The Formalization of Vickrey Auctions: A Comparison of Two Approaches in Isabelle and Theorema

Alexander Maletzky* and Wolfgang Windsteiger

Research Institute for Symbolic Computation (RISC)
Johannes Kepler University Linz (JKU)
4232 Hagenberg, Austria
firstname.lastname@risc.jku.at

Abstract. In earlier work presented at CICM, four theorem provers (Isabelle, Mizar, Hets/CASL/TPTP, and Theorema) were compared based on a case study in theoretical economics, the formalization of the landmark Theorem of Vickrey in auction theory. At the time of this comparison the Theorema system was in a state of transition: The original Theorema system (Theorema 1) had been shut down by the Theorema group and the successor system Theorema 2.0 was just about to be launched. Theorema 2.0 participated in the competition, but only parts of the system were ready for use. In particular, the new reasoning engines had not been set up, so that some of the results in the system comparison had to be extrapolated from experience we had with Theorema 1. In this paper, we now want to compare a complete formalization of Vickrey's Theorem in Theorema 2.0 with the original formalization in Isabelle. On the one hand, we compare the mathematical setup of the two theories and, on the other hand, we also give an overview on statistical indicators, such as number of auxiliary lemmas and the total number of proof steps needed for all proofs in the theory. Last but not least, we present a shorter version of proof of the main theorem in Isabelle.

1 Introduction

The Theorem of Vickrey [13] formulates a key property of so-called *second-price auctions*. In this setting, one considers $n \in \mathbb{N}$ bidders participating in the auction of a single indivisible good. Each bidder submits a sealed bid, a bidder with the highest bid wins, and she has to pay the price given by the maximum of the remaining bids.

The Theorem of Vickrey provides a bidding strategy for participants in a second-price auction, namely it says, informally speaking, that *truthful bidding*, i. e. bidding the true valuation of the good, is a *weakly dominant strategy* for every participant, i. e. for each bidder the payoff is greater or equal to the payoff resulting from a different bid, and that truthful bidding is also *efficient*, i. e. it is guaranteed that the winner is a bidder with maximal valuation of the good.

* The research was funded by the Austrian Science Fund (FWF): P 29498-N31

As part of an effort to implement an auction theory toolbox [5], which should assist auction designers in the formal verification of properties of the auction mechanisms they design, four proof assistant systems of different nature (Isabelle [11, 14], Mizar [4, 1], Hets/CASL/TPTP [10, 9, 12], and Theorema [15]) were compared with respect to their suitability as a common platform for the auction theory toolbox [6]. At the time of publication the formalization in Isabelle was complete, whereas the Theorema formalization only contained the formulation of the theorem and all necessary definitions. Due to the complete redesign and reimplementation of the Theorema system at that time, the proofs were still missing and, in particular, the intermediate lemmas and their proofs were not yet known. The assessment of Theorema 2.0 in this comparison was therefore based on one hand on the user-interface, which allowed to judge the effort needed by a user to formulate the desired statements and the quality in which input and output of the system are presented to a user, and on the other hand on experience we had with the proving mechanisms in the predecessor system Theorema 1. The main motivations for the current paper are to close the gap of the missing formalization in Theorema 2.0 and to demonstrate the suitability of Theorema 2.0 as a platform for future formalization projects.

The definitions of the basic auction theory concepts follow the informal introduction given in [6], which is in turn based on influential auction theory literature [7, 8]. We then compare the Theorema formalization to the known formalization in Isabelle, which was the basis for the original assessment in [6]. We concentrate on differences and similarities in the *structure of the formalization* and in *technical details* concerning the formalization approaches chosen in the two systems. Moreover, *a new proof in Isabelle* is given based on the automatically generated proof of the main theorem in Theorema 2.0.

2 The Two Systems: A User's Point of View

In this section we want to sketch briefly how a task of "formalizing some part of mathematics" is carried out typically in the two systems Theorema and Isabelle. Theorema 2.0 provides a user interface based on Mathematica technology, for its details we refer to [2], for examples see the screenshots in Fig. 3. When it comes to proving, Theorema is designed as a multi-method system, i. e., it provides various proving methods that can be selected by the user depending on the application domain. Typically, a *method* consists of a collection of inference rules and a strategy to apply the rules in order to generate a complete proof in a fully automated fashion. One of the main improvements in Theorema 2.0 is the easy customization of pre-defined prove-methods[1]. Each inference rule can be deactivated by a single mouse-click and rule-priorities can easily be adjusted via drop-down menues in the Theorema-GUI. The primary goal in the Theorema

[1] It should be noted that not all methods that were implemented in Theorema 1 are already available in Theorema 2.0. The standard method available, which was used also in the current formalization, is a natural-deduction-like prover for first-order predicate logic with certain enhancements for Theorema-specific language constructs.

system is to provide well-curated default settings such that the fully automated proofs appear as if written by a well-trained mathematician.

In order to prove a formula G w. r. t. a knowledge base K the user has to just mark the formula G in the notebook, select all formulas that constitute K by mouse-click in the Theorema-GUI, and configure the prove method if necessary. All the rest is then done fully automatically, the resulting proof object is stored in a separate file, and a link to a human-readable presentation of the proof is put into the notebook. The proof search may fail when no applicable inference rule is available anymore, when a predefined search-depth is reached, or when a pre-defined search-time is exhausted. A valuable feature in case of a failed proof is the possibility to inspect failing proofs in order to come up with an improved setup for the next prove-run. By reading a failed proof one often gets the idea, which additional lemma in the knowledge base would help the prover to succeed (or at least to proceed further). Other possibilities to improve the setup include the activation/deactivation of certain inference rules (in order to prevent the prover from running into an undesired path) and the modification of rule-priorities (in order to force the prover into a desired path). This phase of fiddling with the prover setup is interactive, sometimes non-trivial, and very similar to what a mathematician is faced when doing proofs with pencil and paper (searching for auxiliary knowledge, changing one's prove strategy, etc.). A typical approach for getting started with a proof is to let it run with small search-depth and low search-time with only definitions in the knowledge base. The proof will probably fail. Then inspect the failing proof and draw conclusions for an improved setup. If things go well, increase search-depth and search-time.

Proving in Isabelle proceeds interactively, meaning that every step in a proof must be written down explicitly by the user. One of the main differences to Theorema (at least at the moment) is that these steps can be huge: one single application of a powerful proof method like *metis* could potentially solve goals that need dozens of steps in Theorema. In that sense, proving in Isabelle could also be regarded (semi-)automatic: the user outlines the main structure of the proof and then, for each remaining subgoal, invokes tools like Sledgehammer for automatically finding relevant facts from the background theory together with suitable proof methods that close the respective subgoals. We refer to [14] for a more thorough exposition of working with Isabelle.

3 The Two Formalizations: A Structural Comparison

3.1 The Content of the Formalizations

Fig. 1 shows the structure of the entire theory developed in Theorema.[2] The labels in the graph stand for individual formulas (D_i are definitions, L_i are

[2] Theorema knowledge archives, which will be an efficient way of storing Theorema formalizations for later use in a structured hierarchical build-up of theories, are not yet available in the current release of Theorema 2.0. Currently, the formalization of Vickrey's Theorem is written in one Mathematica/Theorema notebook containing the statement of all pieces of formalized maths (definitions, lemmas, and

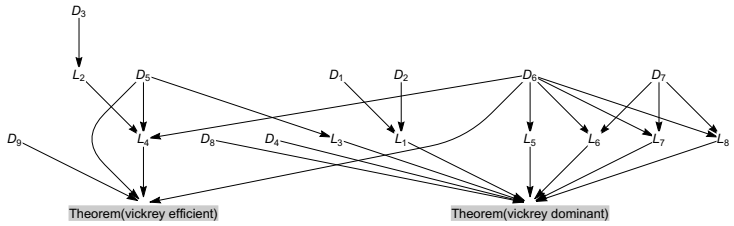

Fig. 1. The dependency graph of the formalization in Theorema.

Lemmas) and an edge $F \to G$ in the graph means 'formula F is needed in the proof of formula G'.

In Theorema we use natural numbers $1, \ldots, n$ for the participants in the auction. We then define the basic auction theory concepts *bids*, *valuations*, *payments*, and *allocations* as n-tuples of numbers (Def. D_1–D_3). In fact, a bid (valuation, payment) tuple b (v, p) contains non-negative numbers, where b_i (v_i, p_i) represents participant i's bid (valuation, payment) of the good. An allocation tuple x contains exactly one 1 and otherwise 0, where $x_i = 1$ means that participants i gets the good. Given a valuation v, an allocation x, and a payment p, participant i's *payoff* (Def. D_4) is then just $v_i x_i - p_i$, i. e. if she gets the good it is the difference of her valuation and the payment, if she does not get the good the payoff is 0.

For truthful bidding we have to use valuations as bids, hence we need a property stating that a valuation is always also permitted as a bid (Lemma L_1 in Fig. 1).[3] Then we show from the definition of allocations that the good cannot be assigned to more than one bidder (Lemma L_2). Next we define the basic ingredients of a second-price auction, such as the *outcome of a second-price auction* and the *participant i being winner (or loser) in a second-price auction* (Def. D_5–D_7). Fig. 1 displays nicely that the two statements in the main theorem are not proved directly from the definitions, but we introduce one layer of intermediate lemmas (L_3–L_8), from which the theorems are then proved. The proofs of the lemmas in this layer only need the definitions of the concepts involved, with the only exception being Lemma L_4, which states that in a second-price auction there can be at most one winner. The proof of this statement needs Lemma L_2 in addition to the definitions of a second-price auction and an auction-

theorems). A proof in Theorema 2.0 is represented in a data-structure called *proof object*, which contains the information about all logical steps the proof consists of. Proof objects and additional statistics about the proof run and user and system settings are stored in separate files. The Theorema formalization thus consists of the Theorema notebook together with its accompanying files, the current formalization being available for download in zip-format from the Theorema homepage at www.risc.jku.at/research/theorema/software/Vickrey.zip.

[3] This lemma could be omitted also, the four steps of its proof could as well be done in an extra branch of the proof of the main theorem. We rather see it as a means to structure the theory.

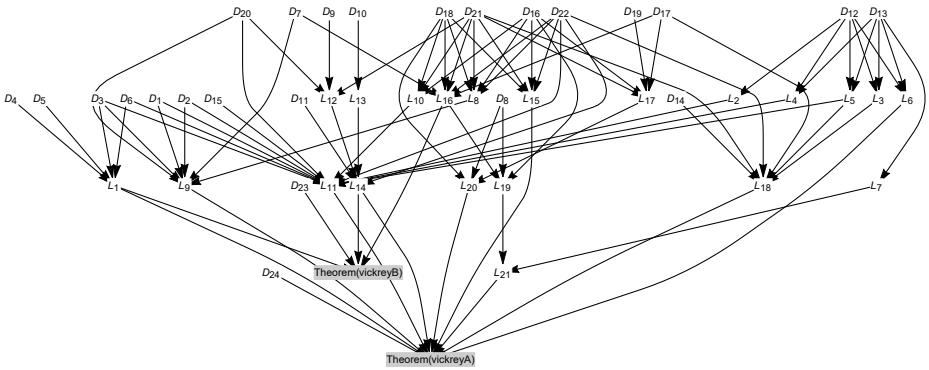

Fig. 2. The dependency graph of the formalization in Isabelle.

winner. The proof of the first part of Vickrey's theorem, truthful bidding being a weakly dominant strategy, uses above-mentioned Lemma L_1, another auxiliary Lemma L_3 (every participant in a second-price auction either wins or loses), and the 4 key Lemmas L_5 to L_8, which encapsulate knowledge about payments and allocations in case of modified bids. Lemmas L_5 and L_6 cover the cases where participant i wins with bid b_i and wins/loses with a modified bid a. Lemmas L_7 and L_8 are their counterparts for the case when i loses with bid b_i. The second part of the theorem, efficiency of truthful bidding in a second-price auction, only needs aforementioned Lemma L_4 and some definitions.

The formalization in Isabelle we used for our comparison is the one obtained from the web-page of the ForMaRE project.[4] It was created with Isabelle2013 and unfortunately contains some auxiliary lemmas whose proofs are not valid in the current version of Isabelle any more.[5] A vastly improved version of the formalization, which is concerned with the more general class of *combinatorial Vickrey-Clarke-Groves* auctions, is contained in the Archive of Formal Proofs (AFP) [3] and thus guaranteed to work in the current version of Isabelle. However, the AFP-version does not include the formulation of Vickrey's theorem that is the subject of the Theorema formalization, thus, it is not suitable as a reference for our present comparison.

The structure of the formalization in Isabelle, shown in Fig. 2, roughly resembles that of the formalization in Theorema: after proving some general facts about the maximum of functions over (finite) sets in theory `Maximum.thy`, and about vectors of real numbers in `Vectors.thy` (which does not have a counterpart in Theorema), the authors first introduce a couple of general concepts related to single-good auctions (e. g. valuations, allocations, bids, etc.) in theory `SingleGoodAuction.thy`. From this theory, the only lemma, that is needed later on in the proofs of the two main theorems, is Lemma L_1; this lemma exactly corresponds to Lemma L_1 in the Theorema formalization. Next comes the core

[4] www.cs.bham.ac.uk/research/projects/formare/code/auction-theory/isabelle/
[5] The proofs of the important theorems are still fine, though.

theory of the whole formalization: `SecondPriceAuction.thy` states the property of being a second-price auction in terms of the vectors of bids and payments and in addition introduces various other auxiliary notions (e. g. winners and losers of second-price auctions, as in Theorema). Furthermore, the authors prove several lemmas, for instance that second-price auctions result in non-negative payments for all participants (L_{11}), that every second-price auction is also a single-good auction (L_{14}), what the payoffs of the winner and the losers, respectively, are, and, most importantly, what the payoff of the winner is if she deviates from her valuation (L_{21}). Finally, theory `Vickrey.thy` contains the statements and proofs of the two main theorems about weakly dominant strategies (Theorem *vickreyA*) and efficiency (Theorem *vickreyB*) in second-price auctions; the definitions of weakly dominant strategies and efficiency are put into a separate theory `SingleGoodAuctionProperties.thy`.

Still, there are also some essential structural differences between the formalizations in Theorema and in Isabelle. First, as already indicated above, the latter one includes theories about vectors of real numbers and about properties of the maximum of functions over sets. One of these properties, which features a key role in the proof of *vickreyA* about weak dominance, is the obvious fact that changing the value of a function f at x does not change the maximum of that function f over the set A, provided that $x \notin A$. In Theorema, in contrast, this property is not stated as a formula on the object level, but on the meta level in a specific set of simplification rules for 'max', which is a built-in construct in the Theorema language. For more details, we refer to Section 4.

The second main difference is that the ultimate goal of the authors of the formalization in Isabelle apparently not only was to prove the two Vickrey-theorems, but to build up an auction theory toolbox that aims at formally checking properties of various—potentially complex—types of auctions. Hence, their formalization not only consists of definitions and results used in the proofs of the two theorems, but also many other definitions, lemmas and theorems that are needed elsewhere. Moreover, several of the definitions/lemmas that *are* needed are stated in a more general form than actually required. For instance, the definition of an equilibrium in weakly-dominant strategies is given for arbitrary classes of single-good auctions in Isabelle, whereas in Theorema it is restricted to second-price auctions. This is the main reason why the dependency graph of the Isabelle formalization, shown in Fig. 2, is considerably more complex than that of the Theorema formalization shown in Fig. 1, although the former already omits all definitions and lemmas irrelevant for the two main theorems.

Furthermore, a minor structural difference is that the four crucial lemmas L_5–L_8 in the Theorema formalization do not have analogues in Isabelle. Instead, the four cases in the proof of *vickreyA* they correspond to are proved directly, without making use of any lemmas of this kind.

3.2 The Size of the Formalizations

In [6] the 'de Bruijn factor', i. e. the formalization size divided by the size of an informal TEX-source, measured after stripping comments and xz-compression,

was used as criterion for the complexity/effort of the formalization. Theorema has already been exempted from that part of the comparison at that time due to the formalization missing the proofs. Still now with all proofs available, we do not think that measuring a Theorema formalization in terms of megabytes is the appropriate thing to do. Proofs are generated automatically including all the formatting of formulas and their nice appearance in the user front-end (including e. g. hyperlinks, tooltips, etc.). Since they need not be typed manually, their length in terms of kilo- or megabytes is irrelevant. Rather, a much more accurate measure for the effort to read and understand a proof is its size in terms of *proof steps*.[6] In Theorema there is a quite natural notion of proof step, it is one application of a proof rule. It should be mentioned, however, that not every proof rule needs to correspond to exactly one inference rule in classical logic. Theorema allows also specialized proof rules, which might combine several elementary inferences into a more complex step, or might involve rather complex computations in the background. Still, proof rules in Theorema are designed according to the principle that one step in a proof should be just as big that it can be perceived easily by a reader of a certain target audience. In the case of Vickrey auctions, we use a *standard predicate logic prover* enhanced with *a few special rules* dealing with *tuple operations* and the *maximum function*, hence targeting an audience that is familiar with the basic rules of logic, tuples and maximum, not more.

The entire formalization of Vickrey's Theorem in Theorema consists of 10 proofs with a total number of 171 proof steps. The most involved one is the proof of the first part of the theorem (truthful bidding is a weakly dominant strategy) with 41 steps.[7] Although the second part of the theorem (efficiency of truthful bidding in a second-price auction) is pretty straight-forward when given informally, it is interesting that the formal proof still needs 20 steps, ranking it as the fourth-biggest of all proofs.

The whole formalization in Isabelle consists of 247 proof steps in total. Note, however, that in Section 6 we illustrate how this number can be decreased to 185 by modeling the proof of *vickreyA* exactly after the proof of the corresponding theorem in Theorema. Counting proof steps in Isabelle is somewhat tricky, especially if the proofs are written in the Isar language: it is not clear how to count individual statements, whether to count them as one proof step, several steps, or not at all. For instance, one could argue that the combination '**unfolding** ⟨*defs*⟩ **by** *simp*', which unfolds definitions of constants in the current goal and then proves it by simplification, counts as two proof steps; the same effect, however,

[6] The number of steps is no measure for the effort needed to generate the proof, since the steps are generated automatically.

[7] To get a feeling, how big a 41-step proof is: Theorema's proof display with natural-language proof explanation consumes ample space because the explanation of every proof step starts in a new line and every formula is printed nicely 2D-formatted in a separate line (see a sample screen-shot of a proof in Fig. 4). In this format the proof is approximately 5 pages. Its fully automated generation took 110 seconds on a standard laptop (4 cores 2.20 GHz each), 106 seconds for proof generation plus 4 seconds for subsequent simplification of the generated proof.

could also be achieved by simply writing '**by** (*simp add:* ⟨*defs*⟩)', which would reasonably only count as one single proof step. Keeping these considerations in mind, it is obvious that there are many different ways of counting proof steps in Isabelle/Isar, and we do not claim that the counting schema we employed to arrive at the aforementioned 247 steps is the best one. Still, we believe that it is the *fairest* one for comparing proofs in Isabelle to proofs in Theorema. Roughly, every occurrence of **fix**, **assume**, **obtain**, **have**, **define**, **unfolding**, **qed**, **next** counts as 1 step; every occurrence of *of*, *OF* etc. counts as 1 step, too; and every occurrence of **proof**, **apply** and **by** counts as n steps, where n is the number of proof methods passed as arguments. Hence, even powerful proof methods like *auto*, *metis* etc. only count as 1 step, although simulating their behavior in Theorema would require lots of steps there.

4 The Two Formalizations: A Technical Comparison

Besides the structural differences between the two formalizations related to the different degrees of generality as discussed in the previous section, there are some technical differences—and similarities—as well.

A *second-price auction* is some mechanism that, given the number of participants n and the bids b, results in an allocation x of the good and a payment p. In Theorema, we decided to model the participants as natural numbers 1 to n with the effect that bids, valuations, allocations, and payments can be modeled as n-tuples of numbers, which are a built-in construct in Theorema. For $1 \leq i \leq n$ the i^{th} entry of such a tuple is then just the bid/valuation/allocation/payment of participant i. The key property of *being a second-price auction outcome* is modeled in Theorema as a 4-ary predicate `secondPriceAuction` on bids b, allocations x, payments p, and the number of participants n, where `secondPriceAuction`$[b, x, p, n]$ expresses that x and p satisfy the properties of a second-price auction for given b and n. The concepts 'being winner/loser in a second-price auction', 'being a weakly dominant strategy in a second-price auction', and 'being an efficient allocation' are also formulated as predicates on the respective tuples.

In contrast, the formalization in Isabelle makes heavy use of *sets*. More concretely, a general single-good auction in Isabelle is defined as a 4-tuple (N, b, x, p) consisting of the set of participants N, their bids b, the allocation x and the payment p, where bids, allocations, and payments are given as functions from type *nat* to type *real*.[8] Special kinds of auctions, like second-price auctions, are then

[8] Note that the authors of the formalization in Isabelle introduced a type-synonym 'vectors' for such function types, which corresponds exactly to what tuples are in Theorema. We want to emphasize, however, that the choice of tuples in Theorema is not system-enforced, it is more a matter of taste. The mathematical representation of the objects to be studied in Theorema can be chosen freely. In general, using built-in structures (like tuples) has the advantage of getting system support in computation and proving. Using non-built structures (like functions as done in Isabelle) would require to prove auxiliary knowledge about these entities. This knowledge would then

simply modeled as the sets containing precisely those tuples with the desired properties—but these properties are now given as 4-ary *relations* (i. e. functions whose result type is *bool*). So, in the end, sets and relations are mixed somewhat randomly, which in our opinion is counter-intuitive and occasionally leads to confusion. Instead, using relations exclusively would perhaps be more 'natural' and probably even fit better into the context of higher-order logic. Nevertheless, we shall emphasize that all this is largely a matter of taste; in the end, the difference between sets and relations boils down to a mere technicality that does not have a major impact on the formalization.

In Isabelle, the set of participants is not fixed to $\{1, \ldots, n\}$, but can be completely arbitrary (as long as it is finite and contains at least two elements). This is obviously more general, but we do not see any immediate advantage of it. After all, it is absolutely irrelevant for auction theory *what* the participants are, as long as one knows *how many* there are. Besides the fact that allocations in Theorema are modeled as tuples and in Isabelle as functions, they differ between the two formalizations in another respect: the definition of allocations in Isabelle in principle allows for *divisible* goods, as the values of an allocation are only required to be non-negative and to sum up to 1. Only the definition of second-price auctions incorporates the condition that allocations must allocate the good to one single participant. In Theorema, allocations need to have this property from the very beginning, making it slightly more difficult to reuse definitions and results from the existing formalization in potential future treatments of other kinds of auctions.

Another technical difference is related to the definition of *losers* of second-price auctions: in Theorema, the definition of a loser (i. e. a participant who is not awarded the good) explicitly requires that her bid be not the *unique* maximum among all bids, but at most the *tied* maximum. In other words: according to the formalization in Theorema, participant i is a loser iff her allocation and payment are both 0, and if additionally the tuple of bids b satisfies $b_i \leq \max(b_{i\leftarrow})$, where b_i is the tuple b at position i, and $b_{i\leftarrow}$ is the tuple b with position i deleted. However, this condition turns out to be redundant, since it automatically follows from the fact that in a second-price auction, by definition, one participant *with the highest bid* must be the winner, and all other participants must be losers. The formalization in Isabelle omits said condition and anything equivalent to it in the definition of losers and instead proves it as an auxiliary lemma.

Usually, it is tacitly assumed that at least two bidders participate in the auction, because otherwise the definition of the price as 'the maximum of the remaining bids' does not make sense. In a formal approach, of course, this assumption has to be made explicit. In [6] it has been suggested as an alternative to overcome this problem by defining $\max(\emptyset) := 0$. We do not consider this a viable choice because it is both unnatural and impractical, because it would mean that in case of only one participant in the auction, the good would be given away for free. Rather, we think that the common way of handling a second-price

be part of the formalization, like the knowledge about 'maximum' in the Isabelle formalization.

auction is to define the outcome in the special case of only one participant in a different way without referring to 'the maximum of the remaining bids'.

In the current state of the Theorema formalization, the assumption $n > 1$ is still omitted, and the prover does not yet check the side-condition $|t| > 0$ when it encounters an expression such as $\max[t]$. We consider this an open issue in the current formalization in Theorema, but it does not affect the principal feasibility of the presented approach. Since 'max' is a built-in construct of the Theorema language, we do not define 'max' as part of the auction formalization (as in Isabelle), rather we rely on available computation and reasoning rules for expressions involving 'max'. The Theorema computation engine already contained algorithms for operations on concrete tuples and, of course, in these algorithms all necessary side-conditions are really checked, e. g. the expression $\max[\langle\rangle]$ does not evaluate. As a generalization, we provide also *symbolic algorithms* on tuples, e. g. $(t_{i \leftarrow x})_i = x$, and the Theorema reasoning engine uses these algorithms in order to simplify expressions. In the case of maximum, these algorithms are then required to check $|t| > 0$ as soon as they encounter an expression $\max[t]$ with symbolic t. In order to do this, the computation engine needs access to the current knowledge base in case the computation occurs as part of a proof.

5 The Two Proofs of the Main Theorem

Fig. 3 shows the formulation of Vickrey's Theorem in the Theorema language. Roughly, it says, that for all[9] $n \in \mathbb{N}$, for all valuations v, allocations x, and payments p

1. using the valuations as bids is an `equilibriumWeaklyDominantStrategy` and
2. if the auction outcome with these bids conforms to the rules of a second-price auction then the allocation is `efficient`.

We want to concentrate on the first part of the theorem. Here b constitutes a *weakly dominant strategy* w.r.t. v and n iff b is a bid-tuple of length n, v is a valuation-tuple of length n, and for all $i = 1, \ldots, n$ and for all $\ldots$, see the definition of `equilibriumWeaklyDominantStrategy` in Fig. 3. The main part of this definition expresses that if i bids differently from b_i the payoff does not increase.

During the first phase of the proof, the universal quantifiers are eliminated. After expanding the definition of `equilibriumWeaklyDominantStrategy` we are left with three branches. The first two (v being a bid-tuple of length n and a

[9] The universal quantifier for n is not visible locally, neither in the theorem nor in the definitions, because we use a 'global universal quantifier' $\underset{n \in \mathbb{N}}{\forall}$ at the beginning of the document. This mechanism in Theorema 2.0 is explained in detail in [2]. The Theorema language is untyped. Quantifiers range over all objects that can be expressed in the Theorema language, i. e. sets, tuples, and various kinds of numbers available in Mathematica. There are special ranges with limited domain, such as $i = 1, \ldots, n$ for finite integer fragments. The domain can also be restricted using conditions. All what is needed in the current formalization is essentially first-order.

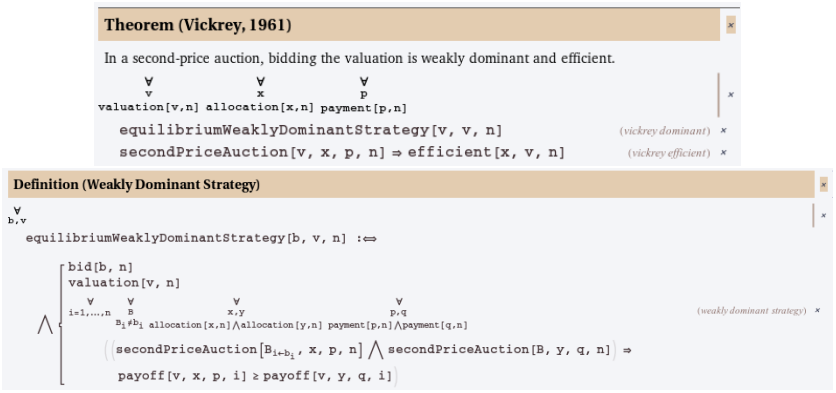

Fig. 3. The main theorem and the required definitions in Theorema.

valuation-tuple of length n, respectively) are trivial, the third one corresponds to the main part of the definition of `equilibriumWeaklyDominantStrategy`. Essentially by Lemma L_3, the two formulas

$$\text{secondPriceAuctionWinner}[B, y, q, i] \vee \text{secondPriceAuctionLoser}[B, y, q, i] \tag{1}$$

$$\text{secondPriceAuctionWinner}[B_{i \leftarrow v_i}, \bar{x}, \bar{p}, i] \vee \text{secondPriceAuctionLoser}[B_{i \leftarrow v_i}, \bar{x}, \bar{p}, i] \tag{2}$$

are derived. In (1) B represents an arbitrary bid, whereas in (2) $B_{i \leftarrow v_i}$ represents the bid B with v_i at position i. Now the disjunction (1) gives rise to a case distinction 'i wins with an arbitrary bid' vs. 'i loses with an arbitrary bid'. In each branch then the disjunction (2) initiates a case distinction 'i wins when bidding her valuation' vs. 'i loses when bidding her valuation'. Note that the order in which (1) and (2) enter the knowledge base is random. Theorema first distinguishes cases based on (1). Both in Isabelle and in the pencil-and-paper proof in [6] the cases when bidding her valuation are treated first. However, all proofs end up with the same four cases. In order to get an impression how a Theorema proof actually looks, we refer to the screen-shot in Fig. 4. The part of the proof displayed there corresponds to the case i loses with an arbitrary bid but wins when bidding her valuation. We see that this branch relies on Lemma *lose-win* (L_7) and then succeeds by basic rewriting. The remaining three branches proceed analogously making use of Lemmas L_5, L_6, and L_8, respectively.

In the Isabelle formalization the proof of *vickreyA* is not very complicated, but lengthy. It starts by introducing the necessary objects, like an arbitrary, but fixed participant i and two vectors of bids (where in one case participant i bids arbitrarily, and in the other case she bids exactly her valuation of the good), and proving a couple of simple properties of these objects. Then comes the key step of the proof: the case distinction depending on whether i is allocated the good if she bids her valuation or not, analogous the hand-written proof in [6]. Here it is worth noting that Lemma L_{15} is used in order to infer that a participant loses if she is not assigned the good, i. e. that $x(i) \neq 1$ implies $x(i) = 0$; this

means that L_{15} roughly corresponds to Lemma L_3 in Theorema. The first case, where i wins, proceeds by expressing i's payoff explicitly in terms of her valuation and the maximum of the remaining bids and by showing that this quantity is certainly non-negative, whereas the second case proceeds by showing that i's payoff is 0. Finally, in either case two further cases are distinguished depending on whether i wins with her alternative bid, yielding the same four cases as in Theorema. These four cases are not tackled using separate lemmas analogous to L_5–L_8 in Theorema, but proved directly taking into account the previously obtained properties of the payoff of participant i if she bids her valuation.

All in all, the proof in Isabelle closely follows the proof presented in [6]. It is considerably longer than the hand-crafted proof mainly because a lot of intermediate proof steps are required for inferring simple facts about bids, allocations, payments and payoffs that are more or less obvious, and hence omitted in [6].

6 A New Proof of *vickreyA* in Isabelle

The Isabelle proof of Vickrey's theorem is considerably longer than the corresponding proof in Theorema (even if one counts all the lemmas that are not proved separately in Isabelle, like L_5–L_8). This lead to the idea of translating the Theorema proof to Isabelle, to see whether the Isabelle formalization could perhaps be shortened; and indeed, with moderate effort we managed to reconstruct the Theorema proof almost one-to-one in Isabelle. Of course, we did not start completely from scratch but built upon the existing formalization as much as possible, which also necessitated a couple of new definitions and lemmas for establishing the connection to results that we wanted to reuse in our proof. In particular, we introduced new definitions of *winners* and *losers* that exactly resemble the definitions in the Theorema formalization, and we also proved another general result about the maximum of a function over a set, namely

```
lemma remaining-maximum-le-maximum:
  fixes A::"'a set"
    and f::"'a ⇒ 'b::linorder"
    and a::'a and b::'b
  assumes "card A > 1"
  shows "maximum (A - {a}) f ≤ maximum A (f(a := b))"
```

This lemma states that when removing an arbitrary element a from a set A, the maximum of a function f over that new set is certainly not larger than over the original set, even if f attains a different value at a. Stating and proving *remaining-maximum-le-maximum* was inspired by the formalization in Theorema where, however, the analogue of this lemma is not stated on the theory level, but instead on the reasoning level as a special inference rule.

Summarizing, the new proof of *vickreyA* depends on two additional definitions and seven additional lemmas. If the original proof of the theorem was replaced by the new one and all lemmas not needed any more were removed, the total number of proof steps would drop by 25% from 247 to 185, which is only slightly more than the 171 steps in Theorema.

```
next
  assume lose1: "spa-loser N b y q i"
  from cases2 show ?thesis
  proof
    assume win2: "spa-winner N (b(i := v i)) x p i"
    from lose1 card-N have "∀y' q' a. spa-winner N (b(i := a)) y' q' i ⟶
                              (q i = 0 ∧ y i = 0 ∧ y' i = 1 ∧ a - q' i ≥ 0)"
      using lose-win by auto
    hence "q i = 0 ∧ y i = 0 ∧ x i = 1 ∧ v i - p i ≥ 0" using win2 by auto
    hence "q i = 0" and "y i = 0" and "x i = 1" and "v i - p i ≥ 0" by simp_all
    show ?thesis unfolding payoff-def
    proof -
      from ‹q i = 0› ‹y i = 0› ‹x i = 1›
        show "v i * y i - q i ≤ v i * x i - p i"
      proof (simp del: diff-ge-0-iff-ge)
        from ‹v i - p i ≥ 0› show "v i - p i ≥ 0" .
      qed
    qed
  next ...
qed
```

Fig. 4. An Isabelle proof and a Theorema proof side by side.

In Fig. 4 we show the part of the proof, where participant i is assumed to lose with some arbitrary bid and wins when bidding her valuation, in both systems.[10] In the Isabelle proof, q and y denote the payment- and allocation vectors, respectively, if i bids arbitrarily, whereas p and x denote the payment- and allocation vectors, respectively, if i bids her valuation; v is the vector of valuations, and b is some arbitrary vector of bids. *cases2* is a local fact corresponding to *A#279* in the Theorema-proof (it expresses that i either wins for loses if she bids her valuation). The full new proof, together with the original proof and all auxiliary concepts from the original formalization the proof depends upon, is available online.[11] It works in Isabelle2016-1. For the Theorema proof, it should be noted that the proof is displayed using Mathematica notebook technology. It employs lots of interactive GUI-features that are hard to resemble in print, e. g. all formula labels[12] are actually hyperlinks that jump to the point where the formula has been introduced, labels have tooltips that show the entire formula when hovered above, and the interface shows a schematic clickable tree representation of the proof, that allows easy navigation through a proof. For the details we refer to [2].

7 Conclusion and Future Work

We presented two formalizations of Vickrey's theorem, one in Theorema and one in Isabelle. The formalizations are based on the same paper elaboration, but were done independently of each other in two different proof assistants, with quite different approaches toward theorem proving (automatic vs. interactive). The results are surprisingly similar—both in terms of structure and size, if one takes into account the increased generality of the formalization in Isabelle.

The paper does not aim at arguing why one of the two systems—Theorema or Isabelle—is better than the other, but we hope to have convinced the reader that despite their apparent differences both systems enable users to efficiently formalize mathematics with comparable effort and similar outcome. The strengths of Isabelle clearly are the extensive knowledge base of formal theories one can build upon, as well as the integrated support by automatic tools like Sledgehammer. Theorema, on the other hand, offers a quite unique way of presenting proofs in a form that even inexperienced users can easily comprehend, and also facilitates interaction with the system through an intuitive GUI. Summarizing, in our opinion Isabelle is better suited for experts in interactive theorem proving who are interested in doing large-scale formalizations efficiently, whereas Theorema 2.0 is more appealing to mathematicians who are not very familiar with

[10] The Isabelle proof is spelled out in more detail than necessary, in order to be easily comparable to the proof in Theorema; a single application of *fastforce* would suffice. The aforementioned 185 steps refer to the short version, which is not shown here.

[11] www.risc.jku.at/people/amaletzk/Vickrey.zip

[12] Definitions, lemmas, and theorems have user-defined labels. Formulas that are generated automatically during a proof have system-generated labels, where $A\#\ldots$ and $G\#\ldots$ refer to assumptions and goals, respectively.

proof assistants and to people (e. g. students) who want to learn the concept of mathematical proof, thanks to its intuitive handling and natural-style presentation of formal content. Of course, Theorema also claims for itself to enable large-scale undertakings.

As for current and future work, we have already mentioned the need for a more sophisticated means for checking side-conditions in symbolic tuple computations. The development of Theorema archives to have the possibility to efficiently store structured formalizations in Theorema language also deserves high priority.

References

1. Bancerek, G., Byliński, C., Grabowski, A., Korniłowicz, A., Matuszewski, R., Naumowicz, A., Pak, K., Urban, J.: Mizar: State-of-the-art and Beyond. In: Kerber, M., Carette, J., Kaliszyk, C., Rabe, F., Sorge, V. (eds.) CICM 2015. LNAI, vol. 9150, pp. 261–279. Springer (2015)
2. Buchberger, B., Jebelean, T., Kutsia, T., Maletzky, A., Windsteiger, W.: Theorema 2.0: Computer-Assisted Natural-Style Mathematics. JFR 9(1), 149–185 (2016)
3. Caminati, M.B., Kerber, M., Lange, C., Rowat, C.: VCG – Combinatorial Vickrey-Clarke-Groves Auctions. Archive of Formal Proofs (April 2015)
4. Grabowski, A., Korniłowicz, A., Naumowicz, A.: Mizar in a Nutshell. Journal of Formalized Reasoning 3(2), 153–245 (2010)
5. Kerber, M., Lange, C., Rowat, C., Windsteiger, W.: Developing an Auction Theory Toolbox. In: Kerber, M., Lange, C., Rowat, C. (eds.) AISB 2013. pp. 1–4 (2013), http://www.cs.bham.ac.uk/research/projects/formare/events/aisb2013/proceedings.php, proceedings available online
6. Lange, C., Caminati, M.B., Kerber, M., Mossakowski, T., Rowat, C., Wenzel, M., Windsteiger, W.: A Qualitative Comparison of the Suitability of Four Theorem Provers for Basic Auction Theory. In: Carette, J. (ed.) CICM 2013. LNAI, vol. 7961, pp. 200–215. Springer (2013)
7. Maskin, E.: The Unity of Auction Theory. Journal of Economic Literature 42(4), 1102–1115 (2004)
8. Milgrom, P.: Putting Auction Theory to Work. Cambridge University Press (2004)
9. Mossakowski, T., Haxthausen, A.E., Sannella, D., Tarlecki, A.: CASL – The Common Algebraic Specification Language, pp. 241–298. Monographs in Theoretical Computer Science, Springer (2008)
10. Mossakowski, T., Maeder, C., Codescu, M.: Hets User Guide. Tech. Rep. version 0.98, DFKI Bremen (2013)
11. Paulson, L.C.: Isabelle: The next 700 Theorem Provers. In: Odifreddi, P. (ed.) Logic and Computer Science. pp. 361–386. Academic Press (1990)
12. Sutcliffe, G.: The TPTP Problem Library and Associated Infrastructure: The FOF and CNF Parts, v3.5.0. Journal of Automated Reasoning 43(4), 337–362 (2009)
13. Vickrey, W.: Counterspeculation, Auctions, and Competitive Sealed Tenders. Journal of Finance XVI, 8–37 (1961)
14. Wenzel, M.: Isabelle/Isar Reference Manual (2017)
15. Windsteiger, W.: Theorema 2.0: A Graphical User Interface for a Mathematical Assistant System. In: Kaliszyk, C., Lueth, C. (eds.) Proceedings 10th International Workshop UITP. EPTCS, vol. 118, pp. 72–82. Open Publishing Association (2012), http://arxiv.org/abs/1307.1945v1, doi 10.4204/EPTCS.118.5