A sequent calculus for Limit Computable Mathematics

Stefano Berardi[1],
Yoriyuki Yamagata[2]

[1] C. S. Dept., Univ. of Turin, Turin,
[2] National Institute of Advanced Industrial Science and Technology, Osaka, Japan,

Abstract. We prove a kind of Curry-Howard isomorphism, with some set of recursive winning strategies taking the place of typed λ-terms, and some set of classical proofs taking the place of intuitionistic proofs.
We introduce an implication-free fragment PA_1 of ω-arithmetic, *having Exchange rule for sequents dropped.* Exchange rule for formulas is, instead, an admissible rule in PA_1. Our main result is that cut-free proofs of PA_1 are tree-isomorphic with recursive winning strategies of a set of games we call "1-backtracking games".
"1-backtracking games" are introduced in [1] as a complete semantics for the implication-free fragment of Limit Computable Mathematics, or LCM ([3], [4]). LCM, in turn, is a subset of Classical Mathematics introduced for software verification whose proofs can be "animated" to detect a formalization error. The goal of this paper is to show that PA_1 is a sound and complete formal system for the implication-free fragment of LCM. A simple syntactical description of this fragment was still missing. No sound and complete formal system is known for LCM itself.

1 Introduction

We first sketch what Limit Computable Mathematics is, then we address the problem of finding a formal system for it. Limit Computable Mathematics (from now on, LCM for short) was introduced in [3] as a way to "animate" (to interpret as a program) a classical proof that a formal specification can be satisfied. The motivation was to run a proof on simple examples, as a toy program, in order to find formalization bugs (like false equational axioms), and to check which kind of input/output behavior the specification is describing.[1]

LCM does not interpret all classical proofs as programs. Yet, it interprets many proofs of common use in applied mathematics. LCM interprets proofs as algorithms "able to learn". These algorithms can do mistakes, they can change their output value if they find out it is wrong, but they can change it only finitely many times. A standard example is the algorithmic interpretation of the proof of Excluded Middle for a semi-decidable statement $\exists y.P(n, y)$ (with P decidable

[1] The general problem addressed by Hayashi is to check how close is a formal specification to the informal idea we want to express through it.

predicate). LCM interprets it as an algorithm learning whether $\exists y.P(n,y)$ is true or false. The algorithm first says that $\exists y.P(n,y)$ is false. Then we are allowed to use the hypothesis $\forall y.\neg P(n,y)$ as many times as we want in the following. If we eventually deduce a false conclusion $\neg P(n,m)$, then we know that $P(n,m)$ (and therefore $\exists y.P(n,y)$) is actually true. In this case the algorithm changes his mind just once, from "$\exists y.P(n,y)$ false" to "$\exists y.P(n,y)$ true". Every computation which used the assumption $\forall y.\neg P(n,y)$ is erased and restarted. More in general, every proof using only Excluded Middle over semi-decidable formulas can be interpreted as a learning algorithm. We conjecture that a kind of converse holds: the only proofs which can be interpreted as "learning algorithms" are ω-proofs using only Excluded Middle over semi-decidable formulas. Many proofs of applied mathematics fall in this class, and can be "animated" using these ideas. In §5.2, we produce an example of a proof (actually, an axiom of classical logic) which, instead, cannot be "animated" using these ideas.

The first semantics proposed for the informal idea of "learning algorithm" was Limit Realization ([3]). However, a limitation of this semantics is the lack of readability for algorithms extracted from proofs. A second semantics proposed was a game semantics called "1-backtracking". ([1], [4]). 1-backtracking game semantics is a restriction of the backtracking semantics proposed by T. Coquand in [2] for proofs in ω-arithmetic. 1-backtracking was proved equivalent to Limit Realization Semantics in [1]. For the viewpoint of proof animation, "1-backtracking" has several nice features: it is very intuitive, and quite close to the idea of "learning algorithm" it comes from. It also explain in details how to "run" a proof as a learning algorithm. What was still missing was a simple syntactical description of LCM, that is, of the set of proofs we can "run" using these techniques.

In this paper we fill this (important) gap, and we show that recursive winning strategy for 1-backtracking are isomorphic with proof-trees of a suitable fragment PA_1 of ω-arithmetic. We conjecture that PA_1 derives exactly all intuitionistic consequences of Excluded Middle for semi-decidable formulas. The fragment PA_1 has a simple description: it is classical ω-arithmetic without Exchange rule for sequents (exchange rule for formulas is, instead, an admissible rule). Therefore PA_1 fulfill its goal, giving a simple syntactical description of proofs of LCM.

The isomorphism result we prove is alike to Curry-Howard isomorphism between λ-terms and proofs in natural deduction. It says that recursive winning strategies restate proofs of PA_1 in term of "learning algorithm", without altering their tree structure. The same result implies that theorems of PA_1 are exactly the implication-free valid formulas of LCM. This is the first formal system proposed for some fragment of LCM. Until now, all descriptions of LCM were semantical in nature. Even though PA_1 is ω-logic, understanding PA_1 is easier than game semantics since game semantics require the subtle notion of 1-backtracking. Furthermore, having such system makes encoding to LCM from other formal systems easier.

From a technical viewpoint, our result is obtain by adapting an isomorphism result by H. Herbelin [5] , between all cut-free proofs of classical ω-arithmetic

from one side, and (full, Coquand's style) backtracking game semantics on the other side. The isomorphism between 1-backtracking and cut-free proofs of PA_1 answers to a question proposed by T. Coquand to the authors.[2]

This is the plan of the paper. In §2 we introduce 1-backtracking games, and in §3 Tarski games. In 4 PA_1, we introduce ω-arithmetic without Exchange rule on sequents, and we state the existence of an isomorphism between cut-free proofs and winning strategies with 1-backtracking. In §5 we introduce some examples of proofs we read as winning strategies with 1-backtracking, and, conversely, of winning strategies with 1-backtracking we read as proofs. We also show a classical proof which cannot be interpreted with 1-backtracking.

The proof of the isomorphism result is postponed to the appendix. In §6 we introduce coding and terminology for trees. Then in §7 we derive some properties of proofs, and in §8 introduce a class of intermediate objects between proofs and winning strategies we call proof-strategies. Eventually, in §9, 10 we show that proofs, proof-strategies, winning strategies are isomorphic (as trees).

2 Games and Strategies

In this section we introduce finite games between two players, $\mathcal{E}$ (Eloise) and $\mathcal{A}$ (Abelard). following [6]. We assume the reader is familiar with tree terminology (see §6). Informally, the set of positions of a game is a tree. The starting position of a play is the root of the tree. There is a "turn" map deciding who moves next from a given position. The player moving next selects a child of the current position of the play, and this becomes the new position of the play. If the play eventually stops, the first player who should move and does not (either because he does not want to, or because he cannot) loses. In some games, the play can continue forever. In this case, some extra rules of the game decide the winner. The formal definition runs as follows.

Definition 1. 1. A game G between two players, $\mathcal{E}$ (Eloise) and $\mathcal{A}$ (Abelard), is any tuple $\langle T, t, W_{\mathcal{E}}, W_{\mathcal{A}}\rangle$. T is a tree (a set of lists, see §6) over some set $E, *_E$. t is a map from T to $\{\mathcal{E}, \mathcal{A}\}$. (W_E, W_A) is a partition of all infinite branches of T.
2. We call the elements of E moves, the elements of T positions, and the branches of T plays. We call t the turn map. We call a position $x \in T$ an $\mathcal{A}$-*position*, an $\mathcal{E}$-*position* according if $t(x) = \mathcal{A}, \mathcal{E}$.
3. G is a *finite* game if all plays in G are finite (that is, if $W_E = W_A = \emptyset$). We identify a finite game G with the list $\langle T, t\rangle$.

We introduce here 1-backtracking game associated to any finite game G. Informally, for a while a 1-backtracking play runs like a play in G. The first

[2] (T. Coquand, letter of 25/01/2005) *Question: if one gives the presentation of cut-free provability for ω-logic (with countable and finite $\wedge$ and $\vee$ as only connectives) one gets general backtracking. Is there a modification of the rules of cut-free provability such that the proofs correspond to learning strategies [i.e., to 1-backtracking]??*

player plays x_0, the next player a child x_1 of x_0, and so forth. Player $\mathcal{E}$ (Eloise), however, can make a new kind of move, called *backtracking*. If $\mathcal{E}$ moves from x_i, she can, instead of choosing a child of x_i, come back to some previous position x_j (with $j < i$) from which she moved x_{j+1}. She can decide, this time, to move x_{i+1} from x_j. In this way, in a 1-backtracking game player $\mathcal{E}$ has the possibility of learning from her mistakes. If she thinks that the move x_{j+1} she did from x_j is a mistake, she can backtrack to the position x_j before such move, and change her move to x_{i+1}. There is a constraint: $\mathcal{E}$ can only backtrack finitely many times to the same move, otherwise she loses. A strategy using 1-backtracking can be considered a learning algorithm in the sense of Hayashi. For more information we refer to [4], [1]. Here is the formal definition. Remark that the moves in a 1-backtracking game associated to a game G are defined as positions of G, which are defined as list of moves of G. Therefore, formally, positions of a 1-backtracking game associated to G are list of lists of moves of G.

Definition 2. A cut-free 1-backtracking play of a finite game $G = \langle T, t\rangle$, or a 1-bck. play for short, is some (finite non-empty or infinite) sequence of $\beta = \langle x_0, x_1, \ldots, x_i \ldots\rangle$ of positions of the play G. We require that, for each x_{i+1} in β,

1. if x_i is an $\mathcal{A}$-position, then x_{i+1} is a child in T of x_i.
2. if x_i is an $\mathcal{E}$-position, then x_{i+1} is a child in T of some $\mathcal{E}$-position x_j, for some $0 \leq j \leq i$, *which is an ancestor of* x_i. We call x_j the position $\mathcal{E}$ backtracks to from x_i, and x_{i+1} the new move of x_i from x_j.

We define now the turn map and a winning condition on infinite plays. If we start from a finite play G, an infinite play can only arise if $\mathcal{E}$ backtracks infinitely many times to the same positions. If she does it she loses, therefore $\mathcal{E}$ loses all infinite plays.

Definition 3. The cut-free 1-backtracking game $bck_{cf}(G)$ of a finite game $G = \langle T, t\rangle$ is a Game $\langle T^{cf}, t^{cf}, W^{cf}_{\mathcal{A}}, W^{cf}_{\mathcal{E}}\rangle$ where,

1. T_{cf} is the set of all finite 1-bck. plays(see Def. 2).
2. $t^{cf}(\langle x_0, \ldots, x_n\rangle) = t(x_n)$
3. $W^{cf}_{\mathcal{A}} = \{\text{all infinite branches in } T^b\}$
4. $W^{cf}_{\mathcal{E}} = \emptyset$

We introduce now winning strategies of $bck_{cf}(G)$. Informally, a winning strategy is some way of selecting one move of $\mathcal{E}$ from any $\mathcal{E}$-position. Besides, if $\mathcal{E}$ always plays, from any $\mathcal{E}$-position, the move selected from the winning strategy, then she always wins, no matter what the moves of $\mathcal{A}$ are. We identify a winning strategy with some set of finite plays, namely, the set of all plays in which $\mathcal{E}$ follows the strategy.

Definition 4. Let $G = \langle T, t\rangle$ be any finite game. A winning strategy $\mathcal{W}$ of $bck_{cf}(G)$ is any well-founded subtree of T_{cf} such that,

1. If $p \in \mathcal{W}$ and p is an $\mathcal{A}$-position, all one-step extension q of p in T are in $\mathcal{W}$,
2. If $p \in \mathcal{W}$ and p is an $\mathcal{E}$-position, there exists one and only one $m \in E$ such that $p::m \in \mathcal{W}$ (i.e., p has a unique one-step extension in $\mathcal{W}$).

3 Formulas and Tarski games

In this section we introduce the set of formulas for ω-arithmetic, and Tarski games for them. For an intuitive motivation of Tarski games we refer to [6], §3. We only consider the standard model of arithmetic to interpret our language.

Definition 5.

1. We call L the language consisting of one symbol of predicate for each recursive predicate, and of one symbol of function for each recursive function.
2. We call L_{pos} the set of closed formulas of L in the connectives $\vee, \wedge, \exists, \forall$. We call any $A \in L_{\text{pos}}$ just a *formula*, for short. We call any $\Gamma \in \mathtt{List}(L_{\text{pos}})$ an ordered sequent, or just a *sequent* for short.
3. We call conjunctive formulas, or $\mathcal{A}$-formulas, all formulas $A_1 \wedge A_2, \forall x A \in L_{\text{pos}}$, and all $a \in L_{\text{pos}}$ true atomic (in the standard model).
4. We call disjunctive formulas, or $\mathcal{E}$-formulas, all formulas $A \vee B, \exists x A \in L_{\text{pos}}$, and all $a \in L_{\text{pos}}$ false atomic (in the standard model).
5. We call the first symbol of $A \in L_{\text{pos}}$ the outermost connective of A, if A is not atomic. If A is atomic true, atomic false the first symbol of A is true, false.

Conjunctive formulas are all formulas with first symbol $\wedge, \forall$, true. Disjunctive formulas are all formulas with first symbol $\vee, \exists$, false. We also call conjunctive and disjunctive formulas $\mathcal{A}$-formulas and $\mathcal{E}$-formulas, because, in Tarski games, formulas are positions, and player $\mathcal{A}$ moves from conjunctive formulas, and player $\mathcal{E}$ moves from disjunctive formulas. We now define a tree whose nodes are (corresponding to) all subformulas of $A \in L_{\text{pos}}$.

Definition 6. The subformula tree

1. For $i = 1, 2$, we say that A_i is the one-step subformula of $A_1 \vee A_2$, $A_1 \wedge A_2$ of *index* i. For $n \in N$, we say that $A(n)$ is the one-step subformula of $\forall x.A(x), \exists x.A(x)$, of *index* $n \in N$. We say that B is a subformula of A if there is a chain of one-step subformulas from A to B.
2. Fix some $* \in N$ (say, $* = 0$). Let $A \in L_{\text{pos}}$. The subformula tree T_A of A is defined as the set of all lists $x = \langle *, i_1, \ldots, i_k \rangle \in \mathtt{List}(N)$, such that there is a sequent $A_0, \ldots, A_k$, with $A = A_0$, and with A_{h+1} subformula of index i_{h+1} of A_h, for all $0 \leq h < k$. We call x *a* position of the subformula A_h of A. We define a map $\mathtt{form} : T_A \to L_{\text{pos}}$ by $\mathtt{form}(x) = A$.

The only position of A in T_A is the root $\langle * \rangle$ of T_A. In general, however, a subformula of A can have many different positions in T_A. T_A is a well-founded tree: all branches of T_A are finite because they correspond to shorter and shorter subformulas of A.

We can now define a Tarski game for $A \in L_{\text{pos}}$. The tree of position is the subformula tree of A. Intuitively, $\mathcal{E}$ defends the truth of A and $\mathcal{A}$ the falsity of A. In order to defend the truth of a disjunctive formula, $\mathcal{E}$ chooses an immediate subformula A_i of A she believes to be true. $\mathcal{E}$ defends the truth of A by defending the truth of A_i. In order to defend the truth of a conjunctive formula, $\mathcal{A}$ chooses an immediate subformula A_i of A he believes to be false. $\mathcal{A}$ defends the falsity of A by defending the falsity of A_i. Eventually, the play ends in some atomic subformula a of A. If a is true then $\mathcal{E}$ wins, if a is false then $\mathcal{A}$ wins.[3] Here is the formal definition.

Definition 7. The Tarski Game G_A of a formula $A \in L_{\text{pos}}$ is the finite game $\langle T_A, t_A \rangle$ defined as follows.

1. T_A is the (well-founded) subformula tree of A.
2. For all $p \in T_A$,we set $t_A(p) = \mathcal{A}$ if p is an $\mathcal{A}$-formula (i.e., the position of a conjunctive formula), and $t_A(p) = \mathcal{E}$ if p is an $\mathcal{E}$-formula (i.e., the position of a disjunctive formula).

We end this section by defining what are list of positions for a sequent of L_{pos}, and we associate to each node $x \in T_A$ a sequent Γ in a "canonical" way. This association will be used to define the isomorphism between proofs and strategies.

Definition 8. Let $x \in T_A$ and $B \in L_{\text{pos}}$. Assume $L = \langle x_0, \ldots, x_m \rangle$ is a list of positions in T_A, and Γ is a sequent (a list of formulas).

- When $B = \mathtt{form}(p)$ is conjunctive, we say that the position p of B in T_A is conjunctive, or is an $\mathcal{A}$-node. When B is disjunctive, we say that p is disjunctive, or is an $\mathcal{E}$-node.
- If $\Gamma = \mathtt{Map}(\mathtt{form})(L) = \langle \mathtt{form}(x_0), \ldots, \mathtt{form}(x_m) \rangle$, we say that L a list of positions for Γ. $\mathtt{Form} = \mathtt{Map}(\mathtt{form})$ is the operator computing Γ out of L.
- *(Canonical labelling of T_A)* For each $x \in T_A$, we define the A-disjunctive sequence $L = \mathtt{disj}(x)$ as the list of positions $\mathcal{E}$ can backtrack to from x (including x itself). We say that a sequent Γ is A-disjunctive if there is some A-disjunctive list $L = \mathtt{disj}(x)$ of positions for Γ. We call Γ the *canonical sequent* labelling the node $x \in T_A$.

If L is the canonical sequent labelling x, then, by definition of 1-bck. play, L is the list of all $\mathcal{E}$-nodes (all disjunctive nodes) which are proper ancestors of x, plus x.[4]

[3] In the definition of Tarski play, this is obtained with a little trick. Nobody can move from an atomic a, because a has no immediate subformula. Therefore if a is true we make a conjunctive and we ask $\mathcal{A}$ to move next, in order to make him lose. If a is false we make a disjunctive and we ask $\mathcal{E}$ to move next, in order to make her lose.

[4] Alternatively, L is the list of ancestors of x with all $\mathcal{A}$-nodes (all conjunctive nodes) $\neq x$ skipped. The last node of L is x, that is, $\mathtt{last}(L) = x$. The previous nodes in L are all disjunctive nodes proper ancestors of x. Alternatively again, we can define $L = \mathtt{disj}(x)$ by induction on x. If $x = \langle * \rangle$, then $\mathtt{disj}(x) = \langle \langle * \rangle \rangle$. Assume $y = x{::}i$, and $\mathtt{disj}(x) = \langle x_0, \ldots, x_{m-1}, x \rangle$. If x is conjunctive, we set $\mathtt{disj}(y) = \langle x_0, \ldots, x_{m-1}, y \rangle$ (we skip x). If x is disjunctive, we set $\mathtt{disj}(y) = \langle x_0, \ldots, x_{m-1}, x, y \rangle$ (we keep x).

4 Proofs

In this section we introduce PA_1, a sub-classical ω-arithmetic whose proof trees will correspond exactly to all recursive winning strategies in Tarski games with 1-backtracking. Rules of PA_1 are introductions of some true atomic formula, or of $\vee, \wedge, \forall, \exists$. All these rules are merged with weakening, adding a sequent Δ of disjunctive formulas to the conclusion (possibly, Δ is empty). Sequents are ordered and Exchange is skipped.

Definition 9. Let $B \in L_{\text{pos}}$. Denote with B_i the one-step subformula of B of index i.

1. A proof π of classical ω-arithmetic without Exchange is a pair $\langle T_\pi, d_\pi \rangle$ of a *well-founded* T_π over $N, *$, and a decoration d_π, associating to each node $p \in T$ a non-empty sequent Θ of L_{pos} decorating it. d_π is a recursive map and T is recursive.
2. In addition, π should satisfy the following condition. For each $p \in \pi$, there is some sequent Γ, B, Δ decorating p, with all formulas in Δ *disjunctive*, and:
   - If B is a true atomic, then p has no descendant.

     $$\overline{\Gamma, B, \Delta}$$

   - If $B = B_1 \wedge B_2$, then the children of the node p are exactly $p{::}1$ and $p{::}2$, one for each B_i. The decoration of each $p{::}i$ is Γ, B_i.

     $$\frac{\Gamma, B_1 \quad \Gamma, B_2}{\Gamma, B_1 \wedge B_2, \Delta}$$

   - If $B = \forall x B(x)$, then the children of the node p are exactly all $p{::}n (n \in N)$, one for each $B(n)$. The decoration of each $p{::}n$ is $\Gamma, B(n)$.

     $$\frac{\ldots \quad \Gamma, B(n) \quad \ldots}{\Gamma, \forall x B(x), \Delta} \text{ (one node for each } n\text{)}$$

   - If $B = B_1 \vee B_2$, then p has only one descendant $p{::}i \in T$, corresponding to some B_i. The decoration of $p{::}i$ is $\Gamma, B_1 \vee B_2, B_i$

     $$\frac{\Gamma, B_1 \vee B_2, B_i}{\Gamma, B_1 \vee B_2, \Delta}$$

   - If $B = \exists x B(x)$, then p has only one descendant $p{::}n (n \in N)$, corresponding to some $B(n)$. The decoration of $p{::}n$ is $\Gamma, B, B(n)$.

     $$\frac{\Gamma, \exists x B(x), B(n)}{\Gamma, \exists x B(x), \Delta}$$

3. In the picture above, we say that in p we have an introduction of B, and a true, $\vee, \exists, \wedge, \forall$-introduction if the first symbol of B is true, $\vee, \exists, \wedge, \forall$.

There is no introduction for atomic false formulas, but you can add them by Weakening, inside Δ. We often insert the rule name, true, $\vee, \exists, \wedge, \forall$, in the right-hand-side of a rule. We stress that a sequent is an (ordered) list, and that there is no rule permuting two formulas.

We briefly discuss the main features of PA_1. The rule for disjunction involves contraction: we infer a disjunctive B_j from $B_j, B_{j,i}$, instead of inferring B_j, B_j. All rules involve weakening, too: we add some *disjunctive* formulas $B_{j+1}, \ldots, B_m$ to the conclusion. Weakening in its most general form (with both conjunctive and disjunctive formulas) is a derived rule, and so are Contraction, Exchange for *formulas* (not sequents), and Cut.

Lemma 10. *Let Γ, Δ be any sequents and A any formula.*

1. *(Weakening) If Γ is derivable, then Γ, Δ is, with a proof of the same (ordinal) height.*
2. *(Contraction) If Γ, A, A, Δ is derivable, then Γ, A, Δ is.*
3. *(Exchange* for formulas only*) Assume Γ can be obtained from Δ by replacing some $A \wedge B$ and some $C \vee D$ with $B \wedge A$ and $D \vee C$. Then Γ is derivable if and only if Δ is.*
4. *(Cut) Γ, A, and $A^{\perp}, \Delta$ are derivable, then Γ, Δ is, with a proof of* lesser height*.*

We skip the proof because it is not required in the rest of the paper. The last point is really unexpected: in full ω-arithmetic, cut elimination involves a super-exponential grow in height of the proof, while in PA_1, cut elimination reduces the height of a proof. Another curious feature concerns proofs of a singleton sequent $\{A\}$. All sequent in these proofs are lists $A_0, \ldots, A_n$ strictly decreasing under the subformula relation: A_{i+1} is a proper subformula of A_i, for all $i < n$. Actually (Lemma 18), all these sequents are A-disjunctive sequents of the Tarski game of A with 1-backtracking. Besides (Lemma 15 in Appendix), in a proof of $\{A\}$ we never merge Weakening and introduction of a conjunctive formula. That is, in any proof of $\{A\}$, all introductions of a conjunctive formula have the simplified form:

$$\frac{\ldots \quad \Gamma, B_i \quad \ldots}{\Gamma, B} \text{ (one node for each } B_i\text{)}$$

The main result of the paper is:

Theorem 11. *(Isomorphism Theorem) For all $A \in L_{pos}$, the class of proof-trees of the sequent $\{A\}$ in PA_1 is pointwise isomorphic to the class of the recursive winning strategy for the 1-backtracking game for A. Besides, if a node of a proof and a position in a winning strategy are in correspondence in such isomorphism, then they are labelled with the same sequent.*

The isomorphism theorem can be informally restated as follows. For any $A \in L_{\text{pos}}$ true in LCM, the proofs of A (in a suitable formal system PA_1) can

be identified with strategies learning by "trial-and-error" the winning moves in the Tarski game for A.

Here, we only sketch the proof of isomorphism theorem. We postpone the rigorous proof of the theorem to the appendix.

From proofs to strategies. Since we only deal with cut-free proofs, all formulas appeared in the proofs are subformulas of A. Moreover, we can identify these subformulas with subformula occurrences of A. Hence, we can treat formulas in the proofs as positions in Tarski game of A. For each sequents, we identify the right-most formula with the current position of the game, and other formulas with the positions which we can backtrack from the current position. Now, if we drop all formulas in the sequent except the right-most formula, then we get a strategy as a tree. The root of the tree is A, which is the starting point of the game. If the current position is $\mathcal{A}$-formula, the tree contains all branches corresponding all $\mathcal{A}$'s move. Note that by Lemma 15, there is no weakening (hence backtracking) occurs here. If the current position is $\mathcal{E}$-formula, there is only one branch with a possible backtracking. It is clear that $\mathcal{E}$ wins on all leaves in the tree.

From strategies to proofs. Let W be a winning strategy for $bck_{cf}(A)$. We decorate each position B in W by the sequent $C_1, \ldots, C_n, B$ where $C_1, \ldots, C_n$ are the sequence of all disjunctive formulas in the path from A to B of the subformula tree of A. Then the decorated tree π we obtained is a proof of A.

In the next section, we include some examples of how the isomorphism works.

5 Examples

In this section we include some example of proofs of A in PA_1 defining 1-backtracking winning strategies for T_A, and the other way round. The general idea is that a sequent $A_1, \ldots, A_n$ corresponds to a list of moves in a 1-bck. play, after we skip all moves to which $\mathcal{E}$ cannot backtrack. The introduction of a conjunctive formula corresponds to all possible moves $\mathcal{A}$ can do from a given position. The introduction of a disjunctive formula corresponds to a *winning* move by $\mathcal{E}$ from a given position. If the rules involves a Weakening of n formulas, this means that $\mathcal{E}$ backtracks n positions before making her move.

Our first example is the proof of Excluded Middle for semi-decidable formulas we quoted in the introduction.

5.1 1-excluded middle

The law of *Excluded Middle*, or EM is the schema $A \vee \neg A$. If only Σ^0_1-formulas are allowed to be substituted into A, the schema is called the law of *1-Excluded Middle*, or EM_1. EM_1 has the following natural proof in PA_1. In the proof tree below, we assume that $P(n-1)$ is a true (hence conjunctive) instance of $P(x)$, and $P(n)$ is a false (hence disjunctive) instance of $P(x)$. Therefore $\neg P(n)$ is true (hence conjunctive). We write boldface all formulas introduced by some rule. We write crossed out all formulas added by some introduction of $A \vee B$ or $\exists x.A$

(recall, these rules can add any sequent Δ to $A \vee B$ or $\exists x.A$). The only formulas of this latter kind in the proof are all $P(n)$. Since our logic is ω-logic, $\forall$-introduction has infinite branches. In the proof below, the premise $\forall x P(x) \vee \exists x \neg P(x), P(m)$ is deduced by true-introduction if $P(m)$ is true. Otherwise, $P(m)$ is derived in a similar from $\forall x P(x) \vee \exists x \neg P(x), \exists x \neg P(x), \neg P(m)$ (note that $\neg P(m)$ is true) by merging $\exists x \neg P(x), \neg P(m)$ to $\exists x \neg P(x)$ using $\exists$-introduction (we obtain $\forall x P(x) \vee \exists x \neg P(x), \exists x \neg P(x)$) and simultaneously merging $\forall x P(x) \vee \exists x \neg P(x), \exists x \neg P(x)$ to $\forall x P(x) \vee \exists x \neg P(x)$ by $\vee$-introduction and introducing $P(m)$ by weakening. Having $\forall x P(x) \vee \exists x \neg P(x), P(m)$ deduced for all $m \in N$, we apply $\forall$-introduction and obtain $\forall x P(x) \vee \exists x \neg P(x), \forall x P(x)$.

$$\dots \dfrac{\dfrac{\dfrac{}{\forall x P(x) \vee \exists x \neg P(x), \mathbf{P(n-1)}}\ \text{true} \quad \dfrac{\dfrac{\dfrac{}{\forall x P(x) \vee \exists x \neg P(x), \exists x \neg \mathbf{P(x)}, \neg \mathbf{P(n)}}\ \text{true}}{\forall x P(x) \vee \exists x \neg P(x), \exists \mathbf{x} \neg \mathbf{P(x)}}\ \exists^n}{\forall \mathbf{x} P(\mathbf{x}) \vee \exists \mathbf{x} \neg \mathbf{P(x)}, \cancel{P(n)}}\ \vee^2}{\forall x P(x) \vee \exists x \neg P(x), \forall \mathbf{x} P(\mathbf{x})}\ \vee^1 \dots_{\text{(one node for each } n\text{)}}$$
$$\forall \mathbf{x} P(\mathbf{x}) \vee \exists \mathbf{x} \neg \mathbf{P(x)}$$

We can translate this proof as a strategy learning the truth value of $\forall x P(x)$ using backtracking. We obtain the same strategy we sketched in the introduction. For the first move, Eloise chooses either $\forall x P(x)$ or $\exists x \neg P(x)$. Since Eloise does not know the witness of $\exists x \neg P(x)$, Eloise postpones the answer the witness, and chooses $\forall x P(x)$. Then, Abelard chooses an integer m for x. The play goes to $P(m)$. If $P(m)$ is true, Eloise wins. If $P(m)$ is false, then $\neg P(m)$ is true. Thus m is a witness of $\exists x \neg P(x)$. Therefore, Eloise backtracks from $P(m)$ to $\forall x P(x) \vee \exists x \neg P(x)$. This corresponds to introducing $P(m)$ by Weakening. Eloise moves $\exists x \neg P(x)$ from $\forall x P(x) \vee \exists x \neg P(x)$ this time. Again, the formula is disjunctive. Hence Eloise moves again.[5] Eloise chooses m and wins because $\neg P(m)$ is true.

5.2 2-excluded middle

We introduce now a proof corresponding to *no* winning strategy of 1-bck. play. Consider the law of excluded middle $A \vee \neg A$. When Σ^0_2-formulas, not only Σ^0_1-formulas are allowed to be substituted into A, the schema is called the law of *2-Excluded Middle*, or EM_2. The proof of EM_2 requires Exchange rule for sequents, as it is illustrated by the proof below (in ω-arithmetic with Exchange, not in PA_1). In the proof, $A = \forall x \exists y Q(x,y)$ and $\neg A$ denotes $\exists x \forall y \neg Q(x,y)$, $B(x) = \exists y Q(x,y)$ and $\neg B(x)$ denotes $\neg B(x) = \forall y \neg Q(x,y)$. Further, we assume that $Q(n, k-1)$ is false while $Q(n,k)$ is true. Formulas introduced by a rule are boldfaced, while formulas added by an introduction of $A \vee B$ or $\exists x.A$ to the left of $A \vee B$ or $\exists x.A$ are crossed out (recall, any introduction of $A \vee B$ or $\exists x.A$ is merged with Weakening). Exchange is labelled by Exch. The only formula subjected to Exchange, $B(n)$, is *not* introduced but switched with $A \vee \neg A$. Note

[5] Remark that we did not ask in the definition of play that player alternate.

that since we have true-introduction only for true atomics, A and $\neg A$ must be decomposed to atomic formulas. Since we are dealing with ω-logic, the proof has infinite branches.

First, we derive $B(n), A \vee \neg A, \neg A, \neg Q(n,k)$ in a way depending on whether $Q(n,k)$ is true or false. If $Q(n,k)$ is true, we derive $B(n), Q(n,k)$ by true-introduction, then simultaneously merging $Q(n,k)$ to $B(n)$ by $\exists$-introduction and introducing $A \vee \neg A, \neg A, \neg Q(n,k)$ by weakening. If $Q(n,k)$ is false, since $\neg Q(n,k)$ is true, we can derive $B(n), A \vee \neg A, \neg A, \neg Q(n,k)$ by true-introduction. Having $B(n), A \vee \neg A, \neg A, \neg Q(n,k)$ for all $k \in N$, we derive $B(n), A \vee \neg A, \neg A, \forall x \neg Q(n,x)$ ($\forall x \neg Q(n,x) \equiv \neg B(n)$). Applying $\exists^n$- and $\vee^2$-introduction, we have $B(n), A \vee \neg A$. Now, exchange $B(n)$ and $A \vee \neg A$ so that $B(n)$ becomes active. Since the proof so far can be carried out for any $n \in N$, we have $A \vee \neg A, B(n)$ for all $n \in N$. By $\forall$-introduction we have $A \vee \neg A, \forall x B(x)$ ($\forall x B(x) \equiv A$). By $\vee^1$-introduction, we have $A \vee \neg A$.

$$\dfrac{\dfrac{\dots \quad \dfrac{}{B(n), A \vee \neg A, \neg A, \neg \mathbf{Q(n, k-1)}}\ \text{true} \quad \dfrac{\dfrac{}{B(n), \mathbf{Q(n,k)}}\ \text{true}}{\mathbf{B(n)}, A \vee \neg A, \neg A, \neg Q(n,k)}\ \exists^k \quad \dots}{B(n), A \vee \neg A, \neg A, \neg \mathbf{B(n)}}\ {\scriptstyle \text{(one node for each } k\text{)}}\ \exists^n}{\dfrac{B(n), A \vee \neg A, \neg \mathbf{A}}{B(n), \mathbf{A} \vee \neg \mathbf{A}}\ \vee^2}$$

$$\dfrac{\dfrac{\dots \quad A \vee \neg A, \mathbf{B(n)} \quad \dots}{A \vee \neg A, \mathbf{A}}\ {\scriptstyle \text{(one node for each } n\text{)}}\ \text{Exch}}{\mathbf{A} \vee \neg \mathbf{A}}\ \vee^1$$

It is essential to exchange $B(n)$ and $A \vee \neg A$. In this stage of the proof, we are supposed to prove $B(n)$, that is, we are suppose to find some y which is a witness $B(n) = \exists y Q(n,y)$. We do not know even whether such a y exists or not. Moreover we cannot introduce $B(n)$ by weakening, otherwise we would stuck in a later stage of the proof, while proving $A \vee \neg A, \neg A, \neg Q(n,k)$ with $Q(n,k)$ true. What Exchange does is to postpone the proof of $B(n)$ after we fail proving $\neg B(n) = \forall y \neg Q(n,y)$. If and when we fail, we find some true $Q(n,k)$, and we use $y = k$ as a witness for it $B(n) = \exists y Q(n,y)$ (look to the top right corner of the proof).

It is easier to understand how the proof works if we interpret it as a winning strategy of a backtracking game of Tarski game $G_{A \vee \neg A}$. However, as we will show, we need a stronger form of backtracking than 1-backtracking for interpreting Exchange: we need backtracking in the sense of Coquand [2]. The interpretation of the proof runs as follow.

As in the case of EM_1, Eloise chooses $\forall x \exists y Q(x,y)$ and Abelard chooses a subformula $\exists y Q(n,y)$. Now the turn is Eloise's, but she does not know which $Q(n,k)$ is true (if any). Therefore, she postpones to answer $\exists y Q(n,y)$ and opens the dual game $\neg A$. She repeats Abelard's play and reaches $\forall y \neg Q(n,y)$. Abelard chooses either some integer k such that $\neg Q(n,k)$ is true, or some integer l such that $\neg Q(n,l)$ is false. In the first case, Eloise wins. In the second case, $Q(n,l)$ is

true. Eloise reopens the play for $\exists y Q(n, y)$, and chooses l for y. The play goes to $Q(n, l)$ which is true. Hence Eloise wins. This play is not a 1-bck. play, since backtracking from $\neg Q(n, k)$ to $\exists y Q(n, y)$ violates the condition 2 of Definition 2. Indeed, $\exists y Q(n, y)$ is not an ancestor of the node $\neg Q(n, k)$ in the subformula tree of $A \vee \neg A$, therefore the backtracking we used is not 1-backtracking. Backtracking necessary for EM_2 is called 2-backtracking in [1]. In 1-bck. play, a position $\exists y Q(n, y)$ becomes unreachable after Eloise backtracks from $\exists y Q(n, y)$ to $A \vee \neg A$, while the winning strategy for EM_2 requires to reactivate it. In the term of a proof, backtracking of 1-bck. play corresponds weakening, since formulas in the conclusions are discarded from the premises after 1-bck. Instead, 2-backtracking and general backtracking correspond to exchange, since de-activated formulas are *not* discarded from the premises, but just pushed back in the sequent, and they can be recovered if we later need them.

5.3 A "learning algorithm" corresponding to a classical proof

In this example, we first define a "learning strategy" (or 1-backtracking strategy) for the Tarski game of some formula A. Then we turn it into a proof in PA_1 of A. In this way we sketch how we can recover a proof from a "learning algorithm" in the sense of Hayashi. This is the formula A we consider. For any functions f, g_1, g_2 from $\mathbf{N}$ (the set of natural numbers) to $\mathbf{N}$, there is some $n \in \mathbf{N}$ such that $f(n) \leq f(g_1(n))$ and $f(n) \leq f(g_2(n))$. We can find an n as required, by the following "learning" algorithm. First we choose n arbitrary, say $n = 0$, then:

Step 1: If $f(n) \leq f(g_1(n)), f(g_2(n))$, then n is a solution. Otherwise, either $f(g_1(n))$ or $f(g_2(n))$ is strictly smaller than $f(n)$.

Step 2: If $f(g_1(n)) < f(n)$, then let n be $g_1(n)$ and return to step 1. If $f(g_2(n)) < f(n)$, then let n be $g_2(n)$ and return to step 1.[6]

Let n_i be the i-the number used as n. Then $f(n_1) > f(n_2) > \ldots > f(n_i) > \ldots$ by its construction. From $f(n_i) \in \mathbf{N}$ we conclude the algorithm eventually reaches some solution.

We can see this "learning" algorithm as a winning strategy for 1-bck. plays of Tarski game of the formula $\exists n. f(n) \leq f(g_1(n)) \wedge f(n) \leq f(g_2(n))$ with 1-backtracking. First, Eloise chooses 0 as n. Abelard chooses either $f(0) \leq f(g_1(0))$ or $f(0) \leq f(g_2(0))$. If the formula Abelard chooses is true, then Eloise wins. Otherwise, Eloise backtracks to $\exists n. f(n) \leq f(g_1(n)) \wedge f(n) \leq f(g_2(n))$ and chooses another n by using Abelard's move, as follows. If Abelard chooses $f(0) \leq f(g_1(0))$, then Eloise chooses $g_1(0)$ as a new n. If Abelard chooses $f(0) \leq f(g_2(0))$, then Eloise chooses $g_2(0)$ as a new n. Next, Abelard chooses either $f(n) \leq f(g_1(n))$ or $f(n) \leq f(g_2(n))$, and Eloise uses this information to refine n further. $\mathcal{E}$ repeats this sequence of moves over and over again. Eloise always wins by the same reason why the algorithm above is terminating.

We can translate this strategy to the proof of $\exists n. f(n) \leq f(g_1(n)) \wedge f(n) \leq f(g_2(n))$. Backtracking corresponds to introducing $f(0) \leq f(g_1(0))$ or $f(0) \leq$

[6] If $f(g_1(n))$ and $f(g_2(n))$ are both smaller than $f(n)$, we can choose $g_1(n)$ or $g_2(n)$ randomly.

$f(g_2(0))$ by Weakening. Let $A(n)$ be $f(n) \leq f(g_1(n)) \wedge f(n) \leq f(g_2(n))$ and $m = g_{i_0}(\ldots g_{i_k}(0)))$, where $i_j = 1, 2$ for $0 \leq j \leq k$. In the picture below, we assume that the only atomic true formula is $(f(m) \leq f(g_2(m)))$. Then, the proof has the following form. We write boldface all formulas introduced by some rule, and crossed out all formulas added by Weakening. We construct our proof by bottom-up fashion. For each step, we try to prove $\exists n A(n), A(m)$ where $m = 0$ at first. $\exists n A(n), A(m)$ is derived from $\exists n A(n), (f(m) \leq f(g_1(m))$ and $\exists n A(n), (f(m) \leq f(g_2(m))$. If, say $f(m) \leq f(g_2 m$ is true, we are done. If, say $f(m) \leq f(g_1(m))$ is false, we try to prove $\exists n A(n), A(g_1(m))$. Since $f(g_1(m)) < f(m)$, this process should be stopped somewhere. If we prove $\exists n A(n), A(g_1(m))$, we can obtain $\exists n A(n), (f(m) \leq f(g_1(m))$ by simultaneously merging $A(g_1(m))$ to $\exists n A(n)$ and introducing $f(m) \leq f(g_1(m))$ by weakening.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\exists n A(n), \mathbf{A(g_1(m))}}}{\exists \mathbf{nA(n)}, \cancel{(f(m) \leq f(g_1(m)))}} \quad \dfrac{}{\exists n A(n), (\mathbf{f(m) \leq f(g_2(m))})}\ \text{true}}{\exists n A(n), \mathbf{A(m)}} \quad \dfrac{\dfrac{\dfrac{\vdots}{\exists n A(n), \mathbf{A(g_1(0))}}}{\exists \mathbf{nA(n)}, \cancel{(f(0) \leq f(g_1(0)))}} \quad \dfrac{\dfrac{\vdots}{\exists n A(n), \mathbf{A(g_2(0))}}}{\exists \mathbf{nA(n)}, \cancel{(f(0) \leq f(g_2(0)))}}}{\exists n A(n), \mathbf{A(0)}}}{\exists \mathbf{nA(n)}}$$

For the reverse direction, it is easy to see that the proof corresponds the strategy given above. In the workshop version of the paper, we cut here the example section for reason of space. In the technical report, instead, we include two more elaborated examples of the correspondence proof-strategy. We take an algorithm learning some minimum point of $f : \mathbf{N} \to \mathbf{N}$, and we turn it into a proof that the minimum point exists. Then we define an algorithm learning an infinite weakly increasing subsequence from any infinite sequence over $\mathbf{N}$, and we turn it into a proof that such infinite subsequence exists.

In appendix we generalize to all proofs and all strategies the correspondence we just defined between some proofs and some strategies.

References

[0] S. Berardi, Y. Yamagata. A sequent calculus for 1-backtracking, Technical Report, Turin University, December 2005. http://www.di.unito.it/ stefano/Yamagata-Berardi-report.pdf.

[1] S. Berardi, Th. Coquand., S. Hayashi. Games with 1-backtracking, Proceedings of GaLop, Edinburgh, April 2005.

[2] Th. Coquand. A semantics of evidence for classical arithmetic. Journal of Symbolic Logic 60 (1995), pp. 325–337.

[3] S. Hayashi, *Mathematics based on Learning*, Algorithmic Learning Theory, LNAI 2533, Springer, pp.7-21.
[4] Hayashi, S.: *Can proofs be animated by games?*, in P. Urzyczyn ed., TLCA 2005, LNCS 3461, pp.11-22, 2005, invited paper.
[5] H. Herbelin, *Sequents qu'on calcule: de l'interpretation du calcul des sequents comme calcul de lambda-termes et comme calcul de strategies gagnantes*, Ph. D. thesis, University of Paris VII, 1995.
[6] W. Hodges, "Logic and Games", The Stanford Encyclopedia of Philosophy (Winter 2004), Edward N. Zalta (ed.).
http://plato.stanford.edu/archives/win2004/entries/logic-games/

6 Trees

In this section we introduce some (routine) coding and some terminology for trees.

We denote with $\texttt{List}(I)$ the set of finite lists of a set I. For any $x, y \in \texttt{List}(I)$, we denote "$x$ is a prefix of y" and "x is a proper prefix of y" with $x \leq y$ and $x < y$. Let $x = \langle i_0, \ldots, i_{k-1} \rangle \in \texttt{List}(I)$ denote any list. If x is not empty, we denote with $\texttt{drop}(x) = \langle x_0, \ldots, x_{k-2} \rangle \in \texttt{List}(I)$ and $\texttt{last}(x) = x_{k-1} \in I$ the list x with the last element dropped, and the last element of x. For any $0 \leq h < k$ we denote $\langle i_0, \ldots, i_h \rangle$ the prefix of x with all elements up to the index h, with $x\lceil h$. If $f : I \to J$, we define a map $\texttt{Map}(f) : \texttt{List}(I) \to \texttt{List}(J)$ by applying f to each item in x: $\texttt{Map}(f)(x) = \langle f(i_0), \ldots, f(i_{k-1}) \rangle$. If $i \in I$, we denote $\langle i_0, \ldots, i_{k-1}, i \rangle$ with $x::i$. We call any $x::i$ a one-step extension of x. We associate :: to the left: $x::i::j$ is short for $(x::i)::j$. We have $\texttt{drop}(x::i) = x$ and $\texttt{last}(x::i) = i$ and $\texttt{Map}(f)(x::i) = \texttt{Map}(f)(x)::f(i)$.

Definition 12. Let E be any set and $*_E \in E$. A tree T over $E, *_E$ is a set of finite non-empty lists $\langle e_0, \ldots, e_k \rangle$, with $e_0, \ldots, e_k \in E$ and $e_0 = *_E$, such that:

1. $\langle *_E \rangle$ is the minimum of T under prefix.
2. T is closed under non-empty prefixes: if $p \in T$ and $\langle\rangle \neq q < p$, then $q \in T$.

When no ambiguity arises we write $*$ for $*_E$. If T is a tree on $E, *$, we call $\langle * \rangle \in T$ the root of T, and $*$ the starting symbol of T.[7] If both $x, x::i \in T$, we say that $x::i$ is the child of x of index i, and that x is the father of $x::i$. We say that x is an ancestor of y if $x \leq y$. An infinite branch of T is any infinite sequence $\langle e_0, e_1, e_2 \ldots \rangle$ whose non-empty finite prefixes are all in T. A subtree U of T is a subset U of T which is still a tree. Assume U is a tree over some $F, *_F$. Then a morphism $\phi : T \to U$ between T, U is any map sending the root of T into the root of U, and compatible with father/child relation (that is, $\phi(\langle *_E \rangle) = \langle *_F \rangle$, and for all $x::a \in T$ there is some $b \in F$ such that $\phi(x::a) = \phi(x)::b$). ϕ is an isomorphism if and only if ϕ is invertible, and ϕ^{-1} is still a morphism.

[7] $*$ is some dummy symbol we introduced in order to avoid empty lists in the coding of a tree. Introducing $*$ is just an arbitrary choice. Proofs and definitions of this paper become a bit simpler skipping empty lists.

Definition 13. We say that two classes $\mathcal{C}, \mathcal{D}$ of trees are pointwise isomorphic if there are opposite bijections $\Phi : \mathcal{C} \to \mathcal{D}$ and $\Psi : \mathcal{D} \to \mathcal{C}$, such that any two trees corresponding through Φ, Ψ are isomorphic.

By definition unfolding, we can now precise the details of the coding of bck_{cf}. [8]

7 First properties of proofs

We list here a few well-known combinatorial properties we will need while proving the main theorem. We use the tree terminology from §6. For proofs we refer to the technical report [0].

Lemma 14. *Let $\phi : T \to U$ be any morphism between two trees T, U. Let $f : A \to B$ and $g : B \to A$. Denote the identities over A, B with $\mathtt{id}_A, \mathtt{id}_B$.*

1. *ϕ is an isomorphism if and only if ϕ is a bijection (i.e.: if ϕ is invertible, then its inverse is a morphism).*
2. *For any $x \in T$, x and $\phi(x)$ have the same length.*
3. *If $gf = \mathtt{id}_A$, then f is an injection and g a surjection.*
4. *If $gf = \mathtt{id}_A$, and either $fg = \mathtt{id}_B$, or f is a surjection, or g is an injection, then f, g are opposite bijections.*

The first property of proofs we derive is the following:

Lemma 15. 1. *If π is a proof of a sequent Γ, and all formulas in Γ before the last one are disjunctive, then the same holds for all sequents in the proof.*
2. *If π is a proof a singleton sequent $\{A\}$, then all introductions of conjunctive formulas in π use no weakening, that is, they always introduce the last formula of the sequent.*

A well-known property of (cut-free) proofs is the following. All sequents Γ decorating a proof π of a formula A consist of subformulas of A (*proof:* by induction over π, using the fact we have no cut rule in $\mathtt{PA}_1$). Our next step is to fix some canonical choice for the list of positions of all sequent Γ labelling a proof π.

Definition 16. Let π be any proof of a formula A. We say that $\psi_\pi : T \to \mathtt{List}(T_A)$ is a decoration of π with list of positions if for all $p \in \pi$, $\psi_\pi(p)$ is a position of $d_\pi(p)$. We say that ψ_π is *canonical* if for all $p{::}i \in \pi$, $M = \psi_\pi(p)$ and $L = \psi_\pi(p{::}i)$, we have $L = M'{::}(x{::}i)$, for some $M' \leq M$ and some x in M.

[8] Let E be the set of moves of G. Then T_{cf} is a tree over the set T of positions of the game G, therefore $T_{cf} \subseteq \mathtt{List}(T)$. Since $T \subseteq \mathtt{List}(E)$, we are coding the set T_{cf} of positions of $bck_{cf}(G)$ with some subset of $\mathtt{List}^2(E)$ (some set of lists of lists of moves of the original game G). The starting symbol of T_{cf} is $\langle *_E \rangle$ (a list), while the root is $\langle\langle *_E \rangle\rangle$ (a list of lists).

Using $\mathtt{drop}, \mathtt{last}$ from §6 , we can reformulate the definition of canonical decoration by: $\mathtt{drop}(L) \leq M$ and $\mathtt{drop}(\mathtt{last}(L))$ is in M, and $\mathtt{last}(\mathtt{last}(L)) = i$, for all L, M as above. For each $\pi = \langle T, d_\pi, d_\pi \rangle$ proof of formula A there is exactly one canonical decoration. We define ψ_π by induction over p, as follows.

Definition 17. 1. $\psi_\pi(\langle * \rangle) = \langle \langle * \rangle \rangle$.

2. Assume $p_0, p_0{::}i \in T$ and $\psi_\pi(p_0) = \langle x_0, \ldots, x_m \rangle$ and $d_\pi(p_0) = B_0, \ldots, B_m$.
   - Suppose B_m is conjunctive and $d_\pi(p) = B_0, \ldots, B_{m-1}, B_{m,i}$, for some $B_{m,i}$ one-step subformula of B_j. Then $\psi_\pi(p) = \langle x_0, \ldots, x_{m-1}, x_m{::}i \rangle$.
   - Suppose $B_j, \ldots, B_m$ are disjunctive and $d_\pi(p) = B_0, \ldots, B_j, B_{j,i}$, for some $B_{j,i}$ one-step subformula of B_j, of index i. Then we set $\psi_\pi(p) = \langle x_0, \ldots, x_j, x_j{::}i \rangle$.

The two cases in the definition of ψ_π cannot superpose, because B_m is either conjunctive or disjunctive. The two cases cover all proofs of a formula A by Lemma 15.2. If p is any node of the proof π of A, we call $\psi_\pi(p)$ the list of positions associated to p in the canonical decoration. Now we derive some property of canonical decorations.

Lemma 18. *Assume that π, π' are any proofs of A. Then:*

1. *A canonical decoration of π is unique.*
2. *For all $p \in \pi$, $\psi_\pi(p)$ is some A-disjunctive chain.*
3. *ψ_π is a canonical decoration of π with lists of positions, and $\mathtt{last}^2(\psi(p)) = \mathtt{last}(p)$ for all $p \in \pi$.*
4. *If π, π' have equal trees, and equal decorations $\psi_\pi, \psi_{\pi'}$ on these trees, then they are equal proofs.*

8 An intermediate step between proofs and strategies

Fix any $A \in L_{\mathrm{pos}}$. We define a class of trees associated to A we call proof-strategy trees. A proof-strategy tree can be seen either as a proof of A, or as a winning strategy for the Tarski game for A. In order to achieve such effect, proof-strategies are heavily cluttered with redundant information. In the next sections we will show that both proofs and proof-strategies, and proof-strategies and winning strategies are pointwise isomorphic classes of trees. Besides, all these isomorphisms preserve the sequent labelling each node. We will conclude the same result for proofs and strategies.

Intuitively, a proof-strategy of A is a proof π of A in which we replaced each sequent Γ of π by its canonical list of positions. The list of positions of any sequent Γ is an extra information we can add to a proof. It is not strictly required, because it can be recovered from the proof. We only consider it in order to make the correspondence between proofs and strategy more evident.

Intuitively again, a proof-strategy can be seen as a strategy in which we replaced each move x with the A-disjunctive chain of x (the list $L = \langle x_0, \ldots, x_m \rangle$ of moves $\mathcal{E}$ can backtrack to, including x itself, which is x_m). This list L is an extra information we can add to a winning strategy, not strictly required in order to code a winning strategy, because it can be recovered from the position x.

Definition 19. A proof-strategy is any tree P over $\mathtt{List}(T_A)$ (over the set of all lists of positions), having $\langle\langle * \rangle\rangle \in \mathtt{List}(T_A)$ as starting symbol, and such that, for all $Y = X::(L::x) \in S$:

- If x is an $\mathcal{A}$-node, then the one-step extensions Z of Y in S are exactly all $Z = Y::(L::(x::i))$, for all $x::i \in T_A$.
- If x is an $\mathcal{E}$-node, then Y has a unique one-step extension Z in S, and $Z = Y::(M::y::(y::i))$, for some prefix $M::y \leq L::x$, and some $y::i \in T_A$.

If $X \in P$, we call the sequent Γ of position $\mathtt{last}(X)$ the canonical sequent labelling X.

By induction on $X \in P$ we can prove that all $X \in P$ are lists of A-disjunctive chain. A remark about coding: A-disjunctive chains are in $\mathtt{List}^2(N)$, therefore P is (coded by) some subset of $\mathtt{List}^3(N)$, or some set of: lists of lists of lists of integers.

The next Lemma states a first relationship between proof-strategies and proofs. Assume that a proof π and a proof-strategy P are isomorphic trees, that the isomorphism preserves the last integer in a node, and that the isomorphism defines a list decoration of π. In this case, the isomorphism defines a canonical list decoration of π.

Lemma 20. *Assume $p \mapsto X_p$ is an isomorphism between some proof-strategy P and some proof-tree π, such that $\mathtt{last}(p) = \mathtt{last}^3(X_p)$ for all $p \in \pi$. Assume furthermore that $p \mapsto \mathtt{last}(X_p)$ is a list decoration of the proof-tree. Then this list decoration is canonical.*

9 proof-strategies and strategies are pointwise isomorphic

In this section we show that we can interpret a proof-strategy for A as a strategy for the cut-free 1-backtracking game for A.

We can define a pair Ψ_2, Φ_2 of pointwise isomorphisms between strategies and proof-strategies for A, as follows. Intuitively, we replace each list of positions with its last element to turn a proof-strategy into a strategy. Conversely, we replace each position of Tarski game T_A with the A-disjunctive list of position associated to it, in order to turn a strategy into a proof-strategy .

Here is the formal definition. Let $\mathtt{last} : \mathtt{List}(T_A) \to T_A$ be the map taking the last element of a list over T_A. Then we set $\psi_2 = \mathtt{Map}(\mathtt{last}) : \mathtt{List}^2(T_A) \to \mathtt{List}(T_A)$. Conversely, let $\mathtt{disj} : T_A \to \mathtt{List}(T_A)$ be the map computing the A-disjunctive chain for a given $x \in T_A$. Then we set $\phi_2 = \mathtt{Map}(\mathtt{disj}) : \mathtt{List}(T_A) \to \mathtt{List}^2(T_A)$. Eventually, for any proof-strategy P we define a strategy $W = \Psi_2(P) = \{\psi_2(X) | X \in P\}$. For any strategy W we define a proof-strategy $P = \Phi_2(W) = \{\phi_2(L) | L \in W\}$.

Lemma 21. *1. $\psi_2\phi_2(L) = L$, for all $L \in \mathtt{List}(T_A)$, and $\psi_2\phi_2(X) = X$, for all lists X of A-disjunctive chains.*

2. *If P is a proof-strategy , then ψ_2, ϕ_2 are opposite tree isomorphisms, and $\Psi(W)$ is a winning strategy.*
3. *If W is a winning strategy, then ϕ_2, ϕ_2 are opposite tree isomorphisms, and $\Phi_2(W)$ is a proof-strategy .*
4. *Ψ_2, Φ_2 are pointwise isomorphisms. If a node $X \in P$ and a play $p \in W$ are in correspondence through ψ_2, ϕ_2, then they are labelled by the same sequent.*

10 Proofs and proof-strategies are pointwise isomorphic

In this section we show that we can interpret any proof-strategy P as the canonical decoration of some proof π with lists of positions.

Fix any $A \in L_{\text{pos}}$. We define a pair Ψ_1, Φ_1 of pointwise isomorphism between proofs and proof-strategies for A, as follows. Intuitively, we get a proof-strategy from a proof by replacing each sequent by its canonical position. We get a proof from a proof-strategy by replacing back each list of position of a sequent by the sequent. Formally, for any proof π of A, if $p = \langle i_0, \ldots, i_k \rangle \in \pi$, we set $\psi_1(p) = \langle \psi_\pi(p\lceil 0), \ldots, \psi_\pi(p\lceil k) \rangle$ (where $p\lceil k$ is p itself). By definition, the last element of $\psi_1(p)$ is $\psi_\pi(p\lceil k) = \psi_\pi(p)$, the canonical list of positions decorating p. Alternatively, we set $\psi_1(\langle * \rangle) = \langle \langle \langle * \rangle \rangle \rangle$, and $\psi_1(p::i) = \psi_1(p)::\psi_\pi(p::i)$. Then we define a proof-strategy $P = \Psi_1(\pi)$ as the set $\{\psi_1(p) | p \in \pi\}$ of lists of A-disjunctive chains. Each A-disjunctive chain is a list of formula, and formulas are coded by lists. Therefore each $\psi_1(p)$ is coded by a lists of lists of lists, a list nested three times.

We define the opposite map ϕ_1 taking a list Y nested three times and returning a list. ϕ_1 takes, for each element of Y (some list nested twice), the last element of its last element: that is, we set $\phi_1 = \texttt{Map}(\texttt{last}^2)$. Alternatively, $\phi_1(\langle\rangle) = \langle\rangle$, and if $Y \in \texttt{List}^3(N)$ is a one-step extension of X, then $\phi(Y) = \phi(X)::\texttt{last}^3(Y)$. For any proof-strategy P, we define a proof π. The proof-tree is $T = \{\phi_1(X) | X \in P\}$. We claim that ϕ_1 is a tree-isomorphism : $P \to T$. Let $X = \phi_1^{-1}(p) = \langle L_0, \ldots, L_m \rangle$. We define the decoration $d_\pi(p)$ of the node $p \in T$ as the only sequent having list of positions $L_m = \texttt{last}(X)$ (i.e., $d_\pi(p) = \texttt{Form}(\texttt{last}(X))$). Eventually, for any proof-strategy P we define the proof $\pi = \Phi_1(P) = \langle T, d_\pi \rangle$.

Lemma 22. *1. If π is any proof, then ψ_1, ϕ_1 are opposite tree isomorphisms, and $\Psi_1(\pi)$ is a proof-strategy .*
2. If P is any proof-strategy , then ϕ_1 is a tree isomorphism, $\pi' = \Phi_1(P)$ is a proof, and $\texttt{last}(\phi_1^{-1}(p))$ is a canonical list of positions for π', and $\psi_1 = \phi_1^{-1}$.
3. Ψ_1, Φ_1 are pointwise isomorphisms. These isomorphisms preserve the sequent labelling a node.

From Lemma 21.4 and Lemma 22.3 we deduce the Main Theorem:

Theorem 23. *(Isomorphism Theorem) For all $A \in L_{pos}$, the class of proof-trees of the sequent $\{A\}$ in PA_1 is pointwise isomorphic to the class of the recursive winning strategy for the 1-backtracking game for A. Besides, if a node of a proof*

and a position in a winning strategy are in correspondence in such isomorphism, then they are labelled with the same sequent.