# A games semantics for reductive logic and proof-search

David Pym[1], Eike Ritter[2]

[1] University of Bath and HP Labs, Bristol
[2] University of Birmingham

**Abstract.** Theorem proving, or algorithmic proof-search, is an essential enabling technology throughout the computational sciences. We explain the mathematical basis of proof-search as the combination of reductive logic together with a control régime. Then we present a games semantics for reductive logic and show how it may be used to model two important examples of control, namely backtracking and uniform proof.

## 1 Introduction to reductive logic and proof-search

Theorem proving, or algorithmic proof-search, is an essential enabling technology throughout the computational sciences. We explain the mathematical basis of proof-search as the combination of reductive logic together with a control régime. Then we present a games semantics for reductive logic and show how it may be used to model two important examples of control, namely backtracking and uniform proof. This paper presents, from our current perspective, a brief introduction to work first published in 2004 in Chapter 6 of [25]:

> D. Pym and E. Ritter. *Reductive Logic and Proof-search: Proof Theory, Semantics, and Control*. Volume 45, Oxford Logic Guides. Oxford University Press, 2004.

We also summarize the necessary mathematical background developed in the earlier chapters of [25].

Axiomatizations of logics as formal systems are usually formulated as calculi for deductive inferences. Deductive inference proceeds from established, or supposed, premisses, or hypotheses, to premisses, or hypotheses, to a conclusion, regulated by the application of *inference rules*, $R$,

$$\Downarrow \quad \frac{\mathbf{Premiss}_1 \ldots \mathbf{Premiss}_m}{\mathbf{Conclusion}} \quad R.$$

A proof is constructed, inductively, by applying instances of rules of this form to proofs of established premisses, thereby constructing a proof of the given conclusion.

A conceptually valuable semantics of proofs is provided by a correspondence between the propositions and proofs of a logic, the types and terms of a $\lambda$-calculus [13] and the objects and arrows of a category [16], *q.v.* Figure 1, in which (*e.g.*, natural deduction) proofs correspond to (*e.g.*, typed $\lambda$-terms) which correspond to classes of arrows in categories with specified structure.

The leading examples of this form of semantics arise in intuitionistic logic [31], in which natural deduction proofs correspond to simply-typed $\lambda$-terms and to arrows in cartesian closed categories [16], in intuitionistic linear logic, in which natural deduction proofs correspond to linear $\lambda$-terms and to the arrows of symmetric monoidal

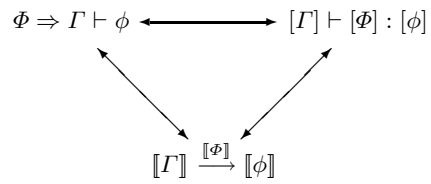



**Fig. 1.** Propositions-as-types-as-objects

closed categories [2], and bunched logic [21, 26, 27], in which natural deduction proofs correspond to $\alpha\lambda$-terms and to arrows of doubly closed categories. Another example, which underlies much of what we do in this paper, is provided by classical natural deduction and the $\lambda\mu\nu$-calculus [23, 28, 24].

Theorem proving, or algorithmic proof-search, is both an essential enabling technology within the computational sciences and of independent philosophical interest. More specifically, in computing, many problems are formulated as judgements about formal texts, typical representable in logical formalisms. For example, well-formedness (parsing), well-typedness (type-checking), as well as logical consequence (*e.g.*, for specification and correctness) itself.

There are, indeed, many useful formal languages and, for each language, typically many useful procedures for judging properties of sentences. As the complexity of the languages and their properties increases, the possibility of obtaining efficient, total procedures recedes, but partial procedures which fail quickly are of great value and interactive theorem provers, such as the Boyer-Moore system [3], the LCF system [11] and its derivatives, such as Paulson's Isabelle system [20], as well as more complex systems, such as Coq [4], based on dependent type theory [18] are used in a wide range of system-critical applications (*e.g.*, [30]).

Although with widely varying complexity and efficiency characteristics, these systems have, however, a common underlying logical basis: *reductive inference*.

Reductive inference proceeds from a putative (*i.e.*, supposed) conclusion to sufficient premisses, regulated by *reduction operators*, $O_R$,

$$\Uparrow \quad \frac{\mathbf{Sufficient\,Premiss}_1 \ldots \mathbf{Sufficient\,Premiss}_m}{\mathbf{Putative\,Conclusion}} O_R,$$

corresponding to (admissible) inference rules, $R$, read from conclusion to premisses.[3] Here the idea is the following:

– The putative conclusion is an assertion, or a *goal*, such as a sequent [10] $\Gamma \vdash \Delta$, the *endsequent*, in our chosen logic. We should like to know whether or not the sequent is provable in our chosen logic. Often, we write $\Gamma$ **?-** $\Delta$, borrowing a notation from Prolog, to indicate $\Gamma \vdash \Delta$ as a putative conclusion;

[3] Henceforth we refer to just $R$ rather than $O_R$.

– Here we are assuming that our given logic comes along with a proof system.[4] Each inference rule in the system, including any admissible rules, gives rise to a reduction operator. To apply a reduction operator to particular assertion we must find an instance of a reduction operator such that instance of the putative conclusion matches the assertion;
– The assertions which must be proved in order to have a proof of the initial assertion, or *subgoals*, are then given by the corresponding instances of the sufficient premisses of the operator.

We believe that this idea of reduction was first explained *in these terms* by Kleene [15].

So, in reductive logic, an attempt to construct a proof, *i.e.*, a *reduction*, proceeds, inductively, by applying instances of reduction operators of this form to putative conclusions of which a proof is desired, thereby yielding a collection of sufficient premisses, proofs of which would be sufficient to imply the existence of a proof, obtainable by deduction, of the putative conclusion.

Note, however, that a reduction may fail to yield a proof: having removed all of the logical structure, *i.e.*, the connectives, by reduction, we may be left with $p$ **?-** $q$, for distinct atoms $p$ and $q$. Such possible failures have no counterpart in deductive logic, but theorem provers have to have strategies which ensure that a proof of a given endsequent is found if there exists one. In general, not only the choice of reduction operators but also the order in which they are applied matters: the application of the same reduction operators in one order may yield a proof, where after applying the same reduction operators in a different order a completion to a proof may be impossible. Defining the order in which reduction operators are applied is only one example of how to control proof-search. These issues are usually very intensional in nature, and so are outside the scope of pure reductive logic. So modelling proof-search requires both modelling a reductive logic and modelling the chosen control régime.

The inherent partiality of reductions presents a clear semantic difficulty: we must be able to interpret those reductions which cannot be completed to be proofs. In particular, we aim to recover a semantics for proofs of utility comparable to that of the propositions-as-types-as-objects triangle for proofs.

The desired set-up is summarized in Figure 2, in which $\Gamma$ **?-** $\phi$ denotes a sequent

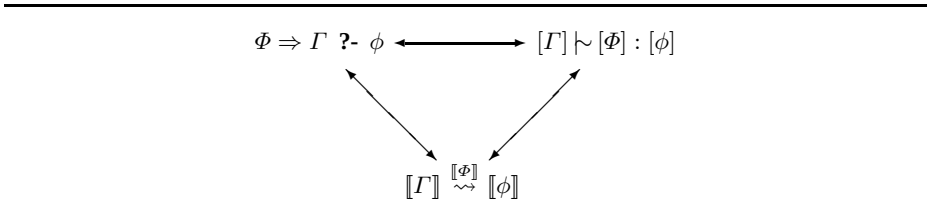



**Fig. 2.** Reductions-as-realizers-as-arrows

[4] We could, however, formulate our subsequent analysis purely semantically.

which is a putative conclusion and $\Phi \Rightarrow \Gamma$ **?-** $\phi$ denotes that $\Phi$ is a search with root $\Gamma$ **?-** $\phi$. The judgement $[\Gamma] \mid\!\sim [\Phi] : [\phi]$ indicates that $[\Phi]$ is a *realizer* of $[\phi]$ with respect to assumptions $[\Gamma]$.

The provision of a model-theoretically adequate such framework is non-trivial. The main difficulty is that the objects constructed during a reduction are, in contrast to the objects, *i.e.*, proofs, constructed during deduction, inherently partial. Whilst any deduction proceeds from axioms to a guaranteed conclusion and so constructs a proof, reductions proceed from a putative conclusion to sufficient premisses. At any intermediate stage, it may be that it is impossible to complete the reduction so as to obtain a proof.

Such a semantic framework is presented in full in [25]. In this paper, we present part of this framework, namely a games semantics for reductive logic and proof-search. We also describe how to use this semantics to model two important concepts of proof-search, which are backtracking and uniform proofs.

In § 2, we introduce the key control régime in proof-search: backtracking. In § 3, we briefly sketch the general proof- and model-theoretic frameworks within which our work resides, though they are suppressed in this paper. In § 4, we introduce our notion of game in the setting of intuitionistic reductive propositional logic and then, in § 5, we generalize it to classical reductive propositional logic. Then, in § 6, we give a games semantics of proof-search in intuitionistic propostional logic via its embedding in classical propositional logic. In § 7, we give a game-theoretic semantics for uniform proofs.

## 2 Backtracking

In this paper, we concentrate on one particular instance of control, namely backtracking in intuitionistic propositional logic. We choose backtracking because it is perhaps the prototypical control régime and because it raises most of the modelling issues of interest. Uniform proofs can be modelled using the same framework, as we show at the end of this paper.

We have shown in previous work [28, 29] that backtracking in intuitionistic propositional logic can be done by embedding intuitionistic propositional logic into classical propositional logic, doing search in classical logic and, finally, deciding whether a classical derivation has an intuitionistic subderivation.

The embedding of intuitionistic propositional logic into classical logic is based on Dummett's multi-conclusioned version of intuitionistic propositional logic [5]. Seen as a system of reductive logic, this embedding works by adding the side formula $\Delta$ also to the premiss of the $\supset R$-rule, thereby obtaining the classical $\supset R$-rule and keeping this side formula $\Delta$ in all reduction operators applied to the modified premiss.

As an example, consider the following reduction, in which the use first of $\supset L$ on $p \supset q$ leaves the subsequent development of the left-hand branch of the reduction

doomed to failure, even though the endsequent is provable:[5]

$$\dfrac{\dfrac{\mathbf{fails}^{(1)}}{\vdots}\quad\dfrac{\dfrac{\mathbf{succeeds}}{\vdots}\quad\dfrac{\mathbf{succeeds}}{\vdots}}{\dfrac{r \supset s\,,\, q\,,\, r \textbf{ ?- } s \qquad r \supset s\,,\, q\,,\, r\,,\, t \textbf{ ?- } t}{r \supset s\,,\, s \supset t\,,\, q\,,\, r \textbf{ ?- } t}\supset L_{s\supset t}}}{\dfrac{p \supset q\,,\, r \supset s \textbf{ ?- } p \qquad r \supset s\,,\, s \supset t\,,\, q \textbf{ ?- } r \supset t}{p \supset q\,,\, r \supset s\,,\, s \supset t \textbf{ ?- } r \supset t}\,(1)\ \supset L_{p\supset q}}\qquad \dfrac{}{}\supset R$$

After the first $\supset L$, we can see that the left-hand branch will fail, and we must backtrack to (1) and make a different choice of reduction. We might try $\supset L_{s\supset t}$ instead. Such a control step lies outside the logical structure we have so far established but we can give a logical account of it by considering the intuitionistic calculus LJ to be embedded in the classical sequent calculus, LK [10].[6]

In general, every intuitionistic sequent derivation arises as a subderivation of a classical sequent derivation via (for example) Dummett's presentation of intuitionistic logic as a multiple-conclusioned sequent calculus [5]. Because the classical $\supset\ R$ rule allows multiple succedents in the premiss, two different intuitionistic sequent derivations, which are not identical up to a permutation of inference rules, can be subderivations of the same classical derivation up to a choice of axioms. For example, consider the following two intuitionistic reductions:

$$\dfrac{\dfrac{}{\psi\,,\, \phi \textbf{ ?- } \psi}\,Ax}{\psi \textbf{ ?- } \phi \supset \psi\,,\, \chi \supset \psi}\supset R \quad \text{and} \quad \dfrac{\dfrac{}{\psi\,,\, \chi \textbf{ ?- } \psi}\,Ax}{\psi \textbf{ ?- } \chi \supset \psi\,,\, \phi \supset \psi}\supset R.$$

They arise as restrictions to intuitionistic logic of the following classical reduction:

$$\dfrac{\dfrac{\dfrac{}{\psi\,,\, \phi\,,\, \chi \textbf{ ?- } \psi\,,\, \psi}\,Ax}{\psi\,,\, \chi \textbf{ ?- } \phi \supset \psi\,,\, \psi}}{\psi \textbf{ ?- } \phi \supset \psi\,,\, \chi \supset \psi}\supset R.$$

Similarly, in LK viewed as reductive system, the $\supset L$ rule has the form

$$\dfrac{\Gamma \textbf{ ?- } \phi, \Delta \quad \Gamma, \psi \textbf{ ?- } \Delta}{\Gamma, \phi \supset \psi \textbf{ ?- } \Delta}\,,$$

in which the $\Delta$ is retained in both premisses. Using this operator instead of its intuitionistic counterpart, we are able to restart the computation at (2), and proceed to apply the

[5] We adopt the notation $R_\phi$ to denote the instance of the operator $O$ generated by the formula $\phi$, *e.g.*, $\supset L_{p\supset q}$.

[6] Our assumed calculi LJ and LK are, in fact, very minor variants of Gentzen's systems. For example, in the premiss of $\wedge L$, in both LJ and LK, we include both of the components of the conjunction introduced in the conclusion; similarly, for $\vee R$ in LK.

necessary $\supset R$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\textbf{succeeds}}{\vdots}}{r \supset s\,,\, s \supset t\,,\, r \textbf{ ?- } t\,,\, p}\supset L_{s\supset t}}{r \supset s\,,\, s \supset t \textbf{ ?- } \boxed{r \supset t}\,,\, p}\supset R}{r \supset s\,,\, s \supset t \textbf{ ?- } p\,,\, \boxed{r \supset t}}(2)\, Exchange \qquad \dfrac{\text{as above}}{\dfrac{\vdots}{r \supset s\,,\, s \supset t\,,\, q \textbf{ ?- } r \supset t}\supset R}}{p \supset q\,,\, r \supset s\,,\, s \supset t \textbf{ ?- } r \supset t}(1)\ \supset L_{p\supset q}$$

We call such a point, at which the exchange rule is used to restart the computation, a *backtracking point*.

In this way we model the backtracking in intuitionistic propositional logic by an exchange on the right-hand side in classical logic. In the following sections, we present a games model which captures this approach.

## 3 The general semantic framework

In the sections following this one, we shall give direct definitions of classes of games for intuitionistic and classical propositional logics. We are then able to give a games semantics for proof-search (*i.e.*, for backtracking and uniform proof) by considering intuitionistic logic to be embedded in classical logic in the way that we have sketched for LJ and LK, with the additional computational structure available in LK being used to represent the control régime.

Before doing so, we mention briefly the general proof- and model-theoretic frameworks within which our analysis formally resides.

We sonsider the proof-objects of propositional LJ to be represented by simply-typed $\lambda$-terms in the usual way. We consider the proof-objects of propositional LK to be represented by $\lambda\mu\nu$-terms [28, 29, 24, 25], the extension, by the present authors, of Parigot's classical $\lambda$-calculus to include an analysis of classical disjunction, that is a representation that analyses directly the rule

$$\frac{\Gamma \vdash \Delta, \phi, \psi, \Delta'}{\Gamma \vdash \Delta, \phi \vee \psi, \Delta'},$$

in which both $\phi$ and $\psi$ are present in the premiss.

Beginning with intuitionistic logic, with proofs represented as $\lambda$-terms, we interpret a proof $t$ of a sequent $\Gamma \vdash \phi$ as a map

$$[\![\Gamma]\!] \xrightarrow{[\![t]\!]} [\![\phi]\!]$$

in a bi-cartesian closed category in the usual way [16].

Turning to classical logic, we consider proof-terms as given by the $\lambda\mu\nu$-calculus, so that we consider sequents of the form $\Gamma \vdash t : \phi, \Delta$. Semantically, such a sequent is interpreted, essentially, in an indexed or fibred category as follows:

– The base interprets $\Delta$. Arrows in the base interpret the structural properties, such as weakening, of the contexts $\Delta$;
– The fibre over (the interpretation of) $\Delta$ is cartesian closed, and is used to interpret $\Gamma \vdash t : \phi$, essentially following the pattern of the intuitionistic semantics;
– The structural binding operator $\mu$ and the binding operator $\nu$ that handles disjunction are both interpreted using certain natural isomorphisms between homsets in fibres determined by weakening maps in the base.

The details of this semantics may be found in [24, 25] (building on [22]). Concrete examples of this semantics are given by continuations [12, 24] and a class of games, to which we turn in the next section.

Turning to reductive logic, we must, as we have suggested, provide an interpretation for reductions, such as those having leaves of the form $p$ **?-** $q$, that cannot be completed to form proofs. Our semantics handles this situation by introducing *indeterminates*, via a polynomial construction over the semantics of the underlying logic, formulated as a fibration over a category 'worlds' corresponding to collections of indeterminates. Then $p$ **?-** $q$ is interpreted by an indeterminate map

$$[\![p]\!] \xrightarrow{\alpha} [\![q]\!].$$

We then establish that a reduction $\Phi$ of $\Gamma$ **?-** $\phi$ can be completed to a proof just in case its semantics determines a suitable instantiation of its indeterminates. The details may be found in [25].

In our presentation of games models for reductive logic, beginning in the next section, we suppress this treatment of indeterminates, concentrating on the structure of the games themselves and their use in interpreting backtracking and uniform proof.

We conclude this brief summary by remarking that semantics we have described admits soundness and completeness theorems in a familiar style [24, 25].

## 4 Games for intuitionistic propositional logic

We describe games for reductive intuitionistic propositional logic. We extend these games in the next section to games for reductive classical logic. As we shall see later, these games are intensional enough that we can model control aspects, such as backtracking and uniform proof, quite simply.

We consider games played between two players, Proponent, $P$, and Opponent, $O$. In such games, for a formula $\phi$, the aim of Opponent is to falsify the given formula $\phi$, and the aim of Proponent is to prove it. A game starts by Opponent challenging the given formula. Proponent wins a game when he can answer Opponent's initial challenge, otherwise he loses. The possible moves of both players in a game for $\phi$ are determined by the structure of $\phi$. A proof of a formula corresponds to a *winning strategy* for Proponent. Such a winning strategy for a formula $\phi$ is a function which for every legal O-move in a game for $\phi$ produces a legal P-move such that if P uses this strategy to determine his moves he wins every game for $\phi$. Such games for proofs have been described for a variety of logics, including classical and intuitionistic logic [17, 6]. Usually, in games for classical logic, Proponent and Opponent are dual to each other, whereas this is not true for games for intuitionistic logic.

These games models for proofs have been adapted to give models of sequential computations in programming languages [14, 1, 9]. There, the intuition is that Opponent asks for the value of a computation, and Proponent performs the computation to produce values as answers. In such games, there is usually a strict alternation between moves by Proponent and Opponent, corresponding to the absence of concurrent computation. As computations have a clear direction (from inputs to outputs) there is usually no duality between Proponent and Opponent in these games.

The key conceptual difference between the games for proofs and the games for computations is that in logic not all propositions are provable, so that in these games not all propositions have strategies, whereas in the programming languages considered, however, all types are inhabited, so that these games have strategies for every type.

The details of how to present games models differ widely, both within games for proofs and within games for computations. The definition of the games considered in this paper uses elements of both approaches. We use one important technical notion from the games introduced by Hyland and Ong, namely the notion of an *arena*: for each formula $\phi$ the possible moves for a game for $\phi$ are listed in a forest[7] called an arena, and the rules of the game use this forest extensively. Ong [22] introduces also the notion of a *scratchpad* to model the multiple conclusions of classical logic. Scrachpads are additional games that Proponent may start at will.

Herein we give a class of games which combines ideas from those for intuitionistic provability and those for programming languages to give a class which models intuitionistic proofs *directly*.[8]

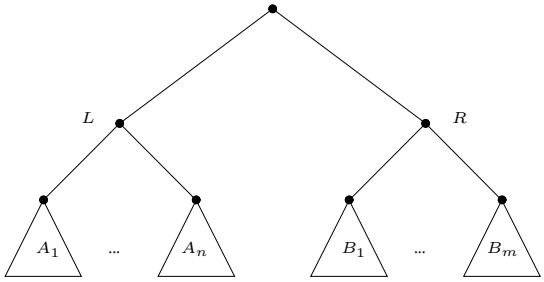
We begin the definition of our games semantics by defining arenas. For each formula $\phi$, we define an arena, that is a forest used to characterize legal moves by both players in our games.

**Definition 1.** *An* arena *of type $\phi$ is a forest with nodes having possibly labels defined inductively by the following:*

- *The arena of $\top$ is the empty forest;*
- *The arena of $\bot$ is the forest with one node labelled $\bot$;*
- *The arena for a propositional atom $p$ is a forest with one node labelled $p$;*
- *The arena for $\phi \wedge \psi$ is the disjoint sum of the arenas for $\phi$ and $\psi$;*
- *Suppose $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are the trees of the arena for $\phi$ and $\mathcal{B}_1, \ldots, \mathcal{B}_m$ are the trees of the arena for $\psi$. Then the arena for $\phi \vee \psi$ is given by*

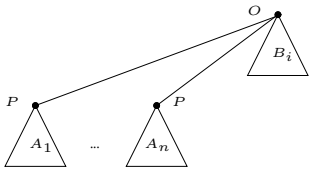
[7] A *forest* is a set of trees.

[8] Games models of intuitionistic proof can be recovered from games models of linear proofs [1] via the exponential ! and, for example, Girard's translation of intuitionistic logic into linear logic.




*Note that there are two special nodes called $L$ and $R$. In the special case that the arena for $\phi$ or the arena for $\psi$ is empty, the arena for $\phi \vee \psi$ is the empty arena too. The root node of the arena for $\phi \vee \psi$ is labelled $\vee$;*

– *Suppose $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are the trees of the arena for $\phi$ and $\mathcal{B}_1, \ldots, \mathcal{B}_m$ are the trees of the arena for $\psi$. Then the arena for $\phi \supset \psi$ is the disjoint union of the following trees*




*In the special case that the arena for $\phi$ is empty, the arena for $\phi \supset \psi$ is the arena for $\psi$. All nodes in the arena for $\phi \supset \psi$ which are root nodes in the arena of $\psi$ are labelled $\supset$ in addition to any other label they might have.*

*We call all root nodes in an arena $O$-nodes, and all children of $O$-nodes $P$-nodes, and all children of $P$-nodes $O$-nodes.*

Arenas are used to define possible plays. The definition of moves and plays makes this precise.

Next, we define possible moves in our games. Each move for a game for $\phi$ is associated with a node in the arena for $\phi$. There are several types of moves. Firstly, we have moves by Proponent and Opponent, and secondly there are question and answer moves. Questions which correspond to $O$-($P$-)nodes are played by Opponent (Proponent), and answers which correspond to $O$-($P$-)nodes are played by Proponent (Opponent). The definition is as follows:

**Definition 2.** *A* move $m$ *for an arena $\mathcal{A}$ is a node which is classified as either question or answer. Questions which correspond to $O$-($P$-)nodes are moves by Opponent (Proponent), and answers which correspond to $O$-($P$-)nodes are moves by Proponent (Opponent). We call a move by Proponent a* P-move *and a move by Opponent an* O-move.

Next, we define plays, which are instances of the game. Each play consists of a sequence of moves satisfying certain conditions. The intuition is that Opponent starts the play by challenging Proponent to verify the given formula. Proponent responds by

asking the Opponent to justify the assumptions which Proponent can make in a sequent calculus proof of $\phi$. Conjunctive choices are made by Opponent, and disjunctive choices by Proponent. Proponent wins a particular game if he can answer Opponent's initial question.

The moves in a play for $\phi$ follow the structure of arena of $\phi$ closely: A $O$ ($P$)-question can be played only if there was already a $P$-($O$)-question corresponding to the parent node. An answer can only be given if a question with the same associated node has already been made.

The precise conditions for a play are as follows:

**Definition 3.** *A* play *for an arena $\mathcal{A}$ is a sequence of moves $m_1, \ldots, m_n$ such that:*

*(i) There exists an index $I \geq 1$ such that all moves $m_1, \ldots, m_I$ are O-questions with position $1, \ldots, I$, respectively, and the corresponding nodes are roots in the forest for $\mathcal{A}$. These moves are called* initial questions*;*

*(ii) For each question $m_i$, with $i > I$, there exists a question $m_k$, with $k < i$, such that the node corresponding to $m_k$ is the immediate predecessor of the node corresponding to $m_i$ in the arena $\mathcal{A}$. We call $m_k$ the* justifying question *for $m_i$;*

*(iii) For each answer $m_i$, with $i > I$, there exists a question $m_k$, with $k < i$, such that $m_k$ and $m_i$ are the same node in $\mathcal{A}$. If $m_j$ is the justifying question for $m_k$, we call $m_j$ the* justifying question *for $m_i$;*

*(iv) Each question can be answered at most once;*

*(v) Any initial questions can only be answered if all non-initial questions have already been answered;*

*(vi) For any P-answer $m_i$ there exists a move $m_j$ such that $m_j$ is an O-answer with the same label or $\bot$ and $j < i$ and that the nodes corresponding to $m_i$ and $m_j$ in the arena are on a path which does not contain a $P$-node $n$ labelled $\supset$ such that the nodes corresponding to $m_i$ and $m_j$ are its children or identical to it;*

*(vii) If $m$ is an $O$-question labelled $\vee$, then at most one $P$-question is justified by $m$.*

Condition (vi) of this definition merits an explanation. During plays we have to ensure that Proponent can answer questions of Opponent only if this answer corresponds to an assumption which Opponent has provided. This matters in the case of Proponent asking a question labelled $\supset$, which corresponds to using an assumption of type $\phi \supset \psi$. The rules of the game work in such a way that in this case two proofs are constructed: one of the original formula using $\psi$ as an additional assumption, and the second one of $\phi$. Now we need to ensure that $\psi$ is not available as an assumption during the proof of $\phi$. Condition (vi) ensures this by making sure that any $O$-answer for $\phi$ cannot be used by Proponent.

Conditions (vii) and (vi) ensure that these games capture intuitionistic proofs: condition (vii) enforces the disjunction property of intuitionistic logic, and condition (vi) makes sure that only one specific formula can be proved at any one given time.

Compared to a games semantics for natural deduction (*e.g.*, [14]), we allow both Opponent and Proponent more freedom: both players can make several moves at a time, which are subject to fewer restrictions. In this way, we capture the possibility of applying reduction operators to several sequents independently. We also capture the possiblity of sequences of blocks of left- and right-rules in a play.

As usual, left-operators involve operations on the premisses: they are initiated by P-questions. Similarly, right-operators involve operations on the conclusions: they are initiated by O-questions. The restriction in Clause (vi) that Proponent can answer questions only if Opponent has answered a P-question with the same label before ensures that the axiom rule can be invoked only if there is the same formula on both sides of the sequent.

Contraction is built in implicitly by allowing both players to ask the same question several times. Moreover, Clause (ii) of the definition of a play allows parallel reductions in different branches of the search tree: A P-question with position $p \cdot n_1 \cdots n_k \cdots m$ with $k > 0$ and $p$ the position of the justifying O-question represents the application of $\supset L$ in all branches which arise by playing moves with position $p \cdot n_1 \cdot n_i$, for $i < k$.

Note also that our games semantics is capable of representing detailed information how searches are executed. The level of detail is sufficient not only to model which reduction operators are applied but also in which order. Some reduction operators are even modelled by several moves, with the possibility of interleaving the moves corresponding to different reduction operators. Hence a mapping from strategies to searches assigns the same search to several strategies.

As an example, we give a possible play for the arena for the formula $((p \supset q) \wedge (r \supset s) \wedge (s \supset t) \wedge r) \supset t$ (see Figure 3 for the arena). The play starts by Opponent

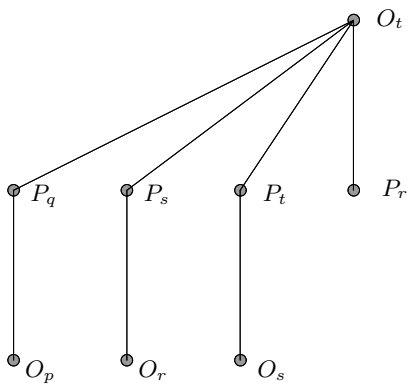



**Fig. 3.** Arena for $((p \supset q) \wedge (r \supset s) \wedge (s \supset t) \wedge r) \supset t$

asking the initial question. Here, this means that Opponent is asking for a proof of the formula $t$. Now Proponent has various choices, namely asking for evidence for one of the assumptions. Let us assume that Proponent asks the question corresponding to the node labelled $P_t$. Now Opponent will ask the question labelled $s$, thereby asking Proponent to prove $r$. Proponent now needs to use the assumption $r \supset s$ and asks the question labelled $s$. Next, Opponent asks the question labelled $r$ and challenges Proponent to prove the formula $r$ in turn, which is the hypothesis in the implication $r \supset s$. Proponent now asks for the final assumption $r$. Opponent now has no choice but to answer this question, thereby making it possible for Proponent to answer outstanding

questions by Opponent. Now Proponent can use this answer and answer Opponent's question $s$. Again, Opponent is now forced to answer the question $t$. This process of answering previously asked questions goes on until finally Opponent is forced to answer the question labelled $t$, and Proponent can answer the initial question.

The key notion of games semantics is that of a *strategy*. A strategy describes how Proponent responds to arbitrary Opponent moves. When related to sequent calculus reductions, a strategy indicates how Proponent answers challenges from Opponent to prove the given formula.

**Definition 4.** *A* strategy *is a function from plays $m_1, \ldots, m_k$, where $m_k$ is an O-move, to a sequence of moves $m_{k+1}, \ldots, m_n$ such that $m_1, \ldots, m_k, m_{k+1}, \ldots, m_n$ is a play, and the sequence $m_{k+1}, \ldots, m_n$ is non-empty if the sequence $m_1, \ldots, m_k$ contains no unanswered P-move which could be answered by Opponent in the next move according to Definition 3.*

Note that this definition makes it possible to force Opponent to answer any unanswered questions by Proponent if such a move was allowed by choosing the empty sequence as a result of the function for sequences with unanswered questions by Proponent.

In the example, a strategy for Proponent would be to answer the initial question by asking the question labelled $t$ and then play as indicated above in response to any Opponent move.

Next we show that each strategy for the arena corresponding to a formula $\phi$ gives rise to an intuitionistic sequent calculus proof of $\phi$. Note that several strategies give rise to the same proof: games make significantly finer distinctions than sequent calculus proofs.

**Theorem 1.** *For any formula $\phi$ and strategy $\Phi$ for $\phi$ there exists an intuitionistic sequent calculus proof of $\phi$.*

## 5 Games for classical logic

We extend the games considered in the previous section to games for classical logic. The main difference between the games for intuitionistic logic and those for classical logic is a consequence of the fact that for classical logic we are working with sequents with multiple conclusions, $\Gamma \vdash \Delta$, with the intuitive meaning that (at least) one of the formulæ in $\Delta$ must to be proved, whereas in intuitionistic logic we work with only one conclusion. This means that, in classical games, when Opponent challenges a formula $\phi$ in $\Delta$, Proponent might choose to defend a different formula $\psi$ in $\Delta$, which has to be accepted also as a valid defence of $\phi$.

The definitions of arena, move, and justification for classical games are the same as those for intuitionistic games. We call a strategy (play) classical if it is the one for classical games. Otherwise we call the strategy (play) intuitionistic.

The conditions for classical plays are not as strong as the conditions for intuitionistic plays. In particular, the rules for disjunction are changed to allow Proponent to select both disjuncts, thereby possibly violating the disjunction property of intuitionistic logic. More precisely, we relax Clause (vi) and Clause (vii). We drop the latter clause, and replace the former as follows:

**Definition 5.** *A* play *for an arena $\mathcal{A}$ is a sequence of moves $m_1, \ldots, m_n$ such that conditions (i) – (v) for intuitionistic plays, and the following additional conditions are satisfied:*

*(vi) For any P-answer $m_i$, there exists a $O$-question $m_k$ and an $O$-answer $m_j$ such that $m_i$ is hereditarily justified by $m_k$, $m_j$ is an O-answer with the same label as $m_k$ or $\perp$, and $k < j < i$ and that the nodes corresponding to $m_k$ and $m_j$ in the arena are on a path which does not contain a $P$-node $n$ labelled $\supset$ such that the nodes corresponding to $m_i$ and $m_j$ are its children or identical to it.*

This relaxation captures the possibility of pending O-questions (arising from the multiple conclusions on the right-hand side) being answered as well as the immediate justifying question.

This games semantics is sound for classical logic:

**Theorem 2.** *For any formula $\phi$ and classical strategy $\Phi$ for $\phi$ there exists a classical sequent calculus proof of $\phi$.*

Two rules are responsible for the fact that we model LK-reduction and not only LJ-reductions. The first rule is the ability of Proponent to play arbitrary moves labelled $L$ and $R$. For modelling LJ-reductions, one would allow Proponent to play only one switching move which is justified by a given O-question. The second rule is the second part of Clause (vi) of the definition of plays. This rule models the possibility of having multiple formulæ on the right-hand side and therefore being able to apply an axiom rule using any formula on the right-hand side. If we omit these two rules, we obtain a representation of LJ-reductions.

The games semantics Ong presents in [22] for the $\lambda\mu$-calculus (without disjunction), which provides proof terms for classical logic with conjunction, negation and implication, uses *scratchpads* to model classical logic. Scratchpads are separate plays to be started by Proponent whenever he chooses. As we consider disjunction as well, we have extended the definition of an arena and introduced the concept of switching moves (the moves labelled $L$ and $R$). Proponent choosing a move labelled $R$ is captured by changing to a scratchpad in Ong's model.

## 6 A games semantics for proof-search

We can now explain, building on the constructions and observations of the previous sectios, how backtracking can be modelled in our games semantics. Backtracking points are captured by the possibility of Proponent making disjunctive choices which are not available when the moves are restricted to intuitionistic games. This is the case when Proponent plays both switching moves and when Proponent plays a P-question $m$ corresponding to a node arising from a $\supset$ $L$-operator. In the first case, playing the other switching move is not allowed in games for LJ, and in the second case no previously pending O-question can be used to justify the P-answer to the O-question which is the immediate successor to the P-question $m$.

Backtracking actually occurs when Proponent plays a different switching move, or actually answers a question with a different label using Clause (vi) of the definition of a play.

One can show that these game-theoretic notions correspond precisely to the proof-theoretic notions introduced in § 2.

**Theorem 3.** *Consider a classical strategy $\Phi$ for the arena $\phi$ which corresponds to a reduction $\Psi$ of $\phi$ in the reductive classical logic LK.*

*(i) The strategy $\Phi$ contains a backtracking point iff the reduction $\Psi$ contains a backtracking point;*
*(ii) The strategy $\Phi$ models backtracking in the game-theoretic sense iff there exist reductions $\Psi_1$ and $\Psi_2$ in the reductive intuitionistic logic LJ such that both $\Psi_1$ and $\Psi_2$ are embedded in $\Psi$ via the embedding of LJ into LK, and $\Psi_2$ arises via backtracking from $\Psi_1$.*

To illustrate this point, consider an example of the previous section, namely the reduction for the sequent $((p \supset q) \wedge (r \supset s) \wedge (s \supset t) \wedge r) \supset t$. The arena is given in Figure 3. The play in the previous section corresponds to the first reduction described in § 2, whereas the following play corresponds to the second reduction:

$$O_t^Q P_q^Q O_r^Q P_t^Q O_s^Q P_s^Q O_r^Q P_r^Q O_r^A P_r^A O_s^A P_s^A O_t^A P_r^A O_q^A P_t^A \ ,$$

where moves by Opponent (Proponent) are denoted by the letter $O$ ($P$) with subscripts and superscripts, and the subscript indicates the label of the move and the superscript indicates whether the move is a question or an answer.

Note first the contraction involved in this play: the move $P_t^Q$ models both instances of the $\supset L$-operator reducing $s \supset t$. The backtracking points are the P-questions labelled $q$, $s$ and $t$, and backtracking is reached with the move $P_r^A$: this move is possible only in games for multiple-conclusioned LK, and models the exchange which is necessary to make the reduction succeed.

## 7 Uniform Proof

In this section, we show that our games semantics also provides a characterization of uniform proofs, which give rise to a simple algorithm for proof-search with relatively little non-determinism.

A uniform proof [19] in (single-conclusioned) LJ is a proof in which, when constructed as a reduction, right-reductions are preferred over left-reductions, so that a left-reduction is applied only if the formula on the right-hand side is atomic. Uniform proof is complete for hereditary Harrop formulæ [19]. In our games semantics, right-reductions correspond to challenges by Opponent and left-reductions to challenges by Proponent, so uniform proofs correspond to strategies in which Opponent always plays as many reductions as possible. The precise definition is as follows:

**Definition 6.** *A strategy for $\phi$ in a game for intuitionistic or classical logic is called a* uniform strategy *if the following conditions hold:*

*(i) Opponent always makes as many moves as possible;*
*(ii) Proponent makes any move labelled $L$ or $R$ if possible.*

If we consider games for intuitionistic logic, then a uniform strategy corresponds to a uniform proof in (single-conclusioned) LJ. If we consider games for classical logic, then a uniform strategy corresponds to a uniform proof in classical LK (in which left-reductions are applied only if *all* formulæ on the right-hand side are atomic).

*Weakly* uniform proofs can be characterized in the same way. Recall that a weakly uniform proof is a uniform proof where, in addition to the conditions for uniform proof, $\vee L$ rules are applied as close to the root as possible. This can be captured in the games semantics by defining a strategy to be a *weakly uniform strategy* if

- it is uniform, and
- moves by Proponent corresponding to the root node in the arena for the interpretation of any formula $\phi \vee \psi$ (on the left) are played in preference to any other moves, and
- moves by Opponent labelled $L$ and $R$ are played in preference to any other move.

It turns out that the embedding of a uniform single-conclusioned LJ-proof $\Phi$ in LK is not necessarily uniform, but there exists a uniform multiple-conclusioned uniform LK-proof $\Phi'$ which contains the LJ-proof as a subproof.

This has an analogue in games: Any strategy for intuitionistic games is also a strategy for classical games. As Opponent has more possibilities of challenging Proponent, a strategy which is uniform for intuitionistic games is not uniform for classical games. However, any uniform strategy for intuitionistic games gives rise in a canonical way to a uniform strategy for classical games: Proponent ignores the additional questions by Opponent and considers only the questions Opponent asked in the original strategy. Proponent is also able to use the answers he gave in the intuitionistic strategy to answer the additional questions by Opponent.

## 8 Directions

The following are obvious extensions to this work:

- Extension to case of the corresponding first- and higer-order predicate logics;
- The case of substructural logics;
- Application to the semantics of logical frameworks;
- More detailed analyses of control régimes (*e.g.*, the order in which reduction operators are applied, or the order of selection of clauses in resolution).

Our current work concerns the construction of an example of a *classical category* based on games and inspired by ideas from proof-search presented herein. Classical categories were introduced by Führmann and Pym [8, 7] as symmetric models of the the sequent calculus LK [10] for propositional classical logic which do not collapse to a Boolean algebra and which model cut-reductions via an order-enrichment.

**Acknowledgements** We are grateful to Didier Galmiche, Daniel Méry, and Edmund Robinson for their careful comments on various aspects of this work. Lincoln Wallen contributed to the background to this work via a joint EPSRC grant with Pym. Pym is partially supported by a Royal Society Industry Fellowship.

## References


1. S. Abramsky, R. Jagadeesan, and P. Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409–470, 2000.
2. P.N. Benton, G.M. Bierman, V.C.V. de Paiva, and J.M.E. Hyland. Term assignment for intuitionistic linear logic (preliminary report). Technical report, University of Cambridge, Computer Laboratory, August 1992. Report 262.
3. R.S. Boyer and J.S. Moore. *A computational logic*. ACM Monograph Series. Academic Press, 1979.
4. Th. Coquand. *Une théorie des constructions, Thèse de Troisième Cycle*. PhD thesis, Université de Paris VII, 1995.
5. M. Dummett. *Elements of Intuitionism*. Oxford University Press, 1977.
6. W. Felscher. Dialogues as a foundation for intuitionistic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic III, Alternatives to Classical Logic*, pages 341–372. Dordrecht, 1986.
7. C. Führmann and D. Pym. On the Geometry of Interaction for Classical Logic. In *Proc. LICS 04*, pages 211–220. IEEE Computer Society Press, 2004.
8. C. Führmann and D. Pym. Order-enriched categorical models of the classical sequent calculus. To appear, *Journal of Pure and Applied Algebra*. Preprint at `http://www.cs.bath.ac.uk/~pym/oecm.pdf`.
9. G. McCusker. Games and Full Abstraction for FPC. In *Proc. of the 11th Annual Symposium on Logic in Computer Science (LICS)*, pages 174–183. IEEE Computer Society Press, 1996.
10. G. Gentzen. Untersuchungen über das logische Schliessen. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1934.
11. M. Gordon, R. Milner, and C. Wadsworth. *Edinburgh* LCF*, A Mechanized Logic of Computation*, volume 78 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1979.
12. M. Hofmann and T. Streicher. Continuation models are universal for $\lambda\mu$-calculus. In *Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science*, pages 387–397. IEEE Computer Society Press, 1997.
13. W. Howard. The formulæ-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays in Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, 1980.
14. Hyland, J.M.E and C.-H.L. Ong. On Full Abstraction for PCF: I, II and III. *Information and Computation*, 163:285–408, 2000.
15. S.C. Kleene. *Mathematical Logic*. Wiley and Sons, 1968.
16. J. Lambek and P. Scott. *Introduction to Higher-Order Categorical Logic*. Cambridge University Press, 1986.
17. P. Lorenzen. Ein dialogisches Konstruktivitätskriterium. In *Infinitistic Methods*, pages 193–200. Pergamon Press, 1961.
18. P. Martin-Löf. Constructive logic and computer programming. In L. J. Cohen et al., editors, *Logic, Methodology and Philosophy of Science VI*, pages 153–175. North-Holland, Amsterdam, 1982.
19. D. Miller, G. Nadathur, F. Pfenning, and A. Ščedrov. Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic*, 51:125–157, 1991.
20. T. Nipkow, L.C. Paulson, and M. Wenzel. *Isabelle/HOL — A proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science*. Springer Verlag, 2002.
21. P.W. O'Hearn and D.J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, June 1999.
22. C.-H. L. Ong. A semantic view of classical proofs: type-theoretic, categorical, and denotational characterizations (preliminary extended abstract). In *Proceedings, 11th Annual IEEE*

*Symposium on Logic in Computer Science*, pages 230–241, New Jersey, USA, 1996. IEEE Computer Society Press, Los Alamitos, California.

23. M. Parigot. $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In *Proceedings of the International Conference on Logic Programming and Automated Reasoning. St. Petersburg*, 1992.
24. D. Pym and E. Ritter. On the semantics of classical disjunction. *Journal of Pure and Applied Algebra*, 159:315–338, 2001.
25. D. Pym and E. Ritter. *Reductive Logic and Proof-search: Proof Theory, Semantics, and Control*. Volume 45, Oxford Logic Guides. Oxford University Press, 2004. Errata and Remarks maintained at `http://www.cs.bath.ac.uk/~pym/reductive-logic-errata.html` and `http://www.cs.bham.ac.uk/~exr/reductive-logic-errata.html`.
26. D.J. Pym. *The Semantics and Proof Theory of the Logic of the Logic of Bunched Implications*, volume 26 of *Applied Logic Series*. Kluwer Academic Publishers, 2002. Errata and Remarks maintained at: `http://www.cs.bath.ac.uk/~pym/BI-monograph-errata.pdf`.
27. D.J. Pym, P.W. O'Hearn, and H. Yang. Possible worlds and resources: The semantics of **BI**. *Theoretical Computer Science*, 315(1):257–305, 2004. Preprint available at `http://www.bath.ac.uk/~pym/recent.html`. Erratum: p. 285, l. -12: ", for some $P', Q \equiv P; P'$ " should be "$P \vdash Q$".
28. E. Ritter, D.J. Pym, and L.A. Wallen. On the intuitionistic force of classical search. *Theoretical Computer Science*, 232:299–333, 2000.
29. E. Ritter, D.J. Pym, and L.A. Wallen. Proof-terms for classical and intuitionistic resolution. *J. Logic Computat.*, 10(2):173–207, 2000.
30. Bruce Schneier. *Secrets and Lies: Digital security in a networked world*. Wiley, 2000.
31. D. van Dalen. Intuitionistic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, vol. III: Alternatives to Classical Logic*, number 166 in Synthese Library, pages 225–339. D. Reidel, Dordrecht, Holland, 1986.