The Dissertation Committee for Zoltán Tibor Hidvégi
certifies that this is the approved version of the following dissertation:

Three Essays on the Interface of Computer Science, Economics and Information Systems

Committee:

Andrew B. Whinston, Supervisor

E. Allen Emerson

J Strother Moore

Maxwell Stinchcombe

John R. Mote

Three Essays on the Interface of Computer Science, Economics and Information Systems

by

Zoltán Tibor Hidvégi, M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2007

Acknowledgments

I wish to thank Hong Xu, Jianqing Chen and Wenli Wang for their comments, ideas and insights, and to my advisor, Andrew B. Whinston for his patience and persistence.

Three Essays on the Interface of Computer Science, Economics and Information Systems

Publication No. _____

Zoltán Tibor Hidvégi, Ph.D.
The University of Texas at Austin, 2007

Supervisor: Andrew B. Whinston

This thesis looks at three aspects related to the design of E-commerce systems, online auctions and distributed grid computing systems. We show how formal verification techniques from computer science can be applied to ensure the correctness of system design and implementation at the code level. Through an e-ticket sales example, we demonstrate that model checking can locate subtle but critical flaws that traditional control and auditing methods (e.g., penetration testing, analytical procedure) most likely miss. Auditors should understand formal verification methods, enforce engineering to use them to create designs with less of a chance of failure, and even practice formal verification themselves in order to offer credible control and assurance for critical e-systems.

Next, we study why many online auctions offer fixed buy prices to understand why sellers and auctioneers voluntarily limit the surplus they can get from an auction. We show when either the seller or the bidders are risk-averse, a properly chosen fixed permanent buy-price can increase the social surplus and does not decrease the expected utility of the sellers and bidders, and we characterize the unique equilibrium strategies of uniformly risk-averse buyers in a buy-price auction.

In the final chapter we look at the design of a distributed grid-computing system. We show how code-instrumentation can be used to generate a witness of program execution, and show how this witness can be used to audit the work of self-interested grid agents. Using a trusted intermediary between grid providers and customers, the audit allows payment to be contingent on the successful audit results, and it creates a verified reputation history of grid providers. We show that enabling the free trade of reputations provides economic incentives to agents to perform the computations assigned, and it induces increasing effort levels as the agents' reputation increases. We show that in such a reputation market only high-type agents would have incentive to purchase a high reputation, and only low-type agents would use low reputations, thus a market works as a natural signaling mechanism about the agents' type.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This dissertation is a sample of the research I have done during my 7 years in the doctoral program. It includes three independent works covering the fields of Economics, Computer Science and Information Systems.

Chapter 2 represents my early research on the use of formal verification methods to audit e-commerce systems. An e-commerce system is in fact a distributed computing environment where several concurrent programs, often running on different computers, interact with each other and with the outside world. Online businesses are highly automated, and are heavily dependent on the correct design and operation of their systems. They must ensure timely responses to consumer requests, they must guard consumers' private information and increasingly often they have to be able to resist malicious attacks. I propose that beyond conventional testing methods firms must also employ formal verification methods, especially to find problems stemming from the interaction of concurrent independent programs. Admittedly, this chapter does not contain any original results, but, through simple examples, it shows how formal methods can be applied in the e-business context. Parts of the work presented in Chapter 2 was published in [1].

In Chapter 3 I study permanent buy-price auctions. A fixed buy-price is posted throughout such an auction, and any bidder at any time can exercise the option of purchasing the item for sale at this fixed price. Clearly, this fixed buy-price puts an upper limit on the maximum surplus which can be achieved by the seller, but I show that despite that the seller's utility increases when either the seller or the buyers are risk-averse, and the

buy-price is not too low. I also calculate the unique equilibrium strategies of bidders in these auctions under the assumption that all bidders share the same linear or concave utility function. The work in this chapter has been published in [2].

In Chapter 4 I describe my current ongoing research focusing on the design of grid computing systems. To be able to build a commercially viable grid infrastructure one must design an economic incentive system which induces self-interested profit-maximizing agents to honestly perform the tasks assigned to them. I introduce a "witness" system which is similar to a message-digest of the execution paths taken while running a program. This system can be implemented in software via code instrumentation, however, in the future it could be easily integrated into CPU hardware to reduce the computation overhead. I use this to design a mechanism where a trusted intermediary audits the agents by assigning some jobs to multiple agents and comparing their results. This enables contracts where payment is contingent on the audit results, and it allows the intermediary to build a reputation history for market participants. I show that allowing the free trade of reputations in an open market induces the agents to put in increasingly more effort than minimally necessary as their reputation increase, and I show that in this market an agent can signal her type by purchasing an expensive good reputation when she is high-type, but only using cheap low-reputations names when she is low-type.

Chapter 2

Model Checking – A Rigorous and Efficient Tool for Preventing E-business Failures

Abstract

An unexpected error in an e-business' processing system may lead to devastating failures. Luckily, model checking, an advanced formal verification method, can thoroughly verify the correctness of critical e-systems [1]. Temporal logic, coupled with automata-theoretic verification, provides a rigorous and efficient means of specifying and assuring correct e-process behaviors. Through an e-ticket sales example, we demonstrate that model checking can locate subtle but critical flaws that traditional control and auditing methods (e.g., penetration testing, analytical procedure) most likely miss. Auditors should understand formal verification methods, enforce engineering to use them to create designs with less of a chance of failure, and even practice formal verification themselves in order to offer credible control and assurance for critical e-systems.

2.1 Introduction

In digital economy, many businesses leverage their critical business operations on Internet-based e-processes. More and more resources are procured, managed, created and consumed over Internet, Intranets and Extranets. Even the world's financial markets, telecommunications, and the supply of water and power partially depend on the operations of massive Internet-based information systems. Any error in a critical information system like those for accounting, stock trading, banking and air traffic control can potentially cause devastating

failures. Not to mention there are hackers or even terrorists aiming to discover and exploit such errors. Tremendous operational uncertainties exist, some may result in disasters.

Many e-businesses have fallen victim to operational problems, ranging from poor security, inadequate controls, to badly designed and unreliable systems [3]. Management lacks reasonable assurance as to the effectiveness and efficiency of its e-operations, the reliability of information for decision-making, and the overall compliance of e-operations with applicable laws and regulations. Security breaches and e-service failures have become a fact of life. The "Code Red" worm has affected world-wide Microsoft IIS servers and Cisco routers causing many businesses to halt temporarily. Toys 'R' Us failed to fulfill customer orders. Ashford labelled zero prices on luxury watches. Even big players may tumble. On June 8, 2001, trading at the New York Stock Exchange halted for almost one and a half hour due to a software glitch caused by changes made in its computing systems.

Current e-business operational incidents are mostly due to the lack of integrity at the low-level computing and networking services. From operating systems to browsers to even anti-virus packages, nearly every major software has had a flaw or two; Windows NT 4.0 had 164 security holes, Internet Explorer 69, Norton Antivirus 7, to name just a few [4]. No wonder most past attacks exploited these easy targets. Consequently, current countermeasures (e.g.,firewalls, encryption and patches) and the EDP auditing primarily focus on these services as well [5].

Unfortunately, even if an ideal computing and networking environment exists in the future, if e-business processes, running on top of the supporting computing and networking services, are not designed or implemented properly, the e-business would still be vulnerable. The incident of Ashford's zero pricing was not a technical failure alone, but rather a lack of input control over Web catalogs, a problem in the process design.

Hence, internal control and assurance over e-operations is essential. As in traditional commerce, e-commerce managers turn to their auditors for support, control and

independent assurance. The auditing profession has already established CobiT[1] for general information system control [6] and introduced WebTrust and SysTrust [7, 8] targeting e-commerce assurance. However, these products largely carry over traditional control concepts and auditing methods [9], which is only a half-right solution. Traditionally auditing methods – observing system behavior, inquiring of employees, examining documents, reprocessing data, and analyzing information through analytical procedures – are useful but limited. One common feature is their generally informal and ad hoc nature; none delivers mathematical certainty. Even their strongest statistics form inherently belongs to the domain of testing and simulation, methods that have limitations.

The future success of auditors in the high-tech arena requires new perspectives and methods suitable for e-commerce. We suggest that correct e-process design and implementation should the first line of defense against errors, fraud and hacking. Minimizing operational faults is critical but not easy. E-business systems are often so complex that they overwhelm the traditional methods. The complexity partially arises from the fact that e-systems are non-stop, non-deterministic computing systems, within which multiple processes with no location constraints execute around the clock in an asynchronous and highly interactive fashion. These concurrent processes execute many tasks in unpredictable order, resulting in virtually unlimited event possibilities and therefore many uncertainties. Like many construction or mechanical engineering disasters, the more possibilities of component interactions, the more chances that there is a whole strain of events that happen in a certain order no one has anticipated may bring systems down. Like builders of bridges have to consider aerodynamic problems, e-business designers need to take into account stresses caused by new Internet operation environment.

Carefully designed and implemented code can handle most expected situations and hence can function well within these defined boundaries. However, it is impossible to con-

[1] CobiT: Control Objectives for Information and related Technology.

sider all potential scenarios due to the complexity and human bounded rationality. Hidden flaws and errors, triggered mostly under unexpected scenarios, lead to potentially devastating disasters. For instance, subtle programming bugs have been identified as the culprit of the explosion of the $500 million European Ariane 5 rocket in 1996 [10] and the loss of the $165 million Mars Polar Lander in 1999 [11]. The chip and telecommunication industries have long experienced the devastating effects of numerous simple but hard-to-detect errors. As we will demonstrate with a rather simple example, a deadlock, occurring only when certain system resources are overloaded, can bring an entire system to a halt.

Time pressure to market and the unfamiliarity with the new technology and economy add more difficulties in system design and implementation, which are often neither thorough nor correct. Interconnectivity of the Internet widens the scope of an attack allowing the remote and anonymous exploitation of hidden flaws. Auditors, no less than management, need to catch these mistakes before hackers and fraudsters. Penetration testing, analytical procedures and test of details, primarily based on sampling and statistical analysis, are inadequate to catch hidden flaws. Similar to the use of formal methods in work-flow [12, 13] and knowledge-based systems [14], formal verification can supplement traditional methods and has its niche in e-commerce. Formal methods is to e-business process engineering can be what fluid dynamics is to aeronautical engineering and what classical mechanics is to civil engineering because their reasoning capabilities are essential to proper designs and more powerful than human authorities or experiences. Like the introduction of statistical methods to auditing in the 1960s, we fuse modern formal verification techniques to internal controls and assurance services. To date, formal verification techniques have been successfully applied to complex microprocessor designs and critical telecommunication protocols. Only recently have a few designers applied them in the business integrity domain [1, 15, 16, 17, 18].

A powerful yet efficient formal verification method is model checking, which can

verify concurrent, non-stop systems. It can locate subtle but critical flaws that are most likely to be missed by conventional control and assurance methods. Unlike the traditional EDP methods which generally only analyze the input-output semantics of a system and are largely based on the manual or computer-aided, post verification of point-in-time states, model checking analyzes both states and state transitions, and provide automated, proactive and continuous control and assurance. In addition, while traditional methods primarily addresses only "safety" properties ("never" or "always" claims, e.g., *debits = credits*), model checking can also verify "liveness" properties ("eventuality" claims like "a system progresses without unwanted halt or infinite loop"), essential for the correctness of non-stop e-systems.

The paper is organized as follows. In Section 2.2, we discuss why model checking has its advantages over testing and simulation. Section 2.3 depicts current industrial applications of formal verification. In Section 2.4, we draw the analogy between formal verification and assurance method spectrums. In Section 2.5, we theoretically explain how model checking works and demonstrate with a traffic control example. In Section 2.6, we build an e-business prototype, an e-ticket sale system, and verify its correctness using two model checkers. In Section 2.7, we discuss the research contributions and limitations.

2.2 Testing and Simulation vs. Formal Verification

Although testing and simulation have been applied extensively in system control and assurance, they have inherent limitations when compared to modern formal verification techniques like model checking (Table 2.1).

The major limitation of testing and simulation (Table 2.1, Item Ia) is that they only check a fraction of system behaviors. Hence, conclusions based on the partial coverage of the state space inherently convey only probability rather than certainty. Research shows that testing and simulation probably need to be conducted over more than half of the entire

Item	Testing & Simulation (a)	Model Checking (b)
I	Impossible to cover all system states, but bugs and errors often hide in hard-to-anticipate scenarios or corner cases.	Operate on logic and cover the entire state space.
II	Need expertise to find test vectors leading to critical execution paths.	Need expertise to model processes and properties, but verification is automated.
III	Tools do not depict what lead to errors.	Graphic tools exist to depict what scenarios lead to errors.
IV	Check only "safety" properties.	Verify both "safety" and "liveness" properties.
V	"Around the system" approaches can be taken advantage of by hackers.	"Through the system" verification requires the knowledge of the system, of which hackers do not have.
VI	Apply traditionally only after system implementation.	Apply in parallel with the system development life cycle.

Table 2.1: Comparison between testing & simulation and model checking

useful life of a system in order to discover just one-third of the total errors [19].

Consider an e-process as a state machine and represent each of the process' states with a vector composed of the simultaneous values of all process variables and the instruction pointer of the current execution. The state space of a system includes all the plausible combinations of the states of all the system processes. Even for a simple system with few processes, the state space can be astronomically large, easily exceeding the number of particles in the Universe. Testing and simulation execute a system with chosen test vectors, i.e., input sequences attempting to exercise critical execution paths. In most cases this effort covers only a microscopic portion of the state space. For example, a Web catalog with a search interface can be tested with selective search patterns, yet it is impossible to try all query combinations.

Testing and simulation become even harder when a system has several interacting components running in parallel, creating uncertainties due to the non-determinism of com-

munication and component performance. Hidden errors like “race conditions” or “deadlocks” are likely to occur in such a system but often triggered only under certain obscure, hard-to-guess situations. It is extremely difficult, if not impossible, to select the test vectors that can catch these errors.

Testing and simulation can provide reasonable system assurance if carefully selected test vectors exercise all important execution paths. However, the creation of a good test requires insightful or even imaginative work by those with an intricate knowledge of the system design and implementation (Item IIa). Although tools exist to measure the coverage of testing and simulation, they do not provide automated means to create test patterns covering missing critical paths that may lead to problematic regions of the state space. Moreover, each time a process changes, test vectors normally have to be modified to cover the new or modified execution paths. Because of the difficulty selecting test vectors, test vectors are sometimes generated randomly. But for complex systems the probability of hitting a bug in a reasonable time with random tests is close to zero [20]. Moreover, in case a mistake is found, testing and simulation tools usually cannot depict how the problem occurs (Item IIIa).

In contrast, formal verification spans the entire state space of a system and prove property-satisfaction with mathematical certainty: if a system is formally verified to have a given attribute, no system behavior can ever be found to contradict this attribute (Item Ib).

The reason why formal verification can achieve the exhaustive examination of a property is because it analyzes the logic among processes rather than executes them. And avoiding examining individual executions makes formal verification efficient. Figure 2.1 illustrates a particular model checking algorithm. It starts with a set of “bad” states (states that violate the property under examination) and repeatedly expands this set by adding states from which a bad state can be reached and marks these states “bad” as well. In a finite state system, this procedure eventually covers all the states with an execution path

leading to bad states. If the initial state is marked "bad," it implies the existence of execution path(s) leading to the violation of the property. This verification procedure, based on set operations, covers states much faster than testing and simulation where each test covers only a few states. It also delivers more reliable results because testing and simulation can find out property violation if and only if one of the execution paths leading to a bad state happens to be chosen as a test vector. There are numerous examples of hardware flaws and software bugs which could have caused significant commercial damages were missed by testing and simulation but located quickly with formal verification [21, 22]. For critical system properties, the thorough examination offered by formal verification should be highly desired or even considered necessary.

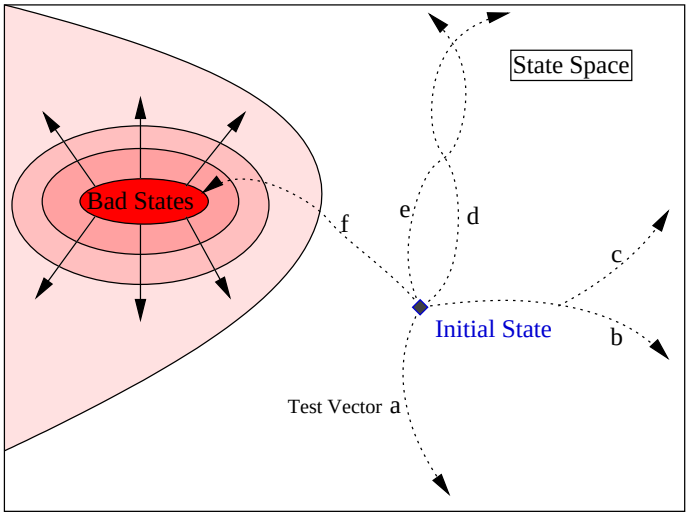


Figure 2.1: Testing and Simulation vs. Model Checking

In addition, model checking can verify both "safety" and "liveness" properties (Item IVb). Safety properties are commonly examined in current debugging practices and testing and simulation can guarantee that safety assertions hold for the set of test patterns considered (Item IVa). In practice a set of test vectors are often run to discover minor and obvious bugs before applying formal verification to find hidden errors. Verifying liveness

properties are generally omitted in testing and simulation because a system simply cannot be exhaustively tested through executions in a finite time. However, when an e-business offers 24*7 services handling numerous service requests, it is important to know that every service request is eventually fulfilled and the non-stop system does not go into an infinite loop which precludes useful behaviors. Model checking can verify liveness properties, as we will explain later in details.

Moreover, testing and simulation are "around the system" approaches (Item Va), which can be used by hackers to "reverse engineer" a victim's system. A "brute force" attack is such an extreme form of testing and simulation. System designers and auditors compete against hackers in a race to find critical test vectors that may lead to system malfunctions. They can improve their odds by using "direct engineer" techniques "through the system," such as formal verification (Item Vb). Although some formal verification tools can be freely downloaded, their effective use still requires full access to system resources (e.g., source code), access not generally available to hackers; hackers can only execute source code but have no read/write privileges.

To achieve high level control and assurance, systems must be originally designed to be testable and verifiable. Testing and simulation traditionally only occur after system implementation is complete (Item VIa). Even though certain software development like fast prototyping supports testing before implementation [23], it is uncertain how much one should rely upon the testing results of an incomplete system when claiming the assurance of the entire system. Research on model checking however, considers decomposition[2] and localization[3] issues and suggests methods to modularize a system with rigorous interfaces that facilitate component-based verification. Formal verification should and can be adopted in parallel to system development to ensure every step is warranted (Item VIb). Even though

[2]Decompose a property to several sub-properties.
[3]Select relevant states and state transitions local to a sub-property that needs to be verified.

proving the correctness of a system is complex, fortunately, once such a proof exists, it is robust to re-examinations.

Auditors should promote the "design for testability and verifiability" in e-business application development and help set up the policies and procedures. If an application is built primarily through integrating off-the-shelf software packages, internal auditors should advise the procurement and IT departments to purchase packages with independent, credible assurance and help examine the integration process. For in-house application development, internal auditors should enforce designers/developers follow the "design for testability and verifiability" policies and procedures. Later on, external auditors can build upon the internal auditors' opinions to further investigate a system and provide independent assurance.

2.3 Industrial Applications of Formal Verification

Proving the correctness of computation (e.g., the Euclidean algorithm) is not new. Turing was among the first to realize its importance. However, it was not until the 1960s and early 1970s that provably correct computation began to attract much research attention [24]. But industrial applications had to wait until 1994, when formal verification was first used for the verification of chip designs. After the discovery of the notorious FDIV error of the Pentium chip, Intel invested heavily in formal verification. Today, most players in the hardware industry, like Intel, AMD, IBM, and Motorola, all have formal verification teams to work on selected chip properties. It is due to both the high cost of a problem fix and the potential legal liability after a chip is on the market; e.g., Intel spent over $400 million to recall its Pentium chips and Toshiba paid $2 billion in an out-of-court settlement of a lawsuit against its faulty design of a simple chip.

Formal verification in the software industry only began in 1997. As might be expected, early applications were in mission-critical systems like telecommunications where a

small bug can disrupt many services. Applying verification to software in general has been promoted, but not well received. This is mainly due to the lack of incentives for software companies to provide rigorous quality assurance. They rely on users and even hackers to debug their software, after which they distribute fixes via the Internet with almost no cost. Ironically, software vendors can even get extra revenue from bug fixes. It is not uncommon to charge customers for upgrades that, apart from bug fixes, provide almost no extra functionality (e.g., the upgrades from Windows 95 to 98, and to ME). Moreover, software bugs have not led to serious litigation threats. Vendors disclaim all warranties in their software license agreements so that customers have no legal power to urge them to provide quality products. Of course, software vendors do not want to sell software that is too buggy to use because of the fear of reputation loss and new market entrants. Their common practice is to use some testing and simulation to debug easy-to-anticipate errors and leave hard-to-detect bugs to users. They often release "beta test" versions of a new product to recruit free testers. But still, the initial release of a software is often rushed to the market without thorough examination.

As more and more critical business applications are implemented in software, we suggest the use of formal verification for e-business assurance. Rather than relying on the unwarranted promises of software vendors and in-house developers, e-businesses should proactively verify that their e-systems conform to the requirements and will function well in both benign and hostile environments. Auditors, promoting, overseeing, and even practicing the use of formal verification, can help e-business managers and shareholders build confidence in e-operations.

2.4 Formal Verification vs. Assurance Method Spectrums

Beginning 1960s, research on formal verification focused on constructing proofs from axioms and inference rules in the same way as constructing mathematics proofs. In the past

four decades, formal verification has developed into a spectrum of methodologies, ranging from manual proof of mathematical arguments, interactive computer-aided theorem proving, to algorithmic and automated model checking [25] (Figure 2.2).

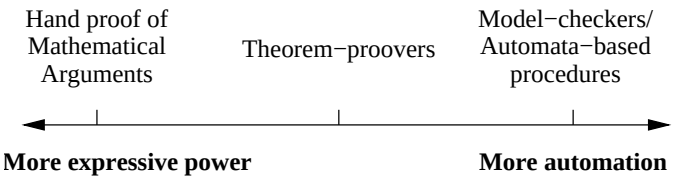


Figure 2.2: The Formal Verification Spectrum

Manual proof of system correctness is very expressive but time-consuming, prone to human errors, hard to verify[4] and often an order of magnitude bigger than the original system. Hence, it is not economically viable to manually prove the correctness of a business software, even for very small ones with few hundred lines of code. Computer-aided theorem provers like ACL2[5] are proof checkers designed to reduce human errors. But a great deal of human expertise and manual hints are still needed to construct proofs and express them in a format acceptable to theorem provers. In addition, theorem provers do not, in most cases, prove a statement false and provide little help in locating the cause leading to a property violation. Therefore, theorem provers can not be easily applied for business software verification.

Unlike manual proof or theorem proving, model checking is automated and relatively efficient in verifying system properties. Users only need to state system models and properties to verify; verification is automatically done by model checkers (Table 2.1, Item IIb). It is the automation that makes model checking attractive to business control and assurance because it does not slow down the pace of application development dramatically. Developers can still strive for short "time-to-market" without sacrificing quality. Moreover,

[4]The reliability of manual proof of mathematical arguments is based upon peer review.
[5]ACL2: A Computational Logic for Applicative Common Lisp.

when a property does not hold, model checkers can help identify the scenarios leading to the violation (Item IIIb) and hence facilitate system refinements.

By analogy, the assurance method spectrum (Figure 2.3) spans from manual collection of evidence and professional analysis, computer-aided checking, to automated verification. However, current auditing practices of system observation, inquiry, document examination, data re-processing and analytical procedures, unlike manual proofs or theorem-proving, cannot be considered as formal methods because they do not convey mathematical certainties. But auditors can still straightly go for modern formal verification to rigorously assure critical system attributes.

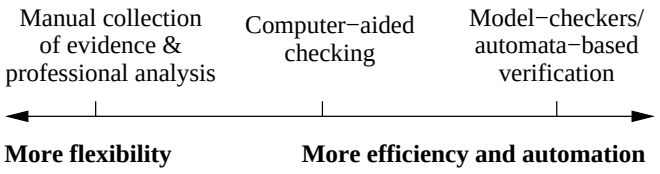


Figure 2.3: The Assurance Method Spectrum

2.5 How Model Checking Works

2.5.1 Theory

Applying model checking to a system consists of three primary tasks:

Modeling Modeling converts a system description into a model accepted by a model checker. Such a formal model is often described in a Kripke structure $M(AP, S, S_0, R, L)$ where

1. AP is the set of atomic properties, properties that can be immediately verified by examining the current states, such as "*credits* = *debits*," "*payment* > 0," etc.

2. S is a finite set of states.

3. $S_0 \subset S$ is the set of initial states.

4. $R = S \times S$ is a transition relation that for every state $s \in S$ there is a state s' such that $(s, s') \in R$.

5. $L : S \mapsto 2^{AP}$ is a function that labels each state with the set of atomic propositions true in that state.

Intuitively, the Kripke structure is a directed graph, where nodes S represent the possible system states, edges R are the possible state transitions, and paths are the possible system executions. This modeling approach allows the use of algorithms in graph theory to verify the system.

It is impractical to directly describe a Kripke structure of a large complex system. Fortunately, most model checkers can create Kripke structures automatically from system models. An automaton is a transformation of a Kripke structure.

Most model checkers have their own modeling languages to facilitate verification. These languages are easy to learn because of their similarity to popular programming languages like C. In addition, most model checkers have built-in constructs (e.g., communication queues and message delays), which make the modeling of distributed systems relatively easy.

Specification Specification is the description of the desired system properties. Temporal logic formulas should be used to specify concurrent e-systems. Temporal logic is a precise mathematical formalism that can express temporal properties, i.e., the ordering of events in time, without introducing time explicitly [26]. For instance, LTL can describe that a property expressed in formula f holds for "all time ($\mathsf{G}f$)," "next time ($\mathsf{X}f$)," "eventually ($\mathsf{F}f$)," "until g holds ($f \mathsf{U} g$)." Temporal logics can be classified according to whether time is assumed to have a linear or branching structure. LTL is Linear Temporal Logic,

CTL is a branching-time logic called Computational Tree Logic, and CTL* combines both branching- and linear-time operators.

Verification Verification is the process of checking whether the system model satisfies its properties. Given a Kripke structure M representing a finite-state concurrent system and a temporal logic formula f expressing a desired property, the checking is $M \models f$, i.e., whether M is a model of f.

There are four types of model checking: LTL, CTL, CTL*, and automata-theoretic model checking. We chose automata-theoretic model checking because automata are more expressive than LTL and are capable expressing eventuality assumptions not expressible in CTL and CTL*. There are two classes of conventional automata: $*$-automata, which has finite accepting runs, and ω-automata, where the accepting runs are countably infinite. We chose ω-automata because that can model the ongoing behavior of systems, and can express not only state invariants but also eventualities; it accepts infinite executions, i.e., the inputs of nonterminating processes.

Automata-theoretic model checking treats individual processes $P_1, P_2, \ldots, P_n$ as separate state machines $M_1, M_2, \ldots, M_n$; the system ω-automaton is the product of these process ω-automata: $M = M_1 \otimes M_2 \otimes \ldots M_n$. A model checker also converts a temporal logic formula f to a property automaton F. Model checking verifies whether the behavior of M is accepted by F. The behavior of M is defined in terms of its (nonterminating) executions, each of the form $x = (x_0, x_1, \ldots)$ where each $x_i = x_{M_1} * x_{M_2} * \cdots * x_{M_n}$. Let the language of M, $\mathcal{L}(M)$, denote the set of all such behaviors x of M and let $\mathcal{L}(F)$ denote the set of all sequences x accepted by F. Verification of M satisfying f consists of proving the first language containment $\mathcal{L}(M) \subset \mathcal{L}(F)$, which is equivalent to checking that $\mathcal{L}(M \otimes \neg F)$ is empty.

Although both $\mathcal{L}(M)$ and $\mathcal{L}(F)$ are infinite sets, it is possible to check the language

containment in finite number of steps: construct $M \otimes \neg F$ and find the cycles in the finite directed graph underlying $M \otimes \neg F$. Although there may be an infinite number of cycles, each cycle is contained in a strongly connected component in the graph. Since $M \otimes \neg F$ is a finite state system, it has only finite number of such components and any (infinite) behavior x must describe a trajectory which eventually cycles within a strongly connected component. It is enough then to check that each strongly connected component is consistent with the acceptance structure of F, and there are efficient automated methods to verify this.

The check $\mathcal{L}(M) \subset \mathcal{L}(F)$ requires searching a state space roughly of size $|M \otimes \neg F|$. In other words, the computational complexity of this check is linear to the size of the state space of the analyzed system and to the size of the negated property automaton. Decomposition and localization methods can further reduce this complexity.

2.5.2 An Example – A Crossroad Traffic Control System

Next we explain how model checking works using a crossroad traffic control system, taken from [21].

Modeling A traffic intersection system has three processes: Avenue A, Boulevard B and traffic Controller C. Figures 2.4 and 2.5 depict the states and state transitions for process A and C. Each of process A, B, C has two states; hence, the system has eight states.

The traffic control system is one of the simplest systems that deal with coordination of concurrent processes, but it can illustrate issues that may arise in larger systems. Harmonic coordination is very important in distributed e-business systems. For example, a ticket sale system that allows reservations from many agents must ensure that the same ticket cannot be reserved twice simultaneously.

In the diagrams, at each state of each process, there are two possible outputs. State transitions are described in Boolean predicates. One sample state transition x of the system

model M is:

x((STOP, STOP, go_A), (GO, STOP, go_A))

= A(STOP, GO) * B(STOP, STOP) * C(go_A, go_A)

=(A:cars_waiting)*(B:no_cars)+(B:cars_waiting))*(C:go_A)

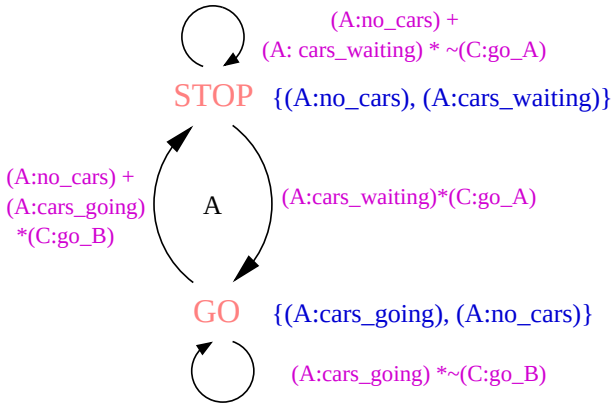


Figure 2.4: States and State Transition Diagrams for Process Avenue A.

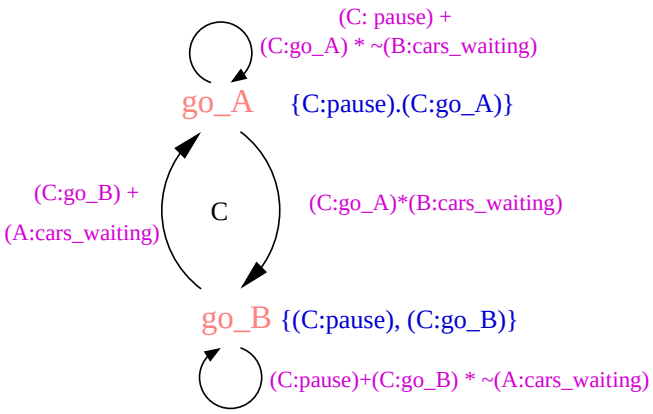


Figure 2.5: States and State Transition Diagrams for Process Controller C.

Properties We define two properties: “no cars collide” (safety) and “all cars on each road eventually get through the intersection” (liveness). The temporal logic formulas are:

1. $\mathsf{G}(\neg((\text{A:cars\_going}) * (\text{B:cars\_going})))$

2. $\mathsf{G}\begin{pmatrix}((\text{A:cars\_waiting}) \Rightarrow \mathsf{F}((\text{A:no\_cars}) + (\text{A:cars\_going})))* \\ ((\text{B:cars\_waiting}) \Rightarrow \mathsf{F}((\text{B:no\_cars}) + (\text{B:cars\_going})))\end{pmatrix}$

Figures 2.6 and 2.7 show the corresponding property automata F_1 and F_2. An initial state, if exists, is designated by the arrow entering the state. "+" marks a recur-edge, meaning that any infinite path (through the transition structure of a property automaton) crossing the edge infinitely often designates a behavior accepted by the automaton. F_1 shows that every behavior of M is accepted by F_1 except for "cars_going" on both roads, in which case F_1 goes to state 2 precluding an infinite number of recur-edge crossings. Every safety property can be modeled by such a 2-state automaton. F_2 depicts the liveness property; a behavior is accepted by F_2 unless on each road at some point cars are waiting, and thereafter neither "no_cars" nor "cars_going" on that road ever become true.

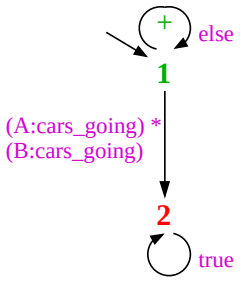


Figure 2.6: The ω-automaton F_1 defining the safety property of "no cars collide."

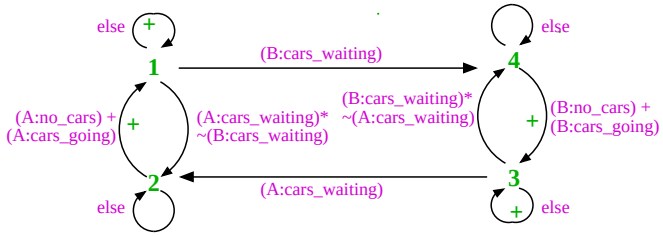


Figure 2.7: The ω-automaton F_2 defining the liveness property of "all cars on each road eventually get through the intersection."

Verification We now verify the crossroad traffic control system satisfies both properties, i.e., $\mathcal{L}(M) \subset \mathcal{L}(F_1) \cap \mathcal{L}(F_2)$. As an illustration, we only verify F_2 here.

To simplify, we decompose the task of verifying F_2 into two local subtasks: "all cars on Avenue A eventually get through the intersection" and likewise for Boulevard B. Figure 2.8 represents the ω-automaton F_A defining the first subtask, and likewise for F_B.

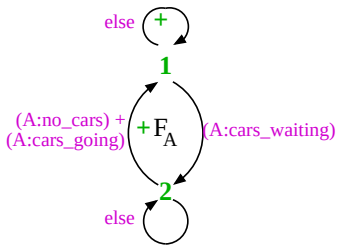


Figure 2.8: The ω-automaton F_A defining the liveness property of "all cars on Avenue A eventually get through the intersection."

M is verified in the following steps:

$$\mathcal{L}(M) \subset \mathcal{L}(F_A);$$

$$\mathcal{L}(M) \subset \mathcal{L}(F_B);$$

$$\mathcal{L}(A) \cap \mathcal{L}(B) \subset \mathcal{L}(F_2)$$

Task localization replaces a system model M with a simplified one by collapsing those portions of M irrelevant to the performance of a task. Since A and F_A do not involve B, we can have a $M' = A \otimes B' \otimes C'$, where B' and C' are derived from B and C respectively by collapsing every appearance of (B: no_cars), (B: car_waiting) or (B: cars_going) to (B: null). Further, we can reduce B' (Figure 2.9) to B'' (Figure 2.10). Now the verification is reduced to $\mathcal{L}(M'') \subset \mathcal{L}(F_A)$, where $M'' = A \otimes B'' \otimes C'$. Once we prove $\mathcal{L}(M) \subset \mathcal{L}(F_A)$ it follows $\mathcal{L}(M) \subset \mathcal{L}(F_B)$ because of the symmetry.

Unfortunately, the verification of $\mathcal{L}(M'') \subset \mathcal{L}(F_A)$ failed. It is possible that when there are "cars_waiting" on A, C may select "pause" instead of "go_A" and the system does

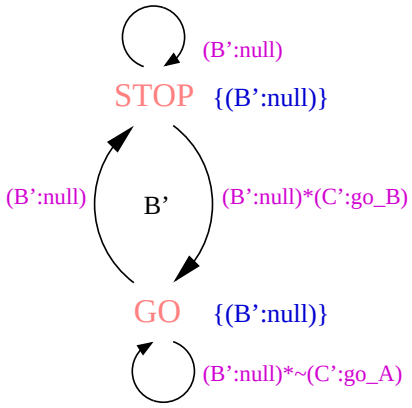


Figure 2.9: State transition diagram for simplified process B' with respect to F_A.

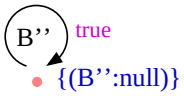


Figure 2.10: State transition diagram for further simplified process B'' with respect to F_A.

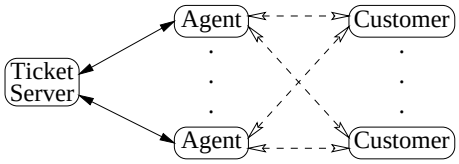
not specify how long C pauses. Therefore, the model should be modified to forbid the possibility that C may pause forever.

2.6 E-business Control and Assurance using Model Checking – An E-Ticket Sales Example

In practice, model checkers can automatically fulfill many verification tasks: a user only needs to describe processes and properties using a high-level programming language and a model checker automatically translates them into automata and verifies the system. There is no need to draw the state and state transition diagrams.

We demonstrate the feasibility of applying model checking to e-business control and assurance through an e-ticket sales example [1], shown in Figure 2.11. Although simple, the example embodies main characteristics of an e-system – distributed and parallel processing, concurrency, asynchronous communications, constraint resources, and non-stop

operations. Many complex e-systems like online stock trading and e-retailing exhibit similar features.



Customer → Agent:	want_to_buy, cancel_res, pay
Agent → Customer:	reserved, canceled, tickets, sold_out, try_later, too_many, table_full
Agent → Ticket Server:	reserve, get, cancel
Ticket Server → Agent:	try_later, sold_out, too_many, table_full, granted, sent, canceled

Figure 2.11: An E-ticket Sales Example.

2.6.1 Modeling

The e-business sells e-tickets in limited quantities over the Internet. The sales are implemented by coordinating e-processes. The following are the three e-process prototypes involved: Customer, Agent and Ticket Server. Each customer/agent is an instance process of the Customer/Agent prototype. There is only one Ticket Server T.

Customer Customers purchase e-tickets through web browsers from agents. Customers can reserve, pay for tickets, and cancel reservations.

Agent Agents, implemented in Perl or Java, are the middlemen forwarding requests and responses between customers and T. Agents are also responsible for verifying customer payments. A customer's payment is accepted only after the customer is notified about a successful reservation. In case the payment is rejected, the customer's agent would send a "not_approved" message to the customer and the reservation is canceled automatically.

Ticket Server The database-implemented Ticket Server T centrally holds all the e-tickets and communicates only with agents. T keeps track of the numbers of available, reserved and sold tickets, denoted as a, r, and s. Let t represent the number of tickets to be reserved in a particular transaction. A new reservation is made if $t \leq a$. If $a < t \leq (a + r)$, T responds "try_later." If $0 < (a + r) < t$, T responds "too_many." If $a = r = 0$, T responds "sold_out."

To reduce the complexity of verifying the system, we made several simplifications to exclude large uninteresting execution paths:

1. The system has a maximum two customers and two agents. This simplification still preserves the distribution, interconnectivity, concurrency and non-determinism of an e-system.

2. Only one ticket is left for sale. This may appear to be a significant restriction however, it retains the most interesting executions as most problems arise only when supply cannot satisfy demand.

3. Computing resources are limited. An agent can handle only two pending reservations and the communication queue of each process can hold only one message at a time. This restriction is realistic because e-processes always have limited resources.

4. Execution paths after Ticket Server's "try_later" or "too_many" responses are not considered because once these responses are delivered to the customer, the system state becomes the same as the state preceding the reservation.

5. Only one reservation cancellation is considered because repeated cancellations do not create new situations.

2.6.2 Properties

There are both safety and liveness properties of the system. Sample safety properties are:

1. When the ticket server closes, the number of tickets sold by the server equals to the number of tickets bought by customers.
2. At all times the sum of the numbers of reserved, available and sold tickets always equals to the total number of tickets.

Sample liveness properties are:

1. Every customer request is eventually responded to.
2. If customers reserve all the tickets and their payments are all approved, the tickets are eventually sold.

2.6.3 Verification

We applied two model checkers, VeriSoft and SPIN[6], and verified the e-ticket sales system in two stages – modeling both well- and ill-behaved customers.

2.6.3.1 Model Checkers Applied

VeriSoft was chosen because it can verify programs written in C/C++, two of the most popular languages. Auditors can simply embed safety assertions in C/C++ code using VS_assert() and model non-deterministic events using VS_toss().

The biggest advantage of VeriSoft is that it allows the verification of an existing C/C++ program with minimal modifications to it. This is important because most current

[6]http://www.bell-labs.com/projects/verisoft/
http://netlib.bell-labs.com/netlib/spin/.

programs are coded without formal verification. VeriSoft can be a handy tool for auditors to perform add-on formal verification of client e-systems.

Moreover, VeriSoft has the minimal memory requirement because it makes no attempt to remember visited states. This gives flexibility but at a price, i.e., inefficiency due to the need for visiting and verifying the same state repeatedly. In extreme, VeriSoft may fail to cover the complete state space in a reasonable time even if the state space is relatively small. To limit the problem, VeriSoft provides VS_abort to manually prune execution paths, but this pruning makes verification less automatic.

Several other model checkers like SPIN avoid the revisit problem by remembering visited states and consequently require more memory capacity. Remembering visited states is only possible if the model checker knows the exact state space structure. C/C++ are too general to meet this specification. Therefore, these model checkers usually have their own languages to specify the system model, e.g., SPIN uses Promela. Since Promela is similar to C, it was relatively easy to convert our C code into the Promela model. Promela is powerful and efficient; the Promela model is richer than the C model yet implemented with fewer coding. SPIN translates the Promela model into an internal Kripke structure and searches for paths leading to bad states or infinite loops.

Similar to VeriSoft, SPIN performs a depth-first search on the state space. But at each state it only considers all possible transitions that lead to not-yet-visited states. SPIN can verify safety properties by checking the assertions embedded in the Promela code and liveness properties by checking if all infinite execution loops go through a “progress” transition infinitely many times. For instance, a customer can keep reserving and canceling reservations, causing infinite loops. A sophisticated system should be designed not to serve such a customer after a few reserve-cancel cycles. If not, SPIN can detect the infinite cycling of reserve-cancel as a livelock unless the reserve-cancel transition is marked with a “progress” label.

VeriSoft can verify C/C++ programs because of its add-on features. SPIN can verify systems in any language but the system model has to be written in Promela. In any case, once the system properties are defined and expressed and the system/system model is implemented, verification by model checkers is automated, efficient and worthwhile. For a detailed discussion on how VeriSoft and SPIN differ, see [1].

2.6.3.2 Finding the Bugs

At stage one, we modeled the customer process as well-behaved. A customer makes one reservation at a time; after a reservation, the customer waits for the confirmation or rejection. In case of confirmation, the customer either pays or cancels the reservation. No bugs were found in this predictable, normal situation.

At stage two, we modeled somewhat abnormal customer behaviors and two bugs were identified: 1) a deadlock occurred and shut down the system when a customer (hacker) constantly made reservations, and 2) the property "every request is eventually responded to" was violated when a customer wanting a single ticket made two reservations simultaneously with two agents, a situation not unusual for Internet users.

VeriSoft and SPIN were both used at stage one. Only SPIN was used in stage two because of its greater power and efficiency.

Deadlock When we modeled a hacker process which submitted reservations continuously, SPIN identified a design flaw resulting in a deadlock. XSPIN, SPIN's graphic tool, helped pinpoint the scenario causing the deadlock (Figure 2.12).

Process 0 initiated the agent, hacker, and ticket server processes (numbered 1, 2 and 3 respectively). In state 46, the agent received the third reservation request from the hacker. Since the agent had limited entry space in his reservation database table to hold pending reservations (only a maximum of two in our model), in state 58, he tried to send a

"table_full" message to the hacker. The message was never read because the hacker, busy sending out reservation requests, refused to process any message in his communication queue. In state 60, the agent received a "granted" message from the ticket server responding to the first reservation and in state 63, the agent tried to notify it to the hacker. But this message cannot be sent because the hacker's communication queue was full (only one message can be held in our model). Also in state 63, the ticket server intended to send a "try_later" message to the agent responding to the second reservation. It reached the agent's communication queue but was not read because the agent was still trying to contact the hacker. Simultaneously, the hacker wanted to send another reservation request but failed because the agent's communication queue was full. So the agent and hacker were waiting for each other in a deadlock. Two reasons for the deadlock: limited resources (common for Web businesses) and the bad design of processes failing to handle multi-tasking.

Deadlocks are typical problems in distributed systems but hard to find or reproduce because they often occur in corner cases. We have seen that deadlocks can result in unexpected system shutdowns. Hackers often exploit a deadlock for Denial-of-Service (DoS) attacks.

After analyzing this deadlock scenario, we introduced "time_out" to break the loop – an agent failing to send out a message within a period of time will handle another task. With this addition the system can proceed even with such a hacker process. Similarly, the system is also modified to ignore the hacker's subsequent reservations if he does not pay for the granted reservation within a reasonable time. "Time_out" is commonly used in networking protocols but not often in e-business applications. This example shows its importance.

Specification flaw SPIN also identified a minor specification flaw. The problem occurred in a corner case when the system has only one customer who wants only one ticket but

makes two reservations simultaneously with two agents and all tickets are sold right after the reservations.

Agent A, handling the customer's first reservation, contacts the ticket server immediately. Knowing that no more ticket is available, A quickly notifies the customer. Hearing the bad news, the customer leaves the system disregarding his another reservation to Agent B. During this period, B, unfortunately overloaded or blocked, does not notice the customer's request. When B does notice and forwards it to the ticket server, the request remains unresponded because the server closes when the system has no more tickets or customers. This violates the property "every request is eventually responded to." To accommodate such a situation, we redefined the specifications.

Our uncovering these flaws in the simplified system model is not hard. VeriSoft considered 331,078 system states and the verification, running on a low-end 400 MHz Intel Xeon server, took about only half an hour. SPIN, which is a more powerful tool, found the problem in seconds, once the model has been expressed in SPIN's language. This demonstrates both the complexity of e-processes and the feasibility of using model checking. Model checking can yield measurable business reliability for a reasonable amount of effort. Traditional auditing methods, in contrast, can hardly do so. Physically observing system behavior is impossible because of fast transaction speed and potentially large transaction volume. Inquiring employees, including designers and developers, will not uncover the aforementioned flaws, because they should have programmed the countermeasures if they had known. Examining documents or reprocessing data are reactive methods, too late to prevent these flaws from resulting in negative impacts. Analytical procedures are limited and time-consuming to pinpoint the execution paths leading to the bad states, considering over 300,000 states in even such a simple model. Only model checking can provide the confidence that e-businesses desire.

2.7 Conclusion

In the digital economy, software applications largely determine how businesses live their digital lives. For those e-processes which have become the brains and nerves of an e-business, their correctness is crucial. Fortunately, it is plausible in both theory and practice to mathematically verify the correctness of e-processes. Model checking, an automated and efficient formal verification method, can perform such a function.

Our research brings a method from another discipline to advance the field of auditing in the context of critical e-commerce systems. As auditing moves from a reactive ex-post audit process to a proactive continuous one, we foresee that model checking will become a valuable and practical tool. It is rigorous because it can verify system correctness under all circumstances. If used correctly, model checking can help locate and correct hard-to-anticipate but potentially crucial flaws that are often impossible to identify using conventional control and auditing methods.

Applying formal methods in auditing is not new. TICOM [27] was a formal method to analyze accounting information systems and detect control problems. Model checking is much more powerful and efficient for analyzing general information systems. For critical e-systems, it should be an essential part of internal control and a supporting tool of external assurance.

The auditors have many key advantages in applying model checking:

1. their potentially strategic roles in advocating formal verification to the management and revolutionizing e-business development practices;

2. their experience with business processes and professional expertise in defining relevant and complete system properties;

3. their well-established reputation for competence and independence in providing credible trust services.

Although model checking is powerful, it has limitations. Model checking can guarantee system correctness with regard to certain verifiable properties but it is difficult to provide assurance over the entire system. Limiting factors include: the complexity of business systems, the difficulty of system modeling, the limited expressiveness of formal presentation languages, and the complexity and cost of current reasoning procedures. To combat these limitations, there are ongoing researches, such as on machine-checkable logics of authentication [28], security property specifications of e-commerce protocols [29], and testbeds to experiment how to assist people without training in formal techniques to effectively apply model checking in business[16].

In summary, this research helps to eradicate two of the three "practice barriers" to internal control quality assurance [30] – the lack of adequate criteria for measuring internal control quality and the lack of methods for auditing a process. We suggest auditors first apply model checking to mission-critical and pervasive business applications (e.g., significant financial trading processes and essential building blocks for e-commerce applications) and focus on the critical properties (e.g., by analogy, in the Titanic case, the strength of steel rather than the color coordination of decorations). Auditors can outsource the verification over hardware, operating systems, application development tools and commercial business systems like SAP, to computer specialists but instead focus on the verification of business applications unique to the client. Although reliance and independence issues will arise, such a practice is not impossible.

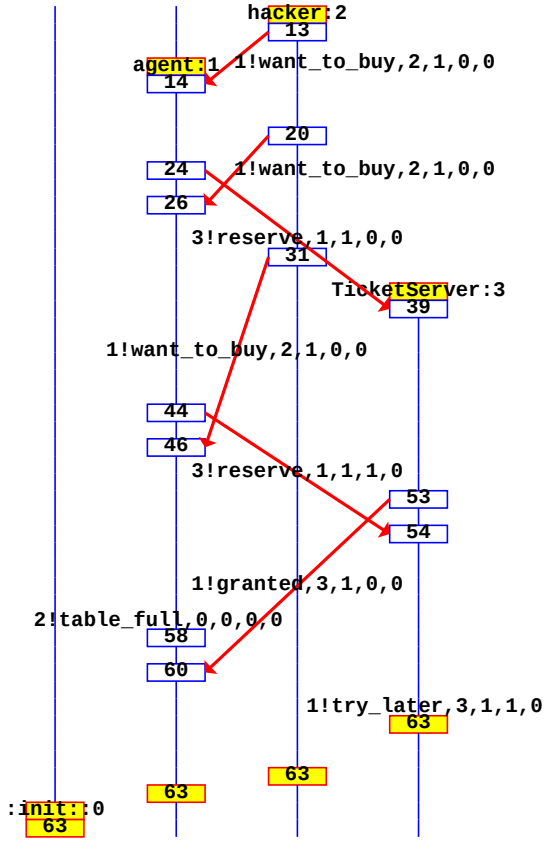


Figure 2.12: A scenario in the e-ticket sales example causes a deadlock. The figure is based on a Message Sequence Chart window of XSPIN. Vertical lines represent processes, boxes represent states. Messages are labelled by "receiving process!request/reply, sending process, and other information." If a message is only sent but never received, i.e., the message is taken out of the communication queue and read by the receiving process, the label appears near the sender. Otherwise, an arrow goes from the sender to the receiver and the label is near the arrow.

Chapter 3

Buy-price English Auction

Abstract

Consider an English auction for a single object in which there is an option for a bidder to guarantee a purchase at a seller-specified buy price b at any time. We show that there exist $\tilde{v}$ and $\hat{v}$ ($\geq \tilde{v}$), such that a bidder purchases at the buy price immediately if his valuation v is no less than $\hat{v}$ or $\tilde{v} \leq v < \hat{v}$ and at least one other bidder is participating in the auction. If $b \leq v < \tilde{v}$, he purchases at the buy price once the current bid reaches a strategically chosen threshold price. A properly set buy price increases expected social welfare and the expected utility of each agent when either buyers or seller are risk-averse.

3.1 Introduction

The popularity, scope, and competitiveness of online auctions have encouraged auctioneers to innovate. Particularly noteworthy is the use of a buy price in English auctions where the seller announces a maximum bid level, at which any bidder can immediately win the auction. Since the start of such auctions in 1999[1], there has emerged a significant portion of sellers who choose to utilize buy prices [31, 32][2]. For example, our preliminary study of over seven thousand sports rookie card auctions at Yahoo! suggests that about half of the

[1] Yahoo! started offering auction sellers the option to utilize buy prices in 1999 whereas eBay implemented its version of the buy price practice in November 2000.

[2] The list of data on the proportion of auctions with buy prices: eBay 2001 data shows 30% in 1Q 2001, 35% in 2Q 2001, and 45% in Dec 2001 [32]. 40% on eBay and 66% on Yahoo! in 2002 [31]. 37% on Bid or Buy in 2001 [32].

auctions utilized buy prices and that approximately one fourth of them ended with bidders exercising that option.

Wang [33] shows that an auction yields higher expected seller revenue than a posted-price sale when the auction is costless. Auctioning online is very affordable, and its popularity indicates that many sellers have recognized the superiority of auctions over posted-price sales. Then why would the seller prefer to specify a posted price when dynamic pricing is in play and consequently restrict her maximum payment? Should bidders also favor buy prices? If so, what are their equilibrium strategies for such auctions?

Before addressing these issues, we need to clarify the categories of buy prices. Currently, there are three types of buy prices: "permanent," "temporary," and "limited." A permanent buy price remains valid during the entire course of the auction. Yahoo!'s "buy now," uBid's "uBuy It," and Amazon's "Take-It" prices all fall into this category. Notably missing from this list is eBay, which offers only a temporary "buy-it-now" feature that disappears as soon as any bid is made at or above the reserve price. A buy price is limited if it is valid only for a restricted period of time during the auction[3].

This paper focuses on the study of permanent buy prices because they allow for an ultimately hybrid model combining posted-price sales and auctions. For simplicity, we generally refer to the "permanent" buy price as the buy price in the following discussions, and we specify other types explicitly.

Budish and Takeyama [34] are the first to have recognized the benefits of buy prices. Using a simple two-bidder, two-value model, they conclude that with a properly set buy price, the seller facing risk-neutral buyers earns the same expected profit as in a standard English auction, but the seller facing risk-averse buyers gains higher expected profit. Reynolds

[3]For example, labx.com, specializing in the auctions of scientific equipment, requires that "an Auction Stop bid must be entered 48 hours prior to the auction close or the Auction Stop feature is dropped and the auction continues to the ending date specified."

and Wooders [31] confirm this result in a model of two bidders with uniform distribution.

But Budish and Takeyama [34] express doubts about the extension of their results to a general setting. They conjecture that "in a more general framework with n valuations, the optimal buy price may be less than the second-highest valuation, which admits the possibility of inefficient outcomes. In this case, revenue equivalence breaks down and the effectiveness of the buy price to enhance sellers' profits when bidders are risk-averse may be diminished."

We show that Budish and Takeyama's doubts are unfounded. We prove that, in a setting of n bidders with arbitrary continuous value distribution, if a buy price is properly set, revenue equivalence still holds when agents are risk-neutral, and the buy price still enhances sellers' profits when bidders are risk-averse. We further prove that if neither sellers nor bidders are risk takers, an English auction augmented with a properly set buy price weakly dominates the standard English auction. And more surprisingly, a buy-price English auction not only increases the expected social welfare, but also ensures that the expected utility of each agent is never lower than it is in a standard auction. Particularly when either the seller or buyers are risk-averse, the seller's expected utility is strictly higher than that in a standard auction and without lowering the buyers' expected utilities.

Utilizing buy prices in auctions can be viewed as providing a form of insurance for risk-averse buyers. If these buyers' valuations are above the buy price, they can bid the buy price to achieve a fixed profit instead of taking the risk of losing the item when bidding below the buy price. These buyers would pay premiums for avoiding the risk. Similar to how insurance companies make money, the seller utilizing buy prices profits by exploiting the risk aversion of these buyers. But unlike insurance, the seller does not have to be risk-neutral to benefit from buy prices; a risk-averse seller can gain even more because utilizing buy prices reduces the variance of seller revenue.

To prove the superiority of a buy-price English auction, we need to define buyers'

equilibrium strategies. We will prove that a unique Bayesian Nash equilibrium exists for buyers when there are unique reference points $\tilde{v}$ and $\hat{v}$ ($\tilde{v} \leq \hat{v}$, and both above the buy price) so that a bidder with a valuation between the buy price and $\tilde{v}$ is a threshold bidder who exercises the buy price once the current high bid reaches a strategically chosen threshold price, a bidder with a valuation between $\tilde{v}$ and $\hat{v}$ is a conditional bidder who bids the buy price immediately on the condition that at least one competing bidder bids at or above the reserve price, and a bidder with a valuation above $\hat{v}$ is an unconditional bidder who selects to purchase at the buy price instantly with no conditions. We will prove that a lower bound exists so that if the buy price is at or above this bound, then all bidders with valuations above the buy price are threshold bidders. In this case the auction is efficient; it guarantees that the bidder with the highest valuation wins, because a bidder with a higher valuation has a lower threshold price. Moreover, the more risk-averse a bidder, the lower his threshold price. The seller thus has higher expected utility from risk-averse buyers than from risk-neutral buyers.

The paper proceeds as follows. In Section 3.2, we lay out the model, state and prove the bidders' unique Bayesian Nash equilibrium strategies. We also compare behaviors of bidders with different degrees of risk aversion. In Section 3.3, we prove that both risk-averse and risk-neutral bidders are not worse off in a buy-price English auction. We analyze the impact of the seller's risk preference on the use of buy prices and prove that the seller is never worse off utilizing properly set buy prices. We also derive the lower bound of a properly set buy price. Section 3.4 provides more intuitions about our results and recommends future research directions.

3.2 Bidders' Equilibrium Strategy

3.2.1 The Model

There is one seller and n bidders in a buy-price English auction of an indivisible good. Only bids at or above the reserve price are valid, and the seller has committed not to relist

the item if no valid bid emerges. This no-resale constraint is a standard assumption and is naturally satisfied in cases of perishable or time-sensitive goods like flowers or tickets. We also assume bidders have independent private valuations. This assumption is restrictive but is closely emulated by auctions of collectibles or used goods. Our empirical study also supports such an assumption because the data show that most sports rookie cards purchased through auctions are for collection rather than for resale – buyers seldom resell the cards they have just purchased.

To simplify the analysis, we use a "modified English clock auction" as our model, which has a set of rules as follows:

- The seller announces both a reserve price and a buy price before the auction starts. The auction starts at a pre-announced time with the auction clock being set at the reserve price. Each bidder controls two buttons: a "bid" button and a "buy" button. A bidder signals his willingness to pay the current clock price by pressing and holding down his "bid" button. Once a bidder releases his "bid" button, he quits the auction and can no longer return. A bidder does not know how many other bidders participate in the auction. At any time, a bidder can press his "buy" button signaling that he bids the buy price. A bidder can start signaling his actions even shortly before the auction starts.

- At the start of the auction, the auctioneer checks the state of bidders' buttons. If only one "buy" button is pressed, the auction ends and the bidder who has pressed his "buy" button wins, paying the buy price. If more than one "buy" button is pressed, the winner is randomly chosen among those who have pressed their "buy" buttons. If none of the "buy" and "bid" buttons is pressed, the auction ends without a sale. If there is no "buy" button being pressed but one "bid" button is being held, the auction ends and the bidder who holds his "bid" button wins, paying the reserve price. If

there is no “buy” button being pressed but more than one “bid” button being held, the auction clock starts ascending from the reserve price.

- The auction terminates when one of the following scenarios occur: 1) There is only one bidder left holding his “bid” button. This bidder wins and pays the current clock price. 2) There is a bidder who has pressed his “buy” button. This bidder wins, paying the buy price. When more than one bidder have pressed his “buy” button simultaneously, the winner is chosen randomly among them. And 3) The auction clock reaches the buy price. The winner is chosen randomly among the bidders who hold their “bid” buttons, and the winner pays the buy price.

3.2.2 Proof of Bidders’ Equilibrium Strategies

In a buy-price English auction, for a bidder with a valuation below or equal to the buy price, his pure and dominant strategy remains the same as in a standard English auction, i.e., to bid up to his valuation. However, the strategy space for a bidder with a valuation above the buy price becomes more complicated. When there are multiple bidders with such valuations, the winner will be the one who first commits to the buy price. If such a bidder thinks that at least one other bidder exists who might bid the buy price, he would find the appropriate moment to bid the buy price before any other bidder. If he thinks that there is no other bidder who might use the buy price, he would simply keep bidding. Consequently, there is no dominant strategy for such a bidder; we could only hope to find a Bayesian Nash equilibrium.

To find such an equilibrium, we first need to characterize all possible pure strategies that a bidder can follow:

- *Traditional*: Bid up to his valuation;

- *Threshold*: Keep bidding until winning or his threshold price is reached. Once the auction clock reaches his threshold price, bid the buy price immediately.
- *Conditional*: Bid, but use the buy price immediately if at least one other bidder bids at or above the reserve price.
- *Unconditional*: Bid the buy price immediately with no conditions.

We can unite the above four strategies under a "generalized threshold strategy" in which each bidder has a threshold price that determines if and when the bidder uses the buy price. If the buy price is above a bidder's valuation, he never uses the buy price and hence we can assume he has a threshold price above his valuation. We can regard that a bidder following the traditional strategy as having a threshold price equal to the buy price. A bidder following the threshold strategy has a threshold price dependent on his valuation and the buy price (as we will show later). A bidder following the conditional strategy can be regarded as having a threshold price equal to the reserve price[4], and a bidder following the unconditional strategy can be regarded as having a threshold price less than the reserve price, say, the lowest bidder valuation.

Let r denote the reserve price and b the buy price. Let us assume bidders' valuations are drawn randomly from the same cumulative probability distribution F, which is strictly increasing and differentiable over the support of bidder valuations, $[\underline{v}, \overline{v}]$. Let $f = F'$ denote the probability density. Let $u(x)$ denote the bidder's von Neumann-Morgenstern utility function, where x is the difference between the buyer's valuation and his payment if he wins and is zero if otherwise. Let $s(p)$ denote the seller's utility when she sells the item

[4]Conditional bidders use the buy price immediately upon learning that they have competition thus they cannot obtain the item at the reserve price. They are different from unconditional bidders, because unconditional bidders give up the chance to obtain the item at the reserve price, but in exchange, they are guaranteed to win over any conditional bidders. Conditional bidders observe the auction clock and bid the buy price as soon as they notice the auction clock departing from the reserve.

and receives the payment p. Both $u(x)$ and $s(p)$, we assume, are linear or concave, twice continuously differentiable, and strictly increasing. Let v_s be the seller's valuation for the item, assuming $u(0) = s(v_s) = 0$ and $u'(0) = s'(v_s) = 1$.

The following theorem defines the bidders' unique Bayesian Nash equilibrium (see Figure 3.1):

Theorem 3.1 *A buy-price English auction has a unique Bayesian Nash equilibrium determined by constants $\tilde{v}$ and $\hat{v}$ and function t, $b < \tilde{v} \leq \hat{v} \leq \overline{v}$, such that all bidders with valuation v have the following strategies:*

- *Use the traditional strategy if $v < b$.*
- *Use the threshold strategy with a threshold price $t(v,b) \in (r,b]$ if $b \leq v < \tilde{v}$, where $t(v,b)$ is the threshold function defined by the differential equation*

$$\frac{u(v - t(v,b))}{u(v-b)} - 1 + \frac{F^{n-1\prime}(v)}{\partial_1 (F^{n-1} \circ t)(v,b)} = 0, \quad \partial_1 t(b,b) = -1, \quad t(b,b) = b$$

- *Use the conditional strategy if $\tilde{v} \leq v < \hat{v}$.*
- *Use the unconditional strategy if $\hat{v} \leq v$.*

All bidders with $v \geq b$ follow the threshold strategy, i.e., $\tilde{v} = \hat{v} = \overline{v}$, if and only if $\lim_{v \to \overline{v}} t(v,b) \geq r$.

PROOF. We need to obtain the necessary and sufficient conditions for t, $\tilde{v}$, and $\hat{v}$ to determine a symmetric Bayesian Nash Equilibrium. First we assume t, $\tilde{v}$, and $\hat{v}$ determine a pure strategy equilibrium. Under this assumption we calculate bidders' expected profits corresponding to the different strategies, prove $t(v,b)$ is both strictly decreasing and continuous in v when $b \leq v < \tilde{v}$, and show how to compute $t(v,b)$. We then prove such a $t(v,b)$

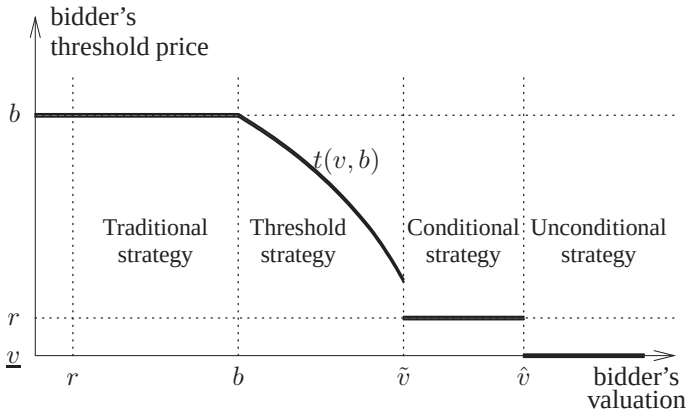


Figure 3.1: Bidders in a buy-price English auction follow one of the four equilibrium strategies dependent on their valuations.

indeed corresponds to a unique symmetric Bayesian Nash Equilibrium. Further, we prove the existence of unique reference points $\tilde{v}$ and $\hat{v}$ and show how to compute them.

Since we are concerned with strategies for a bidder with $v \geq b$, let us first calculate such a bidder's expected profit under the three different strategies:

1. Under the unconditional strategy:

A bidder using the unconditional strategy competes only with other bidders using the same strategy. The probability that a bidder uses the unconditional strategy is $1 - F(\hat{v})$, and the probability that there are exactly $k-1$ other bidders ($1 \leq k \leq n$) who also use the unconditional strategy is $\binom{n-1}{k-1}(1-F(\hat{v}))^{k-1}F^{n-k}(\hat{v})$. Hence, an unconditional bidder's expected profit, denoted by $\Pi_{\mathrm{u}}(v)$, is

$$\begin{aligned} \Pi_{\mathrm{u}}(v) &= \sum_{k=1}^{n} \binom{n-1}{k-1} (1-F(\hat{v}))^{k-1} F^{n-k}(\hat{v}) \frac{u(v-b)}{k} \\ &= \frac{1-F^n(\hat{v})}{n(1-F(\hat{v}))} u(v-b) = \frac{u(v-b)}{n} \sum_{k=0}^{n-1} F^k(\hat{v}) \end{aligned} \tag{3.1}$$

2. Under the conditional strategy:

A bidder using the conditional strategy wins if 1) there are no unconditional bidders and 2) he is chosen randomly among the conditional bidders who bid the buy price simultaneously. A conditional bidder pays b if there is another valid bid and pays r if there are no valid bids. Hence, a conditional bidder's expected profit, denoted by $\Pi_c(v)$, is

$$\begin{aligned}\Pi_c(v) &= \sum_{k=2}^{n} \binom{n-1}{k-1} (F(\hat{v}) - F(\tilde{v}))^{k-1} F^{n-k}(\tilde{v}) \frac{u(v-b)}{k} + \\ &\quad u(v-b)(F^{n-1}(\tilde{v}) - F^{n-1}(r)) + u(v-r)F^{n-1}(r) \\ &= \frac{u(v-b)}{n} \sum_{k=0}^{n-1} F^k(\hat{v}) F^{n-k-1}(\tilde{v}) + (u(v-r) - u(v-b))F^{n-1}(r) \qquad (3.2)\end{aligned}$$

3. Under the threshold strategy:

A bidder with a threshold price p wins and pays b if there are neither unconditional nor conditional bidders and he is chosen randomly among the threshold bidders who bid the buy price simultaneously once the current high bid reaches p. Alternatively, he wins and pays the second-highest bid (or the reserve if he is the only bidder with a valuation at or above the reserve) if all other bidders have valuations below p. Let $G_{n-1}(p)$ be the probability that a bidder with a threshold price p exercises the buy price and wins the auction[5]. A threshold bidder's expected profit, denoted by $\Pi_t(v,p)$, is

$$\Pi_t(v,p) = u(v-b)G_{n-1}(p) + \int_r^p u(v-x)\, \mathrm{d}F^{n-1}(x) + u(v-r)F^{n-1}(r) \qquad (3.3)$$

We now establish two properties of the threshold function $t(v,b)$. To simplify the notations in the following proofs, we define t for the full range of bidder valuations: $t(v,b) = b$ when $v \le b$, $t(v,b) = r$ when $\tilde{v} \le v < \hat{v}$, and $t(v,b) = \underline{v} < r$ when $\hat{v} \le v$.

Proposition 3.2 *$t(v,b)$ is strictly decreasing in v for $b \le v < \tilde{v}$.*

[5]If multiple bidders use the same threshold price p, the winner is selected randomly among them. This will be reflected in $G_{n-1}(p)$. Later we prove that in equilibrium there is zero probability that multiple threshold bidders share the same threshold price.

PROOF. See Section 3.5.1. ■

Proposition 3.2 implies that for any threshold price p, $r < p < b$, there is at most one bidder valuation corresponding to the equilibrium threshold price p; i.e., there is at most one v s.t. $t(v, b) = p$. Thus in equilibrium it is impossible for two threshold bidders to have the same threshold price. This allows us to express $G_{n-1}(p)$, which denotes the probability that a bidder with threshold price p uses the buy price and wins. It occurs if no one else has a threshold price lower than p, excluding the case where the bidder wins without using the buy price because everyone else has a valuation less than p:

$$G_{n-1}(p) = \big(1 - \text{Prob}(t(v, b) \leq p)\big)^{n-1} - F^{n-1}(p)$$

Defining $T(p) = \text{Prob}(t(v, b) \leq p)$, we can write $G_{n-1}(p)$ as

$$G_{n-1}(p) = (1 - T(p))^{n-1} - F^{n-1}(p) \tag{3.4}$$

Note that $t(v, b)$ strictly decreasing for $b \leq v < \tilde{v}$ implies that $T(p)$ is continuous and strictly increasing for $r < p \leq b$, which in turn implies that $G_{n-1}(p)$ is continuous and strictly decreasing for $r < p \leq b$. This can be used to prove the following proposition:

Proposition 3.3 *$t(v, b)$ is continuous in v for $b \leq v < \tilde{v}$.*

PROOF. See Section 3.5.2. ■

Note that the equilibrium threshold function cannot continuously drop to the reserve price if there is a positive probability that there are conditional bidders. This is because a bidder with a threshold slightly above the reserve can switch to the conditional strategy, thus increasing his chance of winning by more than a fixed positive amount while his payment conditional on winning would hardly change.

Using the above two propositions we can now have the following proposition[6].

Proposition 3.4 *Any threshold price function $t(v,b)$ corresponding to an equilibrium satisfies the following differential equation:*

$$\begin{aligned} \frac{u(v-t(v,b))}{u(v-b)} - 1 + \frac{F^{n-1\prime}(v)}{\partial_1(F^{n-1}\circ t)(v,b)} &= 0 \quad \text{if } b \le v < \tilde{v} \\ \partial_1 t(b,b) = -1, \quad t(b,b) &= b \end{aligned} \tag{3.5}$$

PROOF. In equilibrium, when all other bidders follow the threshold strategy determined by t, the optimal threshold price of a bidder with valuation v ($b \le v < \tilde{v}$) is $t(v,b)$, i.e., $p^* = t(v,b)$ maximizes $\Pi_t(v,p)$. Differentiating a threshold bidder's expected utility function (3.3) in p, we get

$$\frac{\partial \Pi_t(v,p)}{\partial p} = u(v-b)G'_{n-1}(p) + u(v-p)F^{n-1\prime}(p) \tag{3.6}$$

Differentiating $G_{n-1}(p)$ gives

$$G'_{n-1}(p) = -(n-1)(1-T(p))^{n-2}T'(p) - F^{n-1\prime}(p)$$

Because $t(v,b)$ is strictly decreasing and continuous in v when $b \le v < \tilde{v}$, its inverse function in the first variable, denoted by $w(\cdot,b)$, exists:

$$t(w(p,b),b) = p$$

Using w, $T(p)$ can be expressed as

$$T(p) = 1 - F(w(p,b))$$

and we have

$$T'(p) = -\frac{f(w(p,b))}{\partial_1 t(w(p,b),b)}$$

[6]In the interest of simplicity, we assume that t is continuously differentiable in both variables v and b so long as it is above r.

Replacing $T(p)$ and $T'(p)$ in $G'_{n-1}(p)$, we get

$$\begin{aligned} G'_{n-1}(p) &= (n-1)F^{n-2}(w(p,b))\frac{f(w(p,b))}{\partial_1 t(w(p,b),b)} - F^{n-1\prime}(p) \\ &= \frac{F^{n-1\prime}(w(p,b))}{\partial_1 t(w(p,b),b)} - F^{n-1\prime}(p) \end{aligned}$$

Therefore, we have

$$\frac{\partial \Pi_t(v,p)}{\partial p} = u(v-p)F^{n-1\prime}(p) + u(v-b)\left(\frac{F^{n-1\prime}(w(p,b))}{\partial_1 t(w(p,b),b)} - F^{n-1\prime}(p)\right)$$

For $v > b$, dividing the above equation by $u(v-b)F^{n-1\prime}(p) > 0$ preserves its sign, and we get

$$\frac{u(v-p)}{u(v-b)} - 1 + \frac{F^{n-1\prime}(w(p,b))}{\partial_1(F^{n-1}\circ t)(w(p,b),b)} \tag{3.7}$$

When t is the equilibrium threshold function, i.e., $p^* = t(v,b)$ and $w(p^*,b) = v$, (3.7) is zero:

$$\frac{u(v-t(v,b))}{u(v-b)} - 1 + \frac{F^{n-1\prime}(v)}{\partial_1(F^{n-1}\circ t)(v,b)} = 0$$

For $v = b$, $t(v,b)$ is continuous in v and $t(b,b) = lim_{v\searrow b} t(v,b) = b$. By our assumption, t is continuously differentiable. Applying the L'Hospital's rule, we have

$$\lim_{v\searrow b}\frac{u(v-t(v,b))}{u(v-b)} = \frac{u'(0)(1-\partial_1 t(b,b))}{u'(0)} = 1 - \partial_1 t(b,b)$$

We can use this to take the limit in (3.5) as $v \searrow b$ to obtain $\partial_1 t(b,b) = \pm 1$. Since t is decreasing, $\partial_1 t(b,b) = -1$ must hold. ■

Equation (3.5) is an ordinary differential equation for $t(\cdot,b)$ with the boundary condition $t(b,b) = b$. The equation always has a unique solution of $t(v,b)$. Although we cannot express the general solution explicitly, bidders in practice can apply (3.5) to calculate their optimal threshold prices once the characteristics of an auction (e.g., bidders' value distribution, utility functions, and the seller's buy price) become known. We will

demonstrate the use of (3.5) when a bidder has Constant Absolute Risk Aversion (CARA) utility [35].

Also using (3.5) we can further verify Proposition 3.2. Since $u(v - t(v,b)) > u(v-b)$, from (3.5) we get

$$\frac{F^{n-1\prime}(v)}{F^{n-1\prime}(t(v,b))\partial_1 t(v,b)} = 1 - \frac{u(v-t(v,b))}{u(v-b)} < 0$$

which implies $\partial_1 t(v,b) < 0$.

Now we prove that the threshold function defined by (3.5) is the best response threshold value.

Proposition 3.5 *Let t be the function defined by (3.5) and $\tilde{v} > b$ satisfy that for all $x < \tilde{v}$ that $t(x,b) > r$. If all other bidders with valuations x, $b \le x < \tilde{v}$, follow the threshold strategy $t(x,b)$, then the optimal threshold strategy of a bidder with valuation v, $b \le v < \tilde{v}$, is to use $t(v,b)$ as his threshold price.*

PROOF. To show that $t(v,b) = p^*$ maximizes the expected profit of a bidder with v ($b \le v < \tilde{v}$), it is enough to show that $\frac{\partial \Pi_t(v,p)}{\partial p}$ is positive if $p < t(v,b)$ and is negative when $p > t(v,b)$. From the detailed proof in Section 3.5.3, we get $\Pi_t(v,p)$ strictly increasing for all $p \in (r, t(v,b))$ and strictly decreasing for $p \in (t(v,b), b)$. This proves that, as long as all other bidders use the threshold strategy, $p^* = t(v,b)$ maximizes $\Pi_t(v,p)$ for a given v, $b \le v < \tilde{v}$; that is, the optimal threshold price of a bidder with valuation v is $t(v,b)$. ■

Corollary 3.6 *If $t(v,b)$ defined by (3.5) satisfies*

$$\lim_{v \to \overline{v}} t(v,b) \ge r$$

then it is an equilibrium that all bidders with valuation $v \ge b$ use the threshold strategy with $t(v,b)$ as their threshold price.

PROOF. Proposition 3.5 with $\tilde{v} = \hat{v} = \overline{v}$ shows that a bidder cannot improve his profit by using a different threshold strategy. We now show that he cannot improve his profit by switching to either the conditional or the unconditional strategies. To do so, it is sufficient to show

$$\Pi_t(v, t(v,b)) > u(v-r)F^{n-1}(r) + u(v-b)(1-F^{n-1}(r)) > u(v-b)$$

The middle of the inequality above is the bidder's maximum possible profit using the conditional strategy: a conditional bidder reaches his maximum possible expected profit when he pays r if everyone else has a valuation below r and pays b otherwise. It is higher than the maximum profit possible using the unconditional strategy, which is $u(v-b)$.

When all other bidders follow the threshold strategy, by Proposition 3.5, we have

$$\Pi_t(v, t(v,b)) > \lim_{p \to r} \Pi_t(v,p)$$

and using (3.3) and $G_{n-1}(p) = F^{n-1}(\tilde{v}) - F^{n-1}(p)$, we have $\lim_{p \to r} \Pi_t(v,p) = u(v-b)(1-F^{n-1}(r)) + u(v-r)F^{n-1}(r)$. ■

Corollary 3.7 *If $t(v,b)$ defined by (3.5) satisfies*

$$\lim_{v \to \overline{v}} t(v,b) \geq r$$

then the bidder with the highest valuation wins. When the buyers and seller are risk-neutral, the buyers' and the seller's expected profits are the same as in a standard English auction.

PROOF. By Corollary 3.6, if $\lim_{v \to \overline{v}} t(v,b) \geq r$, all bidders follow the threshold strategy with threshold prices strictly decreasing with their valuations. Hence the bidder with the highest valuation reaches his threshold price first and wins the auction. By the revenue equivalence theorem [36, 37], a buy-price English auction yields the same expected revenues as a standard English auction when the buyers and the seller are risk-neutral. ■

Corollary 3.6 shows that when $\lim_{v\to\overline{v}} t(v,b) \geq r$, it is an equilibrium for all bidders with $v \geq b$ to follow the threshold strategy, i.e., $\tilde{v} = \hat{v} = \overline{v}$. Now we need to prove $\tilde{v} < \overline{v}$ when $\lim_{v\to\overline{v}} t(v,b) < r$.

We can calculate the equilibrium $\hat{v}$ and $\tilde{v}$ using (3.1) and (3.2). First, find the equation that gives $\hat{v}$ for a given $\tilde{v}$.

One of the following must be true: $\Pi_{\mathrm{u}}(v) \geq \Pi_{\mathrm{c}}(v)$ for all $v \geq \tilde{v}$ (only use the unconditional strategy), $\Pi_{\mathrm{u}}(v) \leq \Pi_{\mathrm{c}}(v)$ for all $v \geq \tilde{v}$ (only use the conditional strategy), or $\hat{v}$ satisfies $\Pi_{\mathrm{u}}(\hat{v}) = \Pi_{\mathrm{c}}(\hat{v})$ (use unconditional or conditional strategy respectively in different value ranges). The last case leads to

$$\sum_{k=0}^{n-1} F^k(\hat{v})(1 - F^{n-k-1}(\tilde{v})) = n\Big(\frac{u(\hat{v}-r)}{u(\hat{v}-b)} - 1\Big)F^{n-1}(r) \tag{3.8}$$

For a given $\tilde{v}$, the right-hand side above is strictly decreasing in $\hat{v}$, while the left-hand side is strictly increasing. Therefore, there is either one unique $\hat{v}$ or no $\hat{v}$ solution ($\hat{v} > \tilde{v}$). If there is no $\hat{v}$ solution, then for all $v \geq \tilde{v}$ either $\Pi_{\mathrm{u}}(v) \leq \Pi_{\mathrm{c}}(v)$ (i.e., $\hat{v} = \overline{v}$) satisfying

$$\sum_{k=0}^{n-1} (1 - F^{n-k-1}(\tilde{v})) \leq n\Big(\frac{u(\overline{v}-r)}{u(\overline{v}-b)} - 1\Big)F^{n-1}(r) \tag{3.9}$$

or $\Pi_{\mathrm{u}}(v) \geq \Pi_{\mathrm{c}}(v)$ (i.e., $\hat{v} = \tilde{v}$) satisfying

$$\sum_{k=0}^{n-1} F^k(\tilde{v})(1 - F^{n-k-1}(\tilde{v})) \geq n\Big(\frac{u(\tilde{v}-r)}{u(\tilde{v}-b)} - 1\Big)F^{n-1}(r) \tag{3.10}$$

Define $\Pi_{\mathrm{d}}(v) = \max\{\Pi_{\mathrm{c}}(v), \Pi_{\mathrm{u}}(v)\}$. $\tilde{v}$ is the valuation limit where bidders switch from the threshold strategy either to the conditional or the unconditional strategies; thus, $\Pi_t(\tilde{v}, t(\tilde{v},b)) = \Pi_{\mathrm{d}}(\tilde{v})$. Since both sides are continuous, in order to demonstrate that there is a solution to this equation, it is sufficient to show that $\Pi_t(\tilde{v}, t(\tilde{v},b))$ is greater for $\tilde{v} \to b$,

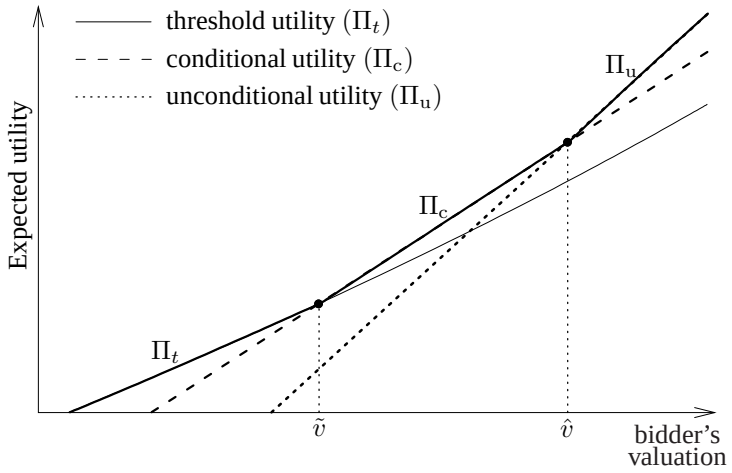


Figure 3.2: The relationship among the slopes of the three expected bidder profit functions under the threshold, conditional, and unconditional strategies guarantees the existence of unique $\tilde{v}$ and $\hat{v}$. For simplicity, we depict the functions as linear. In reality, they are non-linear.

but $\Pi_{\mathrm{d}}(\tilde{v})$ is greater when $\tilde{v}$ is large, s.t., $t(\tilde{v}, b) \to r$. First consider the case where $\tilde{v} \to b$:

$$\lim_{\tilde{v} \searrow b} \Pi_{\mathrm{u}}(\tilde{v}) = 0$$
$$\lim_{\tilde{v} \searrow b} \Pi_{\mathrm{c}}(\tilde{v}) = (b - r)F^{n-1}(r)$$
$$\lim_{\tilde{v} \searrow b} \Pi_t(\tilde{v}, t(\tilde{v}, b)) = \int_r^b u(b - x)\, \mathrm{d}F^{n-1}(x) + (b - r)F^{n-1}(r)$$

which shows that for $\tilde{v} \to b$, $\Pi_t(\tilde{v}, t(\tilde{v}, b)) > \Pi_{\mathrm{d}}(\tilde{v})$ holds.

Let v_x satisfy $t(v_x, b) = r$. Note that $v_x < \overline{v}$ exists here because $\lim_{v \to \overline{v}} t(v, b) < r$. We have

$$\lim_{\tilde{v} \to v_x} \Pi_t(\tilde{v}, t(\tilde{v}, b)) \quad = \quad (u(v_x - r) - u(v_x - b))F^{n-1}(r) + u(v_x - b)F^{n-1}(v_x)$$
$$\leq \quad \Pi_{\mathrm{c}}(v_x) \leq \Pi_{\mathrm{d}}(v_x)$$

Therefore, there exists a $\tilde{v} \in (b, v_x]$ satisfying $\Pi_t(\tilde{v}, t(\tilde{v}, b)) = \Pi_{\mathrm{d}}(\tilde{v})$ and $\tilde{v} < \overline{v}$.

To show that $\tilde{v}$ and $\hat{v}$, together with t, correspond to an equilibrium, we will use the following inequality (see Figure 3.2), which follows easily from equations (3.1–3.3) and the concavity of u:

$$\Pi'_{\mathrm{u}}(v) \geq \Pi'_{\mathrm{c}}(v) \geq \frac{\partial \Pi_t(v,p)}{\partial v} \quad \text{for all } v > b \text{ and } r < p < b \tag{3.11}$$

Intuitively we can think that a bidder's chance of winning decreases in the order of using the unconditional, conditional, and threshold strategies. The marginal expected profit from the unconditional strategy is the highest followed by the conditional and then the threshold strategies. In addition, a bidder's utility conditional on winning the auction is the smallest in the unconditional strategy, followed by the conditional and the threshold strategies. Thus the above inequality holds.

Inequality (3.11) implies that for all p, $\Pi_t(v,p) - \Pi_{\mathrm{d}}(v)$ is non-increasing in v. For any $v < \tilde{v}$,

$$\Pi_t(v, t(v,b)) - \Pi_{\mathrm{d}}(v) > \Pi_t(v, t(\tilde{v},b)) - \Pi_{\mathrm{d}}(v) \geq \Pi_t(\tilde{v}, t(\tilde{v},b)) - \Pi_{\mathrm{d}}(\tilde{v}) = 0$$

which implies that bidders with valuation below $\tilde{v}$ cannot improve their expected utility by switching to the conditional or unconditional strategies. For any $v > \tilde{v}$ and for any p,

$$\Pi_t(v,p) - \Pi_{\mathrm{d}}(v) \leq \Pi_t(\tilde{v},p) - \Pi_{\mathrm{d}}(\tilde{v}) \leq \Pi_t(\tilde{v}, t(\tilde{v},b)) - \Pi_{\mathrm{d}}(\tilde{v}) = 0$$

which implies that bidders with valuation above $\tilde{v}$, following the better of the conditional and unconditional strategies, cannot gain by switching to a threshold strategy.

Inequality (3.11) also implies that $\Pi_{\mathrm{u}}(v) - \Pi_{\mathrm{c}}(v)$ is non-decreasing. Therefore, if $\Pi_{\mathrm{c}}(\hat{v}) = \Pi_{\mathrm{u}}(\hat{v})$, then for all $v \in [\tilde{v}, \hat{v})$ bidders prefer the conditional strategy and, for all $v \geq \hat{v}$, bidders prefer the unconditional strategy. If there is no $\hat{v}$ that satisfies $\Pi_{\mathrm{u}}(\hat{v}) = \Pi_{\mathrm{c}}(\hat{v})$, then either (3.9) or (3.10) holds (i.e., one of the two strategies is always strictly better than the other).

Now we have shown that t, $\tilde{v}$, and $\hat{v}$ indeed determine an equilibrium. ■

We can prove that $\tilde{v}$ and $\hat{v}$ are unique and the equilibrium described in Theorem 3.1 is the only pure strategy symmetric equilibrium of a buy-price English auction. The proof is not difficult and uses similar techniques as in the proof of Theorem 3.1. Since the proof does not provide any more economical insights, we choose to omit it in this paper.

3.2.3 Threshold Prices and the Bidders' Degrees of Risk Aversion

Now we examine the relationship between a bidder's threshold price and his absolute level of risk aversion. Assuming that the bidder's valuation is unchanged, the following theorem proves that the more risk-averse a bidder, the lower his threshold price. In other words, the more risk-averse a bidder is, the sooner he would use the buy price in order to avoid the risk that someone else may use it first.

Theorem 3.8 *Let u_1, u_2 be concave or linear utility functions, t_1, t_2 be the corresponding threshold-price functions, and $a_1 = -u_1''/u_1'$, $a_2 = -u_2''/u_2'$ be the absolute level of risk aversion. If $a_1(x) \leq a_2(x)$ for all $x \geq 0$, then $t_1(v,b) \geq t_2(v,b)$ for all $v \geq b$.*

PROOF. Prove by contradiction: assume that for all $x \geq 0$, $a_1(x) \leq a_2(x)$, but there exists $\beta > b$ such that $t_1(\beta, b) < t_2(\beta, b)$. Let $\alpha = \max\{v : v < \beta \wedge t_1(v,b) = t_2(v,b)\}$. α exists because the set over which we take the maximum is closed, bounded from above, and non-empty (e.g., includes b). Then for all v, $\alpha < v \leq \beta$, $t_1(v,b) < t_2(v,b)$.

Using Lemma 3.14 from Section 3.5.4, the following inequalities hold:

$$\frac{u_1(v - t_1(v,b))}{u_1(v-b)} \geq \frac{u_2(v - t_1(v,b))}{u_2(v-b)} > \frac{u_2(v - t_2(v,b))}{u_2(v-b)}$$

This, combined with (3.5), implies $\partial_1(F^{n-1} \circ t_1)(v,b) > \partial_1(F^{n-1} \circ t_2)(v,b)$, which means that $F^{n-1}(t_1(v,b)) - F^{n-1}(t_2(v,b))$ is strictly increasing in v for $\alpha < v \leq \beta$. Since $F^{n-1}(t_1(\alpha,b)) = F^{n-1}(t_2(\alpha,b))$, $F^{n-1}(t_1(v,b)) > F^{n-1}(t_2(v,b))$ is only possible if $t_1(v,b) > t_2(v,b)$ for all $\alpha < v \leq \beta$, which is a contradiction. ■

3.3 The Expected Social Welfare

3.3.1 Bidders' Choice: Buy-price or Standard English Auction?

When a bidder needs to choose between a buy-price and a standard English auction, which one should he prefer? To decide, we need to compare his expected profits. For a bidder with a valuation below b, the two auctions are equivalent because his equilibrium strategy remains the same. For a bidder with valuation at or above b but below $\tilde{v}$, he follows the same strategy as in the standard auction as long as the second-highest bidder valuation is below the threshold price. Otherwise, his expected extra gain from attending a buy-price English auction, instead of a standard one, is

$$\int_{t(v,b)}^{v} (u(v-b) - u(v-x)) \, \mathrm{d}F^{n-1}(x) \tag{3.12}$$

Next, we calculate the bidder's gains when he is risk-averse or neutral, respectively. Suppose the bidder has CARA utility $u_a(x) = (1 - e^{-ax})/a$, where $a > 0$ is the absolute level of risk aversion. If the bidder is risk-neutral, $a = 0$, $u_0(x) = \lim_{a \to 0} u_a(x) = x$[7]. The CARA utility satisfies the following

$$\frac{u_a(x) - u_a(y)}{u_a'(y)} = u_a(x-y) \quad \text{for all } a \geq 0,\, x, y \in \mathbb{R}$$

$$u_a(-x) = -\frac{u_a(x)}{u_a'(x)} \quad \text{for all } a \geq 0,\, x \in \mathbb{R}$$

Applying them in (3.5), we get

$$u_a(b - t(v,b))\big(F^{n-1} \circ t(\cdot, b)\big)'(v) = u_a(b-v)F^{n-1\prime}(v)$$

Solving it with the boundary condition $t(b,b) = b$ yields

$$\int_b^{t(v,b)} u_a(b-x) \, \mathrm{d}F^{n-1}(x) = \int_b^v u_a(b-x) \, \mathrm{d}F^{n-1}(x)$$

[7] Subsequently, whenever we mention CARA utility we also include the linear utility function.

$$\int_{t(v,b)}^{v} u_a(b-x)\ \mathrm{d}F^{n-1}(x) = 0 \tag{3.13}$$

Using equation (3.13), a bidder with CARA utility can calculate his threshold price $t(v,b)$.

Rewriting (3.12) and using (3.13), we get

$$\int_{t(v,b)}^{v} (u_a(v-b) - u_a(v-x))\ \mathrm{d}F^{n-1}(x) = -u_a'(v-b) \int_{t(v,b)}^{v} u_a(b-x)\ \mathrm{d}F^{n-1}(x) = 0$$

Hence, a bidder with CARA utility and a valuation v, $b \leq v < \tilde{v}$ gains no extra expected profit from attending a buy-price English auction instead of a standard one.

However, bidders with valuation above $\tilde{v}$ are no longer indifferent between a buy-price and a standard English auctions. Most bidders will be better off in a buy-price English auction. If the buy price is low and $\hat{v} \leq \overline{v}$, however, then some bidders with very high valuations will prefer the standard English auction where they do not have to participate in the random draw with other unconditional bidders; thus, their chance of winning is higher. We have not calculated the exact conditions under which a bidder with a high valuation prefers a standard English auction, but we conjecture that this could not happen with most value distributions unless the buy price is set to unreasonably low levels.

3.3.2 Seller's Choice: Buy-price or Standard English Auction?

3.3.2.1 Risk-neutral Sellers

Define $t_{\overline{v}} = \lim_{v \to \overline{v}} t(v,b)$. We have seen that when $t_{\overline{v}} \geq r$ is in equilibrium, all bidders follow the threshold strategy with a strictly decreasing t, ensuring that the bidder with the highest valuation wins. The revenue equivalence theorem, which is true only when both the seller and the bidders are risk-neutral, implies that a risk-neutral seller's expected profit in a buy-price auction is the same as that in a standard one. On the other hand, when $t_{\overline{v}} \geq r$ does not hold, bidders with valuations above $\tilde{v}$ follow different strategies and the auction no longer guarantees that the bidder with the highest valuation wins.

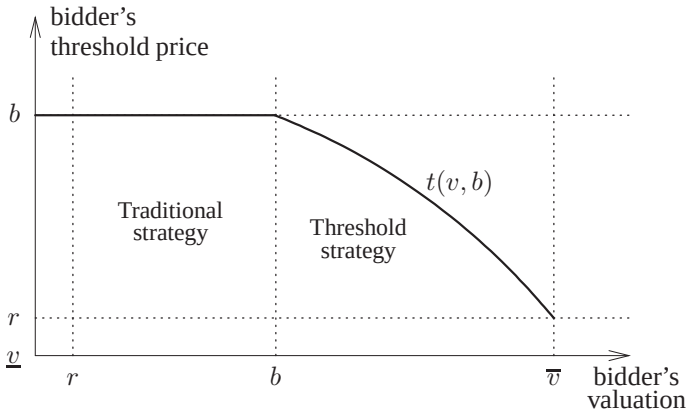


Figure 3.3: The buy-price English auction is efficient if $b \geq \int_r^{\overline{v}} x \ \frac{\mathrm{d}F^{n-1}(x)}{1-F^{n-1}(r)}$.

When bidders have CARA utility and $t_{\overline{v}} \geq r$, we can derive the following from equation (3.13)

$$0 = \int_{t_{\overline{v}}}^{\overline{v}} u_a(b-x)\ \mathrm{d}F^{n-1}(x) \leq \int_r^{\overline{v}} u_a(b-x)\ \mathrm{d}F^{n-1}(x)$$

which implies

$$\lim_{v \to \overline{v}} t(v,b) = t_{\overline{v}} \geq r \iff \int_r^{\overline{v}} u_a(b-x)\ \mathrm{d}F^{n-1}(x) \geq 0 \tag{3.14}$$

When buyers are risk-neutral, i.e., $a = 0$ and $u_a(b-x) = b - x$, the condition of (3.14) is equivalent to

$$\int_r^{\overline{v}} b\, \mathrm{d}F^{n-1}(x) \geq \int_r^{\overline{v}} x\, \mathrm{d}F^{n-1}(x)$$

which can be rewritten as

$$b \geq \int_r^{\overline{v}} x\ \frac{\mathrm{d}F^{n-1}(x)}{1-F^{n-1}(r)} \tag{3.15}$$

This result can be summarized in the following theorem (see Figure 3.3):

Theorem 3.9 *In a buy-price English auction, if both the seller and buyers are risk-neutral and the buy price is set at least as high as the expected maximum valuation among* $(n-1)$

buyers on the condition that at least one of the $(n-1)$ buyers has a valuation at or above the reserve price, then the seller's expected profit is the same as that in a standard English auction, and the buy-price English auction is efficient.

If the second-highest bidder valuation is below the reserve price, the winning bidder would only pay the reserve in both the standard and buy-price auctions. Therefore, to compare the two auctions, it is sufficient to consider the expected seller revenues conditional on having at least two bidders with valuations no lower than the reserve. For the following discussion we assume this condition.

How high should a buy price be? We know that the buy price is the maximum revenue the seller gets from a buy-price auction, thus the expected seller revenue in such an auction is always less than the buy price. This implies that for the seller to have the same expected revenue from the buy-price and standard auctions, she has to set the buy price higher than her expected revenue from a standard auction (i.e., the expected second-highest bidder valuation).

The criterion on how to choose a good buy price in Theorem 3.9 follows the above intuition. The maximum valuation of arbitrary $(n-1)$ bidders is usually higher than the second-highest valuation. They are equal only when the chosen $(n-1)$ bidders happen to be the bidders with the lowest $(n-1)$ valuations, but in all other cases the maximum valuation of arbitrary $(n-1)$ bidders is equvalent to the maximum valuation among all bidders.

Another intuitive way to obtain a lower bound of a well-chosen buy price that guarantees the revenue equivalence between buy-price and standard auctions is to study the expected payment of the bidder with the highest type $\overline{v}$. In the standard auction, the $\overline{v}$-type bidder's expected payment is the expected maximum valuation of all other bidders, which, if the revenue equvalence holds, is also his expected payment in the buy-price auc-

tion. Since his maximum payment is the buy price, the buy price must be at least as high as the expected maximum valuation of all other bidders, which is exactly what is depicted in Inequality (3.15).

Inequality (3.15) is important because it helps the seller to choose an appropriate buy price. Example 3.10 demonstrates how to calculate the lower bound for the buy price.

Example 3.10 *In an auction where there are two risk-neutral bidders with valuation drawn from the uniform* $[0, 1]$ *distribution and the seller's valuation is* 0*, the optimal reserve price is* 0.5 *and the lowest buy price that satisfies the revenue equivalence is* 0.75*.*

In this example, $F(x) = x$ and $f(x) = 1$ for $x \in [0, 1]$. The optimal reserve price satisfies $r = (1 - F(r))/f(r) = 1 - r$, thus $r = 0.5$. From inequality (3.15) we get

$$b \geq \int_r^{\overline{v}} x \, \frac{\mathrm{d}F^{n-1}(x)}{1 - F^{n-1}(r)} = \int_{0.5}^{1} x \frac{\mathrm{d}x}{1 - 0.5} = 1^2 - 0.5^2 = 0.75$$

Note that this lower bound of the buy price only applies to auctions with two bidders. As the number of bidders increases, the buy price should also increase in order to ensure the revenue equivalence.

The same lower bound can also derived from (3.13):

$$0 = \int_{t(v,b)}^{v} u_a(b - x) \, \mathrm{d}F^{n-1}(x) = \int_{t(v,b)}^{v} (b - x) \, \mathrm{d}x = (v - t(v, b)) \frac{2b - v - t(v, b)}{2}$$

which implies that the threshold function $t(v, b) = 2b - v$. v can be at most 1, so the condition for the threshold function staying above the reserve is $t(1, b) = 2b - 1 \geq r = 0.5 \implies b \geq 0.75$. 0.75 is also the expected maximum valuation of one (i.e., $n - 1$) bidder conditional on his valuation (the valuation of one out of $n - 1 = 1$) is at least 0.5. Such a bidder has a uniform $[0.5, 1]$ valuation distribution with an expected valuation of 0.75. ■

We can also characterize the expected profit of a risk-neutral seller facing risk-averse buyers in the following theorem:

Theorem 3.11 *In a buy-price English auction, if the seller is risk-neutral, the buyers are risk-averse, and the buy price is set at least as high as the expected maximum valuation among* $(n-1)$ *buyers on the condition that at least one of the* $(n-1)$ *buyers has a valuation at or above the reserve price, then the seller's expected profit is higher than that in a standard English auction.*

PROOF. Theorem 3.8 says that the more risk-averse the buyers are, the lower their threshold prices. This increases the seller's expected profit because more buyers will pay the buy price instead of the second-highest bidder valuation. But in a standard auction, the seller's expected profit does not change with the buyers' levels of risk aversion, as buyers bid up to their valuations regardless. Therefore, when buyers are risk-averse, a risk-neutral seller is better off in a buy-price English auction. ■

3.3.2.2 Risk-averse Sellers

Let us now calculate the expected profit of a risk-averse seller with risk-neutral buyers. The calculation presented below is similar to that of Riley and Samuelson [37] except that the seller's utility function $s(x)$, where x denotes the sale price, is more general because the seller under analysis can be either risk-neutral or risk-averse: $s(x) = x$ if the seller is risk-neutral and $s(x)$ is strictly concave if the seller is risk-averse.

At the equilibrium, the seller's expected profit from a bidder with a valuation below b is the same as in a standard auction. It is given by the equation (8b) in Riley and Samuelson [37]

$$P_0(v) = s(r)F^{n-1}(r) + \int_r^v s(x)\ \mathrm{d}F^{n-1}(x) = s(v)F^{n-1}(v) - \int_r^v s'(x)F^{n-1}(x)\ \mathrm{d}x$$

The seller's expected profit from a bidder with a valuation $v \in [b, \tilde{v})$ is

$$P(v, b) = P_0(t) + s(b)(F^{n-1}(v) - F^{n-1}(t))$$

Hence, the seller's overall expected profit from all n bidders, denoted by $\Pi_{\mathrm{s}}(v,b)$, is:

$$\Pi_{\mathrm{s}}(v,b) = n\Big(\int_r^b P_0(v)\,\mathrm{d}F(v) + \int_b^{\tilde{v}} P(v,b)\,\mathrm{d}F(v)\Big) + b(1-F^n(\tilde{v})) - n(b-r)F^{n-1}(r)(F(\hat{v})-F(\tilde{v})) \tag{3.16}$$

Since we can regard no buy price in a standard auction as having a very large buy price, i.e., $b \to \infty$, proving that a risk-averse seller is better off in a buy-price English auction with risk-neutral buyers is equivalent to showing that the b-derivative of $\Pi_{\mathrm{s}}(v,b)$ is negative when (3.15) holds. The b-derivative of $\Pi_{\mathrm{s}}(v,b)$, when $\tilde{v} = \hat{v} = \overline{v}$, is

$$\begin{aligned}\partial_2\Pi_{\mathrm{s}}(v,b) &= n\int_b^{\overline{v}} \partial_2 P(v,b)\,\mathrm{d}F(v) \\ &= n\int_b^{\overline{v}} \Big[(s(t)-s(b))\partial_2(F^{n-1}\circ t)(v,b) + \\ &\qquad s'(b)(F^{n-1}(v)-F^{n-1}(t))\Big]\,\mathrm{d}F(v)\end{aligned}$$

Differentiating (3.13) by b with $a=0$, i.e., $u_a(x)=x$, we get

$$F^{n-1}(v) - F^{n-1}(t) = (b-t)\big(\partial_2(F^{n-1}\circ t)\big)(v,b) \tag{3.17}$$

Applying (3.17) and $s(b)-s(t) \geq s'(b)(b-t)$ due to the concavity of s, we get

$$\partial_2\Pi_{\mathrm{s}}(v,b) \leq 0.$$

The inequality shows that as long as (3.15) holds – i.e., the buy price is set high enough that no one uses a conditional or unconditional strategy–a risk-averse seller is strictly better off in a buy-price English auction than in a standard one when buyers are risk-neutral. The equality holds if and only if the seller is risk-neutral, implying that the seller's expected profits from a buy price is the same as in a standard auction. Moreover, risk-averse buyers further increase the seller's expected profit. Hence, the following theorem holds:

Theorem 3.12 *In a buy-price English auction, if either the seller or the buyers are risk-averse and the buy price is set at least as high as the expected maximum valuation among*

$(n-1)$ *buyers on the condition that at least one of the* $(n-1)$ *buyers has a valuation at or above the reserve price, then the seller's expected profit is higher than in a standard English auction.*

Therefore, we can conclude that regardless of whether the seller is risk-neutral or risk-averse, she cannot lose by utilizing buy prices in English auctions.

3.4 Concluding Remarks

This paper has analyzed buy-price English auctions of an indivisible good in the general setting of n bidders with continuous, independently distributed, and private valuations. We have proved that in equilibrium unique reference points $\tilde{v}$ and $\hat{v}$ exist ($\tilde{v} \leq \hat{v}$, with both above the buy price) so that a bidder with a valuation between the buy price and $\tilde{v}$ bids the buy price when the current high bid reaches a threshold price (i.e., the competition among bidders is heated and has reached a level which makes such a bidder unwilling to risk waiting further and thus bids the buy price), a bidder with a valuation between $\tilde{v}$ and $\hat{v}$ bids the buy price on the condition that there already exists a valid bid above or equal to the reserve (i.e., at least one competing bidder exists), and a bidder with a valuation at or above $\hat{v}$ bids the buy price unconditionally (i.e., regardless of whether there is competition or not).

We have proved that the threshold bidders' equilibrium threshold prices can be calculated using the buy price, bidders' utility functions, and their value distribution. These threshold prices are between the reserve and the buy prices and strictly decreasing with valuations. In other words, the higher a threshold bidder's valuation, the lower his threshold price.

We have also shown that if the buy price is set at or above a lower bound, then all bidders with valuations above the buy price are threshold bidders; that is, there are no

conditional or unconditional bidders. Since the threshold prices are strictly decreasing with the bidders' valuations, the bidder with the highest valuation will reach his threshold price first, and thus the auction guarantees that the highest bidder wins and the equilibrium yields full efficiency. In addition, the more risk-averse a threshold bidder, the lower his threshold price. In other words, a more risk-averse bidder tends to bid the buy price earlier, which helps to explain why the seller can gain higher expected profit from risk-averse buyers than from risk-neutral buyers.

Clearly, the buy-price option can reduce a buyer's risk: bidding the buy price, a buyer can obtain the item at a fixed price, and he thus no longer has to worry about losing the auction to a bidder with a higher valuation. Because of this observation one may expect that risk-averse buyers are better off in buy-price English auctions, but this is not true. To reduce the risk of losing the auction, risk-averse buyers bid the buy price more often than risk-neutral bidders. They may bid the buy price even in cases where there are no other bidders with valuation above the buy price, and therefore, pay more than they would have in a standard English auction. In fact, we have proved that when buyers are risk-neutral or uniformly risk-averse, and when the buy price is properly set above a lower bound, the buyers' expected utility in a buy-price English auction is the same as in a standard one.

If risk-averse buyers' expected utility does not increase with the introduction of a buy price even though their risks are reduced, then their expected payment must increase to offset the positive utility of the reduced risk. This insight again explains why the seller's expected revenue is higher in the buy-price English auction than in a standard one when bidders are risk-averse.

In addition to the seller's higher or equivalent expected revenue, the seller's risk is also reduced in a buy-price English auction because the seller will often get the buy price instead of some unpredictable payment either below or above the buy price. This observation in turn implies that a risk-averse seller always prefers a buy-price English auction

because in it her expected revenue is not lower than that in a standard one when the buy price is properly set and, at the same time, her risk is reduced.

The above results do not follow in cases with temporary and limited buy prices, as neither auction can guarantee that the bidder with the highest valuation wins when all agents are risk-neutral. In an English auction with a temporary buy price, bidders have to decide whether or not to use the buy price without observing any bid, and, therefore, their best symmetric strategy is to find a valuation level above which they unconditionally exercise the buy price. This leads to inefficient outcomes reducing the seller's expected profit [38]. Similarly, an English auction with a limited buy price is also inefficient. Although the temporary and limited buy prices can increase the social welfare when players are risk-averse [31, 39], they lower the expected social welfare when players are risk-neutral. Thus temporary and limited buy-price English auctions are generally inferior to the ones with permanent buy prices. This result implies that the permanent buy-price auctions offered by Yahoo!, uBid, and Amazon are, in theory, more beneficial to all agents than the temporary buy-price auctions, like those offered by eBay, or the limited buy-price auctions, like those offered by labx, especially for unique and used goods where buyers have private valuations and face relatively high risks. While the positive network externality has contributed significantly to the popularity of eBay, features in Yahoo!, uBid, and Amazon auctions also have their own competitive advantages[8]. For practice, we recommend that auction houses choose appropriate policies with respect to buy prices, conduct market research on the players' degrees of risk aversion in different markets, and suggest strategies to auction sellers how to use buy prices for additional revenue.

[8] In addition to the good policy of utilizing buy prices, Yahoo!, uBid, and Amazon have some other advanced features. For instance, at the time when this paper is written, Yahoo! and uBid authenticate buyers more rigorously than eBay by requiring valid credit card information for registration. Yahoo! even asks for two passwords for the purpose of authentication, which reduces the number of non-paying winners and the fraud of shill bidding [40]. Yahoo! also allows a seller to specify the automatic extension of an auction if a bid is made within the last five minutes of the auction, and this helps to prevent last-minute bidding [41]. Moreover, Yahoo! auctions charge relatively low intermediation fees.

Relaxing the assumptions of the revenue equivalence theorem [36, 37] leads to differences among the English, Dutch, sealed-bid first-Price, and sealed-bid second-price auction mechanisms[42]. Maskin and Riley [43] provide a detailed analysis of auctions with risk-averse buyers. The most notable result related to our research is that when bidders are risk-averse, first-price sealed-bid and Dutch auctions provide higher expected seller profit than second-price auctions. Using a two-bidder two-type model, Budish and Takeyama [34] conclude that a buy-price English auction can be superior even to the first-price sealed-bid and Dutch auctions when bidders are risk-averse. It would be interesting to investigate whether this result remains true in a general setting of n bidder with arbitrary value distribution.

Another extension of our model would consider the time factor. With the pace of transactions getting faster and the Internet's around-the-clock operations allowing random arrival of traders, the temporal property of a trade becomes increasingly important. We suspect that delay-averse auction sellers and buyers are more likely to use buy prices than delay-neutral or delay-taking buyers, and that the shortened auction cycles would increase the market liquidity. Lucking-Reiley [44] mentioned that the use of a buy price is to "allow buyers to buy an early end to the auction by submitting a sufficiently high bid." Mathews [32] modeled eBay's temporary buy price auction with a time discount and showed that when the seller and buyers are risk-neutral, even temporary buy prices that are exercised with positive probabilities are welcome, because buyers are willing to pay more to get the item sooner and/or the seller is willing to give up some of her expected profit to get the payment sooner. In contrast to an analysis of eBay's temporary buy price model with uniform bidder distribution, we need a more general model to analyze the temporal effect of utilizing permanent buy prices in auctions with arbitrary bidder distribution.

Entry costs to auctions may also affect the comparison between buy-price and standard English auctions. Rothkopf and Harstad [42] note that when potential buyers expect

strong competition for an item, they may not invest efforts to enter the auction because the winner can only expect small profits. A buy price can guarantee a minimum profit for the winner and, hence, may attract more bidders.

Another research direction is to study how the seller uses buy prices as signaling devices. Too high a buy price may alienate buyers from bidding. Too low a buy price may convey information on adverse quality. Ultimately, it might even be possible to embed a Dutch auction within an English auction by allowing buy prices to decrease during the auction.

While we have only modeled private value auctions, buy prices could also prove beneficial in common value models. In a common value auction, the buy price is a strong signal from the seller about the value of the item, which can help reduce the errors in the bidders' value estimates and thus may lower the "winner's curse" effect, in turn increasing the seller's expected profit.

3.5 Detailed proofs

3.5.1 Proof of Proposition 3.2

PROPOSITION 3.2: *$t(v, b)$ is strictly decreasing in v when $b \leq v < \tilde{v}$.*

PROOF. We first prove that $t(v, b)$ is non-increasing in v for all $v \geq b$. Prove by contradiction: assume $t(v, b)$ is increasing in v for all $v \geq b$, that is, for some $v_0 < v_1$, $t_0 = t(v_0, b) < t_1 = t(v_1, b)$. Since we assume that F is strictly increasing, every bidder would have a unique valuation and there will not be random-draw cases. When the auction clock reaches t_0, a bidder with a valuation v_1 can either jump to the buy price immediately for a guaranteed $u(v_1 - b)$ profit, or continue bidding and wait to jump at t_1. Since by our assumption t_1 is this bidder's equilibrium threshold price, jumping at t_1 should give him

expected profit no less than jumping at t_0:

$$u(v_1 - b)G_{n-1}(t_1) + \int_{t_0}^{t_1} u(v_1 - x)\,\mathrm{d}F^{n-1}(x) \geq u(v_1 - b)G_{n-1}(t_0)$$

$$\int_{t_0}^{t_1} u(v_1 - x)\frac{\mathrm{d}F^{n-1}(x)}{G_{n-1}(t_0)} \geq u(v_1 - b)\Big(1 - \frac{G_{n-1}(t_1)}{G_{n-1}(t_0)}\Big) \tag{3.18}$$

On the other hand, for the buyer with valuation v_0, jumping to the buy price at t_0 is at least as good as continuing bidding and jumping at t_1:

$$u(v_0 - b)G_{n-1}(t_0) \geq u(v_0 - b)G_{n-1}(t_1) + \int_{t_0}^{t_1} u(v_0 - x)\,\mathrm{d}F^{n-1}(x)$$

$$u(v_0 - b)\Big(1 - \frac{G_{n-1}(t_1)}{G_{n-1}(t_0)}\Big) \geq \int_{t_0}^{t_1} u(v_0 - x)\frac{\mathrm{d}F^{n-1}(x)}{G_{n-1}(t_0)} \tag{3.19}$$

Since u is concave, $x \leq b \implies u(v_1 - x) - u(v_0 - x) \leq u(v_1 - b) - u(v_0 - b)$. Together with (3.18), (3.19), we have

$$\int_{t_0}^{t_1} \frac{\mathrm{d}F^{n-1}(x)}{G_{n-1}(t_0)} \geq 1 - \frac{G_{n-1}(t_1)}{G_{n-1}(t_0)}$$

$$G_{n-1}(t_0) - G_{n-1}(t_1) \leq F^{n-1}(t_1) - F^{n-1}(t_0)$$

Recall that $G_{n-1}(t)$ is defined to be the probability that a bidder with threshold t exercises the buy price and wins the auction. Lowering the threshold from t_1 to t_0 increases the chance of successfully using the buy price by at least the probability that there is another bidder with a valuation between t_0 and t_1, i.e.,

$$G_{n-1}(t_0) - G_{n-1}(t_1) \geq F^{n-1}(t_1) - F^{n-1}(t_0)$$

Hence $G_{n-1}(t_0) - G_{n-1}(t_1) = F^{n-1}(t_1) - F^{n-1}(t_0)$, and (3.19) becomes

$$u(v_0 - b)(F^{n-1}(t_1) - F^{n-1}(t_0)) \geq \int_{t_0}^{t_1} u(v_0 - x)\ \mathrm{d}F^{n-1}(x)$$

$$0 \geq \int_{t_0}^{t_1} \big(u(v_0 - x) - u(v_0 - b)\big)\, \mathrm{d}F^{n-1}(x)$$

which implies $F(t_0) = F(t_1)$. Since F is strictly increasing, $t_0 = t_1$ must hold, contradictory to our assumption. Therefore, we have proved that $t(v, b)$ is non-increasing in v for all $v \geq b$.

Now we further prove that $t(v, b)$ is strictly decreasing when $b \leq v < \tilde{v}$. Since by definition $t(\tilde{v}, b) = r$, proving t as strictly decreasing indicates $t(v, b) > r$ when $b \leq v < \tilde{v}$. Again, prove by contradiction: assume that for $t_0 > r$ and some $b \leq v_0 < v_1$, $\forall v(v_0 \leq v \leq v_1 \implies t(v, b) = t_0)$, that is, all bidders with valuations between v_0 and v_1 pool at the same threshold price t_0. This implies a positive probability that more than one bidder with valuations between v_0 and v_1 would jump to the buy price simultaneously when t_0 is reached and the winner would be chosen randomly among them. But if one of them decides to jump earlier (i.e., when the auction clock reaches $t_0 - \varepsilon$ instead of t_0 for some arbitrarily small ε), he can avoid this random-draw gamble and increase his chance of winning. By doing so, he can increase his expected profit and only suffer at most ε additional loss. Clearly, if ε is small enough, he can be better off by jumping earlier. Therefore, pooling cannot be an equilibrium. Thus, $t(v, b)$ is strictly decreasing when $b \leq v < \tilde{v}$, i.e., $t(v, b) > r$. ■

3.5.2 Proof of Proposition 3.3

PROPOSITION 3.3: *$t(v, b)$ is continuous in v when $b \leq v < \tilde{v}$.*

PROOF. We first prove $t(v, b)$ is right-continuous in v when $b \leq v < \tilde{v}$, i.e., $t(v, b) > r$. Let $v_0 \geq b$, $t_0 = t(v_0, b)$, $v > v_0$, and $t_+ = \lim_{v \searrow v_0} t(v, b)$. The monotonicity of $t(v, b)$ implies $t_+ \leq t_0$. Since $t(v, b)$ is the equilibrium threshold, for the buyer with a valuation v, jumping to the buy price when the auction clock reaches $t(v, b) \leq t_0$ is at least as good as

jumping at t_0:

$$(3.20)\quad u(v-b)G_{n-1}(t(v,b)) \quad \geq \quad u(v-b)G_{n-1}(t_0) + \int_{t(v,b)}^{t_0} u(v-x)\,\mathrm{d}F^{n-1}(x)$$

Recall that G is continuous. Therefore, we can take the limit in (3.20) as $v \searrow v_0$ to obtain

$$u(v_0-b) \geq u(v_0-b)\frac{G_{n-1}(t_0)}{G_{n-1}(t_+)} + \int_{t_+}^{t_0} u(v_0-x)\frac{\mathrm{d}F^{n-1}(x)}{G_{n-1}(t_+)}$$

Since $t(v,b)$ is non-increasing in v and now $v \searrow v_0$, $t(v,b)$ cannot take any value between t_+ and t_0. $t(v,b)$ is strictly decreasing when it is above r, thus when $t_+ > r$ we have $T(t_+) = T(t_0)$, which by equation (3.4) implies $G_{n-1}(t_0) - G_{n-1}(t_+) = -(F^{n-1}(t_0) - F^{n-1}(t_+))$. Therefore,

$$0 \geq \int_{t_+}^{t_0} u(v_0-x)\frac{\mathrm{d}F^{n-1}(x)}{G_{n-1}(t_+)} - u(v_0-b)\frac{F^{n-1}(t_0)-F^{n-1}(t_+)}{G_{n-1}(t_+)}$$

$$0 \geq \int_{t_+}^{t_0} \Big(u(v_0-x)-u(v_0-b)\Big)\frac{\mathrm{d}F^{n-1}(x)}{G_{n-1}(t_+)}$$

For $x < b$, $u(v_0-x) - u(v_0-b) > 0$ and $t_0 \leq b$, therefore if $t_0 > t_+$, the integral on the right side would be strictly positive. Therefore, the above formula can only hold if $t_+ = t_0$. This completes the proof of the right-continuity of $t(v,b)$ with regard to v when $t(v,b) > r$. Using similar arguments, we can prove the left-continuity of t. ■

Note that the continuity of $t(v,b)$ in v is only true when $t(v,b) > r$; it is possible that t has discontinuity at some point where t suddenly drops to r.

3.5.3 Proof of Proposition 3.5

PROPOSITION 3.5: *Let t be the function defined by (3.5) and let $\tilde{v} > b$ satisfy that for all $x < \tilde{v}$ that $t(x,b) > r$. If all other bidders with valuations x, $b \leq x < \tilde{v}$, follow the threshold strategy $t(x,b)$, then the optimal threshold strategy of a bidder with valuation v, $b \leq v < \tilde{v}$, is to use $t(v,b)$ as his threshold price.*

For the proof we will need the following lemma:

Lemma 3.13 *For any $x > 0$ and $d > 0$, $\frac{u(x+d)}{u(x)}$ is strictly decreasing in x.*

PROOF. Let $x < y$.

$$\begin{aligned}
\frac{u(x+d)}{u(x)} - 1 &= \frac{u(x+d)-u(x)}{u(x)} \geq \frac{u(y+d)-u(y)}{u(x)} \text{ because } u \text{ is concave} \\
&> \frac{u(y+d)-u(y)}{u(y)} \text{ because } u \text{ is increasing} \\
&= \frac{u(y+d)}{u(y)} - 1 \qquad \square
\end{aligned}$$

PROOF. To show that $t(v,b) = p^*$ maximizes the expected profit of a bidder with $b \leq v < \tilde{v}$, it is enough to show that (3.7), having the same sign as $\frac{\partial \Pi_t(v,p)}{\partial p}$, is positive if $p < t(v,b)$ and negative when $p > t(v,b)$. Since $w(\cdot,b)$ is the inverse of $t(\cdot,b)$, $w(t(v,b),b) = v$. $w(\cdot,b)$ is strictly decreasing; therefore, $w(p,b) < v$ if $p > t(v,b)$ and $w(p,b) > v$ if $p < t(v,b)$.

First consider the case when p is in the range of t, i.e., $v = w(p,b) < \tilde{v}$ and $t(w(p,b),b) = p$. Substitute $v = w(p,b)$ into (3.5)

$$\frac{F^{n-1\prime}(w(p,b))}{\partial_1(F^{n-1} \circ t)(w(p,b),b)} = 1 - \frac{u(w(p,b) - t(w(p,b),b))}{u(w(p,b)-b)} = 1 - \frac{u(w(p,b)-p)}{u(w(p,b)-b)}$$

Substituting this into (3.7) we get

$$\frac{u(v-p)}{u(v-b)} - \frac{u(w(p,b)-p)}{u(w(p,b)-b)} \tag{3.21}$$

Applying Lemma 3.13 with $d = b - p$, $x_1 = v - b$, and $x_2 = w(p,b) - b$, we can see that (3.21) is positive if $w(p,b) > v$, i.e., $p < t(v,b)$, and negative if $w(p,b) < v$, i.e., $p > t(v,b)$. This shows that $t(v,b)$ maximizes the expected profit from the threshold strategy as long as the threshold is in the range of t. Since t is continuously decreasing and $t(b,b) = b$, the range of $t(v,b)$ for $b \leq v < \tilde{v}$ is an interval $(\underline{t}, b)$.

When p is not in the range of t, i.e., $r < p \leq \underline{t}$ and no other bidder will use a threshold price below or equal to p, using the threshold price p a bidder can only lose to conditional or unconditional bidders. Therefore,

$$G_{n-1}(p) = F^{n-1}(\tilde{v}) - F^{n-1}(p)$$

Applying it in (3.6) we get

$$\frac{\partial \Pi_t(v,p)}{\partial p} = (u(v-p) - u(v-b))F^{n-1\prime}(p) > 0$$

which implies that $\Pi_t(v,p)$ is strictly increasing for all $p \in (r, \underline{t}]$.

Combining this with the case in which $p \in [\underline{t}, b)$, we get $\Pi_t(v,p)$ strictly increasing for all $p \in (r, t(v,b))$ and strictly decreasing for $p \in (t(v,b), b)$. This proves that, as long as all other bidders use the threshold strategy, $p^* = t(v,b)$ maximizes $\Pi_t(v,p)$ for a given v; that is, the optimal threshold price of a bidder with valuation v is $t(v,b)$. ■

3.5.4 Comparing utility functions based on the level of risk aversion

Lemma 3.14 *Let $u_1 : \mathbb{R}^+ \mapsto \mathbb{R}^+$, $u_2 : \mathbb{R}^+ \mapsto \mathbb{R}^+$ be twice differentiable utility functions, $u_1(0) = u_2(0) = 0$, and $u_1'(x) > 0$, $u_2'(x) > 0$ for all $x \geq 0$. Let $a_1 = -u_1''/u_1'$ and $a_2 = -u_2''/u_2'$ be the absolute level of risk aversion. If $a_1(x) \leq a_2(x)$ for all $x \geq 0$, then the following inequality holds*

$$\forall x, y \left(0 < y < x \implies \frac{u_1(x)}{u_1(y)} \geq \frac{u_2(x)}{u_2(y)} \right)$$

When $\exists y\,(0 < y < x)$ such that the equality holds, there is a constant λ such that $u_1(z) = \lambda u_2(z)$ for all $0 \leq z \leq x$.

PROOF. Let $\lambda = u_1(x)/u_2(x)$. Part I: We want to prove that $\forall x, y, 0 < y < x$

$$\frac{u_1(x)}{u_2(x)} \geq \frac{u_1(y)}{u_2(y)} \iff \lambda \geq \frac{u_1(y)}{u_2(y)} \iff \lambda u_2(y) - u_1(y) \geq 0$$

Prove by contradiction: Suppose $\exists y\big(0 < y < x \ \wedge \ \lambda u_2(y) - u_1(y) < 0\big)$, that is, $\exists y\big(0 < y < x \ \wedge \ \int_0^y (\lambda u_2'(v) - u_1'(v))\ \mathrm{d}v < 0\big)$. Together with $u_1(0) = u_2(0) = 0$, it implies that there is a $z, 0 < z < y$, s.t., $\lambda u_2'(z) - u_1'(z) < 0$, that is, $u_1'(z)/u_2'(z) > \lambda$.

Note that $a_1 = -u_1''/u_1' = (-\ln u_1')'$ and $a_2 = -u_2''/u_2' = (-\ln u_2')'$. Thus for all $v \geq 0$

$$a_2(v) - a_1(v) = \left(\ln \frac{u_1'(v)}{u_2'(v)}\right)' \geq 0$$

This means that u_1'/u_2' is non-decreasing. Therefore, $\forall v\big(v \geq z \implies u_1'(v)/u_2'(v) > \lambda \implies \lambda u_2'(v) - u_1'(v) < 0\big)$. Hence,

$$\lambda u_2(x) - u_1(x) = \lambda u_2(y) - u_1(y) + \int_y^x (\lambda u_2'(v) - u_1'(v))\ \mathrm{d}v < 0$$

which contradicts the definition of $\lambda = u_1(x)/u_2(x)$. Therefore, the following must hold:

$$\forall x, y\left(0 < y < x \implies \frac{u_1(x)}{u_1(y)} \geq \frac{u_2(x)}{u_2(y)}\right)$$

Part II: Assume $\exists y\big(0 < y < x \ \wedge \ \lambda u_2(y) - u_1(y) = 0\big)$, that is

$$\int_0^y (\lambda u_2'(v) - u_1'(v))\ \mathrm{d}v = 0$$

Note that u_1'/u_2' is non-decreasing. In order to satisfy the above, either $\forall z\big(0 < z < y \implies \lambda u_2'(z) - u_1'(z) = 0\big)$ or $\exists z_1, z_2\big(0 < z_1 < z_2 < y \ \wedge \ \lambda u_2'(z_1) - u_1'(z_1) > 0 \ \wedge \ \lambda u_2'(z_2) - u_1'(z_2) < 0\big)$. But the later case means that $\forall v(z_2 \leq v \leq x \implies \lambda u_2'(v) - u_1'(v) < 0)$, thus

$$\lambda u_2(x) - u_1(x) = \lambda u_2(y) - u_1(y) + \int_y^x (\lambda u_2'(v) - u_1'(v))\ \mathrm{d}v < 0$$

which contradicts to the definition of $\lambda = u_1(x)/u_2(x)$. Thus the earlier case $\forall z\big(0 < z < y \implies \lambda u_2'(z) - u_1'(z) = 0\big)$ must be true. Consequently, together with $u_1(0) = u_2(0) = 0$, we get $\forall z\big(0 < z < y \implies \lambda u_2(z) - u_1(z) = 0\big)$.

For $\forall z$, $y < z < x$, applying the inequality result in Part I, we have

$$\frac{u_1(z)}{u_1(y)} \geq \frac{u_2(z)}{u_2(y)} = \frac{\lambda u_2(z)}{\lambda u_2(y)} \implies \frac{u_1(z)}{\lambda u_2(z)} \geq \frac{u_1(y)}{\lambda u_2(y)} = 1 \implies \lambda u_2(z) - u_1(z) \leq 0$$

Also, Part I says that $\forall z\big(0 < z < x \implies \lambda u_2(z) - u_1(z) \geq 0\big)$. Therefore, $\forall z\big(y < z < x, \implies \lambda u_2(z) - u_1(z) = 0\big)$. Thus, $\forall z\big(0 < z < x \implies \lambda = u_1(z)/u_2(z)\big)$. ■

Chapter 4

Mechanism Design for Grid Computing

Abstract

We develop a system for grid computing where the price of computing tasks are determined by an audited market-exchange. We show a method to provide a verifiable certificate, called "witness," of program execution with the following property: if two different agents running the same program on the same input produce the same witness, that proves with certainty very close to 1 that both agents have executed the program correctly. Using these witnesses, a trusted intermediary audits grid agents by dispatching identical work units to different agents and comparing their results. The results of past audits create a reputation history for agents, which is used to offer different prices to consumers based on the expected reliability corresponding to a reputation history. We allow reputations to be traded, instead of being tied to individual agents, and we show that in such a reputation market only high-type agents would have incentive to purchase a high reputation, and only low-type agents would use low reputations.

4.1 Introduction

Many of today's home and office computes are as powerful as supercomputers were a decade ago. However, most users only use a fraction of this computing power. At the same time there is an increasing number of applications that require vast amounts of calculations: movie studios need computers for animation and special effects rendering (`http://news.bbc.co.uk/1/hi/technology/4014333.stm`). Medical research use it to

simulate protein folding (`http://folding.stanford.edu/papers.html`). Engineering firms use it to simulate their designs (`http://www-1.ibm.com/servers/deepcomputing/solutions/functionalverification.html`). Astronomers use it in search of extra-terrestrial life (`http://setiathome.ssl.berkeley.edu/`).

It would be socially beneficial to utilize the idle computing power of desktop machines to solve computation-intensive problems. Companies that need computing power can save floor space, amortization, maintenance and electricity costs by outsourcing these tasks to desktop users. Desktop users, however, suffer minor inconveniences from offering their systems to the grid, such as installation and maintenance of grid applications, increased network traffic, increase in their utility bill or system instability.

Desktop users must receive some monetary compensation in exchange for the resources they contribute to the grid. However, this also opens the door for fraud, as participants may skip the computations and guess the result. This is often easy to do, for example it is very unlikely that a signal sample from a radio-telescope comes from an extra-terrestrial intelligence, thus one can safely report that the signal assigned to him did not come from E.T. without doing any calculation.

Besides malicious errors, incorrect result can also be caused by badly installed software, computer viruses, faulty hardware or even from cosmic radiation particles altering memory or CPU state. One way to catch and fix hardware errors is by redundancy, e.g. using more expensive ECC memory or by running the same calculation on several CPUs, comparing the results at every instruction step. However, this requires expensive hardware and high bandwidth physical connection between the coupled processors, and it does not protect against errors related to software installation, misconfiguration or from opportunistic cheating.

Instead of using expensive custom hardware, our goal is to run the computations on cheap desktop computes, but still find a way to be able to compare two runs from two dif-

ferent, independently owned and maintained systems. Modern computers execute billions of instructions and process gigabytes of data each second. To compare two runs, one would need to produce and compare execution traces, which requires comparing several gigabytes of data each second. But in practice, we could use a compare function which fail with a very low probability. This is a well known problem in data communications and cryptography: we need to calculate a hash function of the large data sets we are comparing. If the hashes match, then we can say with a very high degree of confidence that the original data sets are identical. The advantage of using hashes is that it does not require massive amounts of storage and communication badwidth to hold and transmit execution traces. The hash could be computed on the fly, and requires minimal storage.

Our suggested approach shares some similarities with the Trusted Platform Module (TPM) architecture [45]. The TPM can provide evidence that requested calculation was performed using the original, unmodified software, but it does not allow the detection of calculation errors. Also, implementing TPM-based verification requires significant effort, and incorrect implementation is subject to various attacks; once the protection is circumvented, it can be widely deployed. In contrast, our verification scheme relies on the correct computation of a unique hash, which can only be obtained by performing the requested calculation. The resulting hash is not a secret, which means that the whole system could be implemented with open-source code and without the use of hardware modules.

Witness of execution

A witness is a fixed-length sequence of bits, which is determined by the program being executed, and all of its inputs. The sequence should be long enough to make the probability of guessing it right low, and should be at least as hard to compute as performing the requested task.

We can use the witnesses to verify the correctness of calculations. We do this by

assigning the same job to two different grid agents, and comparing the resulting witness values. When these match, it provides a high level of assurance that the result is correct. This method reliably catches random errors from hardware components (memory corruption, or processor miscalculations due to faulty components, overclocking, bad power supply, failed cooling system etc.), and it can also catch malicious errors, when the agent skips the calculations and returns bogus results.

To generate a witness we can generate an execution trace while running the computations and feed that to a hash function. But how can we obtain such a trace? The most efficient way would be to support this in hardware, the CPU could compute a hash of the execution trace in hardware without any performance loss. Perhaps in the future this feature will be integrated into processors, but until then we need a software solution.

There are several ways this could be implemented in software. The easiest, but most limited method is to modify the compiler to insert extra code collecting execution data and feeding it into the hash function. A more universal solution is to use virtualization techniques, such as on-the-fly code instrumentation (like the ones used in VMWare or the Valgrind debugger). This could be even easier for recent languages like Java or C# which already run on a virtual machine, where the virtual machine implementation could be modified to collect execution hashes. Our prototype implementation uses the first method, a modified C compiler to generate an execution hash. Even though this is not a universal solution, it is sufficient to study the impact of our data gathering on performance.

The are several important implementation details and tradeoffs that can have significant impact on the usability of this method. How long the hash should be? Suppose that an average job takes one hour to complete. If each such job is verified with a 32-bit hash function, the mean time between missing a mistake would be at least 490 thousand years. To get the actual number, we have to divide this time with the probability of failure, which means that in practice a 32-bit hash would miss less than one error in a million

year period. Of course, when many concurrent computation is being verified, the chance of passing an incorrect computation as a success increases, but even with a million concurrent computations, we can expect less than one misdetected failure in a year.

What type of hash function should be used? Cryptographic hash functions are tamper-proof, however they are also more costly to compute. Even though we cannot prove it, we conjecture that for our purpose, simple hash functions are sufficient. The main property we need for the execution hash function is that it must be as hard to compute as running the application which we want to trace. A simple hash function is not suitable in cryptography because for a known hash value it is easy to produce text with almost arbitrary content which hashes to the known value. However, in our case the hash value is the result of the computation, which is not known until the program has run, thus our conjecture is that the use of cryptographic hashes does not improve the trustworthiness of the result. This allows us to use any easy to compute hash function with a good distribution.

The use of execution hashes provide a cheap way to achieve highly reliable computation results without the need for expensive high-end hardware. In fact, the computations can be distributed over the network to external self-interested agents, by breaking up a large job into smaller work units and dispatching each work unit to two independent agents. Two agents returning the same hash result for a work unit guarantees with a very high probability that both agents have performed their job honestly and correctly, provided that the agents do not collude. This helps not only to deter agents to cheat, but it can also catch hardware errors.

Quality differentiation

In reality not all computations need to be 100% correct. For example when a distributed computing farm is used in a random search of some rare information object, such as a bug in a CPU design or a protein sequence with some interesting properties etc., then the main goal

is to maximize the coverage of the search space. On the other hand, there are computations where errors have catastrophic consequences, such as miscalculating the orbit of a space probe. In other words, consumers can be differentiated based on their error tolerance. Some consumers are willing to pay a high price for highly reliable results, but an unreliable result may have no value for them. Other consumers would pay much less for the most reliable results, yet they are still able to compete for resources because their willingness to pay is less sensitive to the reliability of the results.

Our proposed auditing mechanism enables this quality discrimination: it can provide practically 100% assurance for quality-sensitive consumers, and using the past reputation of the agents, it can also measure the expected failure rate of agents.

If there is enough demand for less than 100% reliable calculations, and the desired reliability can be achieved by most computers, then it is enough to implement an economic mechanism which would induce self-interested agents to honestly perform the computations assigned to them. This can be implemented at a much lower cost: instead of duplicating all computations, it is enough to repeat just a small sample of the work units, and "punish" defecting agents enough to make cheating unprofitable.

In addition to discouraging malicious behavior, our mechanism also provides a level of assurance against hardware failures caused by faulty components, and it creates incentives for grid providers to maintain their computers to keep their failure rates low. While errors caused by cosmic radiation are rare, and it equally affects everyone, errors from broken components are less evenly distributed: if there is a problem, it often shows up quickly, so if a program does not fail due to a hardware problem on the first day, then it is unlikely to fail in the future. Therefore, most hardware errors can be eliminated by more thorough checking of new machines. This also helps to eliminate malicious hardware errors coming from greedy agents overclocking their computers.

In the subsequent chapters we will summarize previous related results, we discuss

the technical implementation of the witness system, then we look at economic mechanisms against cheating. First we will show that simple random checking of each work unit does not work because it would require too much checking and too high payment. Then we'll show that unconditionally checking one randomly selected unit of a large bundle works well to discourage cheating, while it is of very low cost. Further, we will look at different types of consumers and suppliers which allow price-discrimination based on quality, and it naturally creates a need for a market for suppliers and consumers. We try to find a steady-state market equilibrium.

4.2 Related research

When the goal of a distributed computing effort is to invert a one-way function, i.e. given a function value, find an input where the function takes that value, the verification of the work is easier. Computing such a function produces a result which is in itself a sufficient witness of the computation, there is no need to add extra code. Golle and Mironov [46] show that for such problems the principal can verify the agents by pre-computing some results and pass these to the agents together with the desired goal value. To receive payment, the agents must find the inverse for all precomputed values. Unfortunately, this method is only practical for problems where the search space is very large and checking a given member of the search space is inexpensive.

Much of the existing research on distributed computing focuses on the optimal allocation of resources using economic mechanism design, where the allocation problem itself is often computationally intractable [47]. They also make simplifications about the agents where it is assumed that an agent is either a rational selfish node with access to fully reliable computing resources, or faulty nodes with no strategic behavior. In contrast, the agents we study here always act strategically, and their computers are not 100% reliable. And instead of using mechanism design to allocate jobs, we rely on an exchange market for reputations

where agents can signal their type and their willingness to lower their computational failure rates by purchasing a reputation.

Similar to the real stock market, to make the market work, we also introduce auditing which uses random checks on the agents' work and ties the payment for the work as well as their future reputation to the outcome of these checks. In the spirit of distributed computing, the auditing itself is distributed by randomly selecting a few jobs which are assigned to several agents, comparing their results and repeating this until two matching results are found. This match confirms the correctness of the matching results and also confirms the incorrect results from the other agents. In effect, we use unreliable distributed agents to perform reliable distributed auditing.

The monitoring we propose here is most similar to accounting rules and auditing of firms. Firms must document their income, expenses and investments, which is an overhead above the basic function of the firm, similar to the code we add to produce a verifiable witness. Firms are periodically audited, where auditors check random samples to verify the firms' operation.

Early papers about reputations assumed that the firm and its reputation is inseparable, thus reputation is not a tradeable asset. Many papers use the model in [48] where a single long-lived player faces opponents who participate only in one round, but can partially observe the previous actions of the long-running player. However, in this framework reputation can have a negative effect [49]. The problem with reputations is that once a good reputation is built, the firm decreases its effort and quality level and achieves higher profits by running down a good reputation. Introducing a market for reputations eliminates these issues, because lower quality is punished by the reduced value of the firm's reputation.

The monitoring together with the market for reputations reduces the issues related to career concerns [50].

The basic idea of using a general equilibrium analysis to analyze the market for firm reputations was first developed in [51], and this model was later substantially extended in [52]. Our reputation market model is similar to the model in [52], but we significantly extend their model. First, in our model a successful outcome cannot be easily distinguished from failure. Furthermore, we introduce an intermediary, who can audit the work of agents and can make payment contingent on a successful audit. Instead of a continuum of agent types, we will assume that there are only two types of agents, however, the types only determine the agents' cost of effort, and the agents choose their effort level, thus indirectly their success rate, strategically. Finally, we do not restrict agents to live for only two periods.

4.3 Technical implementation

We have used the C-Breeze C Compiler Infrastructure [53] to automatically add code to C programs which computes the execution trace hash. We do this by recording the direction of each conditional branch in the code. But recording the complete branch history can consume an unlimited amount of memory, and we are not interested in the actual branch history, we only use that to make sure that two independent computations match. For this it is enough to store a hash function of the complete branch history. Although we cannot prove this, we conjecture that for this purpose we do not need a cryptographic hash function, which would require substantial computation overhead. Our initial implementation uses a simple CRC32 hash function. The prototype implementation increases the run-time of applications by about 50%.

For each branch we assign a 32-bit memory location which holds a shift register with a history of the last 32 branches. At every 32nd branch a hash subroutine is called to merge the content of the shift register to the global branch history hash using CRC32. If a different hash function is desired, only this single hash function has to be modified.

4.4 Economic model

- In our economic model we have "consumers" who need to perform some expensive computations and we have "agents" who have machines with spare CPU time that they are willing to sell. Consumers and agents do not deal with each-other directly, instead they buy/sell services from/to a trusted intermediary.

- We assume that the intermediary deals with a large number of agents and consumers where no single agent has power to change prices.

- To simplify the discussion, we assume that all computation jobs can be divided to identical size units, although our results would hold for heterogeneous job sizes. We call these "Work Units."

- The principal can only verify the correctness of a result returned by an agent by repeating all the calculations. The computation process used to calculate each work unit produces a unique result that can only be obtained by performing all the computations.

- Agents do not collude, and if two agents independently return the same result, then the result is assumed correct.

- Agents can leave and create new identities at no cost.

- The principal has no power to impose penalties on agents returning false results without performing the computation beyond withholding payment to previously returned results.

- The principal keeps record of previously completed work units for each agent identity, and determines the future per unit payment solely based on this history.

- Agent identities are tradeable assets and the principal cannot connect agent identities to physical agents.

- We assume that consumer demand exceeds the available computation power offered by agents. Consumers bid for resources by submitting the per unit price they are willing to pay as a function of the expected failure rate. Based on this the principal calculates $p(\psi)$, the price of a work unit with an expected failure rate ψ, which is the highest bid price for ψ failure rate.

- Agents can reduce their failure rate by choosing their effort level. Let $\phi(w)$ denote the expected failure rate for effort level w and let $\chi_i(w)$ be the cost of effort for agent i. Note that ϕ is universal while χ varies based on the agent ability.

- The principal determines the rationally expected failure rate ψ_i of each agent based on the previous work record of their identities, and sets the per unit price paid to the agent to $p(\psi_i)$. Note, however, that the principal can delay payment and can withhold payment for past items, if an incorrect result is discovered.

The most obvious way to assure error-free calculations is to keep assigning the same job to agents until two result match (i.e. execution hashes match). However, if a small failure rate is acceptable, then we can do much better, as we only need to verify a small number of randomly chosen calculations to keep the agents honest.

4.5 Monitoring

Agents pick up work units from the principal when they are ready to do calculations and are expected to return the completed unit as soon as they finish. To ensure a timely completion of the jobs, each work unit has an expiration time after which the result is no longer accepted, and the principal reassigns the work unit to a different agent.

Agents get paid after every r completed work units subject to passing verification. From the batch of r work units the principal randomly picks one which she also assigns to an other agent. If the results returned by the two agents are identical, then that testifies that both agents have returned correct results, and both agents will get paid after they have completed their r units of calculations. If the results do not match, the principal will keep reassigning the unit to other agents until two of the results match. The two agents with correct calculations are paid for their r unit batches, but the other agents get no payment even if some of the other work units they have returned are correct.

The principal can perform the random draw for the items to be verified in advance, and when enough agents are available, she can use that to minimize the time an agent may have to wait for verification after she has completed a batch of r units. The only time the agent may have to wait is when the verified unit is her last unit in the batch. For these last units the principal can pick a unit which has been computed by an other agent, but if the two computations do not match, the agent will have to wait until her result for the last unit is proven to be correct or incorrect. The principal can minimize the probability of waiting by using a unit which has been previously calculated by an agent with a low expected failure rate. Given these measures we can assume that the effect of waiting for verification is negligible.

The principal always sells the unverified $(r-1)$ units from each batch of r units, but she only pays the agents, if the verification is successful, so the principal can earn money from selling the work of failed agents, however she has expenses to cover the cost of calculating the verified units twice. The principal's revenue from an agent with an expected failure rate ψ and a batch of r units is

$$(r-1)\psi p(\psi) - (1-\psi)\Big(p(\psi) - \frac{p(0)}{2}\Big)$$

r must be large enough to ensure that the principal is not losing money. It is also possible

that the principal earns money from some agents and loses on others depending on their expected success rate.

The agents do not know which unit will be verified, therefore the expected failure rate of the verification is the same as the expected failure rate of the agent. This means that the compensation is fair: the expected payment of the agent is proportional to the expected number of correctly calculated work units returned by the agent.

This is the least amount of verification the principal must do to ensure that agents do the work. If there were a positive probability that no work units are checked from the batch of r units, then the agent could do no work at all and could still expect payment in case the principal skips verification.

4.6 Market

We refer to agents working for the grid as grid providers. Grid consumers have jobs they need to compute. The grid platform provides a market to match consumers with providers. Consumers have a two-dimensional type, $(v, f) \in \mathbb{R}^2$, where v is the consumer's valuation for a correctly calculated work unit and $f - v$ is the consumer's loss, if the result is incorrect. Grid providers choose effort $w \geq 0$ which affects the probability $\phi(w)$ that a work unit is calculated incorrectly. Grid providers have a type $(c, \alpha) \in \mathbb{R}^2_+$. The cost of calculating a unit is $c + \alpha w$. ϕ, c, α and w are unobservable. Each provider has an identity, and each identity has a history of verified successes or failures. Identities can be traded. Consumers can only observe the agents current identity and the history attached to the identity. Consumers cannot observe the trade of identities. It is free to create a new identity with no past history. We assume that the expected future failure rate of a given identity only depends on the number of past successes and failures, i.e. the history can be summarized in a discrete two-dimensional measure $(n, k) \in \mathbb{N}^2$ for an identity with n verified successes and k failures. We search for a steady-state equilibrium where the expected failure rate for each

identity remains unchanged over time, let $\psi(n,k)$ denote this failure rate. The payment function is $p(\psi)$, the consumers utility is $v - \psi f - p(\psi)$.

Unreliable agents can provide reliable results by repeating the calculations until two matches. To get a guaranteed result, the principal would keep assigning the same work unit to different agents until two results match. The principal will only pay for the two successful calculations, so the cost for the principal is always $2p(\phi)$ if the agents' failure rate is ϕ.

Let p_0 be the cost of the cheapest agent. As we have shown above, by repeated calculations even the cheapest agent can deliver reliable result at a cost $2p_0$. The consumer must choose between a reliable calculation at cost $2p_0$ or a randomly checked calculation by an agent with a reputation score (n,k) with an expected failure rate of $\psi(n,k)$. For random checking, the principal assigns $r(n,k)$ work units to an agent, and randomly checks one of them by also assigning one out of the $r(n,k)$ units to an other agent. The calculation of this one work unit is repeated until two calculations match. The market maker can sell this checked work unit as a verified work unit for $2p_0$. The unverified units in a batch are sold to the consumers regardless of the result of the checks using the price corresponding to the success probability (that is, ψ) prior to the verification. The consumer's surplus is:

$$s(\psi) = v - \psi f - p(\psi)$$

The risk-neutral consumer chooses $\psi > 0$ if $s(\psi) > v - 2p_0$, have a positive expected profit, and ψ maximizes this profit, thus the following must hold:

$$p(\psi) \leq \min\{v, 2p_0\} - \psi f \tag{4.1}$$

$$p'(\psi) = -f \quad \text{if} > \text{holds in (4.1)} \tag{4.2}$$

Let $\psi(n,k)$ be the expected failure rate of an agent with a history of n verified successes and k failures. The payment for such an agent is $p(\psi(n,k)) = p(n,k)$ per unit, if he passes

the random verification of one out of r units. Let $V_0(n,k)$ be the current market price for a reputation score of (n,k) and let $V_1(n,k)$ be the market price in the next round. Before starting to work on a new batch agents can trade identities on the market. If there is any identity available which could provide higher expected payoff, then the agent would trade his old identity to this more profitable one. Note that the cost of trading the current identity to the new one must be factored into calculating the payoff. This means that we can assume that each agent trades at every period because they can sell and repurchase their current identity at no cost. We assume that no single agent could affect the market prices. Because of the trade after each round, we do not lose generality by assuming that agents live for one period, and where they purchase an optimal identity at the beginning of the period, execute r work units assigned to them, then receive their payments, update their identity, sell this updated identity on the market and retire. Agents choose their identity and their effort to maximize profit. We assume that there are no budget constraints and agents are risk-neutral. Let $\Delta_1 V(n,k) = \mathrm{E}V_1(n+1,k) - V_0(n,k)$, $\Delta_2 V(n,k) = \mathrm{E}V_1(n,k+1) - V_0(n,k)$. Note that $\Delta_1 V > 0$ and $\Delta_2 V \leq 0$. In fact, $V_1(n+1,\cdot)$ is not known until the next trade, however, it has an expected value, and the previously defined Δ in fact denotes the expected change in the value of the identity. Define

$$s(n,k) = p(n,k) + \frac{\Delta_1 V(n,k) - \Delta_2 V(n,k)}{r} \tag{4.3}$$

$s(n,k)$ the agent's revenue difference between passing and failing verification on a batch of r units accounting for both the lost payment and decreased reputation value. $s(n,k)$ is determined by the market, and it is not affected by the agent's effort. Using this we can express the agent's profit per unit:

$$\begin{aligned}\Pi(\alpha,n,k,w) &= p(n,k) + \frac{\Delta_1 V(n,k)}{r} - c - \alpha w - \phi(w)s(n,k) \\ &= s(n,k) + \frac{\Delta_2 V(n,k)}{r} - c - \alpha w - \phi(w)s(n,k)\end{aligned}$$

The optimal w satisfies:

$$-\phi'(w)s(n,k) = \alpha$$

The agent's effort is increased until the marginal cost of effort (α) equals to the marginal increase in income. Let's assume $\phi(w) = e^{-w}$, this implies $\phi(w)s(n,k) = \alpha$ and $w = \ln s(n,k) - \ln \alpha$, thus the agent's profit is

$$\Pi(\alpha, n, k) = s(n,k) + \frac{\Delta_2 V(n,k)}{r} - c - \alpha(1 + \ln s(n,k) - \ln \alpha) \tag{4.4}$$

$$\Pi(\alpha, n, k) = \alpha(e^w - 1 - w) + \frac{\Delta_2 V(n,k)}{r} - c$$

$\phi(w)s(n,k) = \alpha$ allows us to calculate the expected failure rate for a reputation of type (n,k):

$$\psi(n,k) = \frac{\mathrm{E}(\alpha|n,k)}{s(n,k)} \tag{4.5}$$

4.7 Separation

High-type agents would choose an identity which looses lot of value in case of a failure but it gives higher than average payment plus value increase in the case of success. This in turn implies that these agents choose high effort. Let us explore this idea more formally.

Let us assume that there are two agent types, (α_0, c_0) and (α_1, c_1) with $\alpha_0 > \alpha_1$ and $c_0 > c_1$, i.e. (α_0, c_0) is the low type (higher expenses for the same work). In a market equilibrium all agents of the same type have identical expected profit, even if they have different reputation scores, because if one reputation were to give higher expected profit than an other, the demand for that reputation would increase resulting in a higher price to remove the imbalance. Let π_0 and π_1 denote the expected profit of low and high type agents respectively, and $\Theta = \pi_1 - \pi_0$ be the expected profit difference between high and low type agents. Agents purchase a reputation which maximizes their profit.

If reputation (n,k) is used by low type agents only, then $\Pi(\alpha_0,n,k) = \pi_0$ and $\Pi(\alpha_1,n,k) < \pi_1$, therefore $\Lambda(n,k) = \Pi(\alpha_1,n,k) - \Pi(\alpha_0,n,k) < \Theta$. Similarly $\Lambda(n,k) > \Theta$ is reputation (n,k) is only used by high-type agents and $\Lambda(n,k) = \Theta$ when reputation (n,k) can be used by either types. $\Lambda(n,k)$ can be calculated from equation (4.4)

$$\begin{aligned}\Lambda(n,k) &= c_0 - c_1 + \alpha_0(1-\ln\alpha_0) - \alpha_1(1-\ln\alpha_1) + \\ &\quad (\alpha_0-\alpha_1)\ln s(n,k)\end{aligned}$$

Only the last term above depends on the reputation, all other terms are constant. This means that knowing Θ and $s(n,k)$ is enough to determine the type of agents using the reputation (n,k). There is a threshold Ξ such that reputation (n,k) is used only by low-type agents, if $s(n,k) < \Xi$, only used by high-type agents if $s(n,k) > \Xi$. The only time when the reputation does not reliably predict the type of its owner is when $s(n,k) = \Xi$.

This means that in the worst case we have a separating or semi-separating equilibrium, depending on if there is (n,k) such that $s(n,k) = \Xi$.

Equation (4.5) can be written as

$$\psi(n,k) = \begin{cases} \frac{\alpha_0}{s(n,k)} & \text{when} \quad s(n,k) < \Xi \\ \frac{\mathrm{E}(\alpha|n,k)}{\Xi} & \text{when} \quad s(n,k) = \Xi \\ \frac{\alpha_1}{s(n,k)} & \text{when} \quad s(n,k) > \Xi \end{cases}$$

This implies that even the hardest working low-type agent is using less effort than the lowest effort level of a high-type agent. Also, the expected failure rate from a low reputation (with $s(n,k) < \Xi$) is always strictly higher than the expected failure rate of a mixed reputation (if such exists) which in turn has a strictly higher failure rate than any high-reputation (i.e. $s(n,k) > \Xi$).

4.8 Consumer pricing

Let us assume two types of consumers, high-type consumers, who need verified results, which they either get from the intermediary and they would pay $2v_0$, or get by repeatedly

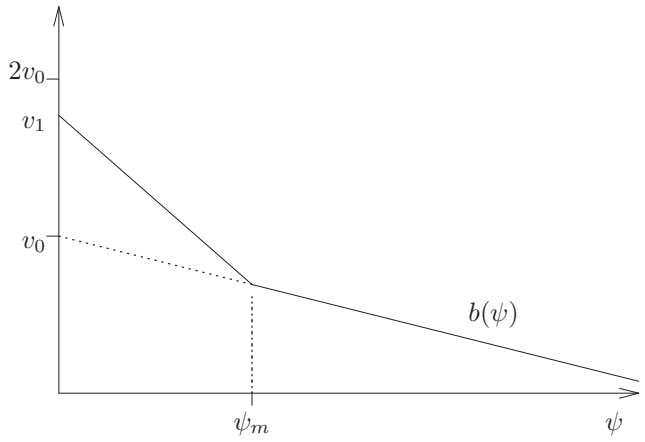


Figure 4.1: The price per unit as a function of the error rate ψ

submitting the same unit until two results match, which means their type is (v_0, v_0). An other consumer type is willing accept unverified results, with type (v_1, f_1), $v_0 < v_1 < 2v_0$. Further, we assume that demand exceeds supply, thus some consumers will not be served, and the consumers' surplus is zero. Consumers of the same type will choose from several different agent identities with differing quality of service, but agent identities with higher expected failure rate will require lower payment, making the consumers indifferent between these agents. If quality ψ is chosen by consumer type i, then $v_i - \psi f_i - p(\psi) = 0$, thus $p(\psi)$ falls on one of two linear functions. The intersection of these linear functions gives the quality level ψ_m that separates the consumption of low-type and high-type consumers: $v_0(1 - \psi_m) = v_1 - f_1\psi_m$:

$$\psi_m = \frac{v_1 - v_0}{f_1 - v_0}$$

Quality levels $\psi < \psi_m$ will only be used by high-type consumers, and $\psi > \psi_m$ will only be use by low-type consumers, therefore

$$p(\psi) = \begin{cases} v_1 - f_1\psi & \text{if} \quad \psi \leq \psi_m \\ v_0(1 - \psi) & \text{if} \quad \psi \geq \psi_m \end{cases}$$

$V(0,0)$ $p(0,0)$	$V(1,0)$ $p(1,0)$	$V(2,0)$ $p(2,0)$	$V(3,0)$ $p(3,0)$	$V(4,0)$ $p(4,0)$
		$V(2,1)$ $p(2,1)$	$V(3,1)$ $p(3,1)$	$V(4,1)$ $p(4,1)$
			$V(3,2)$ $p(3,2)$	$V(4,2)$ $p(4,2)$

Figure 4.2: The game board

$$\begin{aligned} \mathrm{E}(\alpha|n,k) &= \psi(n,k)s(n,k) \\ &= \psi(n,k)\Big(v_i - f_i\psi(n,k) + \\ &\qquad \frac{\Delta_1 V(n,k) - \Delta_2 V(n,k)}{r}\Big) \end{aligned} \tag{4.6}$$

4.9 Steady state

We try to find a steady sate where the portion of identity types do not change over time. Let $\lambda(n,k)$ denote the portion of the (n,k)-type identities.

$$\lambda(n,k) = \psi(n,k-1)\lambda(n,k-1) + (1-\psi(n-1,k))\lambda(n-1,k)$$

For the boundaries, for each n there is a number of failures above which the identity becomes worthless and disappears from the market, denote this by $d(n) \geq 0$. Clearly, d must be non-decreasing and $d(0) = 0$. We define $\lambda(n,k) = 0$ for $k > d(n)$ and $k < 0$. In a steady state equilibrium where the number of agents do not change, each discarded identity must be matched by a new $(0,0)$ identity entering the game:

$$\lambda(0,0) = \sum_{n=0}^{\infty} \psi(n,d(n))\lambda(n,d(n))$$

$$\lambda(n,0) = \lambda(0,0)\prod_{i=0}^{n-1}(1-\psi(i,0))$$

The equilibrium can be described as agents moving on an infinite game-board, where squares of the board correspond to reputation scores. Agents can jump to any square on the board by purchasing a right to move to that square. Agents then perform computations of r work units, one of which is randomly verified, and based on the result of the verification, agents move either one square to the right (success) or one square down (failure). The lower-left portion of the board will have unused squares corresponding to worthless reputations. The upper-right squares are occupied by high-type agents, and the low-type agents live in between, with some squares possibly shared by both low and high types.

In an equilibrium trade will only occur at the boundaries, i.e. after a failure when a high-type agent would have to move to a low-type square or after a success when a low-type agent would have to move to a high-type square. Trading is not prohibited in other cases, but there is no gain that can be made. This means that we only need to know the value of these boundary squares. Probably this equilibrium is not unique.

We can first attempt to find an equilibrium where high type agents will always start at square $(h, 0)$ and will sell their reputation to low types after the first failure, i.e. high-types will always have a reputation $(n, 0)$ with $n \geq h$.

4.10 One-dimensional equilibrium

4.10.1 Pure strategy equilibrium

Or even more extreme, assume any reputation with a history of failure is worthless. This implies that low-types will sell to high types when they reach $(h, 0)$. Note that if there were such equilibrium, that would imply that high-types are subsidizing low-types.

To simplify notation in this one-dimensional case, we use the following:

$$V_i = V(i, 0),\ p_i = p(i, 0),\ \psi_i = \psi(i, 0),\ s_i = s(i, 0)$$

Also assume that the value and payment for all $(n,0),\ n \geq h$ reputations is the same V_h and p_h. This implies $s_h = p_h + V_h/r$ and

$$\frac{V_h}{r} = s_h - p_h = \frac{\alpha_1}{\psi_h} - p(\psi_h)$$

And for $i < h$ we have

$$\frac{V_{i+1}}{r} = s_i - p_i = \frac{\alpha_0}{\psi_i} - p(\psi_i)$$

For $i = h-1$ this implies

$$\frac{\alpha_1}{\psi_h} - p(\psi_h) = \frac{\alpha_0}{\psi_{h-1}} - p(\psi_{h-1})$$

Equation (4.4) can be rewritten as

$$\Pi(\alpha, n) = \frac{\alpha}{\psi_n} - c_0 - \alpha(1 - \ln \psi_n) - \frac{V_n}{r} \tag{4.7}$$

Recall that π_0 (π_1) denotes the expected profit of low-type (high-type) agents:

$$\begin{aligned} \pi_0 &= \frac{\alpha_0}{\psi_0} - c_0 - \alpha_0(1 - \ln \psi_0) \\ \frac{\pi_0 + c_0}{\alpha_0} + 1 &= \frac{1}{\psi_i} + \ln \psi_i - \frac{1}{\psi_{i-1}} + \frac{p(\psi_{i-1})}{\alpha_0} \\ \pi_1 &= p(\psi_h) - c_1 - \alpha_1(1 - \ln \psi_h) \end{aligned}$$

The agents do not like to lose money:

$$0 < \pi_0 = \frac{\alpha_0}{\psi_0} - c_0 - \alpha_0(1 - \ln \psi_0)$$

The high-type agent must prefer reputation $(h,0)$ to $(h-1,0)$:

$$\frac{\alpha_0}{\psi_{h-1}} - \frac{\alpha_0}{\psi_{h-2}} + p(\psi_{h-2}) - c_1 -$$

$$\alpha_1(1 - \ln \psi_{h-1} - \ln \alpha_1 + \ln \alpha_0) < p(\psi_h) - c_1 - \alpha_1(1 - \ln \psi_h)$$

$$\frac{\alpha_0}{\psi_{h-1}} - \frac{\alpha_0}{\psi_{h-2}} + p(\psi_{h-2}) - p(\psi_h) - \alpha_1\left(\ln \frac{\alpha_0}{\psi_{h-1}} - \ln \frac{\alpha_1}{\psi_h}\right) < 0$$

As before, let λ_0 be the number of low-type agents and λ_1 be the number of high-type agents. At every round $\psi_h\lambda_1$ high-type agents fail and start over purchasing a $(h,0)$ reputation. Let λ_z be the number of low-type agents with $(0,0)$ reputation:

$$\psi_h\lambda_1 = \lambda_z \prod_{i=0}^{h-1}(1-\psi_i)$$

$$\lambda_0 = \lambda_z \sum_{k=0}^{h-1}\prod_{i=0}^{k-1}(1-\psi_i)$$

Substituting λ_z we get

$$\frac{\lambda_0}{\lambda_1} = \psi_h \sum_{k=0}^{h-1}\prod_{i=k}^{h-1}(1-\psi_i)^{-1} \tag{4.8}$$

Eliminating π we are left with $h+1$ equations for $h+1$ unknowns:

$$\frac{1}{\psi_0} + \ln\psi_0 = \frac{1}{\psi_i} + \ln\psi_i - \frac{1}{\psi_{i-1}} + \frac{p(\psi_{i-1})}{\alpha_0}, \quad i \in \{1,\dots,h-1\}$$

$$\frac{V_h}{r} = \frac{\alpha_0}{\psi_{h-1}} - p(\psi_{h-1}) = \frac{\alpha_1}{\psi_h} - p(\psi_h)$$

$$\frac{\lambda_0}{\lambda_1} = \psi_h \sum_{k=0}^{h-1}\prod_{i=k}^{h-1}(1-\psi_i)^{-1}$$

Note that $1/x + \ln x$ is strictly decreasing for $x \in [0,1]$, as its derivative is $1/x - 1/x^2 = (x-1)/x^2 < 0$. Also

$$\frac{1}{\psi_{i-1}} - \frac{p(\psi_{i-1})}{\alpha_0} = \frac{V_i}{r\alpha_0} \geq 0$$

This implies $\psi_i \leq \psi_0$ for all $i \in \{1,\dots,h-1\}$. We also want to show $\psi_{i+1} \leq \psi_i$. Prove by contradiction, suppose $\psi_i \leq \psi_{i-1}$ and $\psi_{i+1} > \psi_i$, this would imply $V_{i+1} < V_i$ which leads to contradiction if $\frac{1}{\psi} - \frac{p(\psi)}{\alpha_0}$ is decreasing in ψ, which is true if ψ is small.

Incentive compatibility conditions: high-type agents make more money using high-reputations, low-type agents make more money with low reputations. No switching conditions:

$$\pi_1 = p(\psi_h) - c_1 - \alpha_1(1-\ln\psi_h) \geq$$

$$\frac{\alpha_0}{\psi_{h-1}} - \frac{\alpha_0}{\psi_{h-2}} + p(\psi_{h-2}) - c_1 - \alpha_1\left(1 - \ln\frac{\alpha_1\psi_{h-1}}{\alpha_0}\right)$$

$$\pi_0 = \frac{\alpha_0}{\psi_{h-1}} - \frac{\alpha_0}{\psi_{h-2}} + p(\psi_{h-2}) - c_0 - \alpha_0(1 - \ln\psi_{h-1}) \geq$$

$$p(\psi_h) - c_0 - \alpha_0(1 - \ln\psi_h)$$

All agents want to make a profit, i.e. $\Pi(\alpha, n, k) \geq 0$. Since the no-switching conditions ensure that high-type agents make more than low-types, it is enough to verify this for low-type agents, and since all low-type agents have the same profit, it is enough to look at reputation 0:

$$\frac{1}{\psi_0} + \ln\psi_0 \geq 1 + \frac{c_0}{\alpha_0}$$

Note that this always holds when $c_0 = 0$.

4.10.2 Solving the equations

For each h we can solve the equations above, and we can substitute the solution to the incentive compatibility conditions. Computer experiments show that for small h values the incentive condition which ensures that low-type agents do not use high reputation will not hold, while for large h values the high-type agents would want to switch to using low reputations. Our experiments have showed that there is either a unique h s.t. the solution satisfy the no-switching conditions which leads to a unique pure-strategy equilibrium, or there is a unique h where the solutions lead to a situation where low-type prefer reputation h to $h-1$, while high-types prefer reputation $h-1$ to h. In this later case we need to look for a mixed-strategy equilibrium where reputation h is shared by low and high type agents.

4.10.3 Mixed reputation

Looks like it is necessary in some cases that reputation $(h, 0)$ is shared between low and high type agents. Let σ be the portion of low-type agents using reputation h, and for simplicity

let ψ_h be the low-type failure rate. The expected mixed failure rate is

$$\sigma\psi_i + (1-\sigma)\frac{\alpha_1}{\alpha_0}\psi_i$$

Let $\gamma = \sigma + (1-\sigma)\alpha_1/\alpha_0$. Note that $\gamma < 1$.

$$\frac{1}{\psi_0} + \ln\psi_0 = \frac{1}{\psi_i} + \ln\psi_i - \frac{1}{\psi_{i-1}} + \frac{p(\psi_{i-1})}{\alpha_0}, \quad i \in \{1,\dots,h\}$$

$$\frac{V_{h+1}}{r} = \frac{\alpha_0}{\psi_h} - p(\gamma\psi_h) = \frac{\alpha_1}{\psi_{h+1}} - p(\psi_{h+1})$$

$$\pi_1 = p(\psi_{h+1}) - c_1 - \alpha_1(1 - \ln\psi_{h+1}) =$$

$$\frac{\alpha_0}{\psi_h} - \frac{\alpha_0}{\psi_{h-1}} + p(\psi_{h-1}) - c_1 - \alpha_1\left(1 - \ln\frac{\alpha_1\psi_h}{\alpha_0}\right)$$

Let λ_h be the portion of agents with reputation $(h,0)$.

$$\lambda_h = \lambda_z \prod_{i=0}^{h-1}(1-\psi_i)$$

$$\lambda_0 = \lambda_z \sum_{k=0}^{h-1}\prod_{i=0}^{k-1}(1-\psi_i) + \sigma\lambda_h \tag{4.9}$$

Divide by λ_h:

$$\frac{\lambda_0}{\lambda_h} = \sum_{k=0}^{h-1}\prod_{i=k}^{h-1}(1-\psi_i)^{-1} + \sigma$$

$$\lambda_1\psi_{h+1} + (1-\sigma)\lambda_h\left(\frac{\alpha_1}{\alpha_0}\psi_h - \psi_{h+1}\right) = (1-\sigma\psi_h)\lambda_h$$

$$(\lambda_1 - (1-\sigma)\lambda_h)\psi_{h+1} = (1-\gamma\psi_h)\lambda_h$$

$$\frac{\lambda_1}{\lambda_h} - (1-\sigma) = \frac{1-\gamma\psi_h}{\psi_{h+1}}$$

Divide by ψ_{h+1} and add to (4.9), use $\lambda_0 + \lambda_1 = 1$:

$$1 - \lambda_h = \frac{1-\gamma\psi_h}{\psi_{h+1}}\lambda_h + \lambda_z \sum_{k=0}^{h-1}\prod_{i=0}^{k-1}(1-\psi_i)$$

$$1 = \frac{1-(\psi_{h+1}-\gamma\psi_h)}{\psi_{h+1}}\lambda_h + \lambda_z \sum_{k=0}^{h-1}\prod_{i=0}^{k-1}(1-\psi_i)$$

$$1 = \frac{1-\gamma\psi_h}{\psi_{h+1}}\lambda_h + \lambda_z \sum_{k=0}^{h}\prod_{i=0}^{k-1}(1-\psi_i)$$

4.11 Conclusion

We show that in a reputation market where an intermediary has the power to measure and record the past results of agent identities and withhold payment when the verification has failed, the reputation can act as a good predictor for the expected success rate of an agent using that reputation, even if reputations are tradeable. We have studied a special one-dimensional reputation mechanism where any detected failure during an audit will make the reputation lose all of its value, thus the reputation is simply the number of past successful audits for a given reputation.

Computer simulations have shown that in this game the higher reputation will correspond to more reliable results, and agents self-select a reputation reflecting their types: low reputations are only used by low-type agents, high reputations are only used by high-type agents and there can be at most one reputation level which can be shared by both low and high type agents.

Our conjecture is that this game always leads to a unique equilibrium, which is either fully-separating with low-type agents only using reputation $(h-1)$ or lower, while high-type agents use reputation h or above, or the equilibrium is semi-separating, where reputation h is used by both agent types and the low-type agents use a randomized strategy to decide if they sell a reputation h or take an other batch of calculations, and try to finish that in order to be able to sell reputation $(h+1)$ later.

In our future research we would like characterize the equilibria using the more general two-dimensional reputation measure and, if possible, show that the reputation market

helps to avoid equilibria where high-type agents may sell their reputation to a low-type agent after a successful audit.

Bibliography

[1] W. Wang, Z. Hidvégi, A. D. Bailey, Jr., A. B. Whinston, E-process control and assurance using model checking, IEEE Computer 33 (10) (2000) 48–53.

[2] Z. Hidvégi, W. Wang, A. B. Whinston, Buy-price english auction, Journal of Economic Theory 129 (2006) 31–56.

[3] Ernst & Young, Doing eCommerce Right (Mar. 15 2000).

[4] K. Zetter, Holey software, PC World.

[5] R. Weber, Information Systems Control and Audit, 2nd Edition, Prentice Hall, Upper Saddle River, NJ, 1999.

[6] The Information Systems Audit and Control Foundation, IT Governance, COBIT: Control Objectives for Information and related Technology, 3rd Edition (Jul. 2000).

[7] American Institute of Certified Public Accountants, Report of the Special Committee on Assurance Services (1996).

[8] American Institute of Certified Public Accountants, AICPA/CICA SysTrust Principles and Criteria for System Reliability, exposure draft, version 2.0 (Apr. 2000).

[9] K. D. Nagel, G. L. Gray, Electonic Commerce Assurance Services – Electronic Workpapers and Reference Guide, Harcourt Professional Publishing, San Diego, New York, Chicago, London, 1999.

[10] J.-M. Jézéquel, B. Meyer, Design by contract: The lessons of ariane, IEEE Computer 30 (2) (1997) 129–130.

[11] R. Stenger, Nasa report: Software, tight budget doomed mars lander, http://www.cnn.com/2000/TECH/space/03/28/lander.report.03/index.html.

[12] A. Basu, R. W. Blanning, A formal approach to workflow analysis, Information Systems Research 11 (1) (2000) 17–36.

[13] C. Karamanolis, D. Giannakopoulou, J. Magee, S. M. Wheater, Formal verification of workflow schemas, submitted for publication (Apr. 2000).

[14] S. Sarkar, M. Ramaswamy, Knowledge base decomposition to facilitate verification, Information Systems Research 11 (3) (2000) 260–283.

[15] M. Sighireanu, K. J. Turner, Requirement capture, formal description and verification of an invoicing system, Tech. Rep. Rapport de recherche n3575, INRIA, thème 1 (1998).

[16] W. Janssen, R. Mateescu, S.Mauw, P. Fennema, P. Stappen, Model checking for managers, in: The Sixth International SPIN Workshop on Practical Aspects of Model Checking, 1999.

[17] W. Janssen, R. Mateescu, S. Mauw, Verifying business processes using spin, in: Proceedings of the 4th International SPIN Workshop, Paris, France, 1998, pp. 21–36.

[18] N. Szirbik, G. Wagner, Steps towards formal verification of agent-based e-business applications, in: Workshop on Modelling of Objects, Components, and Agents (MOCA'01), Aargus, Denmark, 2001.

[19] J. C. Westland, Research report: Modeling the incidence of postrelease errors in software, Information Systems Research 11 (3) (2000) 320–324.

[20] R. P. Kurshan, Program verification, Notices of the AMS 47 (5) (2000) 534–544.

[21] R. P. Kurshan, Computer-aided Verification of Coordinating Processes, Princeton University Press, 1995.

[22] R. P. Kurshan, Formal verification in a commercial setting, in: Proceedings of the Design Automation Conference, 1997.

[23] R. V. Alan MacCormack, M. Iansiti, Developing products on 'internet time': The anatomy of a flexible development process, Management Science 47 (1) (2001) 133–150.

[24] E. M. Clarke, R. P. Kurshan, Computer-aided verification'90, DIMACS: Series in Discrete Mathematics and Theoretical Computer Science 3.

[25] M. Kaufmann, P. Manolios, J. Moore, Using the ACL2 Theorem Prover: A tutorial Introduction and Case Studies, Kluwer Academic Publishers, 2000.

[26] E. M. Clarke, O. Grumberg, D. A. Peled, Model Checking, The MIT Press, Cambridge, Massachusetts, 1999.

[27] G. Duke, J. Gerlach, C. Ko, R. Meservy, A. D. Bailey, Jr, A. B. Whinston, Ticom and the analysis of internal control, The Accounting Review (1985) 186–201.

[28] M. Burrows, M. Abadi, R. Needham, A logic of authentication, Technical Report 39, DEC Systems Research Center.

[29] E. M. Clarke, S. Jha, W. Marrero, A machine checkable logic of knowledge for specifying security properties of electronic commerce protocols, in: Proceeding of IFIP Working Conference on Programming Concepts and Methods (PROCOMET), 1998.

[30] W. R. Kinney, Jr., Research opportunities in internal control quality and quality assurance, *working paper* (2001).

[31] S. S. Reynolds, J. Wooders, Ascending bid auctions with a buy-now price, *Working Paper.* (Aug. 2002).

[32] T. Mathews, The impact of discounting on an auction with a buyout option: a theoretical analysis motivated by ebay's buy-it-now feature., Journal of Economics (Zeitschrift für Nationalökonomie) 81 (1) (2004) 25–52.

[33] R. Wang, Auctions versus posted-price selling, The American Economic Review 83 (4) (1993) 838–851.

[34] E. B. Budish, L. N. Takeyama, Buy prices in online auctions: irrationality on the internet?, Economics Letters 72 (2001) 325–333.

[35] M. J. Brennan, The pricing of contingent claims in discrete time models, Journal of Finance 24 (1979) 53–68.

[36] R. B. Myerson, Optimal auction design, Mathematics of Operations Research 6 (1) (1981) 58–73.

[37] J. G. Riley, W. F. Samuelson, Optimal auctions, The American Economic Review (1981) 381–392.

[38] L. M. Ausubel, P. Cramton, The optimality of being efficient, *Working Paper* (Jun. 1999).

[39] T. Mathews, A risk averse seller in a continuous time auction with a buyout option, Brazilian Electronic Journal of Economics 5 (2).

[40] W. Wang, Z. Hidvégi, A. B. Whinston, Shill-proof fee (spf) schedule: the sunscreen against seller self-collusion in online english auctions, *Working Paper* (Oct. 2004).

[41] A. E. Roth, A. Ockenfels, Last-minute bidding and the rules for ending second-price auctions: Evidence from ebay and amazon auctions on the internet, American Economic Review 92 (4) (2002) 1093–1103.

[42] M. H. Rothkopf, R. M. Harstad, Modeling competitive bidding: A critical essay, Management Science 40 (3) (1994) 364–384.

[43] E. Maskin, J. Riley, Optimal auctions with risk averse buyers, Econometrica 52 (6) (1984) 1473–1518.

[44] D. H. Lucking-Reiley, Auctions on the internet: What's being auctioned, and how?, Journal of Industrial Economics 48 (3) (2000) 227–252.

[45] Trusted platform module (TPM) specifications, Tech. rep., Trusted Computing Group, *https://www.trustedcomputinggroup.org/specs/TPM/* (2007).

[46] P. Golle, I. Mironov, Uncheatable distributed computations, Lecture Notes in Computer Science 2020 (2001) 425–441.

[47] J. Feigenbaum, S. Shenker, Distributed algorithmic mechanism design: Recent results and future directions, in: Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, ACM Press, New York, 2002, pp. 1–13.

[48] D. Fudenberg, D. K. Levine, Maintaining a reputation when strategies are imperfectly observed, The Review of Economic Studies 59 (3) (1992) 561–579.

[49] J. Ely, D. Fudenberg, D. K. Levine, When is reputation bad, Games and Economic Behavior (forthcoming), *http://dx.doi.org/10.1016/j.geb.2006.08.007*.

[50] R. Gibbons, K. J. Murphy, Optimal incentive contracts in the presence of career concerns: Theory and evidence, The Journal of Political Economy 100 (3) (1992) 468–505.

[51] S. Tadelis, What's in a name? reputation as a tradeable asset, The American Economic Review 89 (3) (1999) 548–563.

[52] S. Tadelis, Firm reputations with hidden information, Economic Theory 21 (2-3) (2003) 635–651.

[53] C. Lin, S. Z. Guyer, D. Jimenez, The c-breeze compiler infrastructure, Technical Report TR-01-43, The University of Texas at Austin, *http://www.cs.utexas.edu/users/c-breeze/* (Nov. 2001).

Vita

Zoltán Tibor Hidvégi was born in Budapest, Hungary on December 26, 1970. He received the Master of Science (equivalent) degree in Mathematics from the Budapest Eötvös Loránd University in 1995. He has entered the Ph.D. program in Mathematics Eötvös Loránt University, but he has abandoned his studies in 1997 to accept employment at the IBM Corporation Engineering Design Automation group, where is still working full-time designing and implementing the functional logic simulation engine and boolean optimization algorithms used to simulate the microprocessor and system designs developed by IBM and its partners. In the fall of 2000 he has started his graduate studies at the University of Texas at Austin.

Permanent address: 1517 Jerusalem Dr
Round Rock, TX, 78664

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.