Developing an Auction Theory Toolbox

Christoph Lange[1] and **Colin Rowat**[2] and **Wolfgang Windsteiger**[3] and **Manfred Kerber**[4]

Abstract. Auctions allocate trillions of dollars in goods and services every year. Auction design can have significant consequences, but its practice outstrips theory. We seek to advance auction theory with help from mechanised reasoning. To that end we are developing a *toolbox* of formalised representations of key facts of auction theory, which will allow auction designers to have relevant properties of their auctions machine-checked. As a first step, we are investigating the suitability of different mechanised reasoning systems (Isabelle, Theorema, and TPTP) for reproducing a key result of auction theory: Vickrey's celebrated 1961 theorem on the properties of second price auctions – the foundational result in modern auction theory. Based on our formalisation experience, we give tentative recommendations on what system to use for what purpose in auction theory, and outline further steps towards a complete auction theory toolbox.

1 MOTIVATION

Auctions are a widely used mechanism for allocating goods and services (trillions of dollars each year[5]), perhaps second in importance only to market mechanisms. Auctions are used to allocate electromagnetic spectrum, airplane landing slots, bus routes, oil fields, bankrupt firms, internet domains [5], works of art, eBay items, as well as to establish exchange rates, treasury bill yields, and opening prices in stock exchanges. Auction design can have significant consequences. Klemperer attributed the low revenues gained by some governments when auctioning their 3G radio spectrum in 2000 (€20 per capita vs. €600 in other countries) to bad design [11].

Further, the practice of auction design outstrips theory, especially for more complex modern auctions, such as combinatorial auctions in which bids may be submitted on subsets of items (e.g. collections of spectrum, bus routes, landing slots). Important auctions often run 'in the wild' with few formal results [10].

We aim to advance auction theory with help from mechanised reasoning: representing the knowledge underlying auction mechanisms in a formal, explicit way, and verifying these formalisations using computer support. Mechanised reasoning has been successfully applied, e.g., for verifying software that controls commuter rail or payment systems [27]. It has also been applied in economics [9], particularly to social choice theory (cf., e.g., [7]) and game theory (cf., e.g., [22]). However, all of this work has been done by computer scientists, not by economists. The formalisation of (mathematical) theories and the application of mechanised reasoning tools remain novel to economics. Ultimately, we aim to make such techniques more familiar to auction theorists by providing them with a toolbox of basic auction theory formalisations, on top of which they can formalise and verify their own auction designs.

[1] School of Computer Science, University of Birmingham, UK
[2] Department of Economics, University of Birmingham, UK
[3] RISC, Hagenberg, Austria
[4] School of Computer Science, University of Birmingham, UK
[5] Extrapolated from the figure of $268.5 billion reported by the US National Auctioneers Association for 2008 [2].

2 REQUIREMENTS

From the perspective of a domain expert, i.e. an auction designer, the auction theory toolbox (ATT) should satisfy the following requirements:

D1 Provide ready-to-use formalisations of basic concepts of auction theory, including their definitions and their essential properties

D2 Allow for extension and application to custom-designed auctions without requiring expert knowledge of the underlying mechanised reasoning system

From a computer scientist's perspective, these requirements translate to the following, more technical ones:

C1 Identify the right language to formalise auction theory, i.e. a language that is sufficiently expressive for capturing relevant concepts, while supporting efficient proofs for the majority of relevant problems.

C2 Identify a mechanised reasoning system that allows for formalising auction designs in a way that is close to the textbook style economists are used to, and that facilitates reuse of any existing formalisations in the toolbox.

These two requirements cannot be treated independently of each other: a language that is adequate w.r.t. requirement **C1** may not be supported by any mechanised reasoning system that satisfies requirement **C2**.

3 APPROACH

We are building the ATT in parallel to identifying suitable languages and mechanised reasoning systems to avoid a chicken-and-egg problem. The right 'hammers' can only be identified when there are 'nails', i.e. concepts of the application domain to be formalised. In the case of a successful choice of language and system, these initial formalisations will form the core of the future toolbox.

3.1 The Nail: Vickrey's Theorem and Beyond

As the first nail, we chose William Vickrey's 1961 theorem on the properties of second price auctions of a single, indivisible good whose value is not publicly known. In such an auction, each participant submits a sealed bid; one of those with the highest bids wins, and pays the highest of the *remaining* bids; the losers pay nothing.[6] Vickrey proved that 'truth-telling' (i.e. submitting a bid equal to

[6] Wikipedia provides further background, including a discussion of variants used by eBay, Google and Yahoo! [25].

one's actual valuation of the good) was a *weakly dominant* strategy. This means that no bidder could do strictly better by bidding above or below its valuation *whatever* the other bidders did. Thus, the auction is also *efficient*, awarding the item to the bidder with the highest valuation. Vickrey was awarded economics' Nobel prize in 1996 for his work.

We have several reasons for starting our work by redoing an old proof: its formalisation will enable us to prove properties of contemporary related auctions as well; the underlying mathematical theory can be formalised in contemporary systems with reasonable effort; finally, we assume that domain experts being introduced to mechanised reasoning would rather trust this technology, which is new to them, if it is first applied to known results.

Eric Maskin collected high level versions of Vickrey's theorem and 12 others in his 2004 review [13] of Paul Milgrom's influential book on auction theory [14]. This review guides us in building the toolbox; here, however, we focus on Vickrey's theorem (restated as proposition 1 in Maskin's review), as it is the only one we have fully formalised so far.

3.2 Wielding the Hammer

We are formalising auction theory, starting with Vickrey's theorem, in three systems, which differ in logic, syntax and user experience. Before discussing these categories in detail, we discuss how we prepared ourselves for machine formalisation by refining the original paper source; we then introduce the three systems we are using.

3.2.1 Preparing the Paper Formalisation

Maskin's paper states Vickrey's theorem in two sentences of high-level text and proves it in another six sentences. On paper, we elaborated the theorem and its proof into a version that made the details explicit, resulting in eight definitions.[7] Mechanised reasoning systems generally require much more explicit statements than commonly found on paper: Automated theorem provers (cf. the detailed discussion in section 3.2.2) require them to find proofs without running out of search space, whereas proof checkers require proofs to be at a certain level of detail, which in turn requires detailed statements.

An initial attempt to formalise our elaborated proof distinguished cases on the basis of bidders' bids. This generated a multi-level case distinction with nine mostly straightforward leaf cases for Isabelle to check. While feasible, this was tedious, encouraging us to to re-write the cases on the basis of whether a participant won or lost the auction, whether by truthful bidding or otherwise. This resulted in four cases, once more largely straightforward.

We conjecture that such an extra step of elaborating the original paper source will typically be helpful before starting a machine formalisation. The history of proving Arrow's impossibility theorem on the non-existence of a fair aggregation of the preferences of a group of individuals, a central result of social choice theory, supports this claim: Nipkow's Isabelle formalisation [17] as well as Wiedijk's Mizar formalisation [24], both based on Geanakoplos' widely known paper version [6], required such elaborations where the paper source was imprecise. Beyond these formalisations, Tang and Lin obtained insights on the general structure of impossibility results in social choice theory only by studying and formalising a novel, induction-based proof [21].

[7] This 'paper formalisation' is available from the ATT homepage [12].

3.2.2 Choosing a Mechanised Reasoning System

In terms of *logic*, it is not immediately clear whether Vickrey's theorem is inherently a higher-order statement. It does make a statement about the maximum of an arbitrary number n of bids. Defining such an n-vector and proving essential properties of the maximum operation requires induction and thus goes beyond first-order logic. However, if one takes n-vectors and a maximum operation on them for granted, the rest of the formalisation does not require higher-order logic. That is, first-order logic suffices to formalise the concepts that are actually relevant from the perspective of the application domain: single good auctions, second price auctions, and the statement of Vickrey's theorem. First-order logic has the computational advantage that its statements are semi-decidable, and that sound and complete calculi exist. In practice, this means that a number of efficient automated theorem provers are available.

In terms of *syntax*, we assume that our domain experts, i.e. auction designers, will prefer a language that looks like the textbook mathematics they are used to, rather than one that has the flavour of a programming language. We assume that a typed language will support them in defining types for domain concepts (such as bids) and avoiding mistakes in formalisation.

In terms of *user experience*, one has to distinguish between *fully automated proving*, where systems are given a theorem and a knowledge base and they try to automatically find a proof of the theorem w.r.t. the given knowledge base, and *interactive proving*, where the author has to write a proof and have it checked interactively by the system. There are systems that confine themselves to one of these paradigms, but there are also systems that try to combine them.

We roughly describe the features of the languages and systems that we are using:

Isabelle/HOL [8]: higher-order logic (typed), supported by the Isabelle interactive theorem prover, accessible via the ProofGeneral [1] and jEdit [23] text editor interfaces

Theorema [26]: first-order logic with set theory[8], implemented as an add-on package for the Mathematica computer algebra system with its document-oriented notebook interface

TPTP FOF [19, 18]: untyped first-order logic, a machine-oriented language supported by several automated theorem provers, accessible for human authors and developers via the many-sorted first-order logic language CASL[9] [4], which the Hets interface [15] can translate to TPTP and send to provers.[10]

4 STATE, AND EXPERIENCES SO FAR

We performed the first complete formalisation of Vickrey's theorem in Isabelle.[11] We redid the same formalisation in CASL (to obtain TPTP) and thus showed that first-order logic is sufficiently expressive for Vickrey's theorem. Here, the full proof is still work in progress. Finally, we are redoing the formalisation once more in Theorema 2.0. The purpose of redoing the formalisation from scratch is to understand the specific advantages and disadvantages of the different

[8] The Theorema language is actually based on higher-order logic, but in our case we use FOL + set theory.

[9] CASL has a few higher-order features, such as inductive datatypes.

[10] TPTP has a typed (sorted) first-order form (TFF) as well [20]. As Hets is not currently capable of translating CASL to TPTP TFF, we are not using the latter. Note that CASL is more expressive than TPTP TFF in that it supports subsorts.

[11] There was no specific conceptual reason for starting with Isabelle.

languages and systems and to obtain a formalisation that is as idiomatic as possible.[12] The formalisations are available from the ATT homepage [12].

4.1 Theory Structure

Formalising further theorems from Maskin's review paper, some of which make statements about similar types of auctions, should reuse as much of the Vickrey formalisation as possible. To enable this, we have therefore organised them into modular theories. Four of them are essential from the domain perspective:

Single good auctions (basic concepts): the auction as a mechanism that maps bids to an outcome (i.e. uniquely allocating the good to one participant, and defining the participants' payments).

Properties single good auctions may have: efficiency and weak dominance of a given bidding strategy (cf. section 3.1).

Second price auctions: the definition (cf. section 3.1) and some lemmas one can infer from them, e.g. that if there is exactly one highest bidder, that bidder wins; functions to compute the payoff of a winner and a loser.

Vickrey's theorem: truthful bidding in a second price auction is a weakly dominant and efficient strategy.

4.2 Library Coverage

All of our formalisations are structured in terms of the theories listed above.[13] Depending on the system, we had to provide further mathematical foundations as additional theories. Theorema has a built-in tuple datatype which we used to formalise vectors of bids, and supports a maximum operation on such tuples. CASL supports inductive datatypes, and its standard library provides a number of them, including arrays [3, 16], but there is no built-in n-argument maximum operation. The Isabelle/HOL standard library provides a *Max* operation on finite sets; however, given Isabelle's functional programming style syntax, we found it most intuitive to model our own vectors as functions $\mathbb{N} \to \mathbb{R}$ evaluated for arguments up to a given n. Thin wrappers make the set maximum operator usable for these vectors and prove the properties required subsequently.

4.3 Level of Detail Required by the Machine

Even the elaborated paper version introduced in section 3.2.1 turned out to be insufficient for direct machine formalisation. The Isabelle formalisation of the four theories listed above comprises 4 additional, auxiliary definitions, and 7 auxiliary lemmas. We estimate that a similar number of auxiliary statements will be required to guide the automated provers of Theorema and those that support TPTP. On the other hand, our initial steps of extending the formalisation beyond Vickrey's theorem suggest that the auxiliary material makes it easier to formalise further notions.

[12] Hets is, for example, capable of translating CASL to Isabelle, but the resulting Isabelle code would not make use of higher-order features other than inductive datatypes.

[13] In Theorema, which does not support a formal notion of theory, we chose to use notebook sections for structuring. Once further reuse will be required, we may change this to *archives*: collections of definitions, theorems, etc., which a notebook can load for reuse.

4.4 User-friendliness of Input Syntax

While we have not yet collected feedback from actual domain experts, we assume that they will find Theorema's input syntax most accessible. The two-dimensional symbolic notation of Mathematica notebooks is similar to textbook notation; additionally, large parts of our target audience are familiar with Mathematica already. The appearance of Isabelle and CASL is closer to programming languages; however, both allow for defining operators with a custom 'mixfix' notation. The ProofGeneral and jEdit interfaces of Isabelle display the operators built into the language or defined in the standard library in a one-dimensional approximation of textbook style, using the appropriate Unicode characters. Finally, TPTP's pure ASCII syntax is not extensible by custom symbols.

4.5 Interactive or Automated?

Our tentative verdict on interactive vs. automated approaches is that it does not matter: What matters, instead, is that the systems give good error messages, which allow the user to tell where exactly the formalisation is wrong or insufficient, and why exactly the system failed to check or to find a proof. Given that the user provides a proof that proceeds in sufficiently small steps, the jEdit interface for Isabelle gives localised feedback on errors. When Theorema tries to prove a theorem, it develops a structured textbook-style proof at a configurable level of verbosity. The user can navigate it via a tree view and thus quickly identify where Theorema failed to proceed. For CASL, Hets itself performs a type check, and otherwise relies on the output of the theorem provers it invokes.

4.6 Community Support

Community support is another criterion that facilitates adoption of a system. The user communities of the systems considered here mainly comprise computer scientists and therefore may not be ready to answer questions that a domain expert with little computer science background may have, as they often assume previous knowledge about mechanised reasoning, mathematical formalisation, and the specific reasoning approach underlying the respective system. Therefore we simply compare the sizes of the communities for now, assuming that domain experts can expect better assistance from a larger community. Isabelle is developed by multiple institutions and has a large user community; more than 100 posts per month are made to its user mailing list. CASL has been standardised in an international effort and has been the subject of several hundred scientific publications but does not currently have a functional mailing list. Hets is mainly developed and used within a single institution; its user mailing list receives less than 10 posts per month. TPTP has been the subject of more than a thousand publications but does not have a mailing list. Theorema does not have a mailing list either and has so far been developed as closed source software within a single institution, but this is expected to change soon with the open source release of Theorema 2.0.

5 CONCLUSION AND OUTLOOK

Auctions allocate trillions of dollars in goods and services every year, but still it is not well understood how to design a 'good' auction. Our auction theory toolbox (ATT) currently formalises key results about single good second price auctions, a simple, well-understood type of auction. It does so in a modular way, which makes us confident that

we will be able to build formalisations of more complex and new auction types on top of the existing core, ultimately enabling computer-supported verification of auction designs. We are planning to put this hammer right into the hand of domain experts, i.e. auction designers. To that end we are, at the same time as we are building the toolbox, identifying those mechanised reasoning languages and systems whose user experience is closest to the domain experts' mindset.

The logics, languages and systems studied so far satisfy our technical requirements **C1** and, thanks to their support for modular theories, **C2** as well. Our Isabelle formalisation, the most complete one so far, satisfies the domain expert's requirement **D1** for second price auctions but needs to be expanded to other types of auctions. An assessment of how well the ATT satisfies requirement **D2** can only be made once there is a larger core ATT, on top of which we – or rather: auction designers themselves – will have formalised new auctions.

ACKNOWLEDGEMENTS

We would like to thank Makarius Wenzel for considerably improving the Isabelle formalisation of Vickrey's theorem, Till Mossakowski for his help with the CASL formalisation and with using Hets, Marco B. Caminati for a productive exchange of ideas concerning his Mizar formalisation of Vickrey's theorem, as well as Geoff Sutcliffe for his input on TPTP.

References

[1] David Aspinall et al. *Proof General*. 5th Nov. 2012. URL: http://proofgeneral.inf.ed.ac.uk/ (visited on 2012-12-10).

[2] National Auctioneers Association. *Auctions: The Past, Present and Future*. 2010. URL: http://www.sarasotabestauctions.com/resources/History_of_Auctions_3.10.10.ppt.

[3] M. Bidoit and Peter D. Mosses. *CASL — the Common Algebraic Specification Language: User Manual*. LNCS 2900. Springer Verlag, 2004.

[4] *CASL*. CoFI. 12th Feb. 2008. URL: http://www.informatik.uni-bremen.de/cofi/wiki/index.php/CASL (visited on 2012-12-10).

[5] Cramton Associates. *Applicant Auction for Top-Level Domains*. 2012. URL: http://applicantauction.com (visited on 2012-12-05).

[6] John D. Geanakoplos. 'Three brief proofs of Arrow's impossibility theorem'. *Economic Theory* **26**(1) (2005), pp. 211–215.

[7] Christian Geist and Ulle Endriss. 'Automated search for impossibility theorems in social choice theory: ranking sets of objects'. *Journal of Artificial Intelligence Research* **40** (2011), pp. 143–174.

[8] *Isabelle*. 5th May 2012. URL: http://isabelle.in.tum.de (visited on 2012-09-14).

[9] Manfred Kerber, Christoph Lange and Colin Rowat. *An economist's guide to mechanized reasoning or My computer just proved 84 impossibility theorems*. Ed. by Kenneth Judd. Invited lecture at the Initiative for Computational Economics summer school. 25th July 2012. URL: http://cs.bham.ac.uk/research/projects/formare/pubs/ice2012/2012-07-25-ice-mech-reas-econ.pdf.

[10] Paul Klemperer. 'The product-mix auction: a new auction design for differentiated goods'. *Journal of the European Economic Association* **8**(2–3) (2010), pp. 526–536.

[11] Paul Klemperer. 'What Really Matters in Auction Design'. *Journal of Economic Perspectives* **16**(1) (2002), pp. 169–189.

[12] Christoph Lange et al. *Auction Theory Toolbox*. 13th Jan. 2013. URL: http://www.cs.bham.ac.uk/research/projects/formare/code/auction-theory/ (visited on 2013-01-13).

[13] Eric Maskin. 'The unity of auction theory: Milgrom's master class'. *Journal of Economic Literature* **42**(4) (2004), pp. 1102–1115. URL: http://scholar.harvard.edu/files/maskin/files/unity_of_auction_theory.pdf.

[14] Paul Milgrom. *Putting auction theory to work*. Churchill lectures in economics. Cambridge University Press, 2004.

[15] Till Mossakowski. *Hets: the Heterogeneous Tool Set*. URL: http://www.dfki.de/cps/hets (visited on 2012-12-10).

[16] Peter D. Mosses, ed. *CASL Reference Manual*. LNCS 2960. Springer Verlag, 2004.

[17] Tobias Nipkow. 'Social choice theory in HOL: Arrow and Gibbard-Satterthwaite'. *Journal of Automated Reasoning* **43**(3) (2009), pp. 289–304.

[18] Geoff Sutcliffe. 'The TPTP Problem Library and Associated Infrastructure: The FOF and CNF Parts, v3.5.0'. *Journal of Automated Reasoning* **43**(4) (2009), pp. 337–362.

[19] Geoff Sutcliffe and Christian Suttner. *The TPTP Problem Library for Automated Theorem Proving*. URL: http://www.tptp.org (visited on 2012-12-10).

[20] Geoff Sutcliffe et al. 'The TPTP Typed First-order Form with Arithmetic'. In: *Proceedings of the 18th International Conference on Logic for Programming Artificial Intelligence and Reasoning*. (Merida, Venezuela). Ed. by Nikolaj Bjørner and Andrei Voronkov. Lecture Notes in Artificial Intelligence 7180. Springer-Verlag, 2012, pp. 406–419.

[21] Pingzhong Tang and Fangzhen Lin. 'Computer-aided proofs of Arrow's and other impossibility theorems'. *Artificial Intelligence* **173**(11) (2009), pp. 1041–1053.

[22] Pingzhong Tang and Fangzhen Lin. 'Discovering theorems in game theory: two-person games with unique pure Nash equilibrium payoffs'. *Artificial Intelligence* **175**(14–15) (2011), pp. 2010–2020.

[23] Makarius Wenzel. 'Isabelle/jEdit – a Prover IDE within the PIDE framework'. In: *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics (CICM) (Bremen, Germany, 9th–14th July 2012). Ed. by Johan Jeuring et al. LNAI 7362. Berlin and Heidelberg: Springer Verlag, 2012, pp. 468–471. arXiv: 1207.3441 [cs.LO].

[24] Freek Wiedijk. 'Formalizing Arrow's theorem'. *Sādhanā* **34**(1) (2009), pp. 193–220.

[25] Wikimedia Foundation, ed. *Vickrey auction. From Wikipedia, the free encyclopedia*. 15th Nov. 2012. URL: http://en.wikipedia.org/w/index.php?title=Vickrey_auction&oldid=523230741 (visited on 2012-12-05).

[26] Wolfgang Windsteiger. 'Theorema 2.0: A Graphical User Interface for a Mathematical Assistant System'. In: 10th UITP workshop (at CICM 2012) (Bremen, Germany, 11th July 2012). Ed. by Cezary Kaliszyk and Christoph Lüth. 2012, pp. 1–8. URL: http://www.informatik.uni-bremen.de/uitp12/.

[27] Jim Woodcock et al. ‘Formal method: practice and experience’. *ACM Computing Surveys* **41**(4) (2009), pp. 1–40.