.7, we show the expected utility of two types of agents and the principal using the optimum contract for unknown cost and unknown quality, respectively. To sum up:

1. The unknown cost:

   (a) the optimum transfer function for this problem is as same as that the principal would have offered for a single-type high-cost agent with $c_{11} = c_{12} = 0.4$ (moral hazard scenario with no adverse selection);

   (b) the expected utility of the low cost agent (efficient agent) is greater than that of the high cost agent.

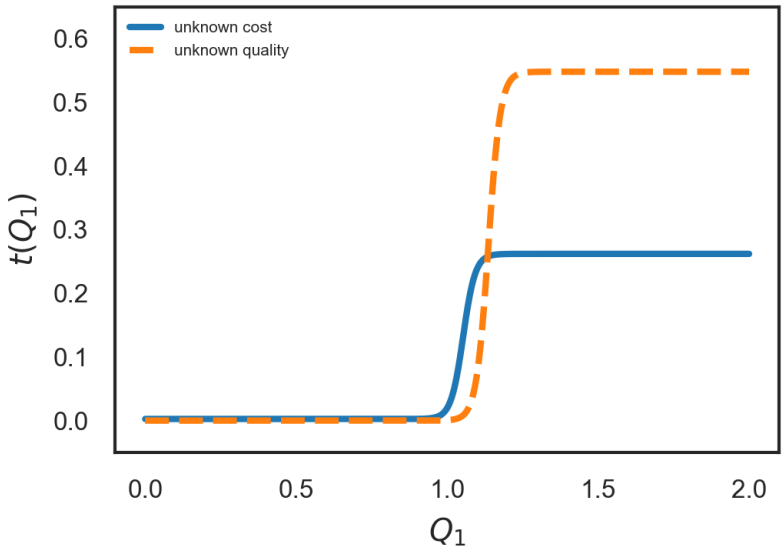
   In this case, the low-cost agent benefits because of information asymmetry. In other words, the principal must pay an information rent to the low-cost agent to reveal their type.

2. The unknown task complexity:

   (a) the optimum contract in this case is the contract that the principal would have offered for the single-type high-complexity task with $\kappa_{11} = \kappa_{12} = 1.5$;

   (b) the expected utility of an agent dealing with a low-complexity task is greater than that of an agent dealing with a high-complexity task.

   Again, due to the information asymmetry, the agent benefits if the task complexity is low. The principal must pay an information rent to reveal the task complexity.



(a)

Fig. 3.6. The transfer function for the adverse selection scenarios with unknown cost (solid line) and unknown quality (dashed line), the agent is risk averse. For unknown cost: $\kappa_{11} = \kappa_{12} = 1.5$ with probability 1, $\sigma_{11} = \sigma_{12} = 0.1$ with probability 1, and $c_{11} = 0.1$ with probability 0.5 and $c_{12} = 0.4$ with probability 0.5. For unknown quality: $\kappa_{11} = 2.5$ with probability 0.5 and $\kappa_{12} = 1.5$ with probability 0.5, $\sigma_{11} = \sigma_{12} = 0.4$ with probability 1, and $c_{11} = c_{12} = 0.4$ with probability 1.

Table 3.6.
The expected utility of the agent with unknown cost for two different contracts.

	$\mathbb{E}[u_1(\cdot)]$	$\mathbb{E}[u_0(\cdot)]$
Low Cost Agent (Type 1)	0.39	0.72
High Cost Agent (Type 2)	0	0.72

Table 3.7.
The expected utility of the agent with unknown quality for two different contracts.

	$\mathbb{E}[u_1(\cdot)]$	$\mathbb{E}[u_0(\cdot)]$
Low Complexity (Type 1)	0.52	0.45
High Complexity (Type 2)	0	0.45

3.3.2 Satellite design

In this section we apply our method on a *simplified* satellite design. Typically a satellite consists of seven different subsystems [82], namely, electrical power subsystem, propulsion, attitude determination and control, on-board processing, telemetry, tracking and command, structures and thermal subsystems. We focus our attention on the propulsion subsystem ($N = 1$). To simplify the analysis, we assume that the design of these subsystems is assigned to a sSE in a one-shot fashion. Note that, the actual systems engineering process of the satellite design is an iterative process and the information and results are exchanged back and forth in each iteration. Our model is a crude approximation of reality. The goal of the SE is to optimally incentivize the sSE to produce subsystem designs that meet the mission's requirements. Furthermore, we assume that the propulsion subsystem is decoupled from the other subsystems, i.e., there is no interactions between them, and that the SE knows the types of each sSE and therefore, there is no information asymmetry.

To extract the parameters of the model, i.e., $a_{11}, \sigma_{11}, c_{11}$, we will use available historical data. To this end, let I_1 be the cumulative, sector-wide investment on the propulsion subsystem and G_1 be the delivered specific impulse of solid propellants (I_{sp}). The specific impulse is defined as the ratio of thrust to weight flow rate of the propellant and is a measure of energy content of the propellants [82].

Historical data, say $\mathcal{D}_1 = \{(I_{1,i}, G_{1,i})\}_{i=1}^{S}$, of these quantities are readily available for many technologies. Of course, cumulative investment and best performance increase with time, i.e., $I_{1,i} \leq I_{1,i+1}$ and $G_{1,i} \leq G_{1,i+1}$. We model the relationship between G_1 and I_1 as:

$$G_1 = G_{1,S} + A_1(I_1 - I_{1,S}) + \Sigma_1 \Xi_1, \tag{3.40}$$

where $G_{1,S}$ and $I_{1,S}$ are the current states of these variables, $\Xi_1 \sim \mathcal{N}(0,1)$, and A_1 and Σ_1 are parameters to be estimated from the all available data, $\mathcal{D}_1$. We use a maximum likelihood estimator for A_1 and Σ_1. This is equivalent to a least squares estimate for A_i:

$$\hat{A}_1 = \arg\min_{A_1} \sum_{i=1}^{S} \left[G_{1,S} + A_1(I_{1,i} - I_{1,S}) - G_{1,i}\right]^2, \tag{3.41}$$

and to setting Σ_1 equal to the mean residual square error:

$$\hat{\Sigma}_1 = \frac{1}{S} \sum_{i=1}^{S} \left[G_{1,S} + \hat{A}_1(I_{1,i} - I_{1,S}) - G_{1,i}\right]^2. \tag{3.42}$$

Now, let G_1^r be the required quality for the propulsion subsystem in physical units. The scaled quality of a subsystem Q_1, can be defined as:

$$Q_1 = \frac{G_1 - G_{1,S}}{G_1^r - G_{1,S}}, \tag{3.43}$$

with this definition, we get $Q_1 = 0$ for the state-of-the-art, and $Q_1 = 1$ for the requirement. Substituting Eq. (3.40) in Eq. (3.43) and using the maximum likelihood estimates for A_1 and Σ_1, we obtain:

$$Q_1 = \frac{\hat{A}_1}{G_1^r - G_{1,S}}(I_1 - I_{1,S}) + \frac{\hat{\Sigma}_1}{G_1^r - G_{1,S}} \Xi_1. \tag{3.44}$$

From this equation, we can identify the uncertainty σ_{11} in the quality function as:

$$\sigma_{11} = \frac{\hat{\Sigma}_1}{G_1^r - G_{1,S}}. \tag{3.45}$$

Finally, we need to define effort. Let T_1 represents the time for which the propulsion engineer is to be hired. The cost of the agent per unit time is C_1. T_1 is just

the duration of the systems engineering process we consider. The value C_1T_1 can be read from the balance sheets of publicly traded firms related to the technology. We can associate the effort variable e_1 with the additional investment required to buy the time of one engineer:

$$e_1 = \frac{I_1 - I_{1,S}}{C_1T_1}, \tag{3.46}$$

that is, $e_1 = 1$ corresponds to the effort of one engineer for time T_1. Let us assume there are Z engineers work on the subsystem. Comparing this equation, Eq. (3.44), and Eq. (3.5), we get that the κ_{11} coefficient is given by:

$$\kappa_{11} = \frac{ZC_1T_1\hat{A}_1}{G_1^r - G_{1,S}}. \tag{3.47}$$

To complete the picture, we need to talk about the value V_0 (in USD) of the system if the requirements are met. We can use this value to normalize all dollar quantities. That is, we set:

$$v_0 = 1, \tag{3.48}$$

and for the cost per square effort of the agent we set:

$$c_{11} = \frac{ZC_1T_1}{V_0}. \tag{3.49}$$

Finally, we use some real data to fix some of the parameters. Trends in delivered I_{sp} (G_1 (sec.)) and investments by NASA (I_1 (millions USD)) in chemical propulsion technology with time are obtained from [83] and [84], respectively. The state-of-the-art solid propellant technology corresponds to a $G_{1,S}$ value of 252 sec. and $I_{1,S}$ value of 149.1 million USD. The maximum likelihood fit of the parameters results in a regression coefficient of $\hat{A}_1 = 0.0133$ sec. per million USD, and standard deviation $\hat{\Sigma}_1 = 0.12$ sec. The corresponding data and the maximum likelihood fit are illustrated in Fig. 3.7. The value of C_1 is the median salary (per time) of a propulsion engineer which is approximately 120,000 USD / year, according to the data obtained from [85]. For simplicity, also assume that $T_1 = 1$ year. Moreover, we assume that there are 200 engineers work on the subsystem, $Z = 200$. We will examine two case studies which is summarized in table 3.8.

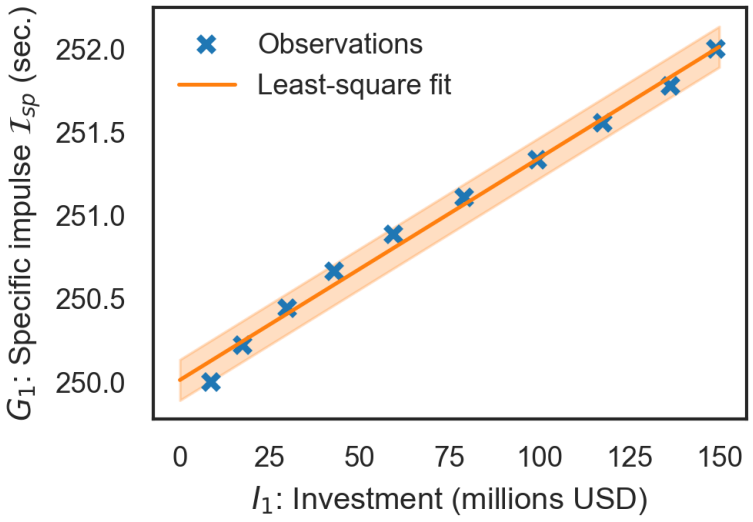
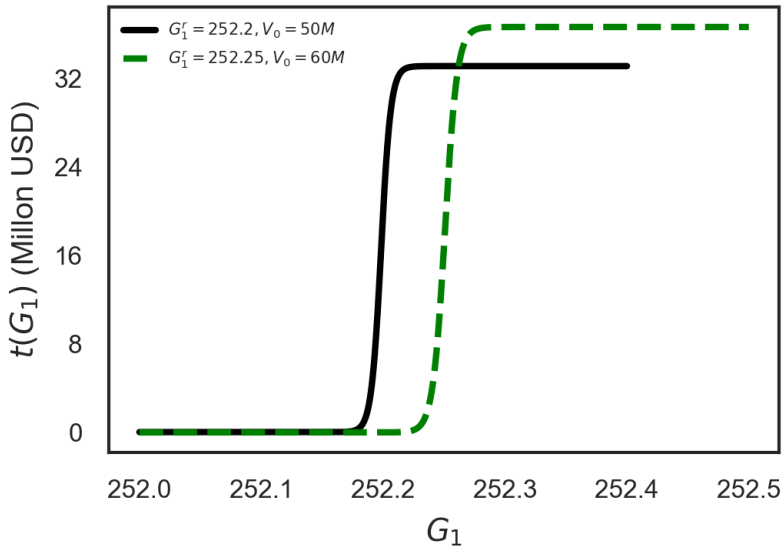


Fig. 3.7. Satellite case study (propulsion subsystem): Historical data (1979–1988) of specific impulse of solid mono-propellants vs cumulative investment per firm. The solid line and the shaded area correspond to the maximum likelihood fit of a linear regression model and the corresponding 95% prediction intervals, respectively.

Table 3.8.
The model parameters for two case studies.

G_1^r	V_0	κ_{11}	σ_{11}	c_{11}
252.2 s	50,000,000 USD	1.6	0.6	0.5
252.25 s	60,000,000 USD	1.28	0.48	0.4

Using RB value function, we depict the contracts for these two scenarios in Fig. 3.8.



(a)

Fig. 3.8. The contracts for two case studies in satellite design.

3.4 Conclusions

In this chapter, we developed a game-theoretic model for a one-shot shallow SEP. We posed and solved the problem of identifying the contract (transfer function) that maximizes the principal's expected utility. Our results show that, the optimum passed-down requirement is different from the real system requirement. For the same level of task complexity and uncertainty, as the agent cost of effort increases, the passed-down requirement decreases and the award to achieving the requirement increases. In this way, the principal makes the contract more attractive to the high-cost agent and ensures that the participation constraint is satisfied. Similarly, for the same level of task uncertainty and cost of effort, increasing task complexity results in lower passed-down requirement and larger award for achieving the requirement. For the same level of task complexity and cost of effort, as the uncertainty increases both the passed-down requirement and the award for achieving the requirement increase. This is because the principal wants to make sure that the system requirements

are achieved. Moreover, by increasing the task complexity, the task uncertainty, or the cost of effort, the principal earns less and the exceedance curve is shifted to the left. Using the RPI contracts, the principal pays smaller amount for achieving the requirement but, instead, they pay for per quality exceeding the requirement.

For the adverse selection scenario with RB value function, we observe that when the principal is maximally uncertain about the cost of the agent, the optimum contracts are equivalent to the contract designed for the high cost agent in the single-type case with no adverse selection. The low-cost agent earns more expected utility than the high-cost agent. This is the information rent that the principal must pay to reveal the agents' types. Similarly, if the principal is maximally uncertain about the task complexity, the two optimum contracts for the unknown quality are equivalent to the contract that is offered to the high-complexity task where there is no adverse selection. Note that, the equivalence of the contracts in adverse selection scenario with the contract that is offered in absence of adverse selection is not universal. If the class of possible contracts is enlarged, e.g., to allow penalties, there may be a set of two contracts that differentiate types.

There are still many remaining questions in modeling SEPs using a game-theoretic approach. First, there is a need to study the hierarchical nature of SEPs with potentially coupled subsystems. Second, true SEPs are dynamic in nature with many iterations corresponding to exchange of information between the various agents. These are the topics of ongoing research towards a theoretical foundation of systems engineering design that accounts for human behavior.

4. USING DEEP REINFORCEMENT LEARNING TO MODEL THE SYSTEM ACQUISITION PROCESS

4.1 Introduction

As we discussed in previous chapters, the acquisition of large-scale complex systems, usually suffers from cost and schedule overruns [86]. To investigate the causes of this problem, we may view the acquisition of a complex system in several different time scales [87]. At the largest time scale one considers the acquisition process as series of actions which are, request for bids, bidding and auctioning, contracting, and finally building and deploying the system, without resolving the fine details that occur within each step. At finer time scales, one may study different stages of the acquisition process from the intricate details of the entire systems engineering process [88, 89] to communication between design teams [90] to how individual designers solve problems [91, 92]. Here, we focus on the largest time scale, i.e., at the entire acquisition process.

Whenever the need arises for the government to acquire a large complex system, e.g., a hospital ship or a communication system, it follows several steps to achieve its goal. In this work, we use the procedure followed by DARPA (Defense Advanced Research Projects Agency) [93] as an archetype of such a process. First, the government publishes a request for bids which describes the type of the system that it wishes to acquire along with certain requirements. Second, in an auction, several private firms make offers and one or more are selected. Third, the government and the winner bidders enter a contract. The contract may be conducted in several phases with certain evaluation criteria and payments. Finally, if all the steps of design and test fulfill the requirements, the system is delivered to the government and deployed.

We model this procedure as a game that is played between several parties with different interests. One actor is the government that seeks to acquire a system with certain requirements. The other actors are the private firms that seek to generate revenue by participating, winning and delivering a successful system. We investigate which strategies the actors should follow to maximize their expected utilities by employing multi-agent reinforcement learning (RL) [25]. RL models each party as an intelligent agent, the goal of which is to learn how to play the game so as to maximize a predefined expected stream of rewards.

There are two categories of RL algorithms: value-based and policy gradient methods. In value-based methods, such as Q-learning, one attempts to estimate the optimal value-to-go as a function of the state-action pair and at each step the agent's policy function chooses the action that has the maximum value. In policy gradient methods, the agent directly learns the policy function. The latter is better suited for situations in which the action space is large as well as for games with mixed optimal strategies, e.g., the rock-paper-scissors game. In large or continuous state spaces, RL algorithms suffer from the curse of dimensionality. To overcome this difficulty, one approximates the values or policy functions using deep neural networks, an approach called deep reinforcement learning (DRL). Deep Q-learning [94], DRL with double Q-learning [95], and dueling deep Q-learning [96] are some algorithms that use deep neural networks to approximate the Q values in Q-learning. Advantage actor critic (A2C) and asynchronous advantage actor critic (A3C) methods [97], are some policy gradient methods that use two neural networks, actor network and critic network. Actor is the policy that predicts the next action and critic is used to criticize the value of the action and guides the optimization direction of the actor. Deep deterministic policy gradient (DDPG) [98], is a hybrid method of Q-learning and policy gradient that is used for continuous action space. In recent years, DRL has achieved great success in very challenging games, e.g., it can learn to play Go better than the best human [99].

In a single-agent RL setting, the state of the environment changes only because of the actions of the agent and the agent's decision only depends on the response of the environment. In multi-agent settings, the actions of each agent affect all other agents' decisions and the environment's response is subjected to all of the actions. The emerging behavior of the agents may be cooperative [100] or competitive [101]. An example of emerging multi-agent cooperation is self-driving cars in crowded situations [102]. In contrast, bidding in an auction to acquire a certain good or service, such as real time bidding for online advertisements [103, 104], results in emerging competitive behavior. Moreover, the competitive multi-agent setting may be used to study the offensive-defensive systems such as cyber-security [105, 106], to predict any malware attacks and plan properly for a cyber-defense. Finally, multi-agent games, can further be categorized into static games where all agents make their decisions simultaneously and observe the result of their actions, and dynamic games where the game evolves in time.

The government acquisition process is a multi-agent, dynamic game which can have a mixture of competitive and cooperative behaviors. For example, it is desirable that the bidders compete with each other during the auction, but the winner and the government may cooperate, each for their own interest, to build a successful system. However, in poorly designed acquisition processes, the bidders may cooperate to increase the price and then compete with the government. As we mentioned earlier, the acquisition process we wish to model consists of an auction and a contract phase. There are several different possibilities for the auction ranging from one-shot first-price and second-price auctions to auctions that also consider the quality of the system to be delivered. For simplicity, we assume that the government awards the contract to the lowest bidder. Similarly, we assume that the contract phase is a single-phase of "evaluation and payment" and we study the effect of contract types such as cost-plus, cost-plus-incentive, cost-not-covered, and cost-not-covered-incentive (a contract that does not cover the cost but pay incentives for exceeding the requirement) on the final outcome of the system. The quality of the deliverable system is correlated to the

amount of effort that the firm devotes on designing and building it. One may think of effort as the amount of investment, time, trials, etc., that needs to be assigned on a certain task. The firm chooses this effort level by considering its potential gains and incentives as captured by the contract. Finally, we investigate the effects of the number of bidders and problem complexity on the game outcome.

The structure of this chapter is as follows. In Sec. 4.2, we explain the methodology and define the parameters that are needed to describe the acquisition game mathematically. In Sec. 4.3, we conduct several numerical experiments to study the outcome of the game. Finally, in Sec. 4.4, we present our conclusions.

4.2 Methodology

4.2.1 Definitions and notations

We idealize the government acquisition process as illustrated in Fig. 4.1. At the

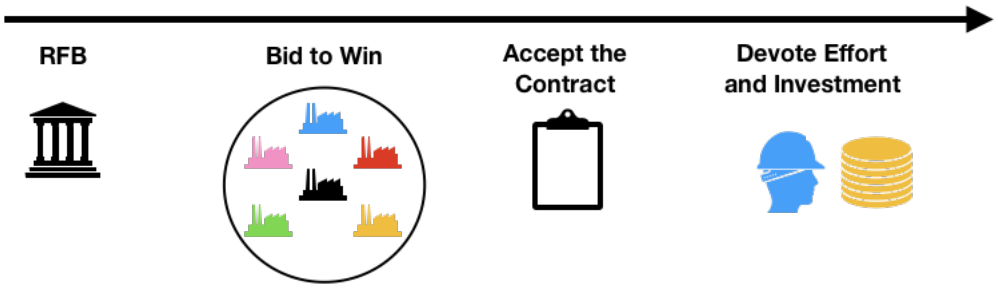


Fig. 4.1. The timeline of acquisition process executed by the government.

very first step of this idealized process, the government puts out a "request for bids (RFB)." The RFB describes the type of system the government wants to acquire. In this study, we restrict our attention to two possible types of system: simple and complex. The second step is the auction. The firms select their bids and the lowest bidder wins. In the third step, the government offers a contract that specifies the requirements of the deliverable system with corresponding deadlines and payments,

and the winner firm either accepts or rejects the contract. In the fourth and final step, the winner firm chooses how much effort to devote to the task and delivers the system. We call this final step, "build-the-system" phase.

Let us refer to the government as principal and show it with agent 0 and the firms as agents $i = 1, \ldots, N$, N being the total number of firms. Let θ^i denote the type of the agent i. The type of an agent specifies their "cost of effort." The cost parameter is defined as the monetary value of the effort that the agent devotes on the assigned task. In general, the cost is an increasing function of effort and for simplicity we assume that it is linear,

$$c^i(e) = C^i e. \tag{4.1}$$

Note that, the the cost of an agent also encompasses their experience level. For instance, a "high-experience" agent, on average, achieves the same level of performance with lower effort level than that for a "low-experience" agent. Therefore, with two different levels of cost, C^i, we can also distinguish between a high-experience and low-experience agent. Therefore, we associate the type of an agent only with cost:

$$\theta^i \in \{\text{"low-cost"}, \text{"high-cost"}\}, \tag{4.2}$$

where C^i is smaller for "low-cost" agent than that of the "high-cost" agent. During the RFB step, the principal decides to begin the acquisition for a system with a type τ. To identify a system's type, we use a parameter we call "complexity." Here complexity captures how difficult it is to improve the quality of a system by devoting effort. We define the quality function to be a stochastic process that maps the effort to a measurable outcome of a system that we call quality. In our previous work [66,92], we argued that a design problem can be modeled as an optimization problem where the goal is to maximize some attribute function and that the designer's search strategy follows certain heuristics that resemble Bayesian global optimization (BGO). The complexity of the system is specified with the smoothness of the underlying attribute function that describes the design problem. Intuitively, finding the maximum of a wiggly function is more difficult than finding the maximum of a smooth function. We

assume that the design problem is a realization of a Gaussian process (GP). In Fig. 4.2, we show two random functions from a GP,

$$a(x) \sim GP(0, k),$$

where the covariance function k is:

$$k(x, x') = \exp\left\{-\frac{(x - x')^2}{2l^2}\right\},$$

which represents the underlying design problems. Finding the maximum of the function with lengthscale $l = 0.1$ is more difficult than finding the maximum of the function with $l = 0.5$. Having said that, we define the type of a system as:

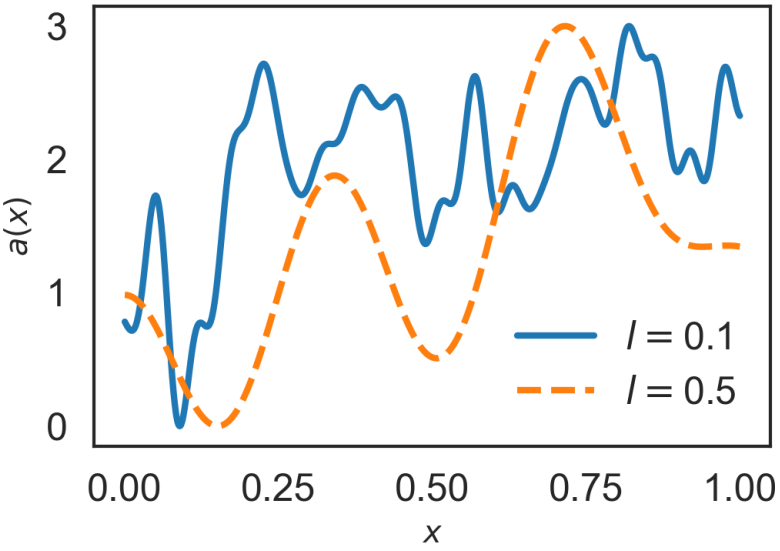


Fig. 4.2. Two random functions from GP, with two different lengthscales, $l = 0.1$ and $l = 0.5$. The function drawn from $l = 0.5$ is smoother and therefore, it is easier to find its maximum.

$$\tau \in \{\text{“low-complexity”}, \text{“high-complexity”}\}, \tag{4.3}$$

where the lengthscale l in the covariance function of the GP, is larger for the system with “low-complexity” than that for the “high-complexity”.

Now, let us focus on the quality function. Assume that, we draw some random functions from a GP with lengthscale $l = 0.01$ and we perform the BGO on them to obtain their maximums. In each iteration, the BGO searches in the parameter space based on a trade-off between exploration and exploitation [107]. Then, it observes the value of the function and makes a decision about the next query point in the parameter space. This process is repeated until convergence. Now, we assume that each iteration of this process resembles the designer's (or firm's) strategy in building the system where they systematically attempt to reach a goal which is maximizing certain attributes of the design problem. By systematic, we mean that the designer (or firm) follows some rules based on physics laws, experiments, prototyping, etc., in the design process to successfully build the system. In Fig. 4.3, we show some realizations of the stochastic process that maps the effort to the quality. The progress in the design process, i.e., making a certain attribute better and better, can follow any realization of the stochastic process that the nature chooses for us. For instance, consider two scenarios. First, let us assume that the design progress will be the green realization in Fig. 4.3. Then, after performing 20 units of effort the quality (attribute of the system) reaches 2 units. In the second scenario, let us assume that the design progress will follow the blue line where the quality of the system after 20 units of effort reaches 3 units. Note that, the quality unit is context-dependent and here, we normalize it such that $q \in [0, 3]$.

In the bidding stage, each agent i either offers a bidding price b^i or decides not to participate. To keep the notation simple, assume that when the agent does not want to participate, they simply set b^i to the special value 'NP' (No Participation). The principal selects the lowest bid, among those who want to participate, as the winner W:

$$W = \arg \min_{1 \leq i \leq N, b^i \neq \text{'NP'}} b^i. \tag{4.4}$$

We denote all bidders who lose (or do not participate) in the bidding process with L.

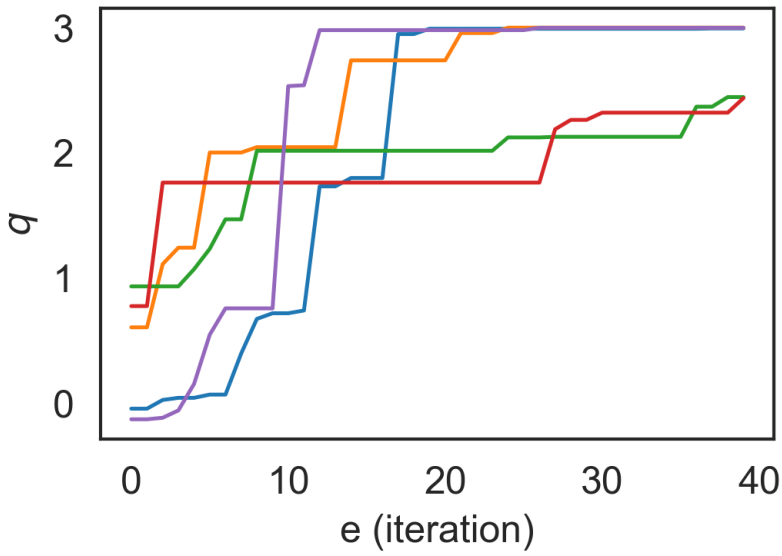


Fig. 4.3. Some realizations of the quality function. The design process can be any realization of a stochastic process that maps the effort to the quality (attribute of the system).

Next, the principal offers a predefined contract, μ. The contract is a function of the winner's bid b^W, delivered system's quality q, and, self-reported cost, $c^W(e)$. We parameterize μ as:

$$\begin{aligned}\mu\left(b^W, q, c^W(e)\right) &= \alpha b^W \\ &+ (1-\alpha) b^W H(q - \mathrm{q_{min}}) \\ &+ \psi\left(q, c^W(e)\right),\end{aligned} \tag{4.5}$$

where $\alpha \in [0,1]$ is the portion of the winner's bid that is paid to the winner prior to starting the task. $H(\cdot)$ is the Heaviside function and $\mathrm{q_{min}}$ is the requested quality by the principal. The second term of the right hand-side simply says that the principal pays $(1-\alpha)b^W$ if the delivered system quality q exceeds the requirement $\mathrm{q_{min}}$. The $\psi(\cdot)$ function depends on the contract type such as "cost-plus", "cost-not-covered", "incentive-cost-plus", and "incentive-cost-not-covered." Using this contract the agent devotes effort, e. As we explained above, the quality is a stochastic function of the

effort of the agent and therefore there is uncertainty about achieving the q_{min}. In the following, we define the function $\psi(\cdot)$ for 4 different types of the contracts:

1. cost-plus (CP):

$$\psi(q, c^W(e)) = c^W(e), \tag{4.6}$$

2. cost-not-covered (CNC):

$$\psi(q, c^W(e)) = 0, \tag{4.7}$$

3. incentive-cost-plus (ICP):

$$\psi(q, c^W(e)) = h\left(q - q_{min}\right) + c^W(e), \tag{4.8}$$

4. incentive-cost-not-covered (ICNC):

$$\psi(q, c^W(e)) = h\left(q - q_{min}\right), \tag{4.9}$$

where $h(\cdot)$ is a function that maps the exceeding quality to a monetary payment. We define $h(x) = xH\left(x\right)$.

Having defined the quality function, cost of effort, agent types, system types, and contract types, our goal is to study the bidding strategy of the agents, the effects of the contract type, problem difficulty, number of agents, and agents' costs. In the following section we explain how we map the described scenario above to the reinforcement learning setting.

4.2.2 Reinforcement Learning Setting of the APG

In this section we describe how to map the game described in Sec. 4.2.1 to the reinforcement learning setting. The goal of a reinforcement learning agent is to learn an optimal policy that maximizes the expected return. Let us call the setting

illustrated in Fig. 4.1, the "**A**cquisition **P**rocess **G**ame" (APG). The actors set in the APG setting include, principal (the government) and the agents (private firms) who bid. Let us define the Markov decision process [25] (MDP) as the following tuple:

$$\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle, \tag{4.10}$$

where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{P}$ is the state transition probability function, $\mathcal{R}$ is the set of rewards, and γ is the discount factor. Our goal is to obtain the optimal policies of the agents that maximize their expected return. In the reinforcement learning setting, we assume that one episode of the game is played T times. In other words, there are T acquisition processes that the government performs and our goal is to obtain the optimum strategies of the bidders. In the following, we describe the states, actions, and rewards. In each episode, the state at time t is shown by $S_t \in \mathcal{S}$ where,

$$S_t = (S_t^0, S_t^1, \ldots, S_t^N), \tag{4.11}$$

and S_t^i is the state of the agent i at time t. Similarly, the action that the agent i takes at time t and gains a reward, is A_t^i and R_t^i, respectively, where $A_t^i \in \mathcal{A}$ and $R_t^i \in \mathcal{R}$. The policy (decision-making rule) of the agent i is:

$$\pi^i(A_t^i | S_t^i), \tag{4.12}$$

which is the probability distribution over actions given states. The policies are time independent and

$$A_t^i \sim \pi^i(.|S_t^i) \quad \forall t > 0. \tag{4.13}$$

The return of the agent i in step t of an episode of the game is defined as:

$$G_t^i = \sum_{k=0}^{T} \gamma^k R_{t+k+1}^i. \tag{4.14}$$

Note that, the discount factor, γ, controls how important the future rewards are compared to the immediate reward of an action. By choosing an appropriate γ, we can make the immediate and future rewards equally effective for the current action.

The goal of the agent is to find the Bayesian Nash equilibrium policy, i.e., the policy $\pi^{*i}(.|.)$ that solves:

$$\pi^{*i}(.|.) = \arg\max_{\pi^i(\cdot|\cdot)} \mathbb{E}_{\pi^{*,-i}}[G_t^i], \tag{4.15}$$

where $\pi^{*,-i}$ are the optimal policies of everyone except agent i.

Having introduced these parameters, let us find the equivalent parameters of the reinforcement learning in the APG setting. To this end, we describe how one step of the game is played. In time step t, the agents $i = 1, \cdots, N$ reveal their bid amounts, b_t^i, based on their previous bid experience, b_{t-1}^i, and the signal of the winning or losing in the previous time step $\sigma_{t-1}^i \in \{0, 1\}$, where 0 denotes the losing and 1 denotes the winning. Therefore, for the agents $i = 1, \cdots, N$ the state S_t^i is:

$$S_t^i = (b_{t-1}^i, \sigma_{t-1}^i), \tag{4.16}$$

and the action is:

$$A_t^i = \begin{cases} b_t^i & \text{if participates} \\ \text{“no-participation”} & \text{otherwise,} \end{cases} \tag{4.17}$$

Now, let us define the state and action of the principal (agent 0). In each time step the principal selects the minimum bid. Therefore, the state of the principal is the received bids:

$$S_t^0 = (b_t^1, \ldots, b_t^N), \tag{4.18}$$

and its action which is selecting the winner, is deterministic:

$$A_t^0 = \arg\min_i \{b_t^i\}_{i=1}^N. \tag{4.19}$$

Finally, the winner agent selects the effort level (number of trials in the BGO) that they need to devote in order to achieve the $q_{\min}$ prior to starting their design efforts. In practice, this is equivalent to allocating the budget to the projects before starting the task.

In the standard reinforcement learning setting, the agents always have the same state space and action space as the game evolves. However, in the APG setting, the

agents experience different state and action spaces. For instance, the agents bid first where the state and action space are shown in Eqs. 4.16 and 4.17, respectively. Then the winner agent decides how much effort to devote as the action. In other words, in APG, we deal with a "changing action set." To build an agent that performs different sets of actions in different stages of the game, we need the concept of a "helper" agent to relate the reward of its action to the reward of the winner agent. To do so, we build a reinforcement learning agent that selects the effort in the "build-the-system" stage and we bind its reward with the reward of the winner agent in the bidding stage. Therefore, the action of the agent in the bidding stage and selecting the effort level will be tied up. In this way, the agents will confront their actions' consequences in the bidding stage later in the "build-the-system" phase. Let us describe the state and the action of the helper agent in the "build-the-system" phase. The state is always the constant tuple of:

$$S_t^H = \left(\mathrm{q}_{\min}, C^W\right), \tag{4.20}$$

and the action is,

$$A_t^H = e_t. \tag{4.21}$$

The reward process of the game is as follows. The principal's reward is:

$$R_t^0 = V(q_t; \mathrm{q}_{\min}) - \mu(b_t^W, q_t, c^W(e_t)), \tag{4.22}$$

where $V(q_t; \mathrm{q}_{\min})$ is the value of the system for the principal. We will assume two types of value functions, requirement-based (RB) and requirement-plus-incentive (RPI) which are defined as,

$$V_{RB}(q; q_{min}) = 10H(q - q_{min}), \tag{4.23}$$

and,

$$V_{RPI}(q; q_{min}) = V_{RB} + (q - q_{min})\, H(q - q_{min}), \tag{4.24}$$

respectively. The loser agents' reward:

$$R_t^L = 0, \tag{4.25}$$

and the winner agent's reward is:

$$R_t^W = \begin{cases} \alpha b_t^W & \text{reward of winning} \\ -c^w(e_t) - V_{rep} & \text{if } q_t < q_{min} \\ (1-\alpha) b_t^W + \psi(q_t, c^W(e_t)) & \text{if } q_t \geq q_{min}, \end{cases} \tag{4.26}$$

where V_{rep} is the monetary value of the reputation. Note the reward of the winner agent in Eq. 4.26 has several temporal stages that correspond to the stages of the contract defined in Eq. 4.5. The winner agent gains α portion of the bid immediately after winning and before starting the "build-the-system" stage. Then, the second part of the reward is contingent on the successful building and deployment of the system. If the winner agent fails to fulfil the requirement of the system, it will lose its cost of effort as well as the monetary value of their reputation, V_{rep}. On the other hand, if the winner agent is successful in meeting the requirements, they will receive the remaining part of the contract in Eq. 4.5. We define the reward for the helper agent as:

$$R_t^H = \begin{cases} -V_{rep} & \text{if } q_t < \mathrm{q_{min}} \\ -c^W(e_t) + \psi(q_t, c^W(e_t)) & \text{if } q_t \geq \mathrm{q_{min}}\,. \end{cases} \tag{4.27}$$

To identify the optimal behavior of the agents, we use the advantage actor critic (A2C) method. We summarize the APG in the Alg. 1.

4.2.3 Advantage actor critic (A2C)

There are generally two different ways of solving the MDP problem that arises in reinforcement learning. The first way is known as value based methods. In these methods, the agent learns the value of each action in each state or a parameterized function that maps the states to the value of the actions. Then, the policy is determined by selecting the action that corresponds to the maximum value. The Q-learning algorithm is one of the value based methods. In the second category, the agent directly learns the parametrized policy, in Eq. 4.12. In this method we still use

Algorithm 1: APG pseudo-code

Input: Agents' types, contract types, number of agents, and system type;

Output: The agents' bids and efforts;

Initialize A2C for the agents $1, \cdots, N$ and helper agent, H;

for *episode* $= 1$ **to** E **do**

  Initialize a trajectory list for each agent;

  for $t = 1$ **to** T **do**

    Sample a random function from GP;

    For each agent $i = 1, \cdots, N$ compute A_t^i;

    Agent 0 selects the agent with minimum bid;

    For the helper agent compute S_t^H;

    For the helper agent compute A_t^H;

    For all the agents compute the rewards R_{t+1}^i;

    For all agents, push (s, a, r, s') into the trajectory list;

    For all agents, $S_{t-1}^i \leftarrow S_t^i$;

  Update the A2C parameters (Sec. 4.2.3);

a value function to learn the parameters of the policy function. But, for action selection we sample from the learned policy. This method is useful when the action space is large or continuous. The A2C is an example of the policy gradient methods which is used here. Let us explain this algorithm in detail. To start, we first describe the policy gradient method and then we will extend it to the A2C. Let us parameterize the policy function as:

$$g(A|S;\beta),$$

furthermore, assume one episode of the game produces the trajectory:

$$\mathcal{T} = S_0, A_0, S_1, A_1, S_2, A_2, \ldots, A_{T-1}, S_T,$$

the total return from this trajectory is:

$$G(\mathcal{T}) = R_1 + \gamma R_2 + \gamma^2 R_3 + \ldots,$$

note how the choice of β affects the probability of the trajectory $\mathcal{T}$, and therefore the return, G. Let us show the probability of the occurrence of the trajectory $\mathcal{T}$ by

$$\mathcal{T} \sim P(\mathcal{T};\beta),$$

and therefore the expected return is:

$$J(\beta) = \sum_{\mathcal{T}} P(\mathcal{T};\beta)G(\mathcal{T}), \tag{4.28}$$

now we can define the optimal policy as

$$g^*(.;\beta^*),$$

where,

$$\beta^* = \arg\max_{\beta} J(\beta).$$

To maximize the $J(\cdot)$ function, we use gradient descent and therefore we need to calculate the gradient of the expected return,

$$\begin{aligned}
\nabla_\beta J(\beta) &= \sum_{\mathcal{T}} \nabla_\beta P(\mathcal{T};\beta)G(\mathcal{T}) \\
&= \sum_{\mathcal{T}} P(\mathcal{T};\theta)\nabla_\beta \log P(\mathcal{T};\beta)G(\mathcal{T}) \\
&= \mathbb{E}_{\mathcal{T}\sim P(\mathcal{T};\beta)}[\nabla_\beta \log P(\mathcal{T};\beta)G(\mathcal{T})],
\end{aligned} \tag{4.29}$$

where,

$$\log P(\mathcal{T};\beta) = \log P(S_0) + \sum_t \log g(A_t|S_t;\beta) + \sum_t \log P(S_{t+1}|S_t),$$

and therefore,

$$\begin{aligned} \nabla_\beta J(\beta) &= \mathbb{E}_{\mathcal{T}\sim P(\mathcal{T};\beta)} \left[\sum_t \nabla_\beta \log g(A_t|S_t;\beta) G\left(t\right) \right] \\ &\approx \frac{1}{T} \sum_t \nabla_\beta \log g(A_t|S_t;\beta) G\left(t\right), \end{aligned} \tag{4.30}$$

where T is the total number of steps in each episode of the game. Therefore, learning the policy function is as follows:

1. In each episode, compute the return G_t.
2. Compute the gradient of log policy.
3. Update the parameter β as, $\beta \leftarrow \beta + \eta \nabla_\theta J(\theta)$, where η is the learning rate.

From Eq. 4.30, the maximization of $J(\beta)$ increases the probability of trajectories with higher return. This makes the algorithm unstable. Because, β converges to the values that maximizes the $g(.)$ for observed trajectories and unseen trajectories which are sample of $g(.)$, will have lower probability to occur. Moreover, in Eq. 4.30, we can replace the $G(t)$ with

$$G(t) \leftarrow G(t) - \mathbb{E}[G(t)] = G(t) - v(s) = a_t,$$

where $v(s)$ is the value of the current state, s,

$$v(s) = \mathbb{E}_\pi[G_t|S_t = s] \tag{4.31}$$

In this way, we can think of Eq. 4.30 as how much advantage, a_t, we gain by following the policy $g(\cdot)$ with respect to the current baseline, $(v(s))$. The actor-critic algorithm

overcome the stability problem. In this method two networks are used. One network is $g(A, S; \beta_a) = \pi(A|S)$ which is called actor, and one network for value function, $k(S; \beta_c) = v(S)$, which is called critic. To calculate the return $G(t)$, the method bootstraps and use N-step of future reward,

$$G(t) = \sum_{k=0}^{K-1} \gamma^k R_{t+k+1} + \gamma^N v(S_{t+N}), \tag{4.32}$$

we use $N = 1$. In Alg. 2, we show the pseudo-code for A2C algorithm.

Algorithm 2: A2C pseudo-code

Input: Initialize β_a and β_c with random values;

Output: Updated policy and critic parameters, β_a and β_c;

for *episode* $= 1$ **to** E **do**

  Initialize a trajectory list for each agent;

  for $t = 0$ **to** T **do**

    Generate S_t, A_t using policy $g(A|S; \theta_a)$ and observe the R_{t+1};

  Compute the returns using Eq. 4.32;

  Compute the advantage, $a_t = G(S_t) - k(S_t; \theta_c)$;

  Compute two losses $L_a(\beta_a) = -\frac{1}{T}\sum_t \log g(A; \beta_a) a_t$, and $L_c(\beta_c) = \frac{1}{T}\sum_t a_t^2$;

  Update the networks parameters:

  $\beta_a \leftarrow \beta_a - \eta_a \nabla_{\beta_a} L_a(\beta_a)$

  $\beta_c \leftarrow \beta_c - \eta_c \nabla_{\beta_c} L_c(\beta_c)$;

4.3 Results

In this section we perform an exhaustive numerical investigation of the effect of several parameters on the outcome of the APG. We will specifically focus on agents' bids (b^i), the effort level of the winner bidder (e), the utility (gain) of the winner agent and the principal which are shown with u^W, u^0, respectively. We study the effects of number of participants in the bidding, the agents' costs, contract types, and

the market value of the system. We assume two types of value functions, RB and RPI.

Note that, the APG is played continuously between the principal and agents. In order to analyze the game we assume that there are T acquisition processes for a given complexity and we select, $T = 5$. We may assume same importance to the immediate and future rewards. We reach this by setting the discount factor $\gamma = 1$ for all the RL agents. We set the game parameters as follows:

1. The minimum market requirement for the principal is $r_{\min} = 2.0$.

2. In all the examples, we assume that the minimum requested quality is equivalent to the minimum market requirement, $q_{\min} = r_{\min}$.

3. If the agents decide to participate, the bid amounts are:

$$b^i \in \{1.0 + 0.2k : 0 \leq k \leq 20\}.$$

   and the effort levels are:

$$e \in \{2 + 2k : 0 \leq k \leq 19\}$$

4. We assign $l = 0.05$ and $l = 0.01$ to the systems with "low-complexity" and "high-complexity", respectively.

5. We categorize the agents' types to 2 levels. $C^i = 0.05, 0.15$ which correspond to "low-cost" and "high-cost" agents, respectively.

6. In the examples with "cost-plus" contracts we assume $c_p = 3$.

7. In all the examples, we assume that $\alpha = 0.2$, i.e., 20% of the bid is paid before the start of the project and 80% of the bid is paid after the successful delivery of the system.

8. In all the examples, we assume $V_{rep} = 30.0$.

4.3.1 The effect of the agent types and number of agents in APG

In this section we investigate the effects of the parameters related to the agents. In all the tables, u^0 and u^W denote the utility (monetary gain) of the principal and winner agent, respectively.

Agents' types

In this section we study the effect of the agents' cost on the outcome of APG. We assume that there are two agents bidding for a system acquisition with "low-complexity" ($l = 0.05$). We assume the contract is "Cost-Not-Covered" and the system has a RB market value function. In Table 4.1, we show the results for following cost scenarios:

1. low-low ($C^1 = C^2 = 0.05$).
2. high-high ($C^1 = C^2 = 0.15$).
3. low-high ($C^1 = 0.05, C^2 = 0.15$).

Table 4.1.
Summary of the observations for different agents' type.

Costs	b^i	e	u^W	u^0
low-low	1.8-1.8	16	1.0	8.2
high-high	3.0-3.0	16	0.6	7.0
low-high	3.0-3.0	14	2.3-0.9	7.0

Number of agents

Here, we study how the number of agents affects the outcome of the APG. First, we assume that there is only one participant in the bidding (no competition). Second, we run the game with 4 low-cost agents and third, we consider the scenario with 2 low-cost and 2 high-cost agents. We show the results in Table 4.2.

Table 4.2.
Summary of the observations for different number of participants in the bidding.

Costs	b^i	e	u^W	u^0
low	5.0	20	4.0	5.0
low-low-low-low	1.2-1.2-1.2-1.2	16	0.4	8.8
low-low-high-high	1.8-1.8-2.6-3.0	14	1.1	8.2

We make the following observations:

- From Table 4.1, if at least one of the agents is high-cost, the principal must pay a higher amount in the bidding than that of the two low-cost agents scenario.
- From Table 4.2, we can clearly see the effect of monopoly and how the principal may gain utility from the competition.
- As the number of low-cost bidders increases, the amount of bid decreases. In a single agent auction, the agent's bid is the maximum possible amount. In case of larger number of bidders there is a larger probability that one of the agents will bid a very low amount. Therefore, the other agents need to reduce their bid amount in order to get a greater chance of winning. An other interpretation of this phenomenon is the labor demand and supply. As the number of laborers grows the wages decrease.

- If there are at least two low-cost agents, they keep their bids low to win the bidding. Therefore, if there are more than one low-cost agent, it is sufficient to evaluate the outcome of the APG with the low-cost agents.

4.3.2 The effect of contract type and system complexity in APG

In this section, we investigate the influence of contracts on the principal's utility. We consider 2 different cases: "low-complexity" ($l = 0.05$) and "high-complexity" ($l = 0.01$) systems. In each case, we assume several different contracts that shown in tables 4.3, 4.4, 4.5, and 4.6 for several scenarios. For all the "cost-plus" and "cost-not-covered" contracts we assume RB value functions. For the "incentive-cost-plus" and "incentive-cost-not-covered" we assume that the value function is RPI.

Table 4.3.
Summary of observations for different contract types for "low-complexity" system with RB value function.

μ/C_i	b^i	e	u^W	u^0
CP/low-low	1.8-1.8	18	1.8	7.3
CP/high-high	1.8-1.8	14	1.8	6.1
CNC/low-low	1.8-1.8	16	1.0	8.2
CNC/high-high	3.0-3.0	16	0.6	7.0

We make the following observations:

- In the "cost-plus" contracts, the agents bid lower than the "cost-not-covered" contracts. However, the principal gains a smaller amount in the "cost-plus" contracts due to the cost coverage. Intuitively, in "cost-plus" contracts the agents low ball in order to win the bid and they strategically misrepresent the costs.

Table 4.4.
Summary of observations for different contract types for "low-complexity" system with RPI value function.

μ/C_i	b^i	e	u^W	u^0
ICP/low-low	1.0-1.0	32	2.0	7.4
ICP/high-high	1.0-1.0	20	2.0	6.0
ICNC/low-low	1.6-1.6	16	1.8	8.4
ICNC/high-high	2.2-2.2	16	0.8	7.8

Table 4.5.
Summary of observations for different contract types for "high-complexity" system with RB value function. NP denotes "NO-PARTICIPATION."

μ/C_i	b^i	e	u^W	u^0
CP/low-low	2.0-2.0	40	2.0	6.0
CP/high-high	4.0-4.0	34	1.9	3.0
CNC/low-low	3.4-3.4	40	1.4	6.6
CNC/high-high	NP-NP	0	0.0	0.0

- For the "high-complexity" system type, the high-cost bidders do *not* participate in a "cost-not-covered" contract. This is because, this type of systems has a higher chance to fail and the agents do not accept such high risks. The principal in order to encourage the agents to participate, must offer a "cost-plus" contract.

- For the "low-complexity" system types with "cost-plus" contracts, the low and high-cost agents offer the same bids. For the "high-complexity" system types, as the agents need to devote more effort and the cost premium is limited, the high-cost agents bid a greater amount.

Table 4.6.
Summary of observations for different contract types for "high-complexity" system with RPI value function. NP denotes "NO-PARTICIPATION."

μ/C_i	b^i	e	u^W	u^0
ICP/low-low	1.2-1.2	40	2.1	6.8
ICP/high-high	3.4-3.4	36	1.85	3.6
ICNC/low-low	2.6-2.6	40	1.5	7.4
ICNC/high-high	NP-NP	0	0.0	0.0

4.3.3 Convergence plots

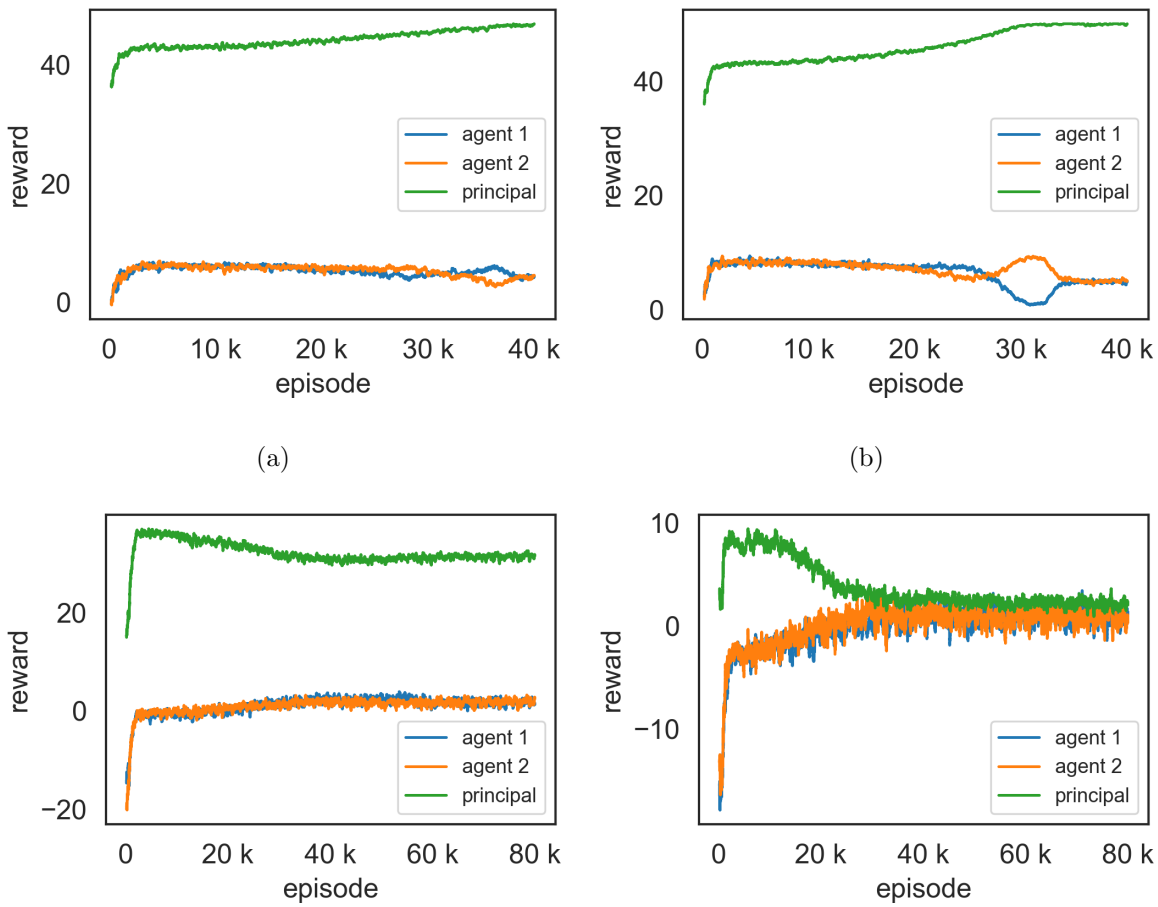
In this section we show the convergence plots for some of the settings described above. We show the rewards of the principal and agents, the convergence plots of the bids and finally the convergence plots of the effort of the winner agent in Figs. 4.4, 4.5, and 4.6, respectively, for 4 different problem settings. For these plots, we use a moving average with window size of 200.

Note that, the reward shown in Fig. 4.4 is the accumulated raw reward from $T = 5$ steps of each episode and it is not the final gains of the agents and principal.

One can see that the bid and effort levels converge.

4.4 Conclusions

In this chapter, we modeled the acquisition process of a complex system as a game that is played between the principal (government) and some agents (private firms). The game includes several stages including, bidding in an auction and "build-the-system." To model the latter stage, we assumed that the designing and building a system is equivalent to maximizing a function with a certain smoothness. Then, we assumed that each trial in building the system is equivalent to one iteration in the



(a) (b)

(c) (d)

Fig. 4.4. The reward of the agents and principal for (a): "low-complexity" system, RPI value function, "incentive-cost-not-covered" contract, two low-cost agents (b) "low-complexity" system, RPI value function, "incentive-cost-plus" contract, two low-cost agents (c) "high-complexity" system, RB value function, "cost-not-covered" contract, two low-cost agents (d) "high-complexity" system, RB value function, "cost-plus" contract, two high-cost agents.

BGO and in this way we resembled the cost of "build-the-system" stage to the cost of iterations in the BGO. We used deep reinforcement learning and in particular the A2C method to discover the optimal behavior of the agents in bidding and "build-the-system" phases. We introduced the concept of helper agent to address the problem of

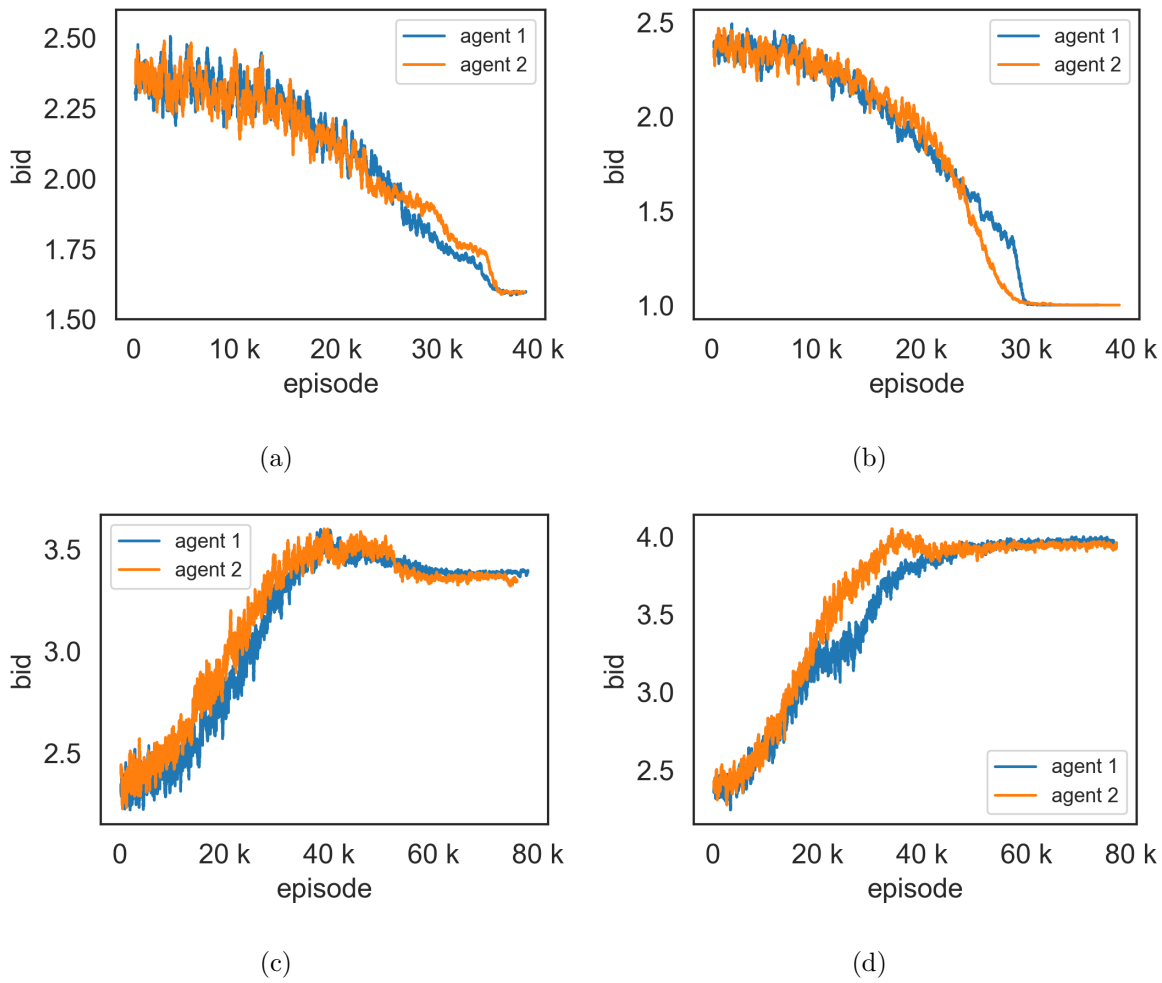


Fig. 4.5. The convergence plot of the bids of the agents and principal for (a): "low-complexity" system, RPI value function, "incentive-cost-not-covered" contract, two low-cost agents (b) "low-complexity" system, RPI value function, "incentive-cost-plus" contract, two low-cost agents (c) "high-complexity" system, RB value function, "cost-not-covered" contract, two low-cost agents (d) "high-complexity" system, RB value function, "cost-plus" contract, two high-cost agents.

"changing action set" in deep reinforcement learning. The action in the bidding stage is selecting the proper amount of the bid. In the "build-the-system" stage, the action is to select the proper amount of effort. We bound the reward of the helper agent which performs the action in "build-the-system" phase, to the reward of the winner

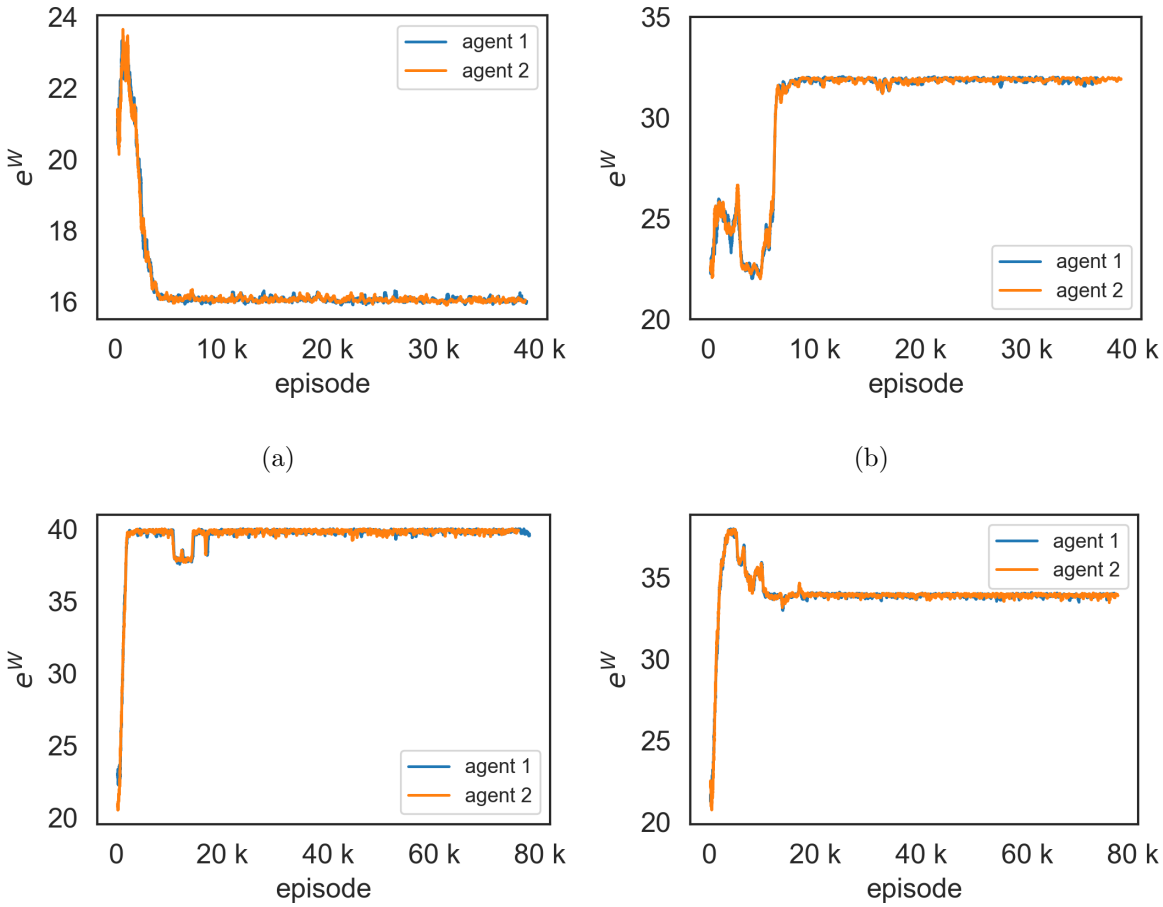


Fig. 4.6. The convergence plot of the effort level of the winner agents for (a): "low-complexity" system, RPI value function, "incentive-cost-not-covered" contract, two low-cost agents (b) "low-complexity" system, RPI value function, "incentive-cost-plus" contract, two low-cost agents (c) "high-complexity" system, RB value function, "cost-not-covered" contract, two low-cost agents (d) "high-complexity" system, RB value function, "cost-plus" contract, two high-cost agents.

agent. In this way, we built a reinforcement learning agent that selects actions with different natures. The optimal policies in the reinforcement learning highly depends on the reward setting in the game. Having set the reputation cost of the winner bidder, which is the penalty of not fulfilling the requirements, to 3 times of the market value,

we studied the effect of different parameters such as number of bidders, contract types, agent types, and problem complexity on the outcome of the APG. In summary our observations were as following. 1) As the number of lowest-cost participants in the bidding process grows, the winning bid becomes smaller and the government benefits from the supply of work phenomenon. 2) A single low-cost firm in the presence of some high-cost firms misrepresents the cost and therefore, offers a higher bid. 3) For the "low-complexity" systems the "cost-not-covered" contracts are more profitable for the government. 4) For the "high-complexity" systems with high-cost firms, the government must offer "cost-plus" contracts to encourage the bidders to participate in the system's acquisition. 5) In "cost-plus" contracts the bidders low ball in order to win the bidding. Therefore, we suggest that the principal avoid the "cost-plus" contracts unless it is inevitable.

The proposed model of the acquisition process has several limitations as it is just a simplified version of the real world procedure. There are several other important effects that we do not explicitly model, e.g., bureaucracy, the cost of auction for the government, the cost of bidding for the agents, the cost of validation and verification of the system, the frictions at the organization level that affect the success of the winner bidder in delivering the system, and the multi-step evaluation of progress which may contain go-no-go decisions. Despite the fact that the current model does not account for these details, it is in principle extendable to something more realistic and reinforcement learning can still be used to discover optimal policies.

Finally, in this work we have not validated our model against real data. Such a task is possible, albeit it requires considerable efforts that go beyond the scope of the present work. First, one would have to gather the necessary data including past projects, contracts, costs, and auction details; an effort that make require the use of the Freedom of Information Act. Second, one may develop an inverse reinforcement learning [108] technique to fit the unobserved part of the agent rewards and disutilities. The government may use these more sensible rewards to design better contracts.

REFERENCES

REFERENCES

[1] D. D. Walden, G. J. Roedler, K. Forsberg, R. D. Hamelin, and T. M. Shortell, Eds., *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, 4th ed. Hoboken, NJ: Wiley, 2015.

[2] Nasa, N. Aeronautics, and S. Administration, *NASA Systems Engineering Handbook*. U.S. Government Printing Office, 2008. [Online]. Available: https://books.google.com/books?id=_8YkuAAACAAJ

[3] J. Austin-Breneman, B. Y. Yu, and M. C. Yang, "Biased information passing between subsystems over time in complex system design," *Journal of Mechanical Design*, vol. 138, no. 1, p. 011101, 2016.

[4] B. Flyvbjerg, "Survival of the unfittest: why the worst infrastructure gets built—and what we can do about it," *Oxford review of economic policy*, vol. 25, no. 3, pp. 344–367, 2009.

[5] C. C. Cantarelli, B. Flybjerg, E. J. Molin, and B. Van Wee, "Cost overruns in large-scale transportation infrastructure projects: Explanations and their theoretical embeddedness," *arXiv preprint arXiv:1307.2176*, 2013.

[6] G. Locatelli, "Why are megaprojects, including nuclear power plants, delivered overbudget and late? reasons and remedies," *arXiv preprint arXiv:1802.07312*, 2018.

[7] U. S. G. A. Office, "Navy shipbuilding past performance provides valuable lessons for future investments," 2018.

[8] N. GAO, "Assessments of major projects," 2018.

[9] P. Collopy, "Adverse impact of extensive attribute requirements on the design of complex systems," in *7th AIAA ATIO Conf, 2nd CEIAT Int'l Conf on Innov and Integr in Aero Sciences, 17th LTA Systems Tech Conf; followed by 2nd TEOS Forum*, p. 7820.

[10] I. Maddox, P. Collopy, and P. A. Farrington, "Value-based assessment of dod acquisitions programs," *Procedia Computer Science*, vol. 16, pp. 1161 – 1169, 2013, 2013 Conference on Systems Engineering Research. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050913001233

[11] P. Collopy, "Economic-based distributed optimal design," in *AIAA Space 2001 Conference and Exposition*, p. 4675.

[12] A. Bandura, "Social cognitive theory: An agentic perspective," *Annual review of psychology*, vol. 52, no. 1, pp. 1–26, 2001.

[13] S. D. Vermillion and R. J. Malak, "Using a principal-agent model to investigate delegation in systems engineering," in *ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2015, pp. V01BT02A046–V01BT02A046.

[14] S. Safarkhani, V. R. Kattakuri, I. Bilionis, and J. Panchal, "A principal-agent model of systems engineering processes with application to satellite design," *arXiv preprint arXiv:1903.06979*, 2019.

[15] J.-J. Laffont and J. Tirole, *A theory of incentives in procurement and regulation*. MIT press, 1993.

[16] A. Lenjani, I. Bilionis, S. J. Dyke, C. M. Yeum, and R. Monteiro, "A resilience-based method for prioritizing post-event building inspections," *Natural Hazards*, vol. 100, no. 2, pp. 877–896, 2020.

[17] A. Lenjani, S. J. Dyke, I. Bilionis, C. M. Yeum, K. Kamiya, J. Choi, X. Liu, and A. G. Chowdhury, "Towards fully automated post-event data collection and analysis: Pre-event and post-event information fusion," *Engineering Structures*, p. 109884, 2020.

[18] D. Fudenberg and J. Tirole, *Game Theory*. MIT Press, 1991. [Online]. Available: https://books.google.com/books?id=pFPHKwXro3QC

[19] L. Hurwicz and S. Reiter, "Designing economic mechanisms," Cambridge University Press, Tech. Rep., 2008.

[20] V. Krishna, *Auction theory*. Academic press, 2009.

[21] L. C. Corchón, "The theory of contests: a survey," *Review of Economic Design*, vol. 11, no. 2, pp. 69–100, 2007.

[22] O. Hart, *Firms, contracts, and financial structure*. Clarendon press, 1995.

[23] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic game theory*. Cambridge university press, 2007.

[24] K. M. Eisenhardt, "Agency theory: An assessment and review," *Academy of management review*, vol. 14, no. 1, pp. 57–74, 1989.

[25] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge, MA, USA: MIT Press, 2018.

[26] S. Kapurch, *NASA Systems Engineering Handbook*. DIANE Publishing Company, 2010. [Online]. Available: https://books.google.com/books?id=2CDrawe5AvEC

[27] I. Maddox, P. Collopy, and P. A. Farrington, "Value-based assessment of dod acquisitions programs," *Procedia Computer Science*, vol. 16, pp. 1161 – 1169, 2013, 2013 Conference on Systems Engineering Research. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050913001233

[28] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. New York, NY, USA: Cambridge University Press, 2008.

[29] S. Safarkhani, V. R. Kattakuri, I. Bilionis, and J. Panchal, "A principal-agent model of systems engineering processes with application to satellite design," *arXiv preprint arXiv:1903.06979*, 2019.

[30] S. Safarkhani, I. Bilionis, and J. Panchal, "Towards a theory of systems engineering processes: A principal-agent model of a one-shot, shallow process," *arXiv preprint arXiv:1903.12086*, 2019.

[31] H. Gintis, "The bounds of reason: Game theory and the unification of the behavioral sciences herbert gintis."

[32] D. Kahneman and A. Tversky, "Prospect theory: An analysis of decision under risk," in *Handbook of the fundamentals of financial decision making: Part I*. World Scientific, 2013, pp. 99–127.

[33] . Gibbons, Robert, *Game theory for applied economists*. Princeton, N.J. : Princeton University Press, [1992] ©1992, [1992]. [Online]. Available: https://search.library.wisc.edu/catalog/999707589902121

[34] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[35] T. L. Griffiths, C. Lucas, J. Williams, and M. L. Kalish, "Modeling human function learning with gaussian processes," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Curran Associates, Inc., 2009, pp. 553–560. [Online]. Available: http://papers.nips.cc/paper/3529-modeling-human-function-learning-with-gaussian-processes.pdf

[36] A. Borji and L. Itti, "Bayesian optimization explains human active search," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 55–63. [Online]. Available: http://papers.nips.cc/paper/4952-bayesian-optimization-explains-human-active-search.pdf

[37] S. Helie and R. Sun, "Incubation, insight, and creative problem solving: A unified theory and a connectionist model," *Psychological Review*, vol. 117, no. 3, pp. 994–1024, July 2010.

[38] J. Bédard and M. T. Chi, "Expertise," *Current Directions in Psychological Science*, vol. 1, no. 4, pp. 135–139, Aug. 1992. [Online]. Available: https://doi.org/10.1111/1467-8721.ep10769799

[39] M. B. Waldron and K. J. Waldron, *The Influence of the Designer's Expertise on the Design Process*. New York, NY: Springer New York, 1996, pp. 5–20. [Online]. Available: https://doi.org/10.1007/978-1-4757-2561-2_2

[40] N. Cross, "Expertise in design: an overview," *Design Studies*, vol. 25, no. 5, pp. 427–441, Sep. 2004. [Online]. Available: https://doi.org/10.1016/j.destud.2004.06.002

[41] S. Ahmed, K. M. Wallace, and L. T. Blessing, "Understanding the differences between how novice and experienced designers approach design tasks," *Research in engineering design*, vol. 14, no. 1, pp. 1–11, 2003.

[42] P. B. Schaub, "Strategies of experts in engineering design: between innovation and routine behaviour," *Journal of Design Research*, vol. 4, no. 2, pp. 125–143, 2004.

[43] R. Ghanem and P. Spanos, *Stochastic Finite Elements: A Spectral Approach*, ser. Civil, Mechanical and Other Engineering Series. Dover Publications, 2003. [Online]. Available: https://books.google.com/books?id=WzgKyTQQcAwC

[44] N. R. Augustine, *Augustine's laws*. American Institute of Aeronautics and Astronautics, 1997.

[45] R. Adler, *The Geometry of Random Fields*, ser. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, Jan. 2010. [Online]. Available: https://epubs.siam.org/doi/book/10.1137/1.9780898718980

[46] B. Øksendal, *Stochastic Differential Equations: An Introduction with Applications*, ser. Hochschultext / Universitext. Springer, 2003. [Online]. Available: https://books.google.com/books?id=kXw9hB4EEpUC

[47] J. H. Panchal, Z. Sha, and K. N. Kannan, "Understanding design decisions under competition using games with information acquisition and a behavioral experiment," *Journal of Mechanical Design*, vol. 139, no. 9, pp. 091 402–091 402–12, 07 2017. [Online]. Available: http://dx.doi.org/10.1115/1.4037253

[48] E. Brochu, V. M. Cora, and N. de Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *CoRR*, vol. abs/1012.2599, 2010. [Online]. Available: http://arxiv.org/abs/1012.2599

[49] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[50] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[51] J.-M. AzaÃs and M. Wschebor, "A general expression for the distribution of the maximum of a gaussian field and the approximation of the tail," *Stochastic Processes and their Applications*, vol. 118, no. 7, pp. 1190 – 1218, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S030441490700138X

[52] P. H. Werhane, "Engineers and management: The challenge of the challenger incident," *Journal of Business Ethics*, vol. 10, no. 8, pp. 605–616, 1991.

[53] P. D. Collopy and P. M. Hollingsworth, "Value-Driven Design," *Journal of Aircraft*, vol. 48, no. 3, pp. 749–759, 2011. [Online]. Available: https://doi.org/10.2514/1.C000311

[54] S. Jung, T. W. Simpson, and C. Bloebaum, "Multi-level value-driven design approaches for product family design," in *ASME 2017 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2017, pp. V02BT03A034–V02BT03A034.

[55] B. D. Lee and C. J. Paredis, "A conceptual framework for value-driven design and systems engineering," *Procedia CIRP*, no. 21, pp. 10–17, 2014.

[56] H. Kannan, E. Tibor, B. Mesmer, and C. L. Bloebaum, "Incorporation of coupling strength models in a value-based systems engineering framework for optimization," in *16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2015, p. 3086.

[57] H. Kannan, B. L. Mesmer, and C. L. Bloebaum, "Increased system consistency through incorporation of coupling in value-based systems engineering," *Systems Engineering*, vol. 20, no. 1, pp. 21–44, 2017.

[58] G. Bhatia and B. Mesmer, "Integrating model-based systems engineering and value-based design with an nea scout small satellite example," in *AIAA SPACE and Astronautics Forum and Exposition*, 2017, p. 5234.

[59] A. Salado, "Integrating design and verification decisions in value-driven design to increase system's expected value," in *Conference on Systems Engineering Research (CSER)*, 2016.

[60] M. Bertoni, A. Bertoni, and O. Isaksson, "Evoke: A value-driven concept selection method for early system design," *Journal of Systems Science and Systems Engineering*, vol. 27, no. 1, pp. 46–77, 2018.

[61] E. Papageorgiou, M. H. Eres, and J. Scanlan, "Value driven conceptual design of unmanned air system for a defence application," *Journal of Aerospace Operations*, vol. 4, no. 3, pp. 141–158, 2017.

[62] L. Vengadasalam, A. Desai, P. Hollingsworth, and K. Smith, "Value-centric/driven design-application for the space industry," in *AIAA SPACE and Astronautics Forum and Exposition*, 2017, p. 5328.

[63] L. Shao, Y. Duan, X. Sun, Q. Zou, R. Jing, and J. Lin, "Bidirectional value driven design between economical planning and technical implementation based on data graph, information graph and knowledge graph," in *Software Engineering Research, Management and Applications (SERA), 2017 IEEE 15th International Conference on*. IEEE, 2017, pp. 339–344.

[64] J. Cheung, J. Scanlan, J. Wong, J. Forrester, H. Eres, P. Collopy, P. Hollingsworth, S. Wiseall, and S. Briceno, "Application of value-driven design to commercial aeroengine systems," *Journal of Aircraft*, vol. 49, no. 3, pp. 688–702, 2012.

[65] M. Bertoni, M. Panarotto, and P. Jonsson, "Value-driven engineering design: Lessons learned from the road construction equipment industry," in *ICED17 21st International Conference on Engineering Design, Vancouver*, vol. 1. The Design Society, 2017, pp. 319–328.

[66] M. Shergadwala, I. Bilionis, K. N. Kannan, and J. H. Panchal, "Quantifying the impact of domain knowledge and problem framing on sequential decisions in engineering design," *Journal of Mechanical Design*, vol. 140, no. 10, p. 101402, 2018.

[67] A. M. Chaudhari, Z. Sha, and J. H. Panchal, "Analyzing participant behaviors in design crowdsourcing contests using causal inference on field data," *Journal of Mechanical Design*, vol. 140, no. 9, p. 091401, 2018.

[68] J. Watson, "Contract, mechanism design, and technological detail," *Econometrica*, vol. 75, no. 1, pp. 55–81, 2007.

[69] S. Safarkhani, I. Bilionis, and J. Panchal, "Understanding the effect of task complexity and problem-solving skills on the design performance of agents in systems engineering," *the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Aug 2018.

[70] J. A. Mirrlees, "The theory of moral hazard and unobservable behaviour: Part i," *The Review of Economic Studies*, vol. 66, no. 1, pp. 3–21, 1999. [Online]. Available: http://www.jstor.org/stable/2566946

[71] W. P. Rogerson, "The first-order approach to principal-agent problems," *Econometrica*, vol. 53, no. 6, pp. 1357–1367, 1985. [Online]. Available: http://www.jstor.org/stable/1913212

[72] R. Ke, C. T. Ryan *et al.*, "A general solution method for moral hazard problems."

[73] R. Myerson, *Game Theory: Analysis of Conflict*. Harvard University Press, 1991. [Online]. Available: https://books.google.com/books?id=1w5PAAAAMAAJ

[74] J. C. Harsanyi, "Games with incomplete information played by Bayesian players. part I: The basic model," *Management Science*, vol. 14, pp. 159–182, 1967.

[75] R. B. Myerson, "Optimal auction design," *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.

[76] D. Kahneman and A. Tversky, "Prospect theory.. an analy-sis of decision under risk," 2008.

[77] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016. [Online]. Available: http://arxiv.org/abs/1605.02688

[78] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001–. [Online]. Available: http://www.scipy.org/

[79] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and computing*, vol. 10, no. 3, pp. 197–208, 2000.

[80] Ilias bilionis, "Sequential Monte Carlo working on top of pymc." [Online]. Available: https://github.com/PredictiveScienceLab/pysmc

[81] T. Gerstner and M. Griebel, "Numerical integration using sparse grids," *Numerical Algorithms*, vol. 18, no. 3, p. 209, Jan 1998. [Online]. Available: https://doi.org/10.1023/A:1019129717644

[82] J. R. Wertz, D. F. Everett, and J. J. Puschell, *Space mission engineering: the new SMAD*. Hawthorne, CA: Microcosm Press : Sold and distributed worldwide by Microcosm Astronautics Books, 2011, oCLC: 747731146.

[83] "Solid." [Online]. Available: http://www.astronautix.com/s/solid.html

[84] "NASA Historical Data Books." [Online]. Available: https://history.nasa.gov/SP-4012/vol6/cover6.html

[85] "Propulsion Engineer Salaries." [Online]. Available: https://www.paysa.com/salaries/propulsion-engineer--t

[86] GAO, NASA, "Assessments of major projects," 2019.

[87] Y. Bar-Yam, "About engineering complex systems: Multiscale analysis and evolutionary engineering," in *Engineering Self-Organising Systems*, S. A. Brueckner, G. Di Marzo Serugendo, A. Karageorgos, and R. Nagpal, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 16–31.

[88]

[89] S. Safarkhani, I. Bilionis, and J. H. Panchal, "Toward a theory of systems engineering processes: A principal–agent model of a one-shot, shallow process," *IEEE Systems Journal*, 2020.

[90] B. Heydari, Z. Szajnfarber, J. Panchal, M.-A. Cardin, K. Holtta-Otto, G. E. Kremer, and W. Chen, "Analysis and design of sociotechnical systems," *Journal of Mechanical Design*, vol. 141, no. 11, 2019.

[91] A. M. Chaudhari and J. H. Panchal, "An experimental study of human decisions in sequential information acquisition in design: Impact of cost and task complexity," in *Research into Design for a Connected World*. Springer, 2019, pp. 321–332.

[92] S. Safarkhani, I. Bilionis, and J. H. Panchal, "Understanding the effect of task complexity and problem-solving skills on the design performance of agents in systems engineering," in *ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers Digital Collection.

[93] Office of the Under Secretary of Defense for Acquisition and Sustainment, "Other transactions guide," 2018. [Online]. Available: https://www.dau.edu/guidebooks/Shared%20Documents/Other%20Transactions%20(OT)%20Guide.pdf

[94] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[95] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Thirtieth AAAI conference on artificial intelligence*, 2016.

[96] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," *arXiv preprint arXiv:1511.06581*, 2015.

[97] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[98] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[99] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[100] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *International Conference on Autonomous Agents and Multiagent Systems*. Springer, 2017, pp. 66–83.

[101] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in neural information processing systems*, 2017, pp. 6379–6390.

[102] G. Bacchiani, D. Molinari, and M. Patander, "Microscopic traffic simulation by cooperative multi-agent deep reinforcement learning," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 1547–1555.

[103] J. Jin, C. Song, H. Li, K. Gai, J. Wang, and W. Zhang, "Real-time bidding with multi-agent reinforcement learning in display advertising," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 2193–2201.

[104] K. N. Kannan, V. Pamuru, and Y. Rosokha, "Using machine learning for modeling human behavior and analyzing friction in generalized second price auctions," *Available at SSRN 3315772*, 2019.

[105] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," *arXiv preprint arXiv:1906.05799*, 2019.

[106] N. Kaloudi and J. Li, "The ai-based cyber threat landscape: A survey," *ACM Computing Surveys (CSUR)*, vol. 53, no. 1, pp. 1–34, 2020.

[107] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[108] A. Y. Ng, S. J. Russell *et al.*, "Algorithms for inverse reinforcement learning."