Logical Construction Games

Fourth Georgian Autumn School in Logic, Language and Information, Borjomi
Johan van Benthem, Amsterdam & Stanford, December 2001

Abstract

We define logic games for model construction tasks, inspired directly by Beth's semantic tableaus, and discuss their general properties and potential impact for logic.

1 Logic games

Many logical tasks can be cast in the form of games between two players. Well-known examples are so-called evaluation games between Verifier and Falsifier, model comparison games between Duplicator and Spoiler, of proof games between Proponent and Opponent. Logic games are discussed extensively in van Benthem 2001, whose Chapter 3 is the main source of this paper. Important logical notions then emerge as *winning strategies* of player in such games. Examples are a *proof* for Proponent in an argumentation, a *distinguishing formula* for Spoiler in a comparison between two models, or a *way of refuting* an assertion under evaluation in some model for Falsifier. The idea in designing such games is to take some logical task, such as evaluating a formula in a model, and pull it apart into two opposing roles that interact. Say, someone is trying to show that the formula is true, and someone else tries to show that it is false.

In the case of argumentation and proof, such attempts have met with some obstacles, though. Paul Lorenzen's pioneering work on logical dialogue dates back to the mid 1950s – but his system is somewhat messy, and in need of ad-hoc procedural conventions. Now Lorenzen corresponded with Evert Beth around 1955, and noted the similarity between what he was trying to achieve, and Beth's *semantic tableaus* (cf. van Ulsen 2001). In this note, we will look at semantic tableaus as an instance of yet one more logic game, whose structure is more elegant than that of pure proof games. The reason is this. In Lorenzen games, as in many people's thinking about argumentation, there is just one brave guy, viz. Proponent sticking his neck out and making a claim – whereas the other player is dragging his heels, or even engaging in sabotage to prevent Proponent from achieving his lofty purpose. But this asymmetry is misleading. In practice, two equally 'positive' roles are involved in argumentation, viz. *inference* and *consistency management*.

Consider a case in court. All the evidence is in, and the prosecutor must now show that the accused has committed the crime. That is his *burden of proof*. Inside that same game, however, it is the lawyer's task to come up with a *scenario* in which all the evidence is compatible with her client's not having committed the crime. The latter can be a positive activity as well. Indeed, even in Lorenzen games, it is much easier to understand what Opponent is doing by thinking of some intuitionistic model which she has in mind, where her moves in the game correspond to stepping form some world in that model to another, when shifting the focus of the discussion. Precisely this duality lies at the heart of tableaus – so much so, that we will even give the scenario player the priority in our description. After all, in real life, we are more often a person telling stories to an audience that have to stay consistent than a prover deriving consequences from given discourse. Moreover, model construction seems closer in spirit to the task of making sense of discourse than proof search – although it must be admitted that the latter paradigm also works, e.g., in the parsing-as-deduction, or the interpretation-as-abduction communities.

2 Tableau construction games

Semantic tableaus are a wide-spread method testing for logical validity of an inference by means of a systematic search for potential counter-examples making the premises true and the conclusion false. We assume that the reader is familiar with some version adequate for first-order logic (cf. Hodges 1977). We will cast tableaus as a game between two players, to be dubbed *Builder* and *Critic*. Both of them will have important roles to play.

> At each stage of the game, there are two finite boxes of formulas
> representing Builder's current tasks. One named YES contains
> the formulas to be made true, and NO those to be made false.

Think of Builder as creating *Paradise*, a situation with existential formulas in YES listing forthcoming attractions like apples and trees, and those in NO the snakes to be avoided. The moves of the game decompose complex formulas, or tasks. Play proceeds in *rounds*, each of which starts as follows:

> Critic schedules some formula for current treatment.

After that, Builder must respond in some manner to be defined. There are several options for gamifying tableaus, and we choose just one of these here. But first, some moves may be considered *automatic*, without any player involved at all:

if $\neg A$ sits in one box,	it changes to A in the other box
if $A\&B$ sits in YES,	it is replaced by both A, B separately
if $A \vee B$ sits in NO,	it is replaced by both A, B separately

if $\exists x\phi$ sits in YES,	it is replaced by $\phi(d)$ for some new object d not yet used in any formula in YES or NO
if $\forall x\phi$ sits in NO,	it is replaced by $\phi(d)$ for some new object d not yet used in any formula in YES or NO

In another version, we could give Builder more control over the latter two moves, allowing her to also choose from already available objects. This makes sense, e.g., if we are interested in finding smallest models, and make that part of the evaluation of the game. (As an incentive, we might give Builder a higher pay-off for smaller models.) Next, we list the rules that require deliberate actions by players:

(1) Disjunctions in YES and conjunctions in NO are a *choice* for Builder.

(2) For existential formulas $\exists x\phi$ in NO , Critic mentions some object in the domain under construction so far, and adds $\phi(d)$ to the NO box.
For universal formulas $\forall x\phi$ in YES , Critic mentions some object in the domain under construction so far, and adds $\phi(d)$ to the YES box.

The latter rule is not precise yet. In standard tableaus, Critic's moves do not *replace* the formulas $\exists x\phi$, $\forall x\phi$: these remain available for further calls later in the game. In that case, there is no special advantage to carefully choosing 'the right challenge', and case (2) may then also be considered 'automatic'.

Next, here is the *winning convention* of this game,

A stage is a *loss* for Builder if some formula occurs *in both boxes,*
Builder *wins* a complete run of the game if no such loss occurs at any stage.

The reason for this stipulation is clear. No model construction can make a formula both true and false: these are conflicting tasks. Finally, one might stipulate some *procedural conventions* – e.g., restricting Critic's scheduling of the game. Here is one that we will adopt, in order to avoid everybody's waste of time:

Critic *may not repeat schedulings* that he has already made.

Every semantic tableau for a propositional or predicate-logical satisfiability problem may be viewed as a record of a game like this – but see below for some divergences. Finally, Builder and Critic are not necessarily antagonistic roles (strife may not always be helpful as a way of understanding a logic game). One can also think of Critic as a *manager* making sure that Builder forgets no tasks, and eventually creates a building of the highest quality.

3 Examples: tableaus and game trees

Here are two logical construction problems demonstrating these ideas.

<u>Example 1</u> Propositional logic

Consider the following initial boxes for Builder:

YES $\neg A \vee B$, $\neg B \vee C$ NO $B \vee \neg(A \vee C)$

First, there is an automatic simplification in the NO box to the disjunction, and then automatic box transfer for one negation, resulting in:

YES $\neg A \vee B$, $\neg B \vee C$, $A \vee C$ NO B

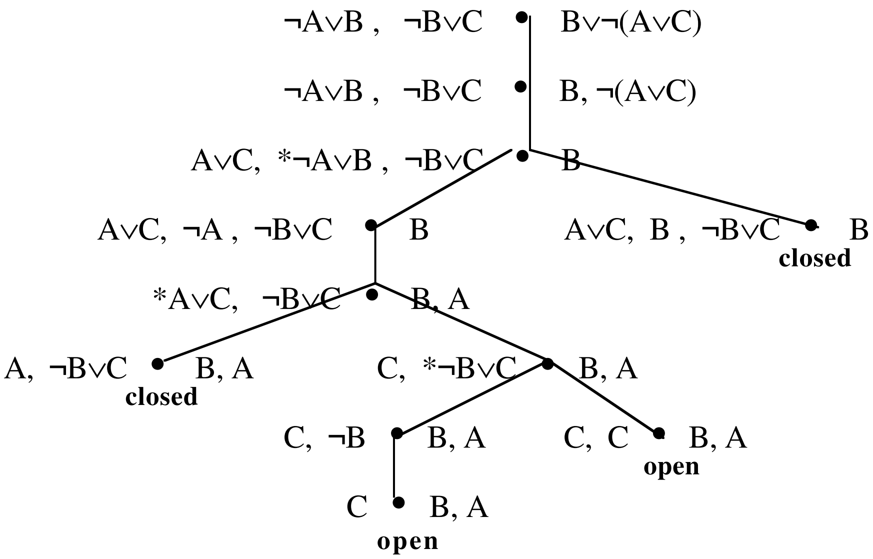
Now, Critic has to schedule assertions in the YES box, but it does not matter how he does this. (We analyze this below.) Say, he selects the first (with the *):

YES * $\neg A \vee B$, $\neg B \vee C$, $A \vee C$ NO B

Builder must choose A or B , giving a branching in the game tree. The second choice is an immediate loss, however, because of the conflict with the NO box. Therefore, the more interesting choice is $\neg A$, which leads automatically to

YES $\neg B \vee C$, $A \vee C$ NO B, A

Suppose Critic now schedules the second disjunct. Builder can choose $\neg B$ which puts B in the NO box: no new task; so perhaps the best move. Alternatively, Builder can put C in the YES box. After that, the last choice is scheduled, which puts either A in the YES box, which is a loss for Builder – or C in the YES box, which ends the game in a win for Builder. Which player (if any) has a *winning strategy* here? We display the complete semantic tableau, which is virtually the same as a game tree:



This tableau remains open, witness the open branches. These yield models solving the original task, making all formulas in YES true, and those in NO false. These models are at the same time Builder's winning strategies in the game tree corresponding to this tableau. Next comes a familiar predicate-logical example.

<u>Example 2</u> Shifting quantifiers

The YES box has the formula $\exists x\ \forall y\ Rxy$ and the NO box has $\forall y\ \exists x\ Rxy$. This should be a hopeless task for Builder, as the inference from $\exists x\ \forall y\ Rxy$ to $\forall y\ \exists x\ Rxy$ is valid. The semantic tableau might unfold as follows, starting with two automatic moves, and then two moves that can be seen as instigated by Critic:

$$\begin{array}{rcl}
\exists x\ \forall y\ Rxy & \bullet & \forall y\ \exists x\ Rxy \\
\forall y\ Rd_1\ y & \bullet & \forall y\ \exists x\ Rxy \\
\forall y\ Rd_1 y & \bullet & \exists x\ Rxd_2 \\
\forall y\ Rd_1 y\ ,\ Rd_1 d_2 & \bullet & \exists x\ Rxd_2 \\
\forall y\ Rd_1 y\ ,\ Rd_1 d_2 & \bullet & \exists x\ Rxd_2\ ,\ Rd_1 d_2 \\
 & \textbf{closed} &
\end{array}$$

In this case, Critic obviously has the winning strategy.

Tableaus are not yet complete games. In particular, complete game trees for the above examples would have to list *other* admissible *scheduling orders* for Critic. So, let us consider scheduling. First, players must keep moving when they still can. In particular, this forces Critic to examine every task in any finite game, as happens indeed in the propositional case. This feature explains why his scheduling order is not crucial to the outcome of the game. Eventually, every relevant task will be performed. But predicate-logical tableaus can be *infinite*, as some formulas have only infinite models, inducing infinite runs of the game. Tableaus normally impose some task scheduling making sure that every formula is eventually considered along such a branch of the tableau. Otherwise, the branch might remain open even though there is a hidden inconsistency. Nevertheless, we can also leave this open. We are mostly interested in players' winning strategies. The fact that Critic can be sloppy, so to speak, does not affect his winning strategies, which will obviously be more harsh qua scheduling.

This freewheeling attitude toward scheduling might suggest that Critic's precise moves are largely irrelevant. This would make construction games ceremonial disputes with one player doing all the work. There is more to these games, however – and we will return to this later.

4 Theory of tableaus and game theory

The usual adequacy result for semantic tableaus states that the method correctly tests for consistency and validity:

Theorem The following two assertions are equivalent in first-order logic:

(a) The set of formulas $\{A_1, .., A_k, \neg B_1, \ldots, \neg B_m\}$ is satisfiable

(b) There is an *open* tableau with top node $A_1, .., A_k \bullet B_1, \ldots, B_m$

Alternatively, one can also state this equivalence in terms of valid consequence. The following two assertions are also equivalent in first-order logic:

(a) $\&\{A_1, .., A_k\}$ logically implies $\vee\{B_1, \ldots, B_m\}$

(b) There is a *closed* tableau with top node $A_1, .., A_k \bullet B_1, \ldots, B_m$

Proofs are in the standard literature. In particular, they involve the observation that closed tableaus for first-order logic are always *finite*: a fact whose game-theoretic import is stated below. As we said, however, open tableaus may have irreducibly infinite branches. This feature reflects the complexity of the satisfiability or consequence problem for first-order logic, which is *undecidable*. Stated purely in terms of construction games, the above results express an equivalence between the following assertions:

(a) The set of formulas $\{A_1, .., A_k, \neg B_1, \ldots, \neg B_m\}$ is satisfiable

(b) Builder has a winning strategy in the above construction game starting with the A's put in the YES box and the B's in NO

Now let us look at the general situation from a game-theoretic point of view. The Adequacy Theorem implies an important property. As the tableau is either closed or open, either Critic of Builder has a winning strategy:

Proposition Model construction games are *determined*.

This can also be seen on general grounds. Even though tableaus can be infinite, the winning condition for Critic (i.e., finite failure by Builder) defines an *open* set of runs in the sense of the *Gale-Stewart Theorem* – which therefore predicts the determinacy. The core observation follows reflects a general game-theoretic

Fact Either Critic has a winning strategy, or Builder has a strategy making sure that only branches result at no stage of which Critic gets a winning strategy from there on.

The latter strategy is bound to produce a set of branches that remain open at every stage (since no win for Critic occurs): i.e., it is winning for Builder.

What players' winning strategies encode by way of independent logical objects becomes clear from a more concrete analysis of tableaus. Builder's winning strategies are related to open branches, which themselves generate *models* satisfying all initial requirements. By contrast, Critic's winning strategies allow him to end every branch of the game tree at some finite stage by a failure for Builder. Thus, the subtree of the complete game tree played according to that strategy *has no infinite branches*. One can think of this as a 'barrier' through the game tree. But then, Critic must have a *finite procedure* for achieving this – by contraposition of the following well-known mathematical result:

<u>König's Lemma</u> Each finitely branching infinite tree has an infinite branch.

This explains why winning strategies for Critic are associated with finite objects, namely 'closed tableaus'. And we know what the latter correspond to. They are proofs of the positive sequent corresponding to the initial formula

$$A_1 \,\&\, \ldots \,\&\, A_k \Rightarrow B_1 \vee \ldots \vee B_m$$

Summing up, we have found the following connection:

<u>Theorem</u> Construction games have an explicit correspondence between
(a) winning strategies for Builder,
(b) *models* for the given formulas
Likewise, construction games have an effective correspondence between
(a') winning strategies for Critic,
(b') *proofs* for the initial sequent.

Thus, the single notion of strategy for two players in one single game 'unifies' the notions of *model* and *proof*. These are normally taken to represent vastly different aspects of logic – while here, they meet directly. This double aspect was also present in Beth's original description of tableaus, where he related the two directions to 'analysis' and 'synthesis' as general practices in mathematics, emphasizing the complementary methods involved.

Our own view is that tableaus are eventually a kind of *task decomposition games*. Their general game-theoretic properties presuppose very little about the details of these tasks – but we shall now discuss some specifics.

5 Making Critic more essential: fragments and variations

In the propositional case, tableau games seem rather lifeless - as Critic is not really doing much. Nothing depends on his precise choice of scheduling. This is borne out by the *complexity* of the propositional satisfiability task, which is in **NP**. This means that, in order to win, Builder has to choose 'something good', which is of course, an open branch, or a lucky valuation satisfying all initial YES and NO demands. In that sense, we might just as well see the problem as a one-person game, where a winning strategy is one branch with this **NP** character. But things are different with full predicate logic, as Critic has to schedule in the right way in order to make every branch end at some finite stage (if possible at all).

But more interesting are intermediate cases. Consider *modal logic*, viewed as a bounded-quantifier fragment of the full first-order language (cf. Andréka, van Benthem & Németi 1998.). We know that the complexity of satisfiability is **PSPACE** for the basic modal logic, which is also the complexity of many genuine two-player games. The reason is as follows. In tableaus for modal logic, quantifiers are treated in a special fashion. Stated in terms of the universal modality only – for convenience – the instruction is as follows:

> Analyze the tableau at any world stage w as far as possible through propositional rules. If no closure occurs, then look at the remaining set of box formulas $[]\phi$ in YES , and the box formulas $[]\alpha$ in NO :
>
> For each of the latter $[]\alpha$, pick a new R–successor t of w , starting there with α in NO , and all ϕ of the first kind in YES

This fan-out of successors is a universal requirement for Builder, and the game gives Critic the choice which new world to open. This time, being able to cope with any challenge by Critic is essential – as is the fact that each challenge be handled separately, without interference from other parts of the tableau.

But also in full first-order tableaus, it is quite possible to let Critic do real work in a finite setting. Consider the following natural variation on the game. All rules become *one-shot*. In particular,

> For existential quantifiers in NO or universal ones in YES , Critic gets to issue one challenge: an object to be plugged in at some stage. The original formula is then discarded.

In purely propositional tableaus, this makes no difference, and neither does it for our modal tableaus – where the above rule had this one-shot character already.

But in full predicate logic, we see great differences now. Note that the second example of Section 3 now depends crucially on Critic's choices. Critic would lose in the following play:

A

$$
\begin{array}{rcl}
\exists x\, \forall y\, Rxy & \bullet & \forall y\, \exists x\;\; Rxy \\
\forall y\, Rd_1\, y & \bullet & \forall y\, \exists x\;\; Rxy \\
\forall y\, Rd_1 y & \bullet & \exists x\;\; Rxd_2 \\
Rd_1 d_2 & \bullet & \exists x\;\; Rxd_2 \\
Rd_1 d_2 & \bullet & Rd_2 d_2
\end{array}
$$

open

Here is are two very simple illustrations in predicate logic that used to be valid, but are no longer guaranteed wins for Critic in the one-shot regime:

B

$$
\begin{array}{rcl}
 & \bullet & \exists x\, (\, \exists y\; Ay \rightarrow Ax) \\
 & \bullet & \exists y\; Ay \rightarrow A\; d_1 \\
\exists y\; Ay & \bullet & Ad_1 \\
Ad_2 & \bullet & Ad_1
\end{array}
$$

open

In standard tableaus, one would *repeat* the instruction for $\exists x\, (\, \exists y\; Ay \rightarrow Ax)$, plugging in the new object d_2 to close the tableau. Here is a similar illustration:

C

$$
\begin{array}{rcl}
\exists x\, \exists y\, \exists z\, (\, Ax\, \&\, Rxy\, \&\, Ryz\, \&\, \neg Az) & \bullet & \exists u\, \exists v\, (\, Au\, \&\, Ruv\, \&\, \neg Av) \\
Ad_1\, \&\, R\, d_1 d_2\, \&\, R\, d_2 d_3\, \&\, \neg Ad_3 & \bullet & \exists u\, \exists v\, (\, Au\, \&\, Ruv\, \&\, \neg Av):
\end{array}
$$

Whatever Critic chooses for u and v here, Builder can always maintain a consistent claim. To demonstrate the classical validity, one needs to make a case distinction for the middle object , and depending on that, make a choice for u and v . In a standard tableau, this intuitive point is washed away by instantiating u and v a number of times, so that Builder will drown in the total requirements.

One-shot games represent a finer level of predicate-logical validity. In particular, it is easy to see that tableaus of this sort remain *finite*. This demonstrates a

Theorem The validities of one-shot predicate logic are *decidable*.

These one-shot validities can be axiomatized as follows. Proof-theoretically, the repetitive character of the quantifier rules has to do with the structural rule of *Contraction*, allowing us to treat more occurrences of the same formula as a single one. This rule is crucial here. E.g., consider an earlier example, but now duplicated. The following tableau does have a winning strategy for Critic!

$$
\begin{array}{rcl}
 & \bullet & \exists x\,(\exists y\,Ay \to Ax),\ \exists x\,(\exists y\,Ay \to Ax) \\
 & \bullet & \exists y\,Ay \to A\,d_1,\ \exists x\,(\exists y\,Ay \to Ax) \\
\exists y\,Ay & \bullet & Ad_1,\ \exists x\,(\exists y\,Ay \to Ax) \\
Ad_2 & \bullet & Ad_1,\ \exists x\,(\exists y\,Ay \to Ax) \\
Ad_2 & \bullet & Ad_1,\ \exists y\,Ay \to A\,d_2 \\
\exists y\,Ay,\ Ad_2 & \bullet & Ad_1,\ A\,d_2 \\
 & \textbf{closed} &
\end{array}
$$

Thus, the valid sequents can be axiomatized with a Gentzen calculus using all the usual 'logical rules' of predicate logic, plus all the standard 'structural rules' minus Contraction Tableau games stay finite, decidable, and axiomatizable with a fixed finite bound on repetitions of the same call in a game (Prijatelj 1995).

One-shot validity has a number of interesting features. First, there are quite a few cases where it does not diverge from classical validity. One example are inferences between existential formulas with only conjunctions, as in 'conceptual graphs' (Kerdiles 2001). In that case, a conclusion of the form

'$\exists x_1 \ldots \exists x_k$: conjunction of atoms'

follows from a set of premises of the same shape iff it holds in the minimal graph model of the premises – which takes just a one-shot choice of witnesses. But even atomic negations already spoil the picture: see the above example C. Another interesting case where one-shot calls or bounded finite numbers of calls are sufficient is *modal logic*. This is in fact clear from the above tableau instruction for modal operators (cf. Andréka, van Benthem & Németi 1998).

Now, one objection to all this might be that one-shot validity has *no semantics*. But this criticism is question-begging. It presupposes that standard semantics is the ultimate test for a game-theoretic account of validity – even though that order of priorities is precisely what is at issue in game-theoretic analysis of basic logical notions. In any case, there is an independent general concept behind one-shot predicate logic, viz. the view of formulas as *resources* that can be consumed only once – in this case, game tasks. This interpretation is close to *linear logic* (cf. Abramsky & Jagadeesan 1994), and indeed, the parallels are striking.

As in linear logic, our new look at the classical instruction for treating quantifiers in tableaus suggests *redesign of the first-order language*. If a task can be rescheduled indefinitely often, this should be marked explicitly by a

iteration operator $!\phi$

creating special versions of ϕ that can be scheduled an arbitrary finite number of times, depending on Critic's decision. This iteration operator makes a decidable system like one-shot predicate logic undecidable – as it leads to the resources for faithfully embedding full predicate logic. The new syntax is richer in effect than the standard first-order language, as we can 'iterate' arbitrary formulas – not just quantified ones, and that in both boxes. But this does not seem to increase expressive power. Thus, games suggest new logical operators.

On the other hand, the operator $!$ *need* not produce undecidability. Added to the modal fragment, it yields a version where modalities with universal force keep recurring through successor worlds, and so on. This leads to a still decidable bimodal system with one minimal modality, and one K4-modality accessing the transitive closure of the alternative relation R. Similar roles for iteration operators arise in *intuitionistic* propositional logic (again of **PSPACE**-complexity with a genuine role for Critic), regulating the action of implications and negations as one proceeds through a Kripke model.

6 Conclusion and further directions

The game view of tableaus provides a nice unified framework for discussing two logical tasks that are intuitively intertwined: proof, and model construction. It is also less ad-hoc than proof games of earlier varieties. Moreover, the framework suggests interesting variations that make sense from a game perspective, such as one-shot predicate logic. This paper has only scratched the surface though. E.g., a closer comparison with the procedural details of Lorenzen games seems of interest – treating Proponent in a gentler guise as a scheduler for Opponent, who is following hunches as to models of the situation. Next, at several points, a richer *pay-off structure* would make sense. For instance, if termination is our main concern, we could reward players for reaching outcomes along shortest paths. By making players prefer wins or losses that are reached quickly to those which take longer, one gets more interesting *strategic equilibria* for logic games. If you have a winning strategy, but I can choose where my defeat occurs, I will go for the quickest one, exercising a real power which I have in the game. On the other hand, infinite branches represent uninterrupted communicative interaction between two players, which would represent a good by another pay-off measure. (Cf. the linear logic games of Abramsky & Jagadeesan 1994.) Finally, infinite runs suggest more complicated pay-offs, as in repeated games with discounts of outcomes (Osborne & Rubinstein 1994). It would be of interest whether this makes sense for tableau games, or other logic games allowing infinite runs.

7 References (to be completed)

S. Abramsky & R. Jagadeesan, 1994, 'Games and Full Completeness for Multiplicative Linear Logic', *Journal of Symbolic Logic* 59:2, 543 – 574.

H. Andréka, J. van Benthem & I. Németi, 'Modal Logics and Bounded Fragments of Predicate Logic', *Journal of Philosophical Logic* 27:3, 1998, 217–274.

J. van Benthem, 2001, *Logic in Games*, lecture notes, ILLC Amsterdam.

W. Hodges, 1977, *Logic*, Pelican Books, Harmondsworth.

W. Hodges, 1985, *Building Models by Games*, Cambridge University Press.

W. Hodges, 2001, 'Elementary Predicate Logic', in D. Gabbay & F. Guenthner, eds., *Handbook of Philosophical Logic*, Volume I, 1–129 , second edition, Kluwer, Dordrecht.

G. Kerdiles, 2001, *Saying it with Pictures*, dissertation, ILLC Amsterdam.

M. Osborne & A. Rubinstein, 1994, *A Course in Game Theory*, MIT Press, Cambridge (Mass.).

A. Prijatelj, 1995, *Studies on Bounded Contraction*, dissertation, ILLC Amsterdam.

P. van Ulsen, 2000, *Scientific Biography of E.W. Beth*, dissertation, ILLC Amsterdam.