Multi-Agent Dialogue Games and Dialogue Sequents for Proof Search and Scheduling

Martin Sticht

University of Bamberg
martin.sticht@uni-bamberg.de

Abstract. Dialogical games as introduced by Lorenzen and Lorenz describe a reasoning technique mainly used for intuitionistic and classical predicate logic: two players (proponent and opponent) argue about the validity of a given formula according to predefined rules. If the proponent has a winning-strategy then the formula is proven to be valid. The underlying game rules can be modified to have an impact on proof search strategies and increase the efficiency of such a searching process. In this paper, a multi-agent version of dialogical logic is introduced that corresponds more to multi-conclusion sequent calculi for propositional intuitionistic logic rather than single-conclusion ones which are more related to two-player dialogues. The rules lead us to a normalization of a proof, let us focus on the proponents' relevant decisions, and therefore give explicit directives that increase compactness of the proof-searching process. This allows us to perform parts of the proof in a parallel way. Soundness and completeness of this multi-agent system is shown.

Keywords: dialogues, proof search, intuitionistic logic, game theories, parallel reasoning

1 Multi-Agent Dialogues

In game-theoretic decision procedures, reasoning happens by means of a play, where usually one player tries to *verify* (called *verifier* or *proponent*) a formula, while the other player's aim is to *falsify* it (therefore called *falsifier* or *opponent*). Considering games as calculi is interesting for two major reasons: first, they provide a different view on the reasoning process and another account to certain systems, e.g., Lorenzen introduced *dialogical games* with the aim of giving an alternative justification for intuitionistic logic than Brouwer's original attempt [20, 19]. Second, modifying the game rules of such systems may lead to new logics, e.g., the semantics of *independence-friendly logic* (IF) [23], is based on the so-called *Hintikka games* [15], while there is also an extension with more than two players leading to special multi-player semantics [1].

Hintikka games are used to valuate the truth of a formula in a fixed model in which the truth values are assigned to atomic formulas. By contrast, in dialogical systems, the subject is the model-independent truth of a formula [15]. As we are more interested in the latter kind of validity, we discuss dialogical games in the following. They have been invented and developed by Paul Lorenzen and Kuno Lorenz [20, 18, 21] especially as a reasoning technique for intuitionistic and classical logic, between the late 1950s and the mid 1970s. The main idea is that the *proponent* (P) states an assertion, the *hypothesis* (usually in terms of a logical formula), which is challenged by the *opponent* (O). A dispute starts: every player performs one move each time, either an *attack* against the other player's assertion or a defence of an attacked assertion. The game rules define which moves are allowed in which situation. At the end, the proponent has a *winning strategy*[1] if and only if the hypothesis, he stated at the beginning, is valid.

As P tries to show validity of the hypothesis, he can be considered as an agent searching for a proof which corresponds to his winning-strategy. Galmiche et al. [12] point out that putting certain restrictions on possible moves due to game rules can modify this searching process. For example, one can put restrictions on allowed repetitions as proposed by Rahman and Keiff [24]. Constructing such rules might improve efficiency of the proof searching process. Alama suggests strategy-preferences for P, but constraining P strictly in this way leads to wrong results in certain cases [2].

Having P as a searcher who takes proof-relevant decisions, it makes sense to supply him with further colleagues to help him finding the proof. We introduce a *syndicate* of agents that indicates that a proponent has different possibilities when reacting on moves by O, but we still keep two *parties* (P and O). The resulting *parallelism* allows concurrent reasoning and joint problem solving among agents. So whenever a proponent agent has different choices to react on a move performed by O, he splits into two or more agents, each doing his own job. Thereby, we obtain a better overview of possibilities for the proponents to analyze the strategies. A similar approach is proposed by Fermüller and Ciabattoni [10, 11] to obtain reasoning systems for intermediate logics, where proponents may *fork* arbitrarily and *join* later to share their information. In our approach, all agents have a global view at any time and fork in a more systematic fashion. In this way, we develop the game-theoretic view to normalize reasoning procedures for propositional intuitionistic logic and simplify proof searches.

We define structural game rules for multi-proponent dialogues and introduce a sequent-based system DiaSeqI which implements these rules in a concrete way. The system is then shown to be sound and complete. At the end, we discuss briefly how to extend it further.

2 From Dialogues to Sequents and Back

In a two-player dialogue, both players take turns one after another. When it is a player's turn, he can either *attack* a formula stated by the other player, or *defend* against such an attack. In an early work, Lorenzen proposed that the proponent may perform several moves at the same time [19]. However this idea is given

[1] This means that the proponent is always able to win, no matter how the opponent behaves.

O	P
	$A \vee \neg A$
?	$\neg A$
? A	—

Fig. 1. A Standard Intuitionistic Dialogue (Excluded Middle)

up in following publications. When looking at intuitionistic dialogues as finally developed by Lorenzen and Lorenz, one notices similarities to Gentzen's sequent calculus LJ [13]: a game state can be interpreted as a sequent $\Phi \Rightarrow \Psi$ where Φ contains assertions by O and Ψ assertions by P. In intuitionistic dialogues, P may only defend against the *last open attack*, i.e., the last attack for which he has not already defended. In LJ-sequents, we always have at most one formula in Ψ which enforces intuitionism.

Figure 1 shows such a dialogue where P tries to prove the *law of the excluded middle* $A \vee \neg A$ in an intuitionistic setting (first shown in a similar way in [19]). This endeavour is doomed to fail, because – as widely known – that formula is *not* valid in intuitionistic logic. In the first line, P states the hypothesis $A \vee \neg A$. This statement is attacked by O (indicated by symbol **?**). P can now defend stating either the left or the right disjunct. However, in Lorenzen's original rules, P is not allowed to state atomic formulas which have not yet been stated by O herself. So he *must not* defend with A and therefore defends with $\neg A$. O attacks this by stating the opposite of the negation, namely assertion A. It is not possible to defend against attacks on negations in Lorenzen games and attacking atomic formulas is also not allowed. As this was the last (open) attack performed by O, P is not allowed to repeat the defence of the second row. He loses, because he is not able to perform another move.[2]

Now, when we compare single-conclusion and the more efficient [9] multi-conclusion sequent calculi (originally introduced in [22]) the idea of more agents on the P side comes up, where each of these agents states another formula.

$$\dfrac{\dfrac{A \Rightarrow}{\Rightarrow \neg A}\neg R}{\Rightarrow A \vee \neg A}\vee R \qquad \dfrac{\dfrac{A \Rightarrow}{\Rightarrow A, \neg A}\neg R}{\Rightarrow A \vee \neg A}\vee R \qquad \dfrac{\dfrac{\dfrac{\Rightarrow A \mid A \Rightarrow \bot}{\Rightarrow A \mid \Rightarrow \neg A}\neg R}{\Rightarrow A \vee \neg A \mid \Rightarrow A \vee \neg A}\vee R}{\Rightarrow A \vee \neg A}EC$$

Fig. 2. Excluded Middle in LJ, LJmc and HLI'

Figure 2 shows a proof attempt in the single-conclusion sequent calculus LJ (left side). The reason for the failing is that one has to decide to keep either A or $\neg A$ when the $\vee R$-rule is applied, while the other disjunct vanishes. In the

[2] In dialogues following *classical logic*, P may defend against any attack more often, so it would be possible to defend against the disjunction-attack with A in the last line, because this atom has just been stated by O herself.

multi-conclusion version LJmc[3] (in the middle of the figure), the decision can be postponed until the *critical rule* ($\neg R$ in this case) is applied. The attempt on the right side is discussed in the next section.

3 Setting up a Focusing Multi-Proponent System

Fermüller [10] presents a variant of Lorenz and Lorenzen's dialogues by introducing multiple players on the proponent's side. The proponents are then allowed to *fork*, i.e., they clone themselves and each proponent follows his own strategy. In classical logic they are then allowed to *merge* again, to collect their information. Allowing different levels of merging leads to different *intermediate logics* [10].

Fermüller's system is very flexible. A proponent may fork when it is his turn, and perform an *additional* move (attack or defence). This leads to parallel games. In each of them, a proponent fights against another opponent with other concessions. These parallel games build proof attempts which correspond to *single-conclusion hypersequents* [10]. As a counterpart for intuitionistic logic, the hypersequent system HLI' [10, 4] is used, where – roughly spoken – rule EC (for *external contraction*) clones a sequent.

Rule EC of HLI' corresponds to P's fork: reading the proof attempt of Figure 2 (right side) from bottom to top, assume that O attacks P's statement $A \vee \neg A$ and P splits into two proponents (EC). So, every concurrent game corresponds to a sub-sequent of a hypersequent and for each of these, O can commit herself to other concessions. For classical and intermediate logics, the proponents can *merge* the contexts again to combine concessions which O committed herself to towards different Ps.

We construct game rules that define concretely in which cases such a fork is allowed. As it turns out, it is not necessary to separate O's commitments towards the different P-agents when we compare the game with a multi-conclusion sequent system. Parallelism is still possible through an extra scheduling mechanism: we extend the idea of forking proponents by a technique for proof searches in sequent calculi called *focusing*, that goes back to Andreoli [3] who introduced it for linear logic. Focused calculi enforce a *normalization* of sequent-proofs.

Focused single-conclusion sequent calculi for intuitionistic logic have been proposed in [17, 25]. The focused *multi-conclusion* calculus LJQ* for intuitionistic propositional logic is presented by Dyckhoff and Lengrand [8] as a variant of LJQ* by Herbelin [14]. Roughly said, in such a calculus, we can have different sorts of sequents, e.g., an *ordinary* sequent $\Phi \Rightarrow \Psi$ for antecedent-rules and a *focused* sequent $\Phi \rightarrow \Psi$, in which succedent-rules can be applied on a single focused formula (called *stoup*).

We use such a focus on the proponents' *decisions* (without stoup) which are significant for the success or failure of the proof. Obviously, such a technique is helpful for proof searches. The combination of multi-proponent games and focusing results in a scheduled and synchronized parallel proof system.

[3] In the following, we use the calculus as it is described in [9]. In literature, it is also often called "m-**G3i**", e.g. c.f [26].

$$\frac{\Gamma, A \Rightarrow B}{\Gamma \Rightarrow A \supset B, \Delta} \supset R \qquad \frac{\Gamma, A \Rightarrow}{\Gamma \Rightarrow \neg A, \Delta} \neg R$$

Fig. 3. Critical Rules in LJMC

Based on the set of *structural rules* provided by Rahman and Keiff [24] and Barth and Krabbe (*constructive Λ-dialectics*, *CΛD*) [5], we construct our new set of rules which can deal with multiple proponent games. A single proponent player is called *P-agent*. All P-agents together form a *party* which fights a single *O-agent* (or simply O), who is the only member of the other party.[4]

1. At the beginning of a dialogue, a single P-agent states the *hypothesis*.
2. A *round* is a sequence of moves performed by O followed by the moves of the P-agents. A dialogue consists of a sequence of such rounds where the first round starts after the statement of the hypothesis.
3. Each agent must perform a move if he can. A P-agent may postpone a move if he is forced to react to a *critical attack* (see rule 6).
4. A dialogue is *won* by the P-agents iff O cannot react on any of the Ps' moves of the previous round. O wins iff no P-agent can react to any of O's statement of the same round (either with an attack or a defence).
5. Only O may attack *atomic formulas*. P-agents may defend against these attacks when O has stated the atom herself towards an P-agent who is not *deactivated* in the same round.
6. Attacks on negations and implications are considered to be *critical attacks*.
7. Whenever a P-agent reacts to a *critical attack*, all other proponent players are immediately *deactivated* and excluded for the rest of the dialogue.
8. A P-agent may repeat a critical attack on the same statement only after any P-agent reacted on a critical attack performed by O. No other repetitions are allowed.

The last three rules implement intuitionism and build the relation to multi-conclusion sequent calculi. Some reader might consider it unnatural to have rules like rule number 6, which refers directly to logical operators. It is usually desired that the rules should be independent of the syntactic elements of the formulas. However, this principle is given up here to increase the flexibility of our system. We can add other connectives to the *critical set* as well, or remove them and therefore generate other logical systems. The concept of critical attacks is based on the non-invertible rules $\supset R$ and $\neg R$ of LJMC, where formulas vanish from the succedent by forced weakening (Figure 3). The usage of these rules is the significant part of the focusing process as, for the proponents, it is highly relevant for winning or losing the game.

To make the rules complete, we need a set of *particle rules*[5] which define how to attack and defend assertions according to the corresponding logical oper-

[4] Note that semantics for dialogical logic are usually defined informally as game rules. We follow that tradition. Formal rules are provided for a special sequent calculus in Section 4.

[5] originally “allgemeine Spielregeln” [18], also called “strip rules” in [5].

Assert	$A \wedge B$		$A \vee B$		$A \supset B$	$\neg A$	$\bot$	a
Attack	$?_L$	$?_R$	$?$		A	A	$?$	$?$
Defence	A	B	A	B	B	—	—	$!!$

Fig. 4. Particle Rules Based on [18] and [5]

ators (Figure 4). These rules are the same for both single-proponent and multi-proponent dialogues.

When attacking a conjunction, the attacker may ask for the left or right conjunct. For the disjunction, the defender may choose the disjunct. Attacking an implication means stating the antecedent. It is defended with the consequent. Attacks on negations or $\bot$ cannot be defended. Note that a represents an arbitrary atomic formula. The double exclamation mark (**!!**) represents the *ipse dixisti* ("you said it yourself") remark [5] which allows the Ps to defend against atomic attacks.

As widely known, a negation $\neg A$ can also be interpreted as implication $A \supset \bot$ in intuitionistic logic, so actually the negation rule is redundant. However, the $\neg$-rule can be seen as a shortcut and therefore we keep it as particle rule.

4 The Calculus DiaSeqI

Barth and Krabbe [5] constructed a sequent system that follows (a variant of) the two-player game rules explicitly. We introduce a sequent-style calculus DiaSeqI that implements the multi-proponent dialogues.

Definition 1 (Dialogue Game). *A* dialogue game *is a path in a* DiaSeqI *sequent tree read from bottom to top.*

We refer to the proponent agents via labels of some set $Propo \subseteq \{pi \mid i \in \mathbb{N}\}$. An *augmented sequent* has the form $\Phi \vdash_X \Psi$ where X is an element of $\{\mathsf{O}, \mathsf{PN}, \mathsf{PD}\}$ indicating the current *phase* of the game (or proof), and Φ and Ψ represent *multi-sets* of *signed formulas*:

$$\Phi \subseteq \{o_p : \varphi,\ \overline{o_p} : \varphi,\ \overline{o_p}^L : \varphi,\ \overline{o_p}^R : \varphi,\ \widetilde{o_p} : \varphi \mid p \in Propo, \varphi \in Form\} \quad (1)$$

$$\Psi \subseteq \{p : \varphi,\ \overline{p} : \varphi,\ \overline{p}^L : \varphi,\ \overline{p}^R : \varphi \mid p \in Propo, \varphi \in Form\} \quad (2)$$

Signed formulas are standard propositional formulas (also called *assertions*; we use the set *Form* as an infinite set of all possible propositional formulas) extended by *announcer labels* (o for opponent and $p \in Propo$ for proponents) with an optional *mark* which is either a horizontal bar or a tilde above the label. For the opponent-label o, we further add an *addressee* as index to indicate the current communication partner. This is not needed for the proponents, as their only communication partner is o.

For example, the signed formula $o_{p1} : A \wedge B$ means that the opponent stated formula (or *assertion*) $A \wedge B$ towards proponent agent $p1$. A bar over the label $\overline{o_{p1}}$ means that the corresponding assertion has been *attacked* by $p1$ and not yet

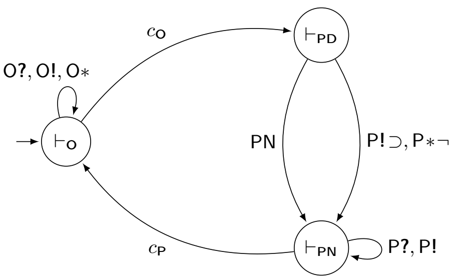

Fig. 5. Proof Cycle in DiaSeqI

been defended. When a conjunction is attacked, we add an L or R referring to the left or right conjunct, because when a player attacks a conjunction, he may ask for the left conjunct ($\overline{o_{p1}}^L : A \wedge B$) or the right one ($\overline{o_{p1}}^R : A \wedge B$). As an extra marker, we use the tilde $\widetilde{o_{p1}}$ to indicate that the statement is *blocked* and can therefore not be attacked or defended. This is used to avoid unnecessary attacks to be carried out by the proponents (see rule 8 from above).

Definition 2 (Round). *A* round *is a section of a dialogue game that starts with an application of* c_{P} *and ends when another* c_{P} *is applied or when the end of the game is reached.*

The three turnstile symbols $\vdash_{\mathsf{O}}$, $\vdash_{\mathsf{PD}}$ and $\vdash_{\mathsf{PN}}$ denote the current *phase*. Every *round* consists of a sequence of these phases (illustrated in Figure 5). In phase $\vdash_{\mathsf{O}}$, it is the opponent's turn who reacts on all of the Ps' moves of the previous round. If O cannot react on all proponents, she loses. In the following $\vdash_{\mathsf{PD}}$ phase (D stands for *decision*), the proponents decide together who of them is going to defend in case that an implication or negation is attacked. This corresponds to the *non-invertible* rules ($\supset R$, $\neg R$) of LJmc. In the last phase $\vdash_{\mathsf{PN}}$ (N for *normal*), the remaining proponent agents react on O's previous moves, each proponent performing at most one move. As soon as no proponent can take a turn, the opponent wins.

A dialogue normally starts with some proponent $p0$ stating the hypothesis φ, so the initial situation is interpreted as sequent $\vdash_{\mathsf{PN}} p0 : \varphi$. Note that the first applied rule would then always be c_{P} which, we omit in the following (start of first round). So our base sequent has the form $\vdash_{\mathsf{O}} p0 : \varphi$.

Figure 6 shows the rules of DiaSeqI. Every rule application, read from bottom to top, represents a possible move. For every logical connective, there are at least four rules, which are in each case two more than in standard sequent calculi. The reason is that we have for both parties rules for attacks and rules for defences. Notice that most attacking rules simply add the bar above the label. Instead of O!⊃, we have a so-called *trigger-rule* O∗⊃ which combines both the defence against an implication-attack and the possibility of a counter-attack, as an extra assertion is added to the proponent-side, i.e., to the right side of the sequent. Because it is not possible to defend against attacks on negations, we also introduce trigger rules for these (O∗¬, P∗¬).

O-rules

$$\frac{\Phi \vdash_{\mathsf{O}} \Psi,\ \overline{p} : A \supset B}{\Phi \vdash_{\mathsf{O}} \Psi,\ p : A \supset B}\ \mathsf{O?}\!\supset \qquad \frac{\widetilde{o_p} : A \supset B,\ \Phi \vdash_{\mathsf{O}} \Psi,\ p : A \qquad o_p : B,\ \Phi \vdash_{\mathsf{O}} \Psi}{\overline{o_p} : A \supset B,\ \Phi \vdash_{\mathsf{O}} \Psi}\ \mathsf{O}{*}\!\supset$$

$$\frac{\Phi \vdash_{\mathsf{O}} \Psi,\ \overline{p} : \neg A}{\Phi \vdash_{\mathsf{O}} \Psi,\ p : \neg A}\ \mathsf{O?}\neg \qquad \frac{\widetilde{o_p} : \neg A,\ \Phi \vdash_{\mathsf{O}} \Psi,\ p : A}{\overline{o_p} : \neg A,\ \Phi \vdash_{\mathsf{O}} \Psi}\ \mathsf{O}{*}\neg$$

$$\frac{\Phi \vdash_{\mathsf{O}} \Psi,\ \overline{p} : A \vee B}{\Phi \vdash_{\mathsf{O}} \Psi,\ p : A \vee B}\ \mathsf{O?}\vee \qquad \frac{o_p : A,\ \Phi \vdash_{\mathsf{O}} \Psi \qquad o_p : B,\ \Phi \vdash_{\mathsf{O}} \Psi}{\overline{o_p} : A \vee B,\ \Phi \vdash_{\mathsf{O}} \Psi}\ \mathsf{O!}\vee$$

$$\frac{\Phi \vdash_{\mathsf{O}} \Psi,\ \overline{p}^L : A \wedge B \qquad \Phi \vdash_{\mathsf{O}} \Psi,\ \overline{p}^R : A \wedge B}{\Phi \vdash_{\mathsf{O}} \Psi,\ p : A \wedge B}\ \mathsf{O?}\wedge$$

$$\frac{o_p : A,\ \Phi \vdash_{\mathsf{O}} \Psi}{\overline{o_p}^L : A \wedge B,\ \Phi \vdash_{\mathsf{O}} \Psi}\ \mathsf{O!}\wedge^L \qquad \frac{o_p : B,\ \Phi \vdash_{\mathsf{O}} \Psi}{\overline{o_p}^R : A \wedge B,\ \Phi \vdash_{\mathsf{O}} \Psi}\ \mathsf{O!}\wedge^R$$

$$\frac{\Phi \vdash_{\mathsf{O}} \Psi,\ \overline{p} : A}{\Phi \vdash_{\mathsf{O}} \Psi,\ p : A}\ \mathsf{O?}A \qquad \frac{\Phi \vdash_{\mathsf{O}} \Psi}{\Phi \vdash_{\mathsf{O}} \Psi,\ p : \bot}\ \mathsf{O?}\bot \qquad \frac{\Phi \vdash_{\mathsf{PD}} \Psi}{\Phi \vdash_{\mathsf{O}} \Psi}\ c_{\mathsf{O}}$$

A is an atom

only applicable if no other rule application is possible

P-rules – decide phase

$$\frac{o_q : A,\ \Phi^\delta \vdash_{\mathsf{PN}} p : B}{\Phi \vdash_{\mathsf{PD}} \Psi,\ \overline{p} : A \supset B}\ \mathsf{P!}\!\supset \qquad \frac{o_p : A,\ \Phi^\delta \vdash_{\mathsf{PN}} \emptyset}{\Phi \vdash_{\mathsf{PD}} \Psi,\ \overline{p} : \neg A}\ \mathsf{P}{*}\neg \qquad \frac{\Phi \vdash_{\mathsf{PN}} \Psi}{\Phi \vdash_{\mathsf{PD}} \Psi}\ \mathsf{PN}$$

$\Phi^\delta =_{df} (\Phi \setminus \{\widetilde{o_p} : f \mid p \in \mathit{Propo}, f \in \mathit{Form}\}) \cup \{o_q : f \mid \widetilde{o_p} : f \in \Phi, p \in \mathit{Propo}, f \in \mathit{Form}\}$,
q is a new agent.

P-rules – normal phase

$$\frac{\overline{o_p} : A \supset B,\ \Phi \vdash_{\mathsf{PN}} \Psi}{o_p : A \supset B,\ \Phi \vdash_{\mathsf{PN}} \Psi}\ \mathsf{P?}\!\supset \qquad \frac{\Phi \vdash_{\mathsf{PN}} \Psi,\ p : A}{\Phi \vdash_{\mathsf{PN}} \Psi,\ \overline{p}^L : A \wedge B}\ \mathsf{P!}\wedge^L$$

$$\frac{\overline{o_p}^L : A \wedge B,\ \overline{o_q}^R : A \wedge B,\ \Phi \vdash_{\mathsf{PN}} \Psi}{o_p : A \wedge B,\ \Phi \vdash_{\mathsf{PN}} \Psi}\ \mathsf{P?}\wedge \qquad \frac{\Phi \vdash_{\mathsf{PN}} \Psi,\ p : B}{\Phi \vdash_{\mathsf{PN}} \Psi,\ \overline{p}^R : A \wedge B}\ \mathsf{P!}\wedge^R$$

$$\frac{\overline{o_p} : A \vee B,\ \Phi \vdash_{\mathsf{PN}} \Psi}{o_p : A \vee B,\ \Phi \vdash_{\mathsf{PN}} \Psi}\ \mathsf{P?}\vee \qquad \frac{\Phi \vdash_{\mathsf{PN}} \Psi,\ p : A,\ q : B}{\Phi \vdash_{\mathsf{PN}} \Psi,\ \overline{p} : A \vee B}\ \mathsf{P!}\vee$$

$$\frac{}{o_r : A,\ \Phi \vdash_{\mathsf{PN}} \Psi,\ \overline{p} : A,\ \Psi}\ \mathsf{P!!} \qquad \frac{}{o_p : \bot,\ \Phi \vdash_{\mathsf{PN}} \Psi,\ r : A}\ \mathsf{P?}\bot$$

$$\frac{\Phi \vdash_{\mathsf{O}} \Psi}{\Phi \vdash_{\mathsf{PN}} \Psi}\ c_{\mathsf{P}}$$

only applicable if no other rule application is possible
and a P-agent performed an attack or defence in the current round

q is a new agent in each case.

Fig. 6. DiaSeqI Rules

$$
\begin{array}{cl}
o_{p1} : A \ \vdash_{\mathsf{PN}} & \\ \hline
\vdash_{\mathsf{PD}} \ \overline{p0} : A,\ \overline{p1} : \neg A & \mathsf{P}{*}\neg \\ \hline
\vdash_{\mathsf{O}} \ \overline{p0} : A,\ \overline{p1} : \neg A & c_{\mathsf{O}} \\ \hline
\vdash_{\mathsf{O}} \ \overline{p0} : A,\ p1 : \neg A & \mathsf{O?}\neg \\ \hline
\vdash_{\mathsf{O}} \ p0 : A,\ p1 : \neg A & \mathsf{O?}A \\ \hline
\vdash_{\mathsf{PN}} \ p0 : A,\ p1 : \neg A & c_{\mathsf{P}} \\ \hline
\vdash_{\mathsf{PN}} \ \overline{p0} : A \vee \neg A & \mathsf{P!}\vee \\ \hline
\vdash_{\mathsf{PD}} \ \overline{p0} : A \vee \neg A & \mathsf{PN} \\ \hline
\vdash_{\mathsf{O}} \ \overline{p0} : A \vee \neg A & c_{\mathsf{O}} \\ \hline
\vdash_{\mathsf{O}} \ p0 : A \vee \neg A & \mathsf{O?}\vee
\end{array}
$$

$$
\begin{array}{cl}
\hline
 & \mathsf{P!!} \\
\widetilde{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{PN}} \ \overline{p3} : A,\ \overline{p4} : \neg A & \\ \hline
\widetilde{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{O}} \ \overline{p3} : A,\ \overline{p4} : \neg A & c_o, \mathsf{PN} \\ \hline
\widetilde{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{O}} \ p3 : A,\ p4 : \neg A & \mathsf{O?}A, \mathsf{O?}\neg \\ \hline
\widetilde{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{PN}} \ p3 : A,\ p4 : \neg A & c_{\mathsf{P}} \\ \hline
\widetilde{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{PN}} \ \overline{p3} : A \vee \neg A & \mathsf{P!}\vee \\ \hline
\widetilde{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{O}} \ \overline{p3} : A \vee \neg A & c_{\mathsf{O}}, \mathsf{PN} \\ \hline
\widetilde{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{O}} \ p3 : A \vee \neg A & \mathsf{O?}\vee \\ \hline
\overline{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{O}} & \mathsf{O}{*}\neg \\ \hline
\overline{o_{p3}} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{PN}} & c_{\mathsf{P}} \\ \hline
o_{p3} : \neg(A \vee \neg A),\ o_{p_2} : A \ \vdash_{\mathsf{PN}} & \mathsf{P?}\neg \\ \hline
\widetilde{o_{p1}} : \neg(A \vee \neg A) \ \vdash_{\mathsf{PD}} \ \overline{p1} : A,\ \overline{p2} : \neg A & \mathsf{P}{*}\neg \\ \hline
\widetilde{o_{p1}} : \neg(A \vee \neg A) \ \vdash_{\mathsf{O}} \ \overline{p1} : A,\ \overline{p2} : \neg A & c_{\mathsf{O}} \\ \hline
\widetilde{o_{p1}} : \neg(A \vee \neg A) \ \vdash_{\mathsf{O}} \ p1 : A,\ p2 : \neg A & \mathsf{O?}A, \mathsf{O?}\neg \\ \hline
\widetilde{o_{p1}} : \neg(A \vee \neg A) \ \vdash_{\mathsf{PN}} \ p1 : A,\ p2 : \neg A & c_{\mathsf{P}} \\ \hline
\widetilde{o_{p1}} : \neg(A \vee \neg A) \ \vdash_{\mathsf{PN}} \ \overline{p1} : A \vee \neg A & \mathsf{P!}\vee \\ \hline
\widetilde{o_{p1}} : \neg(A \vee \neg A) \ \vdash_{\mathsf{O}} \ \overline{p1} : A \vee \neg A & c_{\mathsf{O}}, \mathsf{PN} \\ \hline
\widetilde{o_{p1}} : \neg(A \vee \neg A) \ \vdash_{\mathsf{O}} \ p0 : \bot,\ p1 : A \vee \neg A & \mathsf{O?}\bot, \mathsf{O?}\vee \\ \hline
\overline{o_{p1}} : \neg(A \vee \neg A) \ \vdash_{\mathsf{O}} \ p0 : \bot & \mathsf{O}{*}\neg \\ \hline
o_{p1} : \neg(A \vee \neg A) \ \vdash_{\mathsf{PN}} \ p0 : \bot & c_{\mathsf{P}} \\ \hline
\vdash_{\mathsf{PD}} \ \overline{p0} : \neg(A \vee \neg A) \supset \bot & \mathsf{P!}\supset \\ \hline
\vdash_{\mathsf{O}} \ \overline{p0} : \neg(A \vee \neg A) \supset \bot & c_{\mathsf{O}} \\ \hline
\vdash_{\mathsf{O}} \ p0 : \neg(A \vee \neg A) \supset \bot & \mathsf{O?}\supset
\end{array}
$$

Fig. 7. Two Examples of DiaSeqI-Proofs

A dialogue can then be simply seen as an augmented sequent tree where each branching corresponds to another O-strategy. Every sequent can be interpreted as a state of the dialogue game.

We consider two examples. The first is the law of the excluded middle again, this time in DiaSeqI (Figure 7, left side). The dialogue is read from the bottom to the top. At the beginning, O attacks the thesis stated by $p0$ ($\mathsf{O?}\vee$) and then it is the proponents' turn (c_{O}, decide phase). The proponents (we currently have only one) can now decide whether they defend against critical attacks or skip such defences. As there is no such attack, we proceed directly to normal phase (PN). Agent $p0$ may now decide to defend with the left or right disjunct, so he calls $p1$ as a supporter, defends herself with A and lets $p1$ defend with $\neg A$. There is nothing to attack on the O-side, so it is the opponent's turn again (c_{P}). O has to react to all P-agents. So $p0$'s atom A is attacked ($\mathsf{O?}A$), as well as $p1$'s negation $\neg A$ ($\mathsf{O?}\neg$) and it is the proponents' turn again (c_{O}). One of O's attacks was critical, so they have to decide whether to react on this or not. As it is not possible to defend against the atomic attack in this round (O has not stated A), $p1$ decides to react and triggers this decision ($\mathsf{P}{*}\neg$) which adds A to O's concessions. However, $p0$ is *deactivated* thereby (he vanishes) and is not able to defend anymore. Agent $p1$ is not allowed to attack atoms and therefore cannot react on O's previous attack with a counter-attack. The proponents lose.

When applying $\supset L$ or $\neg L$ in LJmc, the principal formula may not be removed from the sequents. The resulting *duplication problem* can be avoided by replacing the corresponding rules by several others [7]. In DiaSeqI, we reduce the possibilities of repeating such attacks again and again by *blocking* implications and negations on the O-side after an attack, until a critical attack has been answered on the P-side.

The right DiaSeqI-proof of Figure 7 makes clear, how this blocking mechanism works: first, O attacks the implication and $p0$ defends by stating $\bot$, while his colleague $p1$ counter-attacks the negation. O reacts to this attack by triggering and counter-attacking $p1$'s disjunction (O$*\neg$, O?$\vee$). O's negation is blocked and cannot be attacked for now. When $p0$'s $\bot$ is attacked, it disappears. Agent $p1$ introduces $p2$ who states the right disjunct (P!$\vee$). Both $p1$'s and $p2$'s assertions are then attacked by O. Agent $p2$ defends against the critical attack which deactivates $p1$ and triggers O's concession A. Thereby, O's blocked assertion is unlocked and can be attacked again what is done immediately by a new agent $p3$. O triggers $p3$'s assertion $A \vee \neg A$ and counter-attacks it. Agent $p3$ defends with A, his new colleague $p4$ with $\neg A$. Both are attacked by O in the next round, but as O has A in her concessions, $p3$ can defend (P!!) and the proponents win.

5 Equivalence of LJmc and DiaSeqI

In this section, soundness and completeness proofs for DiaSeqI are shown due to a transformation of closed DiaSeqI sequent trees to closed LJmc sequent trees and vice versa. *Closed* means that at all leaves of a sequent tree, a *closing rule* (ax or $\bot ax$ in LJmc; P!! or P?$\bot$ in DiaSeqI) is applied.

Theorem 1 (DiaSeqI Soundness). *Every closed* DiaSeqI *proof tree can be transformed to a closed* LJmc *proof tree.*

Proof. This can be achieved quite easily, as we simply have to remove the announcer-labels and all applications of attacking rules that do nothing more than adding the bars above the labels. We also remove the rules that change the phases. The only exceptions are attacked conjunctions on the P-side which cause a branching in DiaSeqI-trees. They are replaced directly, i.e., $\overline{p}^L : A \wedge B$ becomes A, and $\overline{p}^R : A \wedge B$ becomes B. The defences against these conjunctive attacks are then simply removed, as they do not add any new information to the trees. □

The completeness theorem is more complex. Our aim is to rearrange the rule applications in such a way that we have the round structure of a game. This is not trivial, as rule applications might depend on each other and it is probably necessary to add further applications which are not relevant in the LJmc proof.

We first need some definitions for LJmc sequent trees[6]: in an LJmc-sequent $\Gamma \Rightarrow \Delta$, Γ and Δ are multi-sets of formulas, where Γ is called *antecedent* and Δ

[6] The definitions commonly used can be traced back to [16].

succedent. A rule can be applied on an antecedent-formula (called *left*-rule) or a succedent-formula (*right*-rule). Here, we see the left-rule for disjunctions.

$$\frac{\Gamma, A \Rightarrow \Delta \qquad \Gamma, B \Rightarrow \Delta}{\Gamma, A \vee B \Rightarrow \Delta} \vee L$$

A rule consists of zero, one or two sequents above the line, called *premises*, and one sequent below, called *conclusion*. The formula that is concerned by the rule in the conclusion ($A \vee B$) is called *principal*, the corresponding sub-formulas in the premises (A and B) are called *side formulas*.

Definition 3 (Rule Application). *A rule application is a pair (r, φ) where r is a rule name and φ (the instance) of some* principal formula *in an* LJMC *proof, on which r is applied.*

Definition 4 (Formula Introduction and Reinitialization). *Let t be an* LJMC *sequent tree. Reading the tree from bottom to top, in each branch, a sequent s can be found, where some formula φ occurs for the first time in the branch. This happens* with *a rule application a or φ is already in the root sequent of the tree. We say that φ is* introduced *in sequent s* with the root *or* with application *a. When a critical rule ($\neg R$, $\supset R$) is applied, all implications and negations in the antecedent of the premise are called* reinitialized.[7]

Now, the *movement* of a rule application a below another one a' is only possible if a does not *depend on* a'.

Definition 5 (Rule Application and Dependencies). *A rule application $a_1 = (r_1, \varphi_1)$* depends directly on *another application $a_2 = (r_2, \varphi_2)$, written $a_1 \ltimes a_2$, iff both r_1 and r_2 are applied in the same path of the tree and φ_1 is* introduced *with a_2 or* reinitialized *with a_2.*

A rule application a_1 depends *on an application a_2, written $a_1 \propto a_2$, iff $a_1 \ltimes a_2$, or there is some application a' such that $a_1 \ltimes a'$ and $a' \propto a_2$, i.e., $\propto$ is the transitive closure of $\ltimes$.*

For example, assume we have a section of an LJMC-tree:

$$\frac{\Gamma \Rightarrow A, B, C \wedge D}{\Gamma \Rightarrow A \vee B, C \wedge D} \vee R$$

If there is an application a of $\wedge R$ on $C \wedge D$ above this shown application a', then a is independent of a', because $C \wedge D$ also appears in the shown conclusion, so a' can already be applied there instead of a.

By contrast, if $\vee R$ is applied on a formula $A \vee (C \wedge D)$, then a later application of $\wedge R$ on $C \wedge D$ depends on the former one, which must be done first.

When a non-invertible (i.e., critical) rule is applied, formulas of the succedent vanish. These are therefore critical positions we focus on in the decision phases of dialogues. Parts of the proof tree above and below these applications are

[7] This corresponds to the unblocking mechanism in DiaSeqI.

summarized to *macro blocks*. Within these macro blocks, we collect rule applications which are independent of each other. The movement of rule applications within these *micro blocks* is possible without problems. Micro-blocks correspond to *rounds* of the dialogue that is constructed.

Definition 6 (Macro Block and Micro Block). *A* macro block *is a maximal path of an* LJmc *proof tree that contains only non-critical rule applications.*

A micro block *is a non-empty section of a macro block with only non-critical rule applications, in which every rule application is independent of any other rule application in the same micro block. The first rule application that depends on some other application within a macro block, is the start of a new micro block.*

Definition 7 (Block Height). *The* macro block height *(MBH) of an* LJmc *sequent tree is the maximal number of macro blocks of all paths from the root of the tree to its leaves. The* micro block height *(mbh) of an* LJmc *sequent tree is the maximal number of micro blocks from the root of the tree to its leaves.*

In the following, we assume that an LJmc sequent tree is *closed* if the same formula appears in the antecedent and succedent of some sequent, but *additionally* we assume that this formula is an atom. If this is not the case for some sequent tree t, it can easily be extended to some tree t' by the way of further rule applications, such that the *closing rules* of t' (usually called ax) are only applied on atoms.

Lemma 1 (Rule Application Redundancy). *Let t be a closed* LJmc *proof tree. Then t can be transformed to another tree t' such that for all of its macro blocks M, every rule application appears at most once in M.*

Proof (by induction on the MBH of t). Only the rules $\supset l$ and $\neg l$ are interesting as these cause *duplication*. One has to show that a duplication is not relevant within the same macro block. The application of a critical rule at the end of the macro block makes duplications in succedents vanish, and only after those applications, the implications and negations in the antecedent become interesting again. □

We can now show by induction that a rule application in level n can be moved in any LJmc sequent tree towards the root as long as it is independent of the rule application below, e.g.,

$$\dfrac{\dfrac{\dfrac{— t —}{\Gamma, A \Rightarrow C, D, \Delta}}{\Gamma, A \Rightarrow C \vee D, \Delta}\vee R \qquad \dfrac{— t' —}{\Gamma, B \Rightarrow C \vee D, \Delta}\,?}{\Gamma, A \vee B \Rightarrow C \vee D, \Delta}\vee L \qquad \dfrac{\dfrac{\dfrac{— t —}{\Gamma, A \Rightarrow C, D, \Delta} \qquad \dfrac{— t'' —}{\Gamma, B\, C, D, \Delta}}{\Gamma, A \vee B \Rightarrow C, D, \Delta}\vee L}{\Gamma, A \vee B \Rightarrow C \vee D, \Delta}\vee R$$

So this makes an arbitrary exchange of rule applications within a micro block possible. It is also possible to move applications of left-rules beyond applications of critical rules, as long as the applications are independent of each other. So independent rule applications from one macro block can be moved down to another one (proven by induction on the level of application in sequent tree). The base sequent never changes and tree closure is preserved.

Next, we have to *normalize* the macro and the micro blocks, because every player *must* perform a move whenever this is possible. We have to enrich the

LJMC-trees by further rule applications or by moving rule applications from one block down to another one so that the blocks are finally *saturated*.

Definition 8 (Macro-Normalized Proof Tree). *Let t be an* LJMC *proof tree, then t is* macro-normalized *iff for all non-atomic $\phi \in \Phi$ which appear in a sequent $\Phi \Rightarrow \Psi$ of t (antecedent), if ϕ is introduced or reinitialized in macro block b and a rule is applied on this ϕ then this application happens in the same macro block b.*

Lemma 2 (Macro-Normalization). *Every* LJMC *proof tree t can be macro-normalized to a tree t'. If t is closed then t' is also closed. The MBH of t is the same as that of t'.*

Proof (by induction on MBH h of proof tree t). In the inductive step, considering the macro blocks starting at the root sequent of t, we have to examine the frontiers to the macro blocks above.

$$\dfrac{\begin{array}{c}\vdots\quad\vdots\quad\vdots\\ \dfrac{-\;t_1\;-}{\Gamma_1 \Rightarrow \Delta_1}\,r_{c_1}\\ \vdots\;\vdots\end{array}\qquad\begin{array}{c}\vdots\quad\vdots\quad\vdots\\ \dfrac{-\;t_2\;-}{\Gamma_2 \Rightarrow \Delta_2}\,r_{c_2}\\ \vdots\;\vdots\;\vdots\end{array}\qquad\cdots\qquad\begin{array}{c}\vdots\quad\vdots\quad\vdots\\ \dfrac{-\;t_n\;-}{\Gamma_n \Rightarrow \Delta_n}\,r_{c_n}\\ \vdots\;\vdots\;\vdots\end{array}}{\Gamma_B \Rightarrow \Delta_B}$$

After macro-normalizing all t_is which are not empty (hypothesis) we cut out the section containing $\Gamma_i \Rightarrow \Delta_i$ together with t'_i for each i, and move all left-rule applications, that appear in macro level 1 of this section, below c_i. The resulting tree that starts in $\Gamma_i \Rightarrow \Delta_i$ is again macro-normalized and called t''_i. We obtain an LJMC tree with the following structure:

$$\begin{array}{c}\dfrac{\dfrac{-t''_{i1}-}{\Gamma_{i1} \Rightarrow \Delta_{i1}}\,r_{c_i}\quad\ldots\ldots\quad\dfrac{-t''_{im}-}{\Gamma_{im} \Rightarrow \Delta_{im}}\,r_{c_i}\\ \vdots\qquad\qquad\vdots\qquad\qquad\vdots}{\dfrac{\Gamma_i \Rightarrow \Delta_i}{\vdots}}\end{array}$$

Now for each path i starting in the lowest macro-block, we collect all formulas which are introduced on the left-hand side within this block and store them in a list $\Gamma_i^* \subseteq \Gamma_i$, i.e., Γ_i^* contains all these *fresh* formulas of branch i. For each ϕ_{ik} of Γ_i^*, we check whether a corresponding rule application appears in t''_i, i.e., above $\Gamma_i \Rightarrow \Delta_i$. If it does *not*, we do not need to do anything with ϕ_{ik} and can proceed with the next formula of Γ_i^*. Otherwise, it must be independent of the rules between it and $\Gamma_i \Rightarrow \Delta_i$. As t''_i is also macro-normalized, it must appear below the c_i's. The application on ϕ_{ik} together with all left-hand applications that depend on this and which do not depend on any c_i, are already in the new single macro sequent section starting in the base sequent. □

Definition 9 (Micro Block Saturation). *A* micro block *with $\Phi \Rightarrow \Psi$ as root sequent is* saturated *iff non-critical and non-closing rules are applied on all* *formulas ϕ of Φ and ψ of Ψ for which ϕ is not atomic and ψ is not atomic and not a negation or implication. A* sequent tree *is* micro-saturated *iff all the micro blocks on all of its paths are saturated.*

Lemma 3 (Micro Block Saturation in LJmc Proof Trees). *Let t be a closed* LJmc *proof tree of* micro block height *h. Then t can be transformed so that it is* micro-saturated*. If t is closed then it is still closed after the transformation. Its micro block height h is not increased.*

Proof (by induction on mbh h of t). Use Lemma 2 to *macro-normalize* t. Micro-saturation can be achieved by pulling down independent applications from upper micro-blocks or introducing *irrelevant* rule applications that are not necessary to close the proof tree but which are needed for saturation. □

Lemma 4 (Rule Movement in Saturated Single-Block Sequent Trees). *In a saturated single-micro-block section of an* LJmc*-tree, any non-closing rule application in any level can be moved down within the section any possible number of steps without changing the base sequent, nor the leafs. The micro block height is not increased.*

Proof. As we have already seen, the base sequent never changes because the rule applications are all independent of each other. The rest is done by induction on the height n of the section. □

These lemmas make it possible to prove the completeness theorem.

Theorem 2 (DiaSeqI Completeness). *Every closed* LJmc *proof tree can be transformed to a closed* DiaSeqI *proof tree.*

Proof. We have seen that we can take any LJmc proof tree t and *macro-normalize* it (Lemma 2). The resulting sequent tree t' can then be *micro-saturated* (Lemma 3). So then for each micro block, considering the non-critical and non-atomic formulas, we have a rule application in the block. Now the applications of the rules we want to have at the bottom can be pulled down within these micro blocks (Lemma 4). So we can fix an order for the rule applications in each micro block, e.g., $\wedge r, \vee r, \neg l, \supset l, \wedge l, \vee l$. Each branch of the resulting tree has then the desired form we need: the initial round starts with an attack performed by O. Afterwards, for each micro block, we have a PN-phase followed by an O-phase. This is repeated until the macro-block is finished. Then a critical rule is applied (PD-phase) and the next micro-block starts and so on. The announcer labels can then be simply added. □

As we have seen, we can rearrange the moves within one micro block and therefore within one phase; the result is always the same. Additionally, once a proponent agent is introduced, he is independent of the other P-agents as long as his moves take place in the PN-phase. So, as long as there is no reaction on a critical attack, the proponents may argue with O *in parallel.* There is finally a synchronization taking place when one critically attacked P-agent reacts to that attack. Such a parallelism makes it possible to implement a concurrent and synchronized reasoning procedure for different logics.

6 Conclusion and Further Steps

We have seen a sound and complete game-theoretic proof system which is based on dialogical logic. We introduce more proponent players to parallelize the proof searching process. Further restrictions on the proponents' strategies let us focus on the players' decisions which are significant for winning or losing the game (*decide phase*). This is a normalization of standard multi-conclusion calculi for intuitionistic logic (such as LJmc), similar to sequent systems with focus. The extra *normal phase* of DiaSeqI gives hints of how right-hand rules can be applied in parallel. Besides, the presented game-theoretic approach suggests to implement a concurrent reasoning procedures and provides a highly flexible calculus which can be easily extended due to a modification of the structural rules, to improve efficiency.

Currently, termination of the proof search is not guaranteed, as the repetition rule (rule 8) is still too weak. An elegant way to enforce termination would be a restriction that corresponds to the *a fortiori* rule of sequent system IG by Corsi and Tassi [6]. A suitable structural rule would be "*If a P-agent reacted on a critical attack against some assertion, then, if O attacks the same assertion again, a P-agent may only defend non-critically against this attack*".

An extension would be to consider first-order predicate logic or modal logic instead of propositional logic. Van Dun [27] proposes a system, where a reasoning procedure for modal logic is implemented due to multiple opponents, each opponent representing another Kripke world. Our next step is to combine DiaSeqI with Van Dun's approach to obtain a flexible system which has a considerable potential for proof searches in modal logics. Further extensions for public announcement logics or hybrid logics are possible and fit very well the idea of this dialogical setting.

Acknowledgements

I want to thank Chris Fermüller and Michael Mendler for their comments on an earlier version of the presented work, and especially the CILC'16-reviewers for their helpful comments and remarks.

References

1. Abramsky, S.: A compositional game semantics for multi-agent logics of partial information. In: van Benthem, J., Gabbay, D., Löwe, B. (eds.) Interactive Logic. pp. 11–47. Amsterdam University Press (2007)
2. Alama, J.: Dialogues for proof search. In: Benzmüller, C., Otten, J. (eds.) ARQNL 2014. Automated Reasoning in Quantified Non-Classical Logics. EPiC Series in Computing, vol. 33, pp. 65–70 (2015)
3. Andreoli, J.M.: Logic programming with focusing proofs in linear logic. Journal of Logic and Computation 2(3), 297–347 (1992)
4. Avron, A.: Hypersequents, logical consequence and intermediate logics for concurrency. Annals of Mathematics and Artificial Intelligence 4(3), 225–248 (1991)

5. Barth, E.M., Krabbe, E.C.: From Axiom to Dialogue: A philosophical study of logics and argumentation. Walter de Gruyter (1982)
6. Corsi, G., Tassi, G.: Intuitionistic logic freed of all metarules. The Journal of Symbolic Logic 72, 1204–1218 (12 2007)
7. Dyckhoff, R.: Contraction-free sequent calculi for intuitionistic logic. The Journal of Symbolic Logic 57(3), 795–807 (1992)
8. Dyckhoff, R., Lengrand, S.: LJQ: A Strongly Focused Calculus for Intuitionistic Logic, pp. 173–185. Springer Berlin Heidelberg (2006)
9. Egly, U., Schmitt, S.: On intuitionistic proof transformations, their complexity, and application to constructive program synthesis. Fundamenta Informaticae 39(1-2), 59–83 (1999)
10. Fermüller, C.G.: Parallel dialogue games and hypersequents for intermediate logics. Springer (2003)
11. Fermüller, C.G., Ciabattoni, A.: From Intuitionistic Logic to Gödel-Dummett Logic via Parallel Dialogue Games. In: In Proceedings of the 33rd IEEE International Symposium on Multiple-Valued Logic (2003)
12. Galmiche, D., Larchey-Wendling, D., Vidal-Rosset, J.: Some remarks on relations between proofs and games. Construction-Festschrift for Gerhard Heinzmann (2010)
13. Gentzen, G.: Untersuchungen über das logische Schließen. I. Mathematische Zeitschrift 39, 176–210 (1935)
14. Herbelin, H.: Séquents qu'on calcule: de l'interprétation du calcul des séquents comme calcul de lambda-termes et comme calcul de stratégies gagnantes. Ph.D. thesis, Université Paris-Diderot-Paris VII (1995)
15. Hintikka, J.: Logic, language-games and information: Kantian themes in the philosophy of logic. Oxford Univ Press (1973)
16. Kleene, S.C.: Introduction to Metamathematics. North Holland (1952)
17. Liang, C., Miller, D.: Focusing and polarization in intuitionistic logic. In: Computer Science Logic. pp. 451–465. Springer (2007)
18. Lorenz, K.: Arithmetik und Logik als Spiele. Ph.D. thesis, Christian-Albrechts-Universität zu Kiel (1961), excerpts reprinted in [21]
19. Lorenzen, P.: Ein dialogisches Konstruktivitätskriterium. In: Infinistic Methods. Proceeding of the Symposium on Foundations of Mathematics. pp. 193–200 (1961), reprinted in [21]
20. Lorenzen, P.: Logik und Agon. In: Atti del XII Congresso Internazionale di Filosofia. vol. 4, pp. 187–194 (1958), reprinted in [21]
21. Lorenzen, P., Lorenz, K.: Dialogische Logik. Wissenschaftliche Buchgesellschaft (1978)
22. Maehara, S.: Eine Darstellung der intuitionistischen Logik in der klassischen. Nagoya mathematical journal 7, 45–64 (1954)
23. Mann, A.L., Sandu, G., Sevenster, M.: Independence-Friendly Logic – A Game-Theoretic Approach. London Mathematical Society lecture note series, Cambridge Univ. Press (2011)
24. Rahman, S., Keiff, L.: On how to be a dialogician. In: Vanderveken, D. (ed.) Logic, Thought and Action, Logic, Epistemology, and the Unity of Science, vol. 2, pp. 359–408. Springer Netherlands (2005)
25. Simmons, R.J.: Structural focalization. ACM Transactions on Computational Logic (TOCL) 15(3), 21 (2014)
26. Troelstra, A.S., Schwichtenberg, H.: Basic Proof Theory. Cambridge Univ. Press, second edn. (2000)
27. Van Dun, F.: On the modes of opposition in the formal dialogues of P. Lorenzen. Logique et analyse 15, 103–136 (1972)