LMSO: A Curry-Howard Approach to Church's Synthesis via Linear Logic

Pierre Pradic
ENS de Lyon, Université de Lyon, LIP
Lyon, France
pierre.pradic@ens-lyon.fr
University of Warsaw, Faculty of Mathematics, Informatics and Mechanics
Warsaw, Poland

Colin Riba
ENS de Lyon, Université de Lyon, LIP
Lyon, France
colin.riba@ens-lyon.fr

Abstract

We propose LMSO, a proof system inspired from Linear Logic, as a proof-theoretical framework to extract finite-state stream transducers from linear-constructive proofs of omega-regular specifications. We advocate LMSO as a stepping stone toward semi-automatic approaches to Church's synthesis combining computer assisted proofs with automatic decisions procedures. LMSO is correct in the sense that it comes with an automata-based realizability model in which proofs are interpreted as finite-state stream transducers. It is moreover complete, in the sense that every solvable instance of Church's synthesis problem leads to a linear-constructive proof of the formula specifying the synthesis problem.

CCS Concepts • **Theory of computation** → **Logic**; *Logic and verification*; *Proof theory*; *Linear logic*;

Keywords Categorical Logic, Game Semantics, Realizability, Linear Logic, MSO on Infinite Words.

ACM Reference Format:
Pierre Pradic and Colin Riba. 2018. LMSO: A Curry-Howard Approach to Church's Synthesis via Linear Logic. In *LICS '18: 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, July 9–12, 2018, Oxford, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3209108.3209195

1 Introduction

Church's synthesis [5] consists in the automatic extraction of stream transducers (or *Mealy machines*) from input-output specifications. Ideally, these specifications would be written in *Monadic Second-Order Logic* (MSO) on ω-words [29, 30]. MSO on ω-words is a decidable logic thanks to Büchi's Theorem [3], whose proof is originally based on an effective translation of MSO formulae to non-deterministic Büchi automata (NBA's). It subsumes non-trivial logics used in verification such as LTL (see e.g. [1, 28]). Church's synthesis for (subsystems of) LTL has also been substantially studied (see e.g. [2, 7, 16]).

Traditional theoretical solutions to Church's synthesis start from an ω-word automaton recognizing the specification (typically an NBA), and apply *McNaughton's Theorem* [18] to obtain an equivalent deterministic (say parity) automaton on ω-words. There are then essentially two methods (see e.g. [29, 30]). The first one turns the deterministic automaton into a game graph, in which the *Opponent* O ($\forall$bélard) plays input characters to which the *Proponent* P ($\exists$loïse) replies with output characters. Solutions to Church's synthesis are then given by the Büchi-Landweber Theorem [4], which says that in such games, either P or O has finite-state winning strategy. The second solution goes via infinite trees [23], noting that a causal function from say Σ to Γ can be represented by an infinite Γ-labeled Σ-ary tree.

However, the translation of MSO-formulae to NBA's is non-elementary [10], ruling out any tractable implementation. Moreover, even when restricting to LTL (which has an exponential translation to NBA's, see e.g. [1]), the use of McNaughton's Theorem has long been a major obstacle to Church's synthesis.[1]

In this paper, extending [22], we advocate an approach to Church's synthesis in the framework of program extraction from proofs (in the sense of e.g. [27]). We propose a constructive deduction system for an expressively equivalent variant of MSO, based on a complete axiomatization of MSO on ω-words as a subsystem of second-order Peano arithmetic [26] (see also [24]). The formal proofs in this deduction system are interpreted in an automata-based realizability semantics, along the lines of the Curry-Howard *proofs-as-programs* correspondence. Our system is correct, in the sense that from a proof of a $\forall\exists$-specification one can extract a Mealy machine implementing the specification. It is moreover complete, in the sense that it proves all $\forall\exists$-specifications which are realizable by Mealy machines.

The crux of our approach is that on the one hand the *correctness proof* of our realizability interpretation relies on McNaughton's Theorem, while on the other hand the *extraction* of realizers from formal proofs does not invoke it.

In the context of MSO, using a deduction system may avoid the systematic translation of formulae to automata, and may allow for human intervention and compositional reasoning. In a typical usage scenario, the user interactively performs some proofs steps and delegate the generated subgoals to

LICS '18, July 9–12, 2018, Oxford, United Kingdom

ACM ISBN 978-1-4503-5583-4/18/07...$15.00
https://doi.org/10.1145/3209108.3209195

[1] Interesting workarounds are the *Safraless* approaches of [7, 16].

automatized synthesis procedures. The partial proof tree built by the user is then translated to a combinator able to compose the transducers synthesized by the algorithms.

The deduction system SMSO proposed in [22] was based on intuitionistic logic. While SMSO is correct and complete for Church's synthesis, it suffers from a very limited set of primitive connectives ($\wedge$, $\neg$, $\exists$) so that formal proofs may be cumbersome without resorting to a negative translation from a complete axiomatization of MSO in classical logic. In this paper, we propose a deduction system LMSO inspired from *Intuitionistic Linear Logic* [9] (see also [19]). LMSO has a rich set of connectives (with primitive $\otimes, \parr, \multimap, !, ?, \exists, \forall$), with a straightforward interpretation as usual automata constructions.[2] The system LMSO is moreover based on an extension MSO^+ of MSO with primitive function symbols for Mealy machines, allowing for a much more efficient extraction of realizers from proofs.

Organization of the paper. We begin in §2 by presenting Church's synthesis problem and our extension MSO^+ of MSO. We also briefly discuss there the intuitionistic system SMSO. We then present §3 our linear system LMSO, and show its completeness w.r.t. MSO^+ and Church's synthesis. The realizability interpretation is then split into two parts: §4 recapitulates known material on games and automata from [25], and §5 presents the realizability interpretation of LMSO and states its correctness.

2 Church's Synthesis and MSO^+

Notations. Alphabets (denoted Σ, Γ, etc) are sets of the form $\mathbf{2}^p$ for some $p \in \mathbb{N}$. We see alphabets as being built by following grammar:

$$\Sigma, \Gamma \quad ::= \quad \mathbf{1} \quad | \quad \mathbf{2} \quad | \quad \Sigma \times \Gamma \quad | \quad \Sigma \to \Gamma$$

Concatenation of words u, v is denoted either $u.v$ or $u \cdot v$, and ε is the empty word. We use the vectorial notation both for words and finite sequences, so that e.g. $\overline{B}$ denotes a finite sequence $B_1, \dots, B_n$ and $\bar{\mathtt{a}}$ denotes a word $\mathtt{a}_1 . \cdots . \mathtt{a}_n \in \Sigma^*$. Given an ω-word (or stream) $B \in \Sigma^\omega$ and $n \in \mathbb{N}$ we write $B \restriction n$ for the finite word $B(0). \cdots .B(n-1) \in \Sigma^*$.

Specifications. Specifications for stream functions $\Sigma^\omega \to \Gamma^\omega$ will be given by formulae $\varphi(\overline{Y}; \overline{Z})$ where, assuming $\Sigma = \mathbf{2}^p$ and $\Gamma = \mathbf{2}^q$, $\overline{Y} = Y_1, \dots, Y_p$ and $\overline{Z} = Z_1, \dots, Z_q$ are tuples of (monadic) set variables[3]. For instance, the formula

$$(\exists^\infty k. Y(k)) \quad \Longrightarrow \quad (\exists^\infty k. Z(k)) \qquad (1)$$

specifies functions $f : \mathbf{2}^\omega \to \mathbf{2}^\omega$ such that $f(B) \in \mathbf{2}^\omega \simeq \mathcal{P}(\mathbb{N})$ is infinite whenever $B \in \mathbf{2}^\omega$ is infinite.

Causal Functions and Mealy Machines. We shall actually require our specifications to be realized by stream functions implementable by finite state stream transducers, a.k.a. Mealy machines.

Definition 2.1. A *Mealy machine* $\mathcal{M}$ with input alphabet Σ and output alphabet Γ (notation $\mathcal{M} : \Sigma \to \Gamma$) is given by an alphabet of states Q with a distinguished initial state $q^\imath \in Q$, and a transition function $\partial : Q \times \Sigma \to Q \times \Gamma$.

We write ∂^o for $\pi_2 \circ \partial : Q \times \Sigma \to \Gamma$ and ∂^* for the map $\Sigma^* \to Q$ obtained by iterating ∂ from the initial state: $\partial^*(\varepsilon) := q^\imath$ and $\partial^*(\bar{\mathtt{a}}.\mathtt{a}) := \pi_1(\partial(\partial^*(\bar{\mathtt{a}}), \mathtt{a}))$.

A Mealy machine $\mathcal{M} : \Sigma \to \Gamma$ induces a function $F_{\mathcal{M}} : \Sigma^\omega \to \Gamma^\omega$ defined as $F_{\mathcal{M}}(B)(n) = \partial^o(\partial^*(B \restriction n), B(n))$. Hence $F_{\mathcal{M}}$ can produce a length-n prefix of its output from a length-n prefix of its input. These functions are called *causal.*

Definition 2.2. A function $F : \Sigma^\omega \to \Gamma^\omega$ is *causal* if for all $n \in \mathbb{N}$ and all $B, C \in \Sigma^\omega$ we have $F(B) \restriction n = F(C) \restriction n$ whenever $B \restriction n = C \restriction n$. We say that a causal function F is *finite-state* (f.s.) if it is induced by a Mealy machine.

Example 2.3. (a) The identity function $\Sigma^\omega \to \Sigma^\omega$ is induced by the Mealy machine with state set $\{\bullet\} \simeq \mathbf{1}$ and identity transition function $\partial : (\bullet, \mathtt{a}) \longmapsto (\bullet, \mathtt{a})$.

(b) Causal functions are obviously continuous (taking the product topology on Σ^ω and Γ^ω, with Σ, Γ discrete), but there are continuous functions which are not causal, e.g. $P : \mathbf{2}^\omega \to \mathbf{2}^\omega$ such that $P(A)(n) = 1$ iff $A(n+1) = 1$.

In the context of this paper, it is useful to note that finite-state causal functions form a category with finite products.

Definition 2.4. Let $\mathbf{S}$ be the category whose objects are alphabets and whose maps from Σ to Γ are causal functions $F : \Sigma^\omega \to \Gamma^\omega$. Let $\mathbf{M}$ be the wide subcategory of $\mathbf{S}$ whose maps are *finite-state* causal functions.

Note that ω-words $B \in \Sigma^\omega$ correspond exactly to causal functions from $\mathbf{1}^\omega$ to Σ^ω. We thus identify Σ^ω and $\mathbf{S}[\mathbf{1}, \Sigma]$. Also, functions $\mathtt{f} : \Sigma \to \Gamma$ induce $\mathbf{M}$-maps $[\mathtt{f}] : \Sigma \to_{\mathbf{M}} \Gamma$.

Proposition 2.5. *The categories $\mathbf{S}, \mathbf{M}$ have finite products. The product of $\Sigma_1, \dots, \Sigma_n$ (for $n \geq 0$) is given by the product of sets $\Sigma_1 \times \cdots \times \Sigma_n$ (so that $\mathbf{1}$ is terminal).*

The Logic MSO^+***.*** We now introduce our specification language, the logic MSO^+. It is an extension of (the one-sorted version of) MSO with one function symbol $t_{\mathcal{M}}$ of arity p for each Mealy machine $\mathcal{M} : \mathbf{2}^p \to \mathbf{2}$. The *terms* of MSO^+, ranged over by t, u, etc, are built with these function symbols from (monadic) predicate variables X, Y, Z, etc of arity 0. The formulae of MSO^+ are given in Fig. 1.

MSO^+-formulae are interpreted in the standard model $\mathfrak{N}$ of ω-words. Variables range over sets of natural numbers $B, C, \ldots \in \mathcal{P}(\mathbb{N}) \simeq \mathbf{2}^\omega$. A term t together with a valuation $X_i \mapsto B_i$ of its variables $X_1, \dots, X_n$ is interpreted as $F(B_1, \dots, B_n)$ where $F : \mathbf{2}^n \to \mathbf{2}$ is the f.s. causal function induced by t.[4] The atomic predicates are interpreted as follows: $\doteq$ is equality, $\dot\subseteq$ is set inclusion, E holds on B iff B is empty, N (resp. $\mathsf{0}$) holds on B iff B is a singleton $\{n\}$ (resp. the singleton $\{0\}$), and $\mathsf{S}(B, C)$ (resp. $B \dot\leq C$) holds iff $B = \{n\}$ and $C = \{n+1\}$ for some $n \in \mathbb{N}$ (resp. $B = \{n\}$ and $C = \{m\}$ for some $n \leq m$).

We use the following notational conventions. Lowercase roman letters x, y, z, etc denote variables relativized to N. In particular, $\exists x.\varphi$ and $\forall x.\varphi$ stand respectively for

$$\exists x(\mathsf{N}(x) \wedge \varphi) \qquad \text{and} \qquad \forall x(\mathsf{N}(x) \to \varphi)$$

[2] The usual additive connectives $\oplus, \&$ of Linear Logic have also natural interpretations in automata.

[3] For notational simplicity we work modulo $(\mathbf{2}^k)^\omega \simeq (\mathbf{2}^\omega)^k$.

[4] Recall that $\mathbf{M}$ is a category.

$$\begin{array}{llclclclclclclclcl}
\textbf{Atoms:} & \alpha \in \mathrm{At} & ::= & t \doteq u & \mid & \mathsf{E}(t) & \mid & t \mathrel{\dot\subseteq} t & \mid & \mathsf{N}(t) & \mid & \mathsf{0}(t) & \mid & \mathsf{S}(t,u) & \mid & t \mathrel{\dot\leq} u \\
\textsf{MSO}\ \textbf{formulae:} & \varphi, \psi & ::= & \alpha & \mid & \top & \mid & \bot & \mid & \varphi \vee \psi & \mid & \varphi \wedge \psi & \mid & \varphi \to \psi & \mid & \exists X.\varphi \quad \mid \quad \forall X.\varphi
\end{array}$$

Figure 1. The formulae of MSO and MSO$^+$ (where terms t, u are restricted to variables X, Y for MSO).

Moreover $x \mathrel{\dot\in} t$ stands for $x \mathrel{\dot\subseteq} t$, so that

$$\mathfrak{N} \models \quad X \mathrel{\dot\subseteq} Y \quad \longleftrightarrow \quad \forall x(x \mathrel{\dot\in} X \to x \mathrel{\dot\in} Y)$$

We often write $X(x)$ or even Xx for $x \mathrel{\dot\in} X$. Finally $\exists^\infty n.\varphi$ stands for $\forall m.\exists n \mathrel{\dot\geq} m.\varphi$ and $\forall^\infty n.\varphi$ for $\exists m.\forall n \mathrel{\dot\geq} m.\varphi$.

The logic MSO is MSO$^+$ with terms restricted to monadic variables $X, Y, Z,$ etc. MSO$^+$ is an extension by definition of MSO (and thus conservative over MSO) thanks to the following well-known fact:

Proposition 2.6. *For each Mealy machine $\mathcal{M} : \mathbf{2}^p \to \mathbf{2}$, one can build an* MSO*-formula $\delta_\mathcal{M}(\overline{X}, x)$ such that for all $n \in \mathbb{N}$ and all $\overline{B} \in (\mathbf{2}^\omega)^p$, we have*

$$F_\mathcal{M}(\overline{B})(n) = 1 \qquad \textit{iff} \qquad \mathfrak{N} \models \delta_\mathcal{M}(\{n\}, \overline{B})$$

Example 2.7. MSO directly expresses the specification (1), as well as other typical properties used in software verification, such as the following *safety* and *liveness* properties:

$$\forall x \left(\mathsf{A}[\overline{Y(x)}] \ \longrightarrow\ \mathsf{G}[\overline{Z(x)}]\right) \qquad \text{(safety)}$$
$$\forall x \left(\mathsf{A}[\overline{Y(x)}] \ \longrightarrow\ \exists y > x.\mathsf{G}[\overline{Z(y)}]\right) \qquad \text{(liveness)}$$

Here, A (resp. G) is a propositional formula with say p (resp. q) variables, representing a subset of $\mathbf{2}^p$ (resp. $\mathbf{2}^q$).

Büchi Automata. The logic MSO (and thus MSO$^+$) over $\mathfrak{N}$ is decidable thanks to Büchi's Theorem [3].

Theorem 2.8 (Büchi [3]). MSO *over $\mathfrak{N}$ is decidable.*

In our context, it is pertinent to look at Thm. 2.8 through its original proof method, which consists in translating formulae to Büchi automata. A non-deterministic *Büchi* automaton (NBA) is an NFA, but which accepts an ω-word if there exists an infinite run with infinitely many final states. It is known that *deterministic* Büchi automata are strictly less expressive than NBA's.

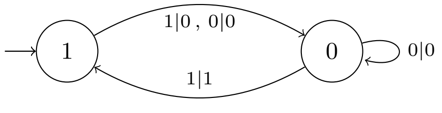
The crux of Büchi's Theorem 2.8 is the effective closure of Büchi automata under complement. Let us recall a few known algorithmic facts (see e.g. [10, 28]). First, the translation of MSO-formulae to automata is non-elementary. Second, it is known that complementation of NBA's is algorithmically hard: there is a family of languages $(\mathcal{L}_n)_{n>0}$ such that each $\mathcal{L}_n$ can be recognized by an NBA with $O(n)$ states, but such that the complement of $\mathcal{L}_n$ can not be recognized by an NBA with less than $n!$ states. Known constructions for complementation produce NBA's with $O(2^{n \log(n)})$ states from NBA's with n states.

Church's Synthesis. Church's synthesis problem for MSO$^+$ is the following. Given as input an MSO$^+$ formula $\varphi(\overline{Y}; \overline{Z})$ (where $\overline{Y} = Y_1, \ldots, Y_p$ and $\overline{Z} = Z_1, \ldots, Z_q$), (1) decide whether there exist f.s. causal $\overline{F} = F_1, \ldots, F_q : \mathbf{2}^p \to_\mathrm{M} \mathbf{2}$ such that $\mathfrak{N} \models \varphi(\overline{B}; \overline{F}(\overline{B}))$ for all $\overline{B} \in (\mathbf{2}^\omega)^p$, and (2), construct $\overline{F}$ whenever they exist.

Example 2.9. The following specification $\phi(Y; Z)$ from [29]

$$\begin{aligned}\forall n\,(\neg Yn \to \neg Zn) \ \wedge\ \forall n, m\,(\mathsf{S}(n,m) \ \to\ Zn \ \to\ \neg Zm) \\ \wedge\ (\exists^\infty n.\, Yn \ \to\ \exists^\infty n.\, Zn)\end{aligned}$$

asks $n \in Z$ whenever $n \in Y$, Z not to contain two consecutive positions, and Z to be infinite whenever Y is infinite. It is realized by the following Mealy machine, where a transition $\mathtt{a}|\mathtt{b}$ outputs $\mathtt{b}$ from input $\mathtt{a}$:



We now briefly sketch general solutions to Church's synthesis. To this end, it is convenient to start from Büchi automata rather than MSO$^+$ formulae. Given an automaton $\mathcal{A} : \Sigma \times \Gamma$, we say that a Mealy machine $\mathcal{M} : \Sigma \to \Gamma$ *realizes* $\mathcal{A}$ if $\mathcal{A}$ accepts $(B, F_\mathcal{M}(B))$ for every $B \in \Sigma^\omega$.

Starting from a Büchi automaton $\mathcal{B} : \Sigma \times \Gamma$, traditional solutions to Church's synthesis (see e.g. [29, 30]) begin by translating $\mathcal{B}$ to an equivalent *deterministic* automaton $\mathcal{D}$ (say equipped with a parity condition[5]). Determinization of automata on ω-words is originally due to McNaughton [18].

Theorem 2.10 (McNaughton [18]). *Each NBA is equivalent to a deterministic parity automaton.*

There are two historical solutions to Church's synthesis. The first one, due to Büchi & Landweber [4], is to turn the automaton $\mathcal{D} : \Sigma \times \Gamma$ into a two-player sequential game, in which the *Opponent* O plays inputs characters in Σ while the *Proponent* P replies with outputs characters in Γ. The game is equipped with the parity condition of $\mathcal{D}$. The solution is then provided by Büchi-Landweber's Theorem [4], which states that ω-regular games on finite graphs are effectively determined, and that the winner has a f.s. winning strategy[6].

A second possibility, due to Rabin [23] (see also [16]), uses *tree* automata. The idea is that causal functions $\Sigma^\omega \to \Gamma^\omega$ can be represented as Γ-labeled Σ-ary trees. The solution is then to build from $\mathcal{D}$ a tree automaton accepting exactly the Γ-labeled Σ-ary trees which represent realizers of $\mathcal{D} : \Sigma \times \Gamma$.[7]

However, neither of these solutions directly lead to applicable algorithms. The best known (and possible) constructions for McNaughton's Theorem (such as *Safra's trees*, see e.g. [10]) give deterministic Muller automata with $2^{O(n \log(n))}$ states from NBA's automata with n states. This may seem no worse than NBA's complementation, but while the latter

[5] There are different expressively equivalent conditions for deterministic automata (parity, *Muller*, *Rabin*, *Streett*, see e.g. [10, 28]). All can specify which states an infinite run *must not* see infinitely often.

[6] This implies that in the setting of this paper, causal realizers can always be assumed to be finite-state.

[7] Actually, emptiness of tree automata is reduced to solving parity games on finite graphs, so this solution also goes via [4].

is amenable to tractable implementations (see e.g. [8]), this is not the case for McNaughton's Theorem (see e.g. [2, 7, 16]). Also, the states of automata obtained from Thm. 2.10 have a complex structure, making difficult implementations of subsequent algorithms (e.g. game solving).

Curry-Howard Approaches. In this paper, extending [22], we advocate semi-automatic approaches in the framework of program extraction from proofs (in the sense of e.g. [27]).

We start with a complete axiomatization of MSO^+, based on a known axiomatization of MSO on infinite words [26] (see also [24]). The theory of MSO^+ is given by deduction for first-order classical logic (with the terms of MSO^+ as individuals), together with the *Arithmetic Rules* of Fig. 2 and the following axiom schemes:[8]

- *Induction.* For each formula φ, the following rule (where y, z are fresh):

$$\frac{\overline{\varphi}, \mathsf{0}(z) \vdash \varphi[z/x], \overline{\varphi}' \qquad \overline{\varphi}, \mathsf{S}(y,z), \varphi[y/x] \vdash \varphi[z/x], \overline{\varphi}'}{\overline{\varphi} \vdash \forall x.\varphi, \overline{\varphi}'}$$

- *Comprehension.* For each formula φ with X not free in φ, the axiom:

$$\overline{\vdash \exists X. \forall x. (Xx \longleftrightarrow \varphi)}$$

- *Definition of Mealy Machines.* For all function symbol $t_{\mathcal{M}}$ representing the machine $\mathcal{M} : \mathbf{2}^p \to \mathbf{2}$, the following axiom (where $\delta_{\mathcal{M}}$ is the formula of Prop. 2.6):

$$\overline{\vdash \forall \overline{X}. \forall x. \left(x \mathrel{\dot{\in}} t_{\mathcal{M}}(\overline{X}) \longleftrightarrow \delta_{\mathcal{M}}(x, \overline{X})\right)}$$

Theorem 2.11. *For each closed formula φ of MSO^+,*

$$\mathfrak{N} \models \varphi \qquad \textit{iff} \qquad \mathsf{MSO}^+ \vdash \varphi$$

In [22] the authors devised a constructive system SMSO which is sound and complete w.r.t. Church's synthesis. The language of SMSO has a restricted set of connectives ($\vee, \exists, \neg$) and a specific set of atomic formulae. The main result of [22] is the following:

Theorem 2.12. *Given an SMSO-formula $\varphi(\overline{X}; \overline{Y})$,*

(a) from a proof of $\exists \overline{Z}.\varphi(\overline{Y}; \overline{Z})$ in SMSO, one can extract a f.s. causal realizer of $\varphi(\overline{Y}; \overline{Z})$,
(b) if $\varphi(\overline{Y}; \overline{Z})$ admits a (f.s.) causal realizer, then SMSO proves $\exists \overline{Z}.\neg\neg\varphi(\overline{Y}; \overline{Z})$.

The idea behind Theorem 2.12 is that for MSO-synthesis, rather than starting by translating a formula to an automaton, one may use a deduction system in order to decompose the problem into subgoals (possibly with semi-automated techniques), on which one may either invoke algorithms, or proceed with other proof steps. The realizability model underlying [22] (based on [25]), then allows to compose realizers obtained by synthesis algorithms with realizers extracted from proofs in SMSO.

One feature of SMSO is that extraction of realizers from formal proofs does not involve McNaughton's Theorem (nor complementation of NBA's) even if the *correctness proof* of the realizability model does invoke it.

The system SMSO has however some limitations. First of all, its set of connectives is very limited, so that proofs in the system SMSO itself may be cumbersome without appealing to a negative translation from the complete axiomatization of MSO in classical logic. Second, the extraction process can itself be quite costly: Mealy machines were represented as usual MSO-formulae (according to Prop. 2.6) so that witness extraction from proofs must pay the price of the translation of these formulae back to automata.

***Linear Variants of* SMSO.** In this paper, we introduce a variant LMSO of SMSO, inspired from *Intuitionistic Linear Logic* [9] (see also [19]). First, linearity allows to have more primitive connectives: implications and disjunction in addition to conjunction, and primitive universal quantifications in addition to existentials.

The logical system LMSO is built on the model of [25], which relies on a variant of alternating automata [20, 21] called *uniform automata* (UA's). UA's are equipped with a monoidal closed structure and with primitive existential and universal quantifications in the categorical sense [25]. Alternating automata allows in some cases better translations of formulae since they are easier to complement. However, it is well-known that existential (resp. universal) quantifications are only correct on ND (resp. universal) automata, so that the full translation of MSO to alternating automata involves the *simulation* of alternating automata by ND ones. This operation, known as the *Simulation Theorem* [21] in the case of infinite trees, actually amounts on ω-words to McNaughton's Theorem (see e.g. [21]).

Similarly as with SMSO, we devise for LMSO a realizability model which involves McNaughton's Theorem in its correctness proof, but *not* for the extraction of realizers from formal proofs. Theorem 2.12 extends to LMSO (for a suitable translation $(-)^L$), so that if $\varphi(\overline{Y}; \overline{Z})$ is realized by $\mathcal{M}$, then

$$\mathsf{LMSO} \;\vdash\; \forall \overline{Y}.\varphi^L(\overline{Y}; \overline{t_{\mathcal{M}}(\overline{Y})})$$

Moreover, the system LMSO is equipped with *polarities* thanks to which the *exponential connectives* of Linear Logic exactly indicate the applications McNaughton's Theorem in translations to UA's. This reflects the fact that the following positive (noted φ^+) and negative (noted φ^-) fragments of MSO^+ have exponential translations to automata:[9]

$$\begin{array}{r c l}
\varphi^+, \psi^+ & ::= & \top \mid \bot \mid \alpha \mid \exists X.\varphi^+ \mid \varphi^- \to \varphi^+ \\
& \mid & \varphi^+ \wedge \psi^+ \mid \varphi^+ \vee \psi^+ \\
\varphi^-, \psi^- & ::= & \top \mid \bot \mid \alpha \mid \forall X.\varphi^- \mid \varphi^+ \to \varphi^- \\
& \mid & \varphi^- \wedge \psi^- \mid \varphi^- \vee \psi^-
\end{array}$$

3 LMSO: A Linear Variant of MSO

We introduce here the formal system LMSO, inspired from the multiplicative-exponential fragment of Linear Logic [9]. LMSO has two layers. The first layer, PLMSO, is a polarized logic whose polarities respect the polarities of automata. The second layer LMSO allows unrestricted polarities *but for the exponential connectives* $!(-)$ *and* $?(-)$.

[8] The rules of Fig. 2 are given in a two-sided sequent calculus format (see e.g. [27]) in order to ease the later presentation of LMSO.

[9] An interesting consequence of [7] is that the negative fragment of MSO^+ is expressively complete.

$$\frac{}{\vdash t \doteq t} \qquad \frac{}{t \doteq u,\, \varphi[t/X] \vdash \varphi[u/X]} \qquad \frac{}{\mathsf{E}(t) \vdash t \,\dot{\subseteq}\, u} \qquad \frac{\overline{\varphi} \vdash t \,\dot{\subseteq}\, Z,\, \overline{\varphi}'}{\overline{\varphi} \vdash \mathsf{E}(t),\, \overline{\varphi}'} \qquad \frac{\overline{\varphi},\, Z \,\dot{\subseteq}\, t \vdash \mathsf{E}(Z),\, Z \doteq t,\, \overline{\varphi}'}{\overline{\varphi} \vdash \mathsf{N}(t),\, \mathsf{E}(t),\, \overline{\varphi}'} \qquad \frac{}{\mathsf{N}(t),\, \mathsf{E}(t) \vdash \bot}$$

$$\frac{}{\vdash t \,\dot{\subseteq}\, u} \qquad \frac{}{t \,\dot{\subseteq}\, u,\, u \,\dot{\subseteq}\, v \vdash t \,\dot{\subseteq}\, v} \qquad \frac{}{t \,\dot{\subseteq}\, u,\, u \,\dot{\subseteq}\, t \vdash t \doteq u} \qquad \frac{\overline{\varphi},\, \mathsf{N}(Z),\, Z \,\dot{\subseteq}\, t \vdash Z \,\dot{\subseteq}\, u,\, \overline{\varphi}'}{\overline{\varphi} \vdash t \,\dot{\subseteq}\, u,\, \overline{\varphi}'} \qquad \frac{}{\mathsf{N}(t),\, u \,\dot{\subseteq}\, t \vdash \mathsf{E}(u),\, u \doteq t}$$

$$\frac{}{\mathsf{N}(t) \vdash t \,\dot{\leq}\, t} \qquad \frac{}{t \,\dot{\leq}\, u,\, u \,\dot{\leq}\, v \vdash t \,\dot{\leq}\, v} \qquad \frac{}{t \,\dot{\leq}\, u,\, u \,\dot{\leq}\, t \vdash t \doteq u} \qquad \frac{}{\mathsf{S}(t,u) \vdash t \,\dot{\leq}\, u} \qquad \frac{}{\mathsf{0}(t) \vdash \mathsf{N}(t)}$$

$$\frac{\overline{\varphi} \vdash \overline{\varphi}'}{\overline{\varphi},\, \mathsf{0}(Z) \vdash \overline{\varphi}'} \qquad \frac{\overline{\varphi} \vdash \overline{\varphi}'}{\overline{\varphi},\, \mathsf{S}(t,Z) \vdash \overline{\varphi}'} \qquad \frac{}{\mathsf{S}(u,v),\, t \,\dot{\leq}\, v \vdash t \doteq v\, ,\, t \,\dot{\leq}\, u} \qquad \frac{}{\mathsf{S}(u,v),\, \mathsf{0}(v) \vdash \bot} \qquad \frac{}{t \,\dot{\leq}\, u \vdash \mathsf{N}(t)} \qquad \frac{}{t \,\dot{\leq}\, u \vdash \mathsf{N}(u)}$$

$$\frac{}{\mathsf{0}(t),\, \mathsf{0}(u) \vdash t \doteq u} \qquad \frac{}{\mathsf{S}(t,u),\, \mathsf{S}(t,v) \vdash u \doteq v} \qquad \frac{}{\mathsf{S}(u,t),\, \mathsf{S}(v,t) \vdash u \doteq v} \qquad \frac{}{\mathsf{S}(t,u) \vdash \mathsf{N}(t)} \qquad \frac{}{\mathsf{S}(t,u) \vdash \mathsf{N}(u)}$$

Figure 2. The Arithmetic Rules of MSO^+ and LMSO (where Z is fresh in each rule mentioning it).

The logic LMSO is only *inspired from* Linear Logic, since both its polarities and linearity are induced by our automata-based realizability model (to be detailed in §4 and §5) rather than by proof-theory (as in e.g. [17]).

3.1 The Language of LMSO

The formulae of PLMSO are divided into the *positive* formulae (written φ^+), the *negative* formulae (written φ^-) and the *deterministic* ones (written $\varphi^\pm$). They are defined on Fig. 3.

The formulae of LMSO are built from the PLMSO-formulae by using with unrestricted polarities all the connectives of PLMSO *but for the exponentials* $!(-), ?(-)$. The formulae of LMSO are formally defined as follows:

$$\begin{array}{rcl} \varphi, \psi & ::= & \varphi^\pm \quad | \quad \varphi \multimap \psi \quad | \quad \varphi \otimes \psi \quad | \quad \varphi \parr \psi \\ & | & \exists X.\varphi \quad | \quad \forall X.\varphi \end{array}$$

In contrast with MSO^+, the formulae of LMSO are not intended to be directly interpreted in the standard model $\mathfrak{N}$ of ω-words. We will instead interpret them in an automaton-based realizability model in the vein of [22, 25]. As such, the connectives of LMSO directly reflect usual operations on alternating ω-word automata:

- LMSO-formulae are to be thought about as representing alternating automata. Positive, negative and deterministic formulae represent resp. non-deterministic, universal and deterministic automata.
- $\otimes$ and $\parr$ are conjunctions and disjunctions based on a direct product of automata[10]. The linear implication $\multimap$ has been introduced in [25].
- The quantifiers $\exists$ and $\forall$ correspond to the usual operations of projection and co-projection (see e.g. [25]).
- The exponentials ? and ! correspond to determinization operations.

As usual with alternating automata, projections only correctly implement existential quantifications on *ND* automata. Dually, co-projections completely implement universal quantifications on *universal* automata. However, both operations can always be defined on alternating automata, and moreover with their intended categorical semantics [25]. This implies that they can be used in a deduction system.

In PLMSO, since existential (resp. universal) quantifications can only be used on positive (resp. negative) formulae, all connectives will indeed have their usual (standard) meaning. This in particular permits the following simple translation from MSO^+-formulae to PLMSO-formulae.

Definition 3.1 (Simple Translation)**.** The translation $(-)^L$ is defined by induction on MSO^+-formulae as follows:

$$\begin{array}{rclcrcl} \top^L & := & \top & \qquad & (\varphi \wedge \psi)^L & := & \varphi^L \otimes \psi^L \\ \bot^L & := & \bot & & (\varphi \vee \psi)^L & := & \varphi^L \parr \psi^L \\ \alpha^L & := & \alpha & & (\varphi \to \psi)^L & := & \varphi^L \multimap \psi^L \\ (\exists X.\varphi)^L & := & ?\exists X.\varphi^L & & (\forall X.\varphi)^L & := & !\forall X.\varphi^L \end{array}$$

Note that φ^L is always a deterministic formula of PLMSO.

Let us stress a couple of important points.

(a) It would have been natural to also allow the usual (additive) non-deterministic disjunction $\oplus$ and its corresponding conjunction &. But this would have required additional structure in our realizability model, while have chosen to keep it as simple as possible in this paper.

(b) Thanks to the *Simulation Theorem* [21][11], it would have been possible to have exponentials $?\varphi$ and $!\varphi$ for all LMSO-formulae. In this case, $!\varphi$ would represent an ND automaton simulating the alternating automaton representing φ. The general exponential ! has been investigated in a Curry-Howard setting [25]. We forbid these operations here because they impose realizers to concretely depend on applications of McNaughton's Thm. 2.10.

3.2 The Deduction System of LMSO

Deduction in LMSO is performed using the rules of Fig. 4, the Arithmetic Rules of Fig. 2 and following axiom schemes:

- *Induction.* For each *negative* φ^-, the following rule (where y, z are fresh, $\overline{\varphi}^+$ are positive and $\overline{\psi}^-$ negative):

$$\frac{\begin{array}{rcl} \overline{\varphi}^+, \mathsf{0}(z) & \vdash & \varphi^-[z/x], \overline{\psi}^- \\ \overline{\varphi}^+, \mathsf{S}(y,z), !(\varphi^-[y/x]) & \vdash & \varphi^-[z/x], \overline{\psi}^- \end{array}}{\overline{\varphi}^+ \vdash \forall x.\varphi^-, \overline{\psi}^-}$$

[10] As such, $\parr$ does not exist on *tree* automata.

[11] Which is *not* required to deal with ω-regular properties.

$$
\begin{array}{rcl}
\varphi^{\pm}, \psi^{\pm} & ::= & \top \mid \bot \mid \alpha \mid !(\varphi^{-}) \mid ?(\varphi^{+}) \mid \varphi^{\pm} \otimes \psi^{\pm} \mid \varphi^{\pm} \parr \psi^{\pm} \mid \varphi^{\pm} \multimap \psi^{\pm} \\
\varphi^{+}, \psi^{+} & ::= & \varphi^{\pm} \mid \exists X. \varphi^{+} \mid \varphi^{+} \otimes \psi^{+} \mid \varphi^{+} \parr \psi^{+} \mid \varphi^{-} \multimap \psi^{+} \\
\varphi^{-}, \psi^{-} & ::= & \varphi^{\pm} \mid \forall X. \varphi^{-} \mid \varphi^{-} \otimes \psi^{-} \mid \varphi^{-} \parr \psi^{-} \mid \varphi^{+} \multimap \psi^{-}
\end{array}
$$

Figure 3. The Formulae of PLMSO (where X is a monadic variable and $\alpha \in \mathrm{At}$ is an atomic formula as in Fig. 1).

- *Deterministic Comprehension.* For each *deterministic* $\varphi^{\pm}$ with X not free in $\varphi^{\pm}$, the axiom
$$\frac{}{\vdash ?\exists X !\forall x \, (Xx \;\; \multimapboth \;\; \varphi^{\pm})}$$
- *Definition of Mealy machines.* For each function symbol $t_{\mathcal{M}}$ representing the machine $\mathcal{M} : \mathbf{2}^{p} \to \mathbf{2}$, the axiom
$$\frac{}{\vdash \forall \overline{X} \forall x \left(x \mathrel{\dot{\in}} t_{\mathcal{M}}(\overline{X}) \;\; \multimapboth \;\; \delta_{\mathcal{M}}(x, \overline{X})^{L} \right)}$$
where $\delta_{\mathcal{M}}$ is the formula of Prop. 2.6.
- *Polarized Double Negation Elimination.* For each PLMSO-formula φ, the axiom
$$\frac{}{\varphi^{\bot\bot} \vdash \varphi}$$
where $\psi^{\bot} := \psi \multimap \bot$ (note that $\varphi^{\bot\bot}$ is a PLMSO-formula of the same polarity as φ).
- *Deterministic Exponentials.* For each *deterministic* formula $\varphi^{\pm}$, the axioms
$$\frac{}{\varphi^{\pm} \vdash !(\varphi^{\pm})} \qquad \text{and} \qquad \frac{}{?(\varphi^{\pm}) \vdash \varphi^{\pm}}$$

Even if LMSO has a multiplicative disjunction $\parr$, which naturally leads to sequents with multiple formulae on the right of $\vdash$, the proof-system is actually constructive, as witnessed by the right rules for $\forall$ and $\multimap$, and by the polarity condition in the right contraction rule. Similarly, double negation elimination is only assumed for polarized formulae, so that $\varphi \multimap \psi$ is not isomorphic to $\varphi^{\bot} \parr \psi$. In Ex. 5.11 we will see that double negation elimination is realizable for all formulae, but is not an isomorphism.

Our first (expected) result on LMSO is that the translation $(-)^{L}$ correctly interprets MSO^{+} in LMSO.

Theorem 3.2. *Given an* MSO^{+} *formula* φ, *if* $\mathsf{MSO}^{+} \vdash \varphi$ *then* $\mathsf{LMSO} \vdash \varphi^{L}$.

Thanks to the completeness of MSO^{+} (Thm. 2.11), Thm. 3.2 implies that LMSO is complete w.r.t. Church's synthesis.

Corollary 3.3. *If an* MSO^{+}*-formula* $\varphi(\overline{Y}; \overline{Z})$ *admits a (f.s.) causal realizer, then* $\mathsf{LMSO} \vdash \forall \overline{Y}. \exists \overline{Z}. \varphi^{L}(\overline{Y}; \overline{Z})$.

4 Games and Automata

We present here our games and automata model. It is a simplification to ω-words of the model presented in [25] for infinite trees. All statements of this Section (excepted those relative to the $\parr$ connective) are proved in [25].

This model targets two indexed categories: The categories Aut_{Σ} of automata over alphabet Σ are indexed over $\mathbf{M}$, while the categories DA_{Σ} of (substituted) acceptance games over Σ are indexed over $\mathbf{S}$. Moreover, the categories $\mathsf{Aut}_{(-)}$ are equipped with the required categorical structure to interpret the linear part of LMSO.

All the structure we use on $\mathsf{DA}_{(-)}$ and $\mathsf{Aut}_{(-)}$ is ultimately induced by the symmetric monoidal closed structure of a category $\mathbf{DZ}$ of *zig-zag games*, that we detail first.

4.1 Simple Zig-Zag Games

We present here the category $\mathbf{DZ}$ of *zig-zag* games, and discuss some structure relevant to us: symmetric monoidal closure and a construction of indexed categories inspired from [13] and generalizing *simple fibrations* (see e.g. [14]).

Definition 4.1. A *game* A has the form $A = (A_{\mathsf{P}}, A_{\mathsf{O}})$ where A_{P} and A_{O} are alphabets of resp. P and O-*moves*.

We are interested in a very specific form of strategies.

Definition 4.2. Given games A and B, a (total zig-zag) *strategy* $\sigma : A \to_{\mathbf{DZ}} B$ is a pair of functions $\sigma = (f, F)$ where

$$
\begin{array}{rclcl}
f & : & \bigcup_{n \in \mathbb{N}} \left(A_{\mathsf{P}}^{n+1} \times B_{\mathsf{O}}^{n} \right) & \longrightarrow & B_{\mathsf{P}} \\
F & : & \bigcup_{n \in \mathbb{N}} \left(A_{\mathsf{P}}^{n+1} \times B_{\mathsf{O}}^{n+1} \right) & \longrightarrow & A_{\mathsf{O}}
\end{array}
$$

Intuitively, a total zig-zag strategy $\sigma : A \to_{\mathbf{DZ}} B$ amounts to a strategy for P in an infinite game which consists in $\mathbb{N}$-indexed sequences of rounds. In a single round $n \in \mathbb{N}$, four moves occur in succession:

(1) O plays a move $a_{n}^{\mathsf{P}} \in A_{\mathsf{P}}$,
(2) P plays a move $b_{n}^{\mathsf{P}} \in B_{\mathsf{P}}$,
(3) O answers with a move $b_{n}^{\mathsf{O}} \in B_{\mathsf{O}}$,
(4) P concludes with a move $a_{n}^{\mathsf{O}} \in A_{\mathsf{O}}$.[12]

Proposition 4.3. *Games and (total zig-zag) strategies form a category* $\mathbf{DZ}$.

Proposition 4.4 (Symmetric Monoidal-Closed Structure)**.** *The category* $\mathbf{DZ}$ *is symmetric monoidal-closed. The monoidal unit is* $\mathbf{I} = (\mathbf{1}, \mathbf{1})$. *Given games* A *and* B, *the monoidal product* $A \boxdot B$ *and the internal hom* $A \boxdot\!\!\rightarrow B$ *are given by*

$$
\begin{array}{rrcl}
 & A \boxdot B & := & (A_{\mathsf{P}} \times B_{\mathsf{P}}, A_{\mathsf{O}} \times B_{\mathsf{O}}) \\
\textit{and} & A \boxdot\!\!\rightarrow B & := & (B_{\mathsf{P}}^{A_{\mathsf{P}}} \times A_{\mathsf{O}}^{A_{\mathsf{P}} \times B_{\mathsf{O}}}, A_{\mathsf{P}} \times B_{\mathsf{O}})
\end{array}
$$

Monoids and Comonoids. A commutative monoid in a symmetric monoidal category (SMC) $(\mathbb{C}, \otimes, \mathbf{I})$ is an object M equipped with structure maps $m : M \otimes M \to M$ and $u : \mathbf{I} \to M$ subject to coherence conditions (see e.g. [19]). A (commutative) comonoid in $\mathbb{C}$ is a (commutative) monoid in $\mathbb{C}^{\mathrm{op}}$. In this paper, by (co)monoid we always mean *commutative* (co)monoid. The category of (co)monoids in $(\mathbb{C}, \otimes, \mathbf{I})$ has (co)monoids as objects and maps are $\mathbb{C}$-maps which commute with the (co)monoid structure. We write $\mathbf{Comon}(\mathbb{C})$ for the category of comonoids of $(\mathbb{C}, \otimes, \mathbf{I})$.

Games A with $A_{\mathsf{P}} \simeq \mathbf{1}$ ($A_{\mathsf{O}} \simeq \mathbf{1}$) induce (co)monoids in $(\mathbf{DZ}, \boxdot, \mathbf{I})$. The following is well-known (see [19, §6.5]).

[12] $\mathbf{DZ}$-maps are actually total zig-zag strategies in the *simple game* $A \multimap B$ with $\multimap$ negative and A, B positive ([11], see also [19]).

$$\frac{}{\varphi \vdash \varphi} \qquad \frac{\overline{\varphi} \vdash \gamma, \overline{\varphi}' \qquad \overline{\psi}, \gamma \vdash \overline{\psi}'}{\overline{\varphi}, \overline{\psi} \vdash \overline{\varphi}', \overline{\psi}'} \qquad \frac{\overline{\varphi}, \varphi, \psi, \overline{\psi} \vdash \overline{\varphi}'}{\overline{\varphi}, \psi, \varphi, \overline{\psi} \vdash \overline{\varphi}'} \qquad \frac{\overline{\varphi} \vdash \overline{\varphi}', \varphi, \psi, \overline{\psi}'}{\overline{\varphi} \vdash \overline{\varphi}', \psi, \varphi, \overline{\psi}'}$$

$$\frac{}{\overline{\varphi} \vdash \top, \overline{\varphi}'} \qquad \frac{\overline{\psi} \vdash \overline{\psi}'}{\overline{\psi}, \varphi \vdash \overline{\psi}'} \qquad \frac{\overline{\psi}, \varphi^+, \varphi^+ \vdash \overline{\psi}'}{\overline{\psi}, \varphi^+ \vdash \overline{\psi}'} \qquad \frac{\overline{\varphi}, \varphi^- \vdash \overline{\varphi}'}{\overline{\varphi}, !(\varphi^-) \vdash \overline{\varphi}'} \qquad \frac{\overline{\varphi}^+ \vdash \varphi^-, \overline{\psi}^-}{\overline{\varphi}^+ \vdash !(\varphi^-), \overline{\psi}^-}$$

$$\frac{}{\overline{\varphi}, \bot \vdash \overline{\varphi}'} \qquad \frac{\overline{\psi} \vdash \overline{\psi}'}{\overline{\psi} \vdash \varphi, \overline{\psi}'} \qquad \frac{\overline{\psi} \vdash \varphi^-, \varphi^-, \overline{\psi}'}{\overline{\psi} \vdash \varphi^-, \overline{\psi}'} \qquad \frac{\overline{\varphi} \vdash \varphi^+, \overline{\varphi}}{\overline{\varphi} \vdash ?(\varphi^+), \overline{\varphi}} \qquad \frac{\overline{\varphi}^+, \varphi^+ \vdash \overline{\psi}^-}{\overline{\varphi}^+, ?(\varphi^+) \vdash \overline{\psi}^-}$$

$$\frac{\overline{\varphi}, \varphi_0, \varphi_1 \vdash \overline{\varphi}'}{\overline{\varphi}, \varphi_0 \otimes \varphi_1 \vdash \overline{\varphi}'} \qquad \frac{\overline{\varphi} \vdash \varphi, \overline{\varphi}' \qquad \overline{\psi} \vdash \psi, \overline{\psi}'}{\overline{\varphi}, \overline{\psi} \vdash \varphi \otimes \psi, \overline{\varphi}', \overline{\psi}'} \qquad \frac{\overline{\varphi}, \varphi \vdash \overline{\varphi}'}{\overline{\varphi}, \exists Z.\varphi \vdash \overline{\varphi}'} \qquad \frac{\overline{\varphi} \vdash \varphi[t/X], \overline{\varphi}'}{\overline{\varphi} \vdash \exists X.\varphi, \overline{\varphi}'} \qquad \frac{\overline{\varphi}, \varphi \vdash \psi}{\overline{\varphi} \vdash \varphi \multimap \psi}$$

$$\frac{\overline{\varphi}, \varphi \vdash \overline{\varphi}' \qquad \overline{\psi}, \psi \vdash \overline{\psi}'}{\overline{\varphi}, \overline{\psi}, \varphi \parr \psi \vdash \overline{\varphi}', \overline{\psi}'} \qquad \frac{\overline{\varphi} \vdash \varphi_0, \varphi_1, \overline{\varphi}'}{\overline{\varphi} \vdash \varphi_0 \parr \varphi_1, \overline{\varphi}'} \qquad \frac{\overline{\varphi}, \varphi[t/X] \vdash \overline{\varphi}'}{\overline{\varphi}, \forall X.\varphi \vdash \overline{\varphi}'} \qquad \frac{\overline{\varphi} \vdash \varphi}{\overline{\varphi} \vdash \forall Z.\varphi} \qquad \frac{\overline{\varphi} \vdash \varphi, \overline{\varphi}' \qquad \overline{\psi}, \psi \vdash \overline{\psi}'}{\overline{\varphi}, \overline{\psi}, \varphi \multimap \psi \vdash \overline{\varphi}', \overline{\psi}'}$$

Figure 4. The Deduction Rules of LMSO (where Z is fresh for $\overline{\varphi}, \overline{\varphi}'$ in each rule mentioning it).

Proposition 4.5. *The category of comonoids (resp. monoids) of an SMC $(\mathbb{C}, \otimes, \mathbf{I})$ has finite products (resp. coproducts) induced by $(\otimes, \mathbf{I})$.*

Given an alphabet Σ, we write Σ for the $\mathbf{DZ}$-object $(\Sigma, \mathbf{1})$.

Proposition 4.6. $\mathbf{S}$ *is equivalent to the full subcategory of* $\mathbf{Comon}(\mathbf{DZ})$ *whose objects are induced by alphabets.*

Comonoid Indexing. Given a comonoid K in an SMC $(\mathbb{C}, \otimes, \mathbf{I})$, the functor $A \mapsto K \otimes A$ of *comonoid indexing with K* is equipped with a comonad structure whose co-Kleisli category $\mathbf{Kl}(K)$ is an SMC for $\mathbf{I}_{\mathbf{Kl}(K)} = \mathbf{I}$ and (on objects) $A \otimes_{\mathbf{Kl}(K)} B = A \otimes B$ [13, Prop. 5].[13] Note that each $\mathbf{Comon}(\mathbb{C})$-map $f : K \to L$ induces a functor $f^\star : \mathbf{Kl}(L) \to \mathbf{Kl}(K)$. This extends to a functor $(-)^\star : \mathbf{Comon}(\mathbb{C})^{\mathrm{op}} \to \mathbf{Cat}$ taking K to $\mathbf{Kl}(K)$.

We let $\mathbf{DZ}(\Sigma)$ be the co-Kleisli category of comonoid indexing with Σ. It has the same objects as $\mathbf{DZ}$, and its maps from A to B are the $\mathbf{DZ}$-maps $\Sigma \boxdot A \to_{\mathbf{DZ}} B$. The following is [25, Prop. 5.2 & 5.3] (see also [19, §6.10]).

Proposition 4.7. $\mathbf{DZ}(\Sigma)$ *is symmetric monoidal-closed for $A \boxdot_{\mathbf{DZ}(\Sigma)} B := A \boxdot B$ (with unit $\mathbf{I}$) and internal hom $A \boxdot\!\!\to_{\mathbf{DZ}(\Sigma)} B := (A \boxdot\!\!\to B)$.*

We write $(-)^\uparrow$ for the canonical functor $\mathbf{DZ}(\Sigma) \to \mathbf{DZ}$ taking $\sigma : A \to_{\mathbf{DZ}(\Sigma)} B$ to $\sigma^\uparrow : \Sigma \boxdot A \to_{\mathbf{DZ}} \Sigma \boxdot B$.

4.2 Games with Winning

As usual, acceptance will be defined using games equipped with *winning conditions.*

Definition 4.8. A *game with winning* is a game A equipped with a *winning condition* $\mathcal{W}_A \subseteq (A_{\mathsf{P}} \times A_{\mathsf{O}})^\omega$.

Definition 4.9. Consider games with winning $(A, \mathcal{W}_A)$ and $(B, \mathcal{W}_B)$, and a strategy $\sigma = (f, F) : A \to_{\mathbf{DZ}} B$.

Given sequences $(a_n^{\mathsf{P}})_n \in A_{\mathsf{P}}^\omega$ and $(b_n^{\mathsf{O}})_n \in B_{\mathsf{O}}^\omega$, the strategy σ induces sequences $(b_n^{\mathsf{P}})_n \in B_{\mathsf{P}}^\omega$ and $(a_n^{\mathsf{O}})_n \in A_{\mathsf{O}}^\omega$ defined as

$$\begin{array}{rrcl} & b_n^{\mathsf{P}} & := & f(a_0^{\mathsf{P}} \cdots a_n^{\mathsf{P}}\,,\, b_0^{\mathsf{O}} \cdots b_{n-1}^{\mathsf{P}}) \in B_{\mathsf{P}} \\ \text{and} & a_n^{\mathsf{O}} & := & F(a_0^{\mathsf{P}} \cdots a_n^{\mathsf{P}}\,,\, b_0^{\mathsf{O}} \cdots b_n^{\mathsf{P}}) \in A_{\mathsf{O}} \end{array}$$

Then σ is *winning* if for all $(a_n^{\mathsf{P}})_n \in A_{\mathsf{P}}^\omega$ and all $(b_n^{\mathsf{O}})_n \in B_{\mathsf{O}}^\omega$, we have $(b_n^{\mathsf{P}}\,,\, b_n^{\mathsf{O}})_n \in \mathcal{W}_B$ whenever $(a_n^{\mathsf{P}}\,,\, a_n^{\mathsf{O}})_n \in \mathcal{W}_A$.

Proposition 4.10. *Games and winning strategies form a category* $\mathbf{DZ}^{\mathrm{W}}$.

The SMC structure $(\boxdot, \mathbf{I}, \boxdot\!\!\to)$ of $\mathbf{DZ}$ lifts to $\mathbf{DZ}^{\mathrm{W}}$, inducing a model of IMLL.

Proposition 4.11 (Symmetric Monoidal-Closed Structure)**.** *The category $\mathbf{DZ}^{\mathrm{W}}$ is symmetric monoidal-closed. The unit is $\top := (\mathbf{I}, \mathbf{1}^\omega)$. Given $(A, \mathcal{W}_A)$ and $(B, \mathcal{W}_B)$, the* tensor product *$(A, \mathcal{W}_A) \otimes (B, \mathcal{W}_B)$ and the* linear arrow *$(A, \mathcal{W}_A) \multimap (B, \mathcal{W}_B)$ are given by*

$$\begin{array}{rrcl} & (A, \mathcal{W}_A) \otimes (B, \mathcal{W}_B) & := & (A \boxdot B, \mathcal{W}_{A \otimes B}) \\ \textit{and} & (A, \mathcal{W}_A) \multimap (B, \mathcal{W}_B) & := & (A \boxdot\!\!\to B, \mathcal{W}_{A \multimap B}) \end{array}$$

$$\begin{array}{rl} \textit{where} & (a_n^{\mathsf{P}}, b_n^{\mathsf{P}}, a_n^{\mathsf{O}}, b_n^{\mathsf{O}})_n \in \mathcal{W}_{A \otimes B} \quad \textit{iff} \\ & \qquad \big((a_n^{\mathsf{P}}, a_n^{\mathsf{O}})_n \in \mathcal{W}_A \quad \textit{and} \quad (b_n^{\mathsf{P}}, b_n^{\mathsf{O}})_n \in \mathcal{W}_B\big) \end{array}$$

$$\begin{array}{rl} \textit{and} & (f_n, F_n, a_n^{\mathsf{P}}, b_n^{\mathsf{O}})_n \in \mathcal{W}_{A \multimap B} \quad \textit{iff} \\ & (a_n^{\mathsf{P}}\,,\, F(a_n^{\mathsf{P}}, b_n^{\mathsf{O}}))_n \in \mathcal{W}_A \quad \Longrightarrow \quad (f_n(a_n^{\mathsf{P}})\,,\, b_n^{\mathsf{O}})_n \in \mathcal{W}_B \end{array}$$

Winning also induces a disjunctive symmetric monoidal structure on $\mathbf{DZ}^{\mathrm{W}}$.

Proposition 4.12 (Multiplicative Disjunction)**.** *The category $\mathbf{DZ}^{\mathrm{W}}$ is symmetric monoidal with unit $\bot := (\mathbf{I}, \emptyset)$ and*

$$(A, \mathcal{W}_A) \parr (B, \mathcal{W}_B) \;:=\; (A \boxdot B, \mathcal{W}_{A \parr B})$$

where $(a_n^{\mathsf{P}}, b_n^{\mathsf{P}}, a_n^{\mathsf{O}}, b_n^{\mathsf{O}})_n \in \mathcal{W}_{A \parr B}$ iff either $(a_n^{\mathsf{P}}, a_n^{\mathsf{O}})_n \in \mathcal{W}_A$ or $(b_n^{\mathsf{P}}, b_n^{\mathsf{O}})_n \in \mathcal{W}_B$.

We write A for $(A, \mathcal{W}_A)$ when no confusion arises. Note that since sets of moves are assumed to be non-empty, $\mathbf{DZ}^{\mathrm{W}}$ is equipped with (non-canonical) *weakening maps* $A \to_{\mathbf{DZ}^{\mathrm{W}}} \top$ and $\bot \to_{\mathbf{DZ}^{\mathrm{W}}} A$.

4.3 Uniform Automata

The adequacy of realizability will be proved using the notion of *uniform automata* (UA's), adapted from [25]. UA's are essentially usual alternating automata, but in which alternation is expressed via an explicitly given set of *moves*.

[13] See also [25, Prop. 4.4] and [19, §6.5 & 6.6].

Definition 4.13. A *uniform automaton* (UA) $\mathcal{A}$ over Σ (notation $\mathcal{A} : \Sigma$) has the form

$$\mathcal{A} \quad = \quad (\mathcal{A}_{\mathsf{P}}\,,\, \mathcal{A}_{\mathsf{O}}\,,\, Q_{\mathcal{A}}\,,\, q^{\imath}_{\mathcal{A}}\,,\, \partial_{\mathcal{A}}\,,\, \Omega_{\mathcal{A}})$$

where $\mathcal{A}_{\mathsf{P}}$ (resp. $\mathcal{A}_{\mathsf{O}}$) is the alphabet of P (resp. O) moves, $Q_{\mathcal{A}}$ is the alphabet *states* (with $q^{\imath}_{\mathcal{A}} \in Q_{\mathcal{A}}$ *initial*), the transition function $\partial_{\mathcal{A}}$ has the form

$$\partial_{\mathcal{A}} \;:\; Q_{\mathcal{A}} \times \Sigma \;\longrightarrow\; \mathcal{A}_{\mathsf{P}} \times \mathcal{A}_{\mathsf{O}} \;\longrightarrow\; Q_{\mathcal{A}}$$

and the *acceptance condition* $\Omega_{\mathcal{A}}$ is an ω-regular subset of $Q^{\omega}_{\mathcal{A}}$. $\mathcal{A}$ is called *positive* (or, in accordance with automata-theoretic terminology, *non-deterministic*) if $\mathcal{A}_{\mathsf{O}} \simeq \mathbf{1}$, *negative* (or, in accordance with automata-theoretic terminology, *universal*) if $\mathcal{A}_{\mathsf{P}} \simeq \mathbf{1}$ and *deterministic* if it is both positive and negative.

The interpretation of PLMSO formulae will respect polarities.

An automaton $\mathcal{A} : \Gamma$ together with a causal function $F \in \mathbf{S}[\Sigma, \Gamma]$ induce a *substituted acceptance game* $\mathcal{A}(F) : \Sigma$ with P-moves $\Sigma \times \mathcal{A}_{\mathsf{P}}$ and O-moves $\mathcal{A}_{\mathsf{O}}$. We equip $\mathcal{A}(F)$ with the winning condition consisting of the ω-sequences $((\mathsf{b}_n, a^{\mathsf{P}}_n), a^{\mathsf{O}}_n)_n \in ((\Sigma \times \mathcal{A}_{\mathsf{P}}) \times \mathcal{A}_{\mathsf{O}})^{\omega}$ such that $(q_n)_n \in \Omega_{\mathcal{A}}$ where $q_0 := q^{\imath}_{\mathcal{A}}$ and $q_{n+1} := \partial_{\mathcal{A}}(q_n, F(\mathsf{b}_0, \dots, \mathsf{b}_n), a^{\mathsf{P}}_n, a^{\mathsf{O}}_n)$.

The substituted acceptance game $\mathcal{A}(F) : \Sigma$ is an object of $\mathbf{DZ}^{\mathrm{W}}$. Acceptance of $B \in \Sigma^{\omega} \simeq \mathbf{S}[\mathbf{1}, \Sigma]$ by $\mathcal{A} : \Sigma$ is defined via the game $\mathcal{A}(B) : \mathbf{1}$.

Definition 4.14. An automaton $\mathcal{A} : \Sigma$ *accepts* the ω-word $B \in \Sigma^{\omega}$ if there is a winning strategy $\sigma \in \mathbf{DZ}^{\mathrm{W}}[\top, \mathcal{A}(B)]$. We let $\mathcal{L}(\mathcal{A})$ be the set of ω-words accepted by $\mathcal{A}$.

4.4 Some Structure on Automata

Our goal here is to display some structure on automata that will produce a realizability model of LMSO.

Definition 4.15. The category DA_{Σ} has games of the form $\mathcal{A}(F) : \Sigma$ as objects. Maps from $\mathcal{A}(F)$ to $\mathcal{B}(G)$ are $\mathbf{DZ}(\Sigma)$-maps σ from $(\mathcal{A}_{\mathsf{P}}, \mathcal{A}_{\mathsf{O}})$ to $(\mathcal{B}_{\mathsf{P}}, \mathcal{B}_{\mathsf{O}})$, which lift to winning strategies $\sigma^{\uparrow} : \mathcal{A}(F) \to_{\mathbf{DZ}^{\mathrm{W}}} \mathcal{B}(G)$.

Aut_{Σ} is the full subcategory of DA_{Σ} whose objects are of the form $\mathcal{A}(\mathrm{Id}_{\Sigma}) : \Sigma$ (denoted $\mathcal{A} : \Sigma$) where Id_{Σ} is the $\mathbf{S}$-identity on Σ. We let $\mathsf{Aut}^{+}_{\Sigma}$ (resp. $\mathsf{Aut}^{-}_{\Sigma}$, $\mathsf{Aut}^{\pm}_{\Sigma}$) be the restriction of Aut_{Σ} to positive (resp. negative, deterministic) automata.

We write $\sigma : \mathcal{A}(F) \multimap \mathcal{B}(G)$ when σ is a DA_{Σ}-map from $\mathcal{A}(F)$ to $\mathcal{B}(G)$. Note that substituted acceptance games $\mathcal{A}(F) : \Sigma$ are $\mathbf{DZ}^{\mathrm{W}}$-objects with P-moves $\Sigma \times \mathcal{A}_{\mathsf{P}}$ and O-moves $\mathcal{A}_{\mathsf{O}}$, while a DA_{Σ}-map from $\mathcal{A}(F) \multimap \mathcal{B}(G)$ is in fact a $\mathbf{DZ}$-map $\Sigma \boxdot (\mathcal{A}_{\mathsf{P}}, \mathcal{A}_{\mathsf{O}}) \to_{\mathbf{DZ}} (\mathcal{B}_{\mathsf{P}}, \mathcal{B}_{\mathsf{O}})$.

Substitution. The categories DA_{Σ} are indexed over $\mathbf{S}$, while the categories Aut_{Σ} are indexed over $\mathbf{M}$. An immediate consequence is that the realizability arrow $\multimap$ is correct w.r.t. language inclusion.

Proposition 4.16. *Given $\mathcal{A}, \mathcal{B} : \Sigma$, if there is an Aut_{Σ}-map $\mathcal{A} \multimap \mathcal{B}$, then $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$.*

Notice that a crucial point in the game $\mathcal{A} \multimap \mathcal{B}$ is that the word w is progressively given by the plays of O; if it were a data given before the first round, the previous proposition would be an equivalence.

Monoidal-Closed Structure. The symmetric monoidal-closed structures of $\mathbf{DZ}(\Sigma)$ and $\mathbf{DZ}^{\mathrm{W}}$ lift to each category Aut_{Σ}.

The unit automaton $\top$ has P-moves, O-moves and states $\mathbf{1} \simeq \{\bullet\}$ (with thus $\bullet$ initial) and acceptance condition $\mathbf{1}^{\omega}$. Given $\mathcal{A}, \mathcal{B} : \Sigma$, the monoidal product $(\mathcal{A} \otimes \mathcal{B}) : \Sigma$ and the linear arrow $(\mathcal{A} \multimap \mathcal{B}) : \Sigma$ both have states sets $Q_{\mathcal{A}} \times Q_{\mathcal{B}}$ with $(q^{\imath}_{\mathcal{A}}, q^{\imath}_{\mathcal{B}})$ initial. Their moves are given (via Prop. 4.7) by Prop. 4.4:

$$\begin{array}{rclrcl} (\mathcal{A} \otimes \mathcal{B})_{\mathsf{P}} & := & \mathcal{A}_{\mathsf{P}} \times \mathcal{B}_{\mathsf{P}} & (\mathcal{A} \otimes \mathcal{B})_{\mathsf{O}} & := & \mathcal{A}_{\mathsf{O}} \times \mathcal{B}_{\mathsf{O}} \\ (\mathcal{A} \multimap \mathcal{B})_{\mathsf{P}} & := & \mathcal{B}_{\mathsf{P}}^{\mathcal{A}_{\mathsf{P}}} \times \mathcal{A}_{\mathsf{O}}^{\mathcal{A}_{\mathsf{P}} \times \mathcal{B}_{\mathsf{O}}} & (\mathcal{A} \multimap \mathcal{B})_{\mathsf{O}} & := & \mathcal{A}_{\mathsf{P}} \times \mathcal{B}_{\mathsf{O}} \end{array}$$

For the transition functions, we let

$$\begin{array}{rcl} \partial_{\mathcal{A} \otimes \mathcal{B}}((q_{\mathcal{A}}, q_{\mathcal{B}}),\, \mathsf{a},\, (a^{\mathsf{P}}, b^{\mathsf{P}}), (a^{\mathsf{O}}, b^{\mathsf{O}})) & := & (q_0, q_1) \\ \partial_{\mathcal{A} \multimap \mathcal{B}}((q_{\mathcal{A}}, q_{\mathcal{B}}),\, \mathsf{a},\, (f, F), (a^{\mathsf{P}}, b^{\mathsf{O}})) & := & (q'_0, q'_1) \end{array}$$

where

$$\begin{array}{ll} q_0 := \partial_{\mathcal{A}}(q_{\mathcal{A}}, \mathsf{a}, a^{\mathsf{P}}, a^{\mathsf{O}}) & q_1 := \partial_{\mathcal{B}}(q_{\mathcal{B}}, \mathsf{a}, b^{\mathsf{P}}, b^{\mathsf{O}}) \\ q'_0 := \partial_{\mathcal{A}}(q_{\mathcal{A}}, \mathsf{a}, a^{\mathsf{P}}, F(a^{\mathsf{P}}, b^{\mathsf{O}})) & q'_1 := \partial_{\mathcal{B}}(q_{\mathcal{B}}, \mathsf{a}, f(a^{\mathsf{P}}), b^{\mathsf{O}}) \end{array}$$

Finally, we let $(q_n, q'_n)_n \in \Omega_{\mathcal{A} \otimes \mathcal{B}}$ iff $((q_n)_n \in \Omega_{\mathcal{A}}$ and $(q'_n)_n \in \Omega_{\mathcal{B}})$, and we let $(q_n, q'_n)_n \in \Omega_{\mathcal{A} \multimap \mathcal{B}}$ iff $((q_n)_n \in \Omega_{\mathcal{A}}$ implies $(q'_n)_n \in \Omega_{\mathcal{B}})$.

Proposition 4.17.

(a) $(\mathsf{Aut}_{\Sigma}, \otimes, \top, \multimap)$ *is symmetric monoidal closed.*
(b) We have $\mathcal{L}(\top : \Sigma) = \Sigma^{\omega}$ *and* $\mathcal{L}(\mathcal{A} \otimes \mathcal{B}) = \mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{B})$.
(c) By determinacy of ω-regular games, $B \in \mathcal{L}(\mathcal{A} \multimap \bot)$ iff $B \notin \mathcal{L}(\mathcal{A})$.

Multiplicative Disjunction. The multiplicative disjunction $(\parr, \bot)$ of $\mathbf{DZ}^{\mathrm{W}}$ induces a symmetric monoidal structure in each Aut_{Σ}. The *falsity automaton* $\bot$, is defined as $\top$ above, but with acceptance condition $\emptyset \subseteq \mathbf{1}^{\omega}$. Given $\mathcal{A}, \mathcal{B} : \Sigma$, the UA $\mathcal{A} \parr \mathcal{B}$ is defined as $\mathcal{A} \otimes \mathcal{B}$, but with $(q_n, q'_n)_n \in \Omega_{\mathcal{A} \parr \mathcal{B}}$ iff either $(q_n)_n \in \Omega_{\mathcal{A}}$ or $(q'_n)_n \in \Omega_{\mathcal{B}}$.

Proposition 4.18. $\mathcal{L}(\bot) = \emptyset$ *and* $\mathcal{L}(\mathcal{A} \parr \mathcal{B}) = \mathcal{L}(\mathcal{A}) \cup \mathcal{L}(\mathcal{B})$.

It follows from §4.1 that $(\otimes, \top)$ induces finite products in $\mathsf{Aut}^{+}_{\Sigma}$ and that $(\parr, \bot)$ induces finite co-products in $\mathsf{Aut}^{-}_{\Sigma}$

Quantifications. Quantifications in UA's can be seen as simple adaptations of quantifications in usual alternating automata. They are actually induced, via the generalization of simple fibrations to comonoid indexing (§4.1), by the categorical quantifiers in simple fibrations (see e.g. [14, Prop. 1.9.3]).[14] In this paper, we only need the following.

Definition 4.19. Given $\mathcal{A} : \Sigma \times \Gamma$, let

$$\begin{array}{rcl} (\exists_{\Gamma}\mathcal{A} : \Sigma) & := & (\Gamma \times \mathcal{A}_{\mathsf{P}}\,,\, \mathcal{A}_{\mathsf{O}}\,,\, Q_{\mathcal{A}}\,,\, q^{\imath}_{\mathcal{A}}\,,\, \partial_{\exists_{\Gamma}\mathcal{A}}\,,\, \Omega_{\mathcal{A}}) \\ (\forall_{\Gamma}\mathcal{A} : \Sigma) & := & ({\mathcal{A}_{\mathsf{P}}}^{\Gamma}\,,\, \Gamma \times \mathcal{A}_{\mathsf{O}}\,,\, Q_{\mathcal{A}}\,,\, q^{\imath}_{\mathcal{A}}\,,\, \partial_{\forall_{\Gamma}\mathcal{A}}\,,\, \Omega_{\mathcal{A}}) \end{array}$$

where

$$\begin{array}{rcl} \partial_{\exists_{\Gamma}\mathcal{A}}(q,\, \mathsf{a},\, (\mathsf{b}, a^{\mathsf{P}}),\, a^{\mathsf{O}}) & := & \partial_{\mathcal{A}}(q,\, (\mathsf{a}, \mathsf{b}),\, a^{\mathsf{P}},\, a^{\mathsf{O}}) \\ \partial_{\forall_{\Gamma}\mathcal{A}}(q,\, \mathsf{a},\, f,\, (\mathsf{b}, a^{\mathsf{O}})) & := & \partial_{\mathcal{A}}(q,\, (\mathsf{a}, \mathsf{b}),\, f(\mathsf{b}),\, a^{\mathsf{O}}) \end{array}$$

Proposition 4.20. *Given $\mathcal{A} : \Sigma \times \Gamma$ and $\mathcal{B}(C) : \Sigma$, we have*

$$\begin{array}{l} \mathsf{DA}_{\Sigma}[(\exists_{\Gamma}\mathcal{A})(B)\,,\, \mathcal{B}(C)] \quad \simeq \\ \qquad\qquad \mathsf{DA}_{\Sigma \times \Gamma}[\mathcal{A}(B \times \mathrm{Id}_{\Gamma})\,,\, \mathcal{B}(C \circ [\pi_{\Sigma}])] \\ \mathsf{DA}_{\Sigma}[\mathcal{B}(C)\,,\, (\forall_{\Gamma}\mathcal{A})(B)] \quad \simeq \\ \qquad\qquad \mathsf{DA}_{\Sigma \times \Gamma}[\mathcal{B}(B \circ [\pi_{\Sigma}])\,,\, \mathcal{A}(C \times \mathrm{Id}_{\Gamma})] \end{array}$$

[14] Which correspond to quantifications in Dialectica categories [6, 12].

Moreover, there are canonical maps $(\forall_\Gamma \mathcal{A})([\pi_\Sigma]) \multimap \mathcal{A}$ *and* $\mathcal{A} \multimap (\exists_\Gamma \mathcal{A})([\pi_\Sigma])$.

Proposition 4.21. *If* $\mathcal{A} : \Sigma \times \Gamma$ *is positive then* $B \in \mathcal{L}(\exists_\Gamma \mathcal{A})$ *iff there exists* $C \in \Gamma^\omega$ *such that* $\langle B, C\rangle \in \mathcal{L}(\mathcal{A})$.

If $\mathcal{A} : \Sigma \times \Gamma$ *is negative then* $B \in \mathcal{L}(\forall_\Gamma \mathcal{A})$ *iff for all* $C \in \Gamma^\omega$ *we have* $\langle B, C\rangle \in \mathcal{L}(\mathcal{A})$.

4.5 Polarities and Duality

Say that $\mathcal{A}$ is *polarized* if $\mathcal{A}$ is positive or negative. Polarized automata are particularly well-behaved. First, they are equipped with a direct complementation operation $(-)^\perp$, which is defined as

$$\mathcal{A}^\perp \quad := \quad (\mathcal{A}_{\mathsf{O}}\,,\, \mathcal{A}_{\mathsf{P}}\,,\, Q_{\mathcal{A}}\,,\, q^{\imath}_{\mathcal{A}}\,,\, \partial_{\mathcal{A}^\perp}\,,\, Q^\omega_{\mathcal{A}} \setminus \Omega_{\mathcal{A}})$$

where $\partial_{\mathcal{A}^\perp}(q, \mathsf{a}, a^{\mathsf{O}}, a^{\mathsf{P}}) := \partial_{\mathcal{A}}(q, \mathsf{a}, a^{\mathsf{P}}, a^{\mathsf{O}})$. For $\mathcal{A}, \mathcal{B} : \Sigma$ of opposite polarities, we have the following iso in Aut_Σ:

$$\mathcal{A} \multimap \mathcal{B} \quad \simeq \quad \mathcal{A}^\perp \parr \mathcal{B} \tag{2}$$

This in particular implies $\mathcal{A} \multimap \perp \simeq \mathcal{A}^\perp$, which has the following interesting consequence.

Corollary 4.22. *If* $\mathcal{A}$ *is polarized, then for all* B*, we have* $B \in \mathcal{L}(\mathcal{A}^\perp)$ *iff* $B \notin \mathcal{L}(\mathcal{A})$.

Second, det. automata are closed under $\otimes, \parr, (-)^\perp$, and positive and negative automata are closed under the following productions (where $\mathcal{A}^+$ is positive, $\mathcal{A}^-$ is negative and $\mathcal{D}$ is deterministic):

$$\begin{array}{rcl}
\mathcal{A}^+, \mathcal{B}^+ & ::= & \mathcal{D} \;\mid\; (\mathcal{A}^-)^\perp \;\mid\; \exists X.\mathcal{A}^+ \\
& \mid & \mathcal{A}^+ \otimes \mathcal{B}^+ \;\mid\; \mathcal{A}^+ \parr \mathcal{B}^+ \\
\mathcal{A}^-, \mathcal{B}^- & ::= & \mathcal{D} \;\mid\; (\mathcal{A}^+)^\perp \;\mid\; \forall X.\mathcal{A}^- \\
& \mid & \mathcal{A}^- \otimes \mathcal{B}^- \;\mid\; \mathcal{A}^- \parr \mathcal{B}^-
\end{array}$$

This, together with the fact that exponentials ? and ! always give det. automata (see Prop. 5.2 below), will imply that the interpretation of PLMSO-formulae preserves polarities.

Third, we have the following isos in Aut_Σ (where $\mathcal{A}$ and $\mathcal{B}$ have the same polarity):

$$(\mathcal{A}^\perp)^\perp \simeq \mathcal{A} \qquad (\exists_\Sigma \mathcal{A})^\perp \simeq \forall_\Sigma \mathcal{A}^\perp \qquad (\forall_\Sigma \mathcal{A})^\perp \simeq \exists_\Sigma \mathcal{A}^\perp$$
$$(\mathcal{A} \otimes \mathcal{B})^\perp \simeq \mathcal{A}^\perp \parr \mathcal{B}^\perp \qquad (\mathcal{A} \parr \mathcal{B})^\perp \simeq \mathcal{A}^\perp \otimes \mathcal{B}^\perp$$

5 The Realizability Model of LMSO

This last Section connects the two previous ones: we show that the categories $\mathsf{Aut}_{(-)}$ provide a realizability model of LMSO. This in particular implies the correctness of LMSO w.r.t. Church's synthesis.

5.1 The Realizability Interpretation of LMSO

We have seen in §4.4 that uniform automata are equipped with categorical structure for the interpretation of the connectives $\top, \otimes, \perp, \parr, \multimap, \forall, \exists$ of LMSO. In order to devise an interpretation of all LMSO-formulae, it remains to deal with the atoms $\alpha \in \mathrm{At}$, and with the exponential modalities $!(-)$ and $?(-)$. Atoms are dealt-with as usual (see e.g. [28]), and we rely on McNaughton's Thm. 2.10 for $!(-)$ and $?(-)$.

Proposition 5.1. *For each atom* $\alpha \in \mathrm{At}$ *with free variables among* $\overline{X} = X_1, \ldots, X_n$*, there is a deterministic automaton* $\mathcal{A}_\alpha$ *such that* $\overline{B} \in \mathcal{L}(\mathcal{A}_\alpha)$ *iff* $\mathfrak{N} \models \alpha(\overline{B})$.

Proposition 5.2. *Given a positive (resp. negative) automaton* $\mathcal{A} : \Sigma$*, there is a deterministic automaton* $?\mathcal{A} : \Sigma$ *(resp.* $!\mathcal{A} : \Sigma$*) which recognizes the same language as* $\mathcal{A}$.

Note that Prop. 5.2 produces deterministic automata, which therefore only have trivial realizers. Hence, no construction for McNaughton's Thm. 2.10 is actually required for the extraction of realizers from proofs.

Interpretation of* PLMSO*-Formulae. We are now ready to define the interpretation of LMSO-formulae as automata.

Definition 5.3. The automaton $[\![\varphi]\!] : \mathbf{2}^n$, for φ an LMSO-formula with free variables among $\overline{X} = X_1, \ldots, X_n$, is defined by induction on φ as follows:

$$\begin{array}{rclrcl}
[\![\top]\!] & := & \top & [\![\varphi \otimes \psi]\!] & := & [\![\varphi]\!] \otimes [\![\psi]\!] \\
[\![\perp]\!] & := & \perp & [\![\varphi \parr \psi]\!] & := & [\![\varphi]\!] \parr [\![\psi]\!] \\
[\![\alpha]\!] & := & \mathcal{A}_\alpha & [\![\varphi \multimap \psi]\!] & := & [\![\varphi]\!] \multimap [\![\psi]\!] \\
[\![\exists X.\varphi]\!] & := & \exists_{\mathbf{2}}[\![\varphi]\!] & [\![\forall X.\varphi]\!] & := & \forall_{\mathbf{2}}[\![\varphi]\!] \\
[\![?\varphi^+]\!] & := & ?[\![\varphi^+]\!] & [\![!\varphi^-]\!] & := & ![\![\varphi^-]\!]
\end{array}$$

This definition requires some comments. First, it follows from §4.5 that $[\![-]\!]$ respects polarities of PLMSO-formulae, so that $[\![?\varphi^+]\!]$ and $[\![!\varphi^-]\!]$ are well-defined. Also, in the cases of $\otimes, \parr, \multimap$ we assume that both compound formulae have free variables among $X_1, \ldots, X_n$. For $\forall X.\varphi$ and $\exists X.\varphi$, we assume that φ has free variables among $X, X_1, \ldots, X_n$.

5.2 Correctness of PLMSO w.r.t. $\mathfrak{N}$

Since $[\![-]\!]$ respects polarities of PLMSO-formulae, it follows from the properties of §4.4 that for PLMSO-formulae, the quantifiers $\exists$ and $\forall$ are correctly implemented by the corresponding operations on automata. As a consequence, the interpretation $[\![-]\!]$ of PLMSO-formulae is correct w.r.t. the following erasure map.

Definition 5.4. The *erasure map* $\lfloor\cdot\rfloor$ is defined by induction on LMSO-formulae as follows:

$$\begin{array}{rclrcl}
\lfloor\top\rfloor & := & \top & \lfloor\varphi \otimes \psi\rfloor & := & \lfloor\varphi\rfloor \wedge \lfloor\psi\rfloor \\
\lfloor\perp\rfloor & := & \perp & \lfloor\varphi \parr \psi\rfloor & := & \lfloor\varphi\rfloor \vee \lfloor\psi\rfloor \\
\lfloor\alpha\rfloor & := & \alpha & \lfloor\varphi \multimap \psi\rfloor & := & \lfloor\varphi\rfloor \to \lfloor\psi\rfloor \\
\lfloor\exists X.\varphi\rfloor & := & \exists X.\lfloor\varphi\rfloor & \lfloor\forall X.\varphi\rfloor & := & \forall X.\lfloor\varphi\rfloor \\
\lfloor!\varphi\rfloor & := & \lfloor\varphi\rfloor & \lfloor?\varphi\rfloor & := & \lfloor\varphi\rfloor
\end{array}$$

Proposition 5.5. *For a* PLMSO*-formula* φ, $\overline{B} \in \mathcal{L}([\![\varphi]\!])$ *if and only if* $\mathfrak{N} \models \lfloor\varphi\rfloor(\overline{B})$.

Proposition 5.6. $[\![\varphi^+]\!] \multimap [\![\psi^-]\!]$ *is realized if and only if* $\mathfrak{N} \models \lfloor\varphi^+\rfloor \to \lfloor\psi^-\rfloor$.

5.3 Adequacy and Realized Axioms

The central result on $[\![-]\!]$ is its *adequacy*:

Theorem 5.7. *Given* LMSO*-formulae* $\overline{\varphi} = \varphi_1, \ldots, \varphi_n$ *and* $\overline{\psi} = \psi_1, \ldots, \psi_m$*, from a derivation of* $\overline{\varphi} \vdash_{\mathsf{LMSO}} \overline{\psi}$*, one can extract a f.s. realizer of* $[\![\varphi_1]\!] \otimes \ldots \otimes [\![\varphi_n]\!] \longrightarrow\!\!\circ\; [\![\psi_1]\!] \parr \cdots \parr [\![\psi_n]\!]$.

In the statement of Thm. 5.7, the interpretation $[\![-]\!]$ is taken for $\overline{X}$ containing all the free variables of $\overline{\varphi}$ and $\overline{\psi}$.

An immediate consequence of Thm. 5.7 is that from a proof of $\mathsf{LMSO} \vdash \forall\overline{Y}.\exists\overline{Z}.\varphi(\overline{Y};\overline{Z})$ one can extract Mealy machines $\overline{\mathcal{M}}$ together with f.s. realizer of $\top \longrightarrow\!\!\circ\; [\![\varphi(\overline{Y}; \overline{t_{\mathcal{M}}(\overline{Y})})]\!]$. When φ is a PLMSO formula, by Prop. 5.5 this gives a f.s. causal realizer of $\varphi^L(\overline{Y};\overline{Z})$.

Corollary 5.8. *Given a* PLMSO*-formula* $\varphi(\overline{Y};\overline{Z})$*, from a proof of* LMSO $\vdash \forall\overline{Y}.\exists\overline{Z}.\varphi(\overline{Y};\overline{Z})$ *one can extract a f.s. causal realizer of* $\varphi^L(\overline{Y};\overline{Z})$*.*

A general feature of realizability models is that they may realize more statements than those provable in the theory they interpret (see e.g. [15]). Here are some examples. Examples 5.9 and 5.10 are counterparts of usual additional axioms for intuitionistic arithmetic (see e.g. [15]).

Example 5.9. The following versions of "independence of premises" and Markov's principle are realizable (where φ is negative, δ, δ' are deterministic and X is not free in φ, δ'):

$$\begin{array}{rcl}(\varphi \multimap \exists X.\psi) & \multimap & \exists X(\varphi \multimap \psi)\\ {[(\forall X.\delta) \multimap \delta']} & \multimap & \exists X(\delta \multimap \delta')\end{array}$$

Example 5.10. The following form of *causal (functional) choice* is realizable, for each negative formula φ:

$$\forall\overline{X}.\exists Y.\varphi(\overline{X},Y) \quad \multimap \quad \exists\overline{Z}.\forall\overline{X}.\varphi(\overline{X},\mathrm{app}_{\mathbf{2}^n,\mathbf{2}}(\overline{Z},\overline{X}))$$

where $\overline{X} = X_1,\ldots,X_n$ (representing $\mathbf{2}^n$), $\overline{Z} = Z_1,\ldots,Z_{2^n}$ (representing $(\mathbf{2} \to \mathbf{2}^n)$) and $\mathrm{app}_{\Sigma,\Gamma}$ is a term for the Mealy machine $\Gamma^\Sigma \times \Sigma \to \Gamma$ computing pointwise application.

Example 5.11. LMSO has an axiom for the elimination of double negation for PLMSO-formulae. One can actually realize $[(\mathcal{A} \multimap \bot) \multimap \bot] \multimap \mathcal{A}$ for any $\mathcal{A}$ by a non-trivial combinatorial argument.[15]

6 Conclusion

We introduced LMSO, a constructive linear proof system for MSO on ω-words. LMSO is complete w.r.t. the translation $(-)^L$ for both MSO and Church's synthesis. We devised an automata-based realizability semantics for LMSO. This model implies that LMSO is correct for Church's synthesis. It also validates some extra axioms, counterparts of usual additional axioms to intuitionistic arithmetic.

Further Works. LMSO has a polarized subsystem PLMSO which reflects the usual polarities of alternating ω-word automata and the corresponding polarized fragments of MSO.

We plan in future work to build on this feature, in particular because it corresponds to the polarities underlying the Safraless approaches of [7, 16]. Actually, universal automata are reminiscent from Gödel's Functional (*Dialectica*) interpretation (see e.g. [15]).[16] The Dialectica interpretation maps formulae to $\exists\forall$-formulae, in such a way that proofs induce witnesses for the prenex existential. This format is very close to the one obtained by reformulating the approach of [7] in MSO^+. We plan to investigate this in further works.

Acknowledgments

This work was partially supported by the ANR-14-CE25-0007-RAPIDO and by the LABEX MILYON (ANR-10-LABX-0070) of Université de Lyon.

[15] But for cardinality reasons, in general this can not be an iso. In particular, Aut_Σ and $\mathbf{DZ}$ are not $*$-autonomous categories.

[16] Besides, Markov's principle (Ex. 5.9) is validated by Dialectica, while it is usually *not* validated by typed realizability models.

References

[1] C. Baier and J.-P. Katoen. 2008. *Principles of Model Checking.* MIT Press.

[2] R. Bloem, B. Jobstmann, N. Piterman, A. Pnueli, and Y. Sa'ar. 2012. Synthesis of reactive (1) designs. *J. Comput. System Sci.* 78, 3 (2012), 911–938.

[3] J. R. Büchi. 1962. On a Decision Methond in Restricted Second-Order Arithmetic. In *Logic, Methodology and Philosophy of Science (Proc. 1960 Intern. Congr.)*, E. Nagel et al. (Ed.). Stanford Univ. Press, 1–11.

[4] J. R. Büchi and L. H. Landweber. 1969. Solving Sequential Conditions by Finite-State Strategies. *Trans. Amer. Math. Soc.* 138 (1969), 367–378.

[5] A. Church. 1957. Applications of recursive arithmetic to the problem of circuit synthesis. In *Summaries of the SISL*, Vol. 1. Cornell Univ., 3–50.

[6] V. de Paiva. 1991. *The Dialectica categories.* Technical Report 213. University of Cambridge Computer Laboratory.

[7] E. Filiot, N. Jin, and J.-F. Raskin. 2011. Antichains and compositional algorithms for LTL synthesis. *Form. Method. Syst. Des.* 39, 3 (Dec 2011), 261–296.

[8] O. Friedmann and M. Lange. 2012. Ramsey-Based Analysis of Parity Automata. In *Proceedings of TACAS 2012 (LNCS)*, Vol. 7214. Springer, 64–78.

[9] J.-Y. Girard. 1987. Linear Logic. *Theor. Comput. Sci.* 50 (1987), 1–102.

[10] E. Grädel, W. Thomas, and T. Wilke (Eds.). 2002. *Automata, Logics, and Infinite Games: A Guide to Current Research.* LNCS, Vol. 2500. Springer.

[11] J. M. E. Hyland. 1997. Game Semantics. In *Semantics and Logics of Computation*, A. M. Pitts and P. Dybjer (Eds.). Publications of the Newton Institute, Vol. 14. CUP, 131.

[12] J. M. E. Hyland. 2002. Proof theory in the abstract. *Ann. Pure Appl. Logic* 114, 1-3 (2002), 43–78.

[13] J. M. E. Hyland and A. Schalk. 2003. Glueing and orthogonality for models of linear logic. *Theor. Comput. Sci.* 294, 1/2 (2003), 183–231.

[14] B. Jacobs. 2001. *Categorical Logic and Type Theory.* Elsevier.

[15] U. Kohlenbach. 2008. *Applied Proof Theory: Proof Interpretations and Their Use in Mathematics.* Springer.

[16] O. Kupferman, N. Piterman, and Y. Vardi, M. 2006. Safraless Compositional Synthesis. In *Proceedings of CAV'06*, T. Ball and R. B. Jones (Eds.). Springer, 31–44.

[17] O. Laurent. 2005. Classical isomorphisms of types. *Math. Struct. Comput. Sci.* 15, 5 (2005), 969–1004.

[18] R. McNaughton. 1966. Testing and generating infinite sequences by a finite automaton. *Inf. Control* 9, 5 (1966), 521 – 530.

[19] P.-A. Melliès. 2009. Categorical semantics of linear logic. In *Interactive models of computation and program behaviour.* Panoramas et Synthèses, Vol. 27. SMF.

[20] D. E. Muller and P. E. Schupp. 1987. Alternating Automata on Infinite Trees. *Theor. Comput. Sci.* 54 (1987), 267–276.

[21] D. E. Muller and P. E Schupp. 1995. Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of the Theorems of Rabin, McNaughton and Safra. *Theor. Comput. Sci.* 141, 1&2 (1995), 69–107.

[22] P. Pradic and C. Riba. 2017. A Curry-Howard Approach to Church's Synthesis. In *Proceedings ot FSCD'17 (LIPIcs)*, Vol. 84. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 30:1–30:16.

[23] M. O. Rabin. 1972. *Automata on Infinite Objects and Church's Problem.* Amer. Math. Soc.

[24] C. Riba. 2012. A Model Theoretic Proof of Completeness of an Axiomatization of Monadic Second-Order Logic on Infinite Words. In *Proceedings of IFIP-TCS'12.*

[25] C. Riba. 2017. Monoidal-Closed Categories of Tree Automata. (2017). Submitted. Available on HAL (hal-01261183), https://hal.archives-ouvertes.fr/hal-01261183.

[26] D. Siefkes. 1970. *Decidable Theories I : Büchi's Monadic Second Order Successor Arithmetic.* LNM, Vol. 120. Springer.

[27] M. H. Sørensen and P. Urzyczyn. 2006. *Lectures on the Curry-Howard Isomorphism.* Studies in Logic and the Foundations of Mathematics, Vol. 149. Elsevier Science Inc.

[28] W. Thomas. 1997. Languages, Automata, and Logic. In *Handbook of Formal Languages*, G. Rozenberg and A. Salomaa (Eds.). Vol. III. Springer, 389–455.

[29] W. Thomas. 2008. Solution of Church's Problem: A tutorial. *New Perspectives on Games and Interaction* 5 (2008), 23.

[30] W. Thomas. 2009. Facets of Synthesis: Revisiting Church's Problem. In *Proceedings of FOSSACS'09*, L. de Alfaro (Ed.). Springer, 1–14.