Adjunct Elimination Through Games in Static Ambient Logic (*Extended Abstract*) ⋆

Anuj Dawar[1], Philippa Gardner[2], and Giorgio Ghelli[3]

[1]Cambridge University; [2]Imperial College, London; [3]Pisa University

Abstract. Spatial logics are used to reason locally about disjoint data structures. They consist of standard first-order logic constructs, spatial (structural) connectives and their corresponding adjuncts. Lozes has shown that the adjuncts add no expressive power to a spatial logic for analysing tree structures, a surprising and important result. He also showed that a related logic does not have this adjunct elimination property. His proofs yield little information on the generality of adjunct elimination. We present a new proof of these results based on model-comparison games, and strengthen Lozes' results. Our proof is directed by the intuition that adjuncts can be eliminated when the corresponding moves are not useful in winning the game. The proof is modular with respect to the operators of the logic, providing a general technique for determining which combinations of operators admit adjunct elimination.

1 Introduction

Spatial logics have been introduced to provide local reasoning about disjoint data structures: O'Hearn and Reynolds have developed a new program logic (the separation logic) for low-level programs that manipulate RAM data structures, based on the bunched logic of O'Hearn and Pym [1]; Cardelli, Gardner and Ghelli have developed techniques for analysing and manipulating tree structures (such as XML), based on the ambient logic of Cardelli and Gordon [2–4]. These logics extend first-order logic with "spatial" connectives and their corresponding adjuncts. The spatial connectives allow us to reason locally about disjoint substructures. The adjuncts are used to obtain weakest pre-conditions for a Hoare logic for updating heaps [5], an elegant proof of the Schorr-Waite algorithm [6], and specifications of security properties of ambients [4].

We study adjunct elimination results for spatial logics. Lozes has recently proved that adjuncts add no expressive power to the ambient logic for specifying properties about trees with hidden names [7]. This result is fascinating as, for the logic without adjuncts, validity is undecidable while model-checking is in PSPACE, while for the logic with adjuncts, validity can be reduced to model-checking, suggesting that adjuncts are powerful. However, Lozes proof is not

⋆ A preliminary version of this work was presented at the LICS'04 workshop LRPP.

modular with respect to the operators of the logic. This means that the proof is not particularly illuminating and it is difficult to determine which variants of the logic enjoy the adjunct-elimination property.

We take a different approach. We provide a natural adaptation of Ehrenfeucht-Fraïssé games to the ambient logic, and use these games to provide a modular proof and an intuitive explanation of Lozes' results. Ehrenfeucht-Fraïssé games are two-players games, played on a pair of structures (in our case trees) T_1 and T_2, where one player Spoiler aims to show that the two structures are different while the other player Duplicator aims to show that they are similar. The number of moves in the game is determined by a fixed rank. Each move corresponds to an operator in the logic. At each move, Spoiler selects one of the trees and makes a move to which Duplicator must respond in the other tree. Spoiler wins if Duplicator has no reply. Duplicator wins if Spoiler runs out of moves without having forced a win. A winning strategy for Duplicator implies that the two trees cannot be distinguished by any sentence of the corresponding rank. Such games have previously been used for proving inexpressivity results for a variety of logics (see, for instance [8]). Here, we use them as tool for establishing a translation.

Our adaptation of the game to spatial logics is natural, reinforcing our view that spatial logics are themselves natural. For example, the standard composition operators $A \mid B$ or $A * B$ declare that the data structure can be split into two parts, one part satisfying A and the other B. The corresponding game move is: Spoiler chooses one of the two structures (here called *boards*) and splits it into two disjoint boards; Duplicator answers by splitting the other board into two corresponding boards; Spoiler then chooses on which pair to continue playing the game. The standard adjoint operators $A \triangleright B$ and $A -\!\!* B$ declare that whenever the data structure is put into a composition context that satisfies A then the result satisfies B. The corresponding game move is based on choosing a context to add to each board and going on either with the contexts or with the extended boards. Our proof is based on the intuition that adjuncts can be eliminated if extending a tree does not help Spoiler win, as Duplicator can respond by extending the other tree identically.

We prove soundness and completeness results for our games: that is, Spoiler has a winning strategy with rank r if and only if there is a logical sentence of rank r that can distinguish between the two trees. One feature of the games we define is that the rank (of a formula or a game) is more refined than just a number. This helps us to extend Lozes' result, by showing that any sentence admits an adjunct-less equivalent of *the same rank*. This preservation of rank is intriguing, as model-checking for the logic without adjuncts is decidable while that for the logic with adjuncts is undecidable. This implies that the translation from a formula with adjuncts to an equivalent one without adjuncts is not computable. However, the preservation of rank implies that the uncomputability is not due to an unbounded increase in size of the formula.

Our elimination results focus on a spatial logic for analysing tree structures with private names (using the hiding quantification and appears construct). A natural question is whether the result holds in the analogous logic with existential quantification. We prove adjunct non-elimination in the presence of existential quantification, regardless of the additional logical operators present. In contrast, Lozes simply provides a specific counterexample for a logic with existential quantification and *appears* operator, and, moreover, the counterexample relies on the absence of equality. Indeed, our game approach provides an intuitive insight into the interaction of existential and hiding quantifiers with adjuncts.

2 Tree Model and Logic

2.1 Trees

We give a simple algebra to describe unordered, edge-labelled trees, where the labels may be free (public) or hidden (private). These trees have been used to form the basic structure of ambients [9] for describing public or private firewalls, and web data [3] (similar to XML) for describing public or private information.

We assume a set $\mathcal{N}$ of names, ranged over by $n, m, \ldots$. The set of trees, denoted $\mathcal{T}$, is defined by the grammar

$T ::=$	$\mathbf{0}$	the tree consisting of a single root node
	$n[T]$	the tree with a single edge from root, labelled with free name n, leading to T
	$T \mid T$	the root-merge of two trees (commutative and associative)
	$(\nu n)T$	the tree T where label n is hidden.

The set of free names of a term is given by $\mathrm{fn}(T)$: for example, $\mathrm{fn}(n[T]) = \{n\} \cup \mathrm{fn}(\mathrm{T})$ and $\mathrm{fn}((\nu n)T) = \mathrm{fn}(T) \setminus \{n\}$. The congruence on trees, analogous to that given for π-processes, is an equivalence relation generated by the axioms in Table 2.1 and closed with respect to the tree constructors. It says that edges are unordered, and that the actual name of a private name is irrelevant, provided that its difference with other names (private and public) is preserved.

Table 2.1. *Congruence*

$T \mid U \equiv U \mid T \qquad (T \mid U) \mid V \equiv T \mid (U \mid V) \qquad T \mid \mathbf{0} \equiv T$

$m \notin \mathrm{fn}(T) \Rightarrow (\nu n)T \equiv (\nu m)T\{n \leftarrow m\} \qquad (\nu n_1)(\nu n_2)T \equiv (\nu n_2)(\nu n_1)T$

$n \notin \mathrm{fn}(T) \Rightarrow T \mid (\nu n)U \equiv (\nu n)(T \mid U) \qquad (\nu n)\mathbf{0} \equiv \mathbf{0}$

$n_1 \neq n_2 \Rightarrow n_1[(\nu n_2)T] \equiv (\nu n_2)n_1[T]$

The following decomposition properties are standard.

Lemma 1 (Decomposition).

1. *If* $T \mid U \equiv n[V]$ *then either* $T \equiv n[V]$ *and* $U \equiv \mathbf{0}$*, or* $U \equiv n[V]$ *and* $T \equiv \mathbf{0}$*.*
2. *If* $T \mid U \equiv V_1 \mid V_2$*, then* $\exists T_1, T_2, U_1, U_2.\ T_1 \mid T_2 \equiv T$*,* $U_1 \mid U_2 \equiv U$*,* $T_1 \mid U_1 \equiv V_1$*,* $T_2 \mid U_2 \equiv V_2$*.*

2.2 Logic

We describe the (static) ambient logic for specifying properties about trees with hidden names, which we denote in this paper by $\mathcal{L}$. It has been used to analyse security properties for ambients [4], and to declare typing properties in a pattern-matching language for manipulating web data [3]. It consists of the Boolean connectives, additional spatial (structural) connectives and their corresponding adjuncts from the propositional ambient logic, and the less familiar hiding quantifier $\mathsf{H}x.\,\_$ for analysing private names and *appears* operator $©n$ declaring that n occurs free [10].

Definition 1 *The set $\mathcal{A}$ of the formulae of $\mathcal{L}$ is defined by the following grammar, where pebble η stands for either a name $n \in \mathcal{N}$ or a name variable $x \in \mathcal{X}$:*

$$A, B ::= \mathbf{0} \mid \eta[A] \mid A \,|\, B \mid A \wedge B \mid \neg A \mid \mathsf{T} \mid A@\eta \mid A \triangleright B \mid \mathsf{H}x.\,A \mid ©\eta$$

The satisfaction relation $T \vDash A$ between trees in $\mathcal{T}$ and closed formulae in $\mathcal{L}$ is defined in Table 2.2. The relation $T \vDash A \,|\, B$ specifies that T can be split into two trees satisfying A and B respectively. For example, the formula $n[\mathsf{T}] \,|\, \neg\mathbf{0}$ means that a tree can be split into an edge n with an unspecified subtree satisfying the true formula T, and a non-empty tree satisfying the formula $\neg\mathbf{0}$. The order of edges is irrelevant, since satisfaction is closed with respect to tree isomorphism.

The location adjunct $A@n$ states that property A holds when the tree is put under edge n. The composition adjunct $A \triangleright B$ specifies that whenever we compose a tree satisfying A to the tree being analysed, then the result satisfies B. For example, if formula $\mathsf{attacker}$ specifies what an attacker can do, then $T \vDash \mathsf{attacker} \triangleright A$ states that, for any attacker O described by $\mathsf{attacker}$, the system $O \,|\, T$ must satisfy A (for example, secret names are not communicated).

A tree T satisfies $\mathsf{H}x.\,A$ if $T \equiv (\nu n)T'$ and $T'\{n \leftarrow m\} \vDash A\{x \leftarrow m\}$, for some fresh m. Hence, $\mathsf{H}x.\,\_$ quantifies x over the private names in T. However, m may be also bound to a private name that is not in fact used in T, since $T \equiv (\nu n)T$ when $n \notin \mathrm{fn}(T)$. The appears construct $©n$ can be used to prevent this possibility. In particular, $T \vDash \mathsf{H}x.\,(©x \wedge A)$ states that $T \equiv (\nu m)T'$, $T'\{n \leftarrow m\} \vDash A\{x \leftarrow m\}$, and $n \in \mathrm{fn}(T')$. Thus, the private name structure can be fully analysed by the logic.

The definition of free variables is standard: variable x is free in $x[A]$, $©x$, $A@x$, and the hiding quantification $\mathsf{H}x.\,A$ binds x in A. A *sentence* is a formula where no variable is free. We use $\mathrm{fv}(A)$ to denote all the free variables in A, and $\mathrm{fn}(A)$ to denote all the free names in A. Notice that, while name occurrences can be bound in a term by $(\nu n)\_$, only variables can be bound in formulae.

Lemma 2 (Basic Properties).

1. *Satisfaction relation is closed wrt congruence: $T \vDash A \wedge T \equiv U \Rightarrow U \vDash A$.*
2. *Logical equivalence $\equiv_L$ equals structural congruence: $T \equiv_L T' \Leftrightarrow T \equiv T'$.*

With the interpretation of hiding quantification $\mathsf{H}x.\,A$, it is intuitively clear the property $A\{x \leftarrow m\}$ holds regardless of which fresh m is chosen. This universal property is formally stated in the following lemma, mimicking a previous result in Gabbay and Pitts' seminal work on abstract syntax with binders [11].

Table 2.2. *Satisfaction*

$T \vDash \mathbf{0}$	$\overset{\text{def}}{\Leftrightarrow}$	$T \equiv \mathbf{0}$
$T \vDash n[A]$	$\overset{\text{def}}{\Leftrightarrow}$	$\exists U \in \mathcal{T}.\ T \equiv n[U] \ \wedge\ U \vDash A$
$T \vDash A \mid B$	$\overset{\text{def}}{\Leftrightarrow}$	$\exists T_1, T_2 \in \mathcal{T}.\ T \equiv T_1 \mid T_2 \ \wedge\ T_1 \vDash A \ \wedge\ T_2 \vDash B$
$T \vDash A \wedge B$	$\overset{\text{def}}{\Leftrightarrow}$	$T \vDash A \ \wedge\ T \vDash B$
$T \vDash \neg A$	$\overset{\text{def}}{\Leftrightarrow}$	$T \nvDash A$
$T \vDash \mathsf{T}$	always	
$T \vDash A@n$	$\overset{\text{def}}{\Leftrightarrow}$	$n[T] \vDash A$
$T \vDash A \triangleright B$	$\overset{\text{def}}{\Leftrightarrow}$	$\forall U \in \mathcal{T}.\ U \vDash A \ \Rightarrow\ T \mid U \vDash B$
$T \vDash \mathsf{H}x.\ A$	$\overset{\text{def}}{\Leftrightarrow}$	$\exists n \in (\mathcal{N} \setminus \text{fn}(A)), U \in \mathcal{T}.\ T \equiv (\nu n)U \wedge U \vDash A\{x \leftarrow n\}$
$T \vDash ©n$	$\overset{\text{def}}{\Leftrightarrow}$	$n \in \text{fn}(T)$

Lemma 3 (Universal Characterization of H).

$T \vDash \mathsf{H}x.\ A \ \Leftrightarrow \forall n \in \mathcal{N} \setminus (fn(A) \cup fn(T)).\ \exists U \in \mathcal{T}.\ T \equiv (\nu n)U \ \ \wedge\ \ U \vDash A\{x \leftarrow n\}$

The hiding quantifier $\mathsf{H}x.\ A$ and $©\eta$ are taken here as primitive in the original spirit of [12]. Lozes focuses on the alternative formulation [10, 13], using freshness quantification $\mathsf{N}x.\ A$ and revelation $\eta®A$ introduced in [4]. The two pairs can be mutually encoded, as we prove in the full paper [14]. Throughout the paper, we comment on how our results adapt to the case with revelation. In particular, revelation ® has an accompanying adjunct $A \oslash \eta$. As part of our adjunct-elimination results, we show that the revelation adjunct is also eliminable (Corollary 2). We report here the definition of $\mathsf{N}x.\ A$, $n®A$, and $A \oslash n$. The interdefinability of $\mathsf{N}x.\ A$, $n®A$ and $\mathsf{H}x.\ A$ and $©\eta$ is described in the full paper.

Definition 2 (Alternative Operators)

$$T \vDash \mathsf{N}x.\ A \ \overset{def}{\Leftrightarrow}\ \exists n \in (\mathcal{N} \setminus (fn(T) \cup fn(A))).\ T \vDash A\{x \leftarrow n\}$$
$$T \vDash n®A \ \overset{def}{\Leftrightarrow}\ \exists U \in \mathcal{T}.\ T \equiv (\nu n)U \ \textit{and}\ U \vDash A$$
$$T \vDash A \oslash n \ \overset{def}{\Leftrightarrow}\ (\nu n)T \vDash A$$

3 Games

We define an Ehrenfeucht-Fraïssé style game for $\mathcal{L}$. We prove that the game is sound and complete: that is, Spoiler has a winning strategy for a game on (T_1, T_2) with rank r if and only if there is a sentence of rank r that distinguishes T_1 from T_2. Each move in the game is associated with a specific operator from the logic. Our results are modular with respect to these moves, which means that they automatically extend to sublogics of $\mathcal{L}$ (as long as $\wedge$, $\neg$ and T are present).

3.1 Ranks, Valuation, and Discrimination

The rank of a formula A is a function $|A|$ that maps each operator (other than $\wedge$, $\neg$, T) to the depth of nesting of that operator in A. For example, the rank $|n[\mathsf{T}] \triangleright (n[\mathsf{T}] \triangleright \mathbf{0})|$ is the tuple $\{\mathbf{0} \mapsto 1; \_[] \mapsto 1; \triangleright \mapsto 2; \mathit{else} \mapsto 0\}$. The operators $\wedge$, $\neg$, and T are not in the rank domain, since there are no associated game moves. The leaf operators $\mathbf{0}$ and ⓒ may only be mapped to 0 or 1, since they do not nest.

We write $r + r'$, $r - r'$, $r \sqcup r'$, $r \geq r'$ to denote pointwise sum, subtraction, lub, and comparison between ranks r and r'. We write $\delta(\mathit{Op})$ for the Kronecker delta function: $\delta(\mathit{Op})$ is the tuple $\{\mathit{Op} \mapsto 1; \mathit{else} \mapsto 0\}$. Hence, a rank $\{\triangleright \mapsto 2; n \mapsto 1; \mathit{else} \mapsto 0\}$ can be written $2\delta(\triangleright) + \delta(n)$.

Table 3.3. *Examples of Ranks*

$\lvert n[\mathbf{0}] \mid (n[\mathbf{0}] \mid n[\mathbf{0}])\rvert$	$=$	$2\delta(\mid) + \delta(\_[]) + \delta(\mathbf{0})$
$\lvert \mathsf{H}x. \neg\mathbf{0} \wedge m[x[\mathbf{0}]]\rvert$	$=$	$\delta(\mathsf{H}) + \delta(\mathbf{0}) + 2\delta(\_[])$

For rank r, $\mathit{Ops}(r) \stackrel{\text{def}}{=} \{\mathit{Op} \ : \ r(\mathit{Op}) > 0\}$. For rank r, set of names N, and set of variables $\mathcal{Y}$, $\mathcal{L}(r, N, \mathcal{Y})$ are the formulae of rank r which only use names and variables in N and $\mathcal{Y}$, i.e. $\mathcal{L}(r, N, \mathcal{Y}) \stackrel{\text{def}}{=} \{A \ : \ |A| \leq r, \ \mathrm{fn}(A) \subseteq N, \ \mathrm{fv}(A) \subseteq \mathcal{Y}\}$.

We say that a tree T is distinguished from U by a sentence A when $T \vDash A$ and $U \nvDash A$. A sentence identifies a set of trees (those that satisfy it). We therefore say that two trees are distinguished by a set P if one is in the set and the other is not. To deal with open formulae, we define a *valuation* to be a finite partial function f from $\mathcal{N} \cup \mathcal{X}$ into $\mathcal{N}$, such that, for every $n \in \mathcal{N}$, either $f(n) = n$ or $f(n)$ is undefined. (This extension of valuations to names as well as variables is used in Section 3.2.) For any valuation $f : \mathcal{N} \cup \mathcal{X} \to \mathcal{N}$, let $A\{f\}$ denote the result of substituting x with $f(x)$ in A for every $x \in \mathrm{fv}(A)$ for which f is defined. We use $\mathit{dom}(f)$ and $\mathit{ran}(f)$ to denote the domain and range of f.

Definition 3 *For any valuation f, T is f-discriminated from U by a formula A with $\mathit{fv}(A) \subseteq \mathit{dom}(f)$ iff $T \vDash A\{f\}$ and $U \nvDash A\{f\}$.*

The next lemma is standard, but crucial.

Lemma 4. *For each rank r, finite set of names N, and finite set of variables $\mathcal{Y}$, a finite subset $\mathcal{A}_{r,N,\mathcal{Y}}$ of $\mathcal{L}(r, N, \mathcal{Y})$ exists such that any formula in $\mathcal{L}(r, N, \mathcal{Y})$ is equivalent to some formula in $\mathcal{A}_{r,N,\mathcal{Y}}$.*

For $f : \mathcal{N} \cup \mathcal{X} \to \mathcal{N}$, the formula $D_T^{r,N,f} = \bigwedge\{A \in \mathcal{A}_{r,N,\mathit{dom}(f)} \ : \ T \vDash A\{f\}\}$ is itself a formula of rank r and has the property that if $U \vDash D_T^{r,N,f}\{f\}$ then U and T cannot be f-discriminated by a formula of rank r (Lemma 5). Hence, $D_T^{r,N,f}$ describes T as seen at rank r.

Lemma 5. *For any T, U, f, r, N:*

$$(\forall A \in \mathcal{L}(r, N, \mathit{dom}(f)).\ T \vDash A\{f\} \Leftrightarrow U \vDash A\{f\}) \quad \Leftrightarrow \quad U \vDash D_T^{r,N,f}\{f\}$$

Before proceeding to define the games, we present a final lemma, which will be crucial in the proof of Adjunct Elimination (Corollary 2).

Lemma 6. *Let P be a set of trees such that, for any P-discriminated pair (T, U), there is a sentence $A_{T,U} \in \mathcal{L}(r, N, \emptyset)$ that discriminates T from U. Then, P is defined by a sentence $A \in \mathcal{L}(r, N, \emptyset)$.*

Proof Hint The disjunction of the descriptors $D_U^{r;N}$ of the trees U in P defines P: $A \stackrel{\text{def}}{=} \bigvee\{B \in \mathcal{A}_{r,N,\emptyset} \mid \exists U \in P.\ B \Leftrightarrow D_U^{r;N}\}$.

3.2 Games

We define a game parametrised by a finite rank r. The game is played by two players, Spoiler and Duplicator. At any stage of the game, the position consists of a quadruple (T_1, T_2, f, r) where T_1 and T_2 are trees, f is an injective valuation, and r is a rank. Initially, for some set of names N, f coincides with f_N, the function that sends every $n \in N$ to itself and is undefined otherwise. While a complete game position is given by (T_1, T_2, f, r), we will just write (T_1, T_2, f) or (T_1, T_2) when the rest is clear, or irrelevant.

At each turn, Spoiler makes a move and Duplicator responds. Spoiler can choose any move Op such that $r(Op) > 0$, provided that the move preconditions are met. Either the Op move terminates the game, as described below, or the game goes on with the T_i's and f updated as prescribed by the move and with $r(Op)$ decreased by one. Spoiler wins if it plays a move which Duplicator cannot answer ($\mathbf{0}$, ©, and sometimes $\_[]$). Duplicator wins when Spoiler has no move left to play, because r has become zero on every Op which can be played.

In the description below, most moves begin with Spoiler choosing a tree T between T_1 and T_2; in these cases, U is used for the other tree.

$\mathbf{0}$ move Spoiler chooses T so that $T \equiv \mathbf{0}$ and $U \not\equiv \mathbf{0}$, and wins.

$\_[]$ move Spoiler chooses a tree T and a pebble η such that $T \equiv f(\eta)[T']$. If $U \equiv f(\eta)[U']$, the game continues with (T', U'); otherwise, Spoiler wins.

$|$ move Spoiler chooses T, and two trees T' and T'' such that $T \equiv T' \mid T''$. Duplicator chooses U' and U'' such that $U \equiv U' \mid U''$. Spoiler decides whether the game will continue with (T', U'), or with (T'', U'').

$\triangleright$ move Spoiler chooses T and new tree T'; Duplicator chooses new tree U'. Spoiler decides whether the game will continue with $(T \mid T', U \mid U')$ or (T', U').

@ move Spoiler chooses a pebble η, and replaces T with $f(\eta)[T]$ and U with $f(\eta)[U]$.

H move Spoiler chooses T, a name n not in $\mathrm{fn}(T) \cup \mathrm{fn}(U) \cup ran(f)$, a variable $x \notin dom(f)$, and a tree T' such that $(\nu n)T' \equiv T$. Duplicator chooses a tree U' such that $(\nu n)U' \equiv U$. The game continues with $(T', U', (f; x \mapsto n))$.

© move Spoiler chooses T and η so that $f(\eta)$ is in T but not in U, and wins.

The definition is easily extended to the operators for freshness, revelation and the revelation adjunct (see [14]).

We may classify the moves according to their effect on the state of the game:

– $\_[]$, $\mathbf{0}$, ⓒ may end the game;
– H may extend f and change h-names to names;
– $|$, $\_[]$ reduce the size of the board; @ and $\triangleright$ may increase the board.

Indeed, one begins to see why adjunct moves may be useless. Spoiler is trying to show that the two boards are different, while Duplicator aims to show that they are similar enough. In a challenging game, Spoiler plays with a small rank over two large boards with a small difference buried somewhere. A typical strategy for Spoiler is "zooming in": splitting the boards, removing edges, until the small difference is exposed. In this setting, adjunct moves are quite useless: $\triangleright$ and @ blur the difference between the two boards by extending both with isomorphic trees (in a $\triangleright$ move, Duplicator will typically choose a U' isomorphic to the T' chosen by Spoiler). This is the intuition that we are going to exploit in our adjunct-elimination proof.

3.3 Soundness and Completeness

We state soundness and completeness results for our game. The proofs are in the full paper [14]. The proofs are completely "modular"; for each move, they only depend on the properties of the corresponding operator in the logic. This means that the result holds for any sublogic of $\mathcal{L}$, provided that it includes all the operators that appear in r. Similarly, our results easily extend to the logic with operators N, ⓡ and $\oslash$.

Lemma 7 (Game Soundness). *If a sentence $A \in \mathcal{L}(r, N, \emptyset)$ exists such that $T \vDash A \;\wedge\; U \nvDash A$, then Spoiler has a winning strategy for the game (T, U, f_N, r).*

Lemma 8 (Game Completeness). *If Spoiler has a winning strategy for the game (T, U, f_N, r), then there exists $A \in \mathcal{L}(r, N, \emptyset)$ such that $T \vDash A \;\wedge\; U \nvDash A$.*

4 Adjunct Elimination

We prove that any sentence can be transformed to an equivalent adjunct-free sentence of the same rank, hence extending Lozes result which does not express rank preservation. The basic idea is that, when Spoiler adds a context around one board, Duplicator can answer by adding the same context around the other board; whatever Spoiler does on the new context, Duplicator can mimic on the other copy. Our result requires that $r(\mathbf{0})$ is non-zero. This condition is not surprising, since, for example, the formula $n[\mathsf{T}] \triangleright n[\mathsf{T}]$ is logically equivalent to $\mathbf{0}$, and cannot be expressed without adjuncts and without $\mathbf{0}$ itself. Recall that we focus on the logic $\mathcal{L}$ with hiding and appears. Since our proofs are modular, the results also hold for the logic without these constructs. We include hiding and appears to link more closely to Lozes' original work, and to make the comparison with the non-eliminability of adjuncts in the presence of existential quantification (Section 5). Our results simply extend to the logic $\mathcal{L}$ with the additional revelation adjunct $\oslash$ [14]. We use DW (and SW) to denote the sets of game positions such that Duplicator (and Spoiler) has a winning strategy.

Lemma 9. *If* $(T, U, f, r) \in DW$ *and* $r(\mathbf{0}) > 0$*, then* $T \equiv \mathbf{0} \Leftrightarrow U \equiv \mathbf{0}$*.*

Theorem 1. *If* $(T, U, f, r) \in DW$ *and* $(T', U', f, r) \in DW$ *for* $\{\mathbf{0}\} \subseteq Ops(r)$ *and* $\eta \in dom(f)$*, then:*

$$(f(\eta)[T], f(\eta)[U], f, r) \in DW \quad (1)$$
$$(T \,|\, T', U \,|\, U', f, r) \in DW \quad (2)$$

Proof. (Sketch). The proof is by induction on r, and by cases on the possible moves of Spoiler. We analyse each move Op that Spoiler may make on the bigger board, and show that he cannot win under the hypothesis that he could not win on the original boards. We only show here the cases $Op = |$ and $Op = \triangleright$, assuming that Spoiler chooses T; the complete proof is in the full paper. When we analyse a move Op, we write r^- for $r - \delta(Op)$.

$|$, property (1): Spoiler splits $f(\eta)[T]$ into two trees, which must be congruent to $f(\eta)[T]$ and $\mathbf{0}$ by Lemma 1(1). Duplicator splits $f(\eta)[U]$ into $f(\eta)[U]$ and $\mathbf{0}$. The game $(\mathbf{0}, \mathbf{0}, f, r^-)$ is in DW by game completeness (Lemma 8) ($\mathbf{0}$ is logically equivalent to $\mathbf{0}$). $(T, U, f, r) \in DW$ implies that $(T, U, f, r^-) \in DW$, hence $(f(\eta)[T], f(\eta)[U], f, r^-) \in DW$ by induction.

$|$, (2): Spoiler splits $T \,|\, T'$ into two trees T_1 and T_2 which, by Lemma 1(2), can be written expressed as $T_1 \equiv T_1' \,|\, T_1''$ and $T_2 \equiv T_2' \,|\, T_2''$ such that $T_1' \,|\, T_2' \equiv T$ and $T_1'' \,|\, T_2'' \equiv T'$. Since $(T, U, f, r) \in DW$ and $(T', U', f, r) \in DW$, Duplicator has a response to a move by Spoiler in the game (T, U, f, r) where Spoiler splits T into T_1' and T_2' and similarly for the game (T', U', f, r). Suppose the moves for Duplicator in these two games involve splitting U into $U_1' \,|\, U_2'$ (respectively U' into $U_1'' \,|\, U_2''$), then by hypothesis Duplicator wins each of the four games (T_1', U_1', f, r^-), (T_2', U_2', f, r^-), (T_1'', U_1'', f, r^-) and (T_2'', U_2'', f, r^-). By induction hypothesis, this means that $(T_1' \,|\, T_1'', U_1' \,|\, U_1'', f, r^-) \in DW$ and $(T_2' \,|\, T_2'', U_2' \,|\, U_2'', f, r^-) \in DW$. Thus, splitting the tree $U \,|\, U'$ as $(U_1' \,|\, U_1'') \,|\, (U_2' \,|\, U_2'')$ is a winning move for Duplicator as required.

$\triangleright$ (1,2): Let $C\{T\}$ be either $T \,|\, T'$, or $f(\eta)[T]$ and $C\{U\}$ denote $U \,|\, U'$, or $f(\eta)[U]$, respectively. Spoiler chooses a tree V to compose with $C\{T\}$. Duplicator responds by adding the same tree to $C\{U\}$. If Spoiler chooses to proceed with (V, V), then Duplicator wins by game completeness (Lemma 8). Assume that Spoiler chooses to proceed with $(C\{T\} \,|\, V, C'\{U\} \,|\, V, f, r^-)$. $(T, U, f, r) \in DW$ and $(T', U', f, r) \in DW$ imply that $(T, U, f, r^-) \in DW$ and $(T', U', f, r^-) \in DW$, hence $(C\{T\}, C'\{U\}, f, r^-) \in DW$ follows by induction, and hence $(C\{T\} \,|\, V, C'\{U\} \,|\, V, f, r^-) \in DW$ also follows by induction.

Corollary 1 (Move Elimination) *If* $(T, U, f, r) \in DW$, $r^{\sqcup} \stackrel{\text{def}}{=} r \sqcup \delta(\mathbf{0})$*, and* $\{\triangleright, @\} \supseteq Ops(r^{adj})$*, then:*

$$(T, U, f, r^{\sqcup}) \in DW \;\Rightarrow\; (T, U, f, r + r^{adj}) \in DW$$
$$(T, U, f, r + r^{adj}) \in SW \;\Rightarrow\; (T, U, f, r^{\sqcup}) \in SW$$

We can finally show that adjuncts do not add expressive power to the logic. Not only that but, for each sentence containing adjuncts, there is an equivalent adjunct-less sentence of a related rank. There are only a finite number of inequivalent sentences for each rank (Lemma 4), but it remains an undecidable problem to determine which one is equivalent to a given sentence with adjuncts.

Corollary 2 (Adjunct Elimination) *Any property that can be expressed by a sentence in $\mathcal{L}(r + r^{adj}, N, \emptyset)$, where $\{\triangleright, @\} \supseteq Ops(r^{adj})$, can be expressed by a sentence in $\mathcal{L}(r \sqcup \delta(\mathbf{0}), N, \emptyset)$.*

Proof. Let P be defined by a sentence A in $\mathcal{L}(r + r^{adj}, N, \emptyset)$. For each $T \in P$ and $U \notin P$, by Game Soundness (Lemma 7), $(T, U, f_N, r + r^{adj}) \in SW$. By Corollary 1, $(T, U, f_N, r \sqcup \delta(\mathbf{0})) \in SW$. By Game Completeness (Lemma 8), this implies that, for each P-discriminated pair T, U, there is a sentence B_{TU} in $\mathcal{L}(r \sqcup \delta(\mathbf{0}), N, \emptyset)$ that discriminates T from U. By Lemma 6, there is a sentence B in $\mathcal{L}(r \sqcup \delta(\mathbf{0}), N, \emptyset)$ that defines P.

In the full paper we use the same technique to prove adjunct elimination for the logic extended with revelation adjunct. Revelation adjunct allows $\copyright n$ to be expressed as $(n[\mathbf{0}] \triangleright ((\neg(\neg\mathbf{0} \mid \neg\mathbf{0})) \oslash n))@m$ (for any $m \neq n$). For this reason, in the revelation-adjunct version of Theorem 1 the hypothesis $\{\mathbf{0}\} \subseteq Ops(r)$ must be strengthened to $\{\mathbf{0}, \copyright\} \subseteq Ops(r)$, and $\delta(\copyright)$ appears in the statement of the adjunct elimination result, as follows.

Theorem 2 (Adjunct Elimination With $\oslash$). *Any property that can be expressed by a sentence in $\mathcal{L}(r + r^{adj}, N, \emptyset)$, where $\{\triangleright, @, \oslash\} \supseteq Ops(r^{adj})$, can be expressed by a sentence in $\mathcal{L}(r \sqcup \delta(\copyright) \sqcup \delta(\mathbf{0}), N, \emptyset)$.*

5 Adjunct Non-Eliminability for ∃

The hiding quantifier H is similar to existential quantification $\exists$. A natural question is whether a similar adjunct elimination result holds for the logic with existential quantification. In [7], Lozes gives a counterexample to show that adjuncts cannot be eliminated in a logic with both existential quantification *and* $\copyright$. This result, although interesting, is weak since existential quantification is not usually associated with $\copyright$ and, moreover, the counterexample relies on the absence of primitive equality from the logic. Here we complete the analysis, by proving that adjuncts cannot be eliminated in a logic with $\exists$ and without $\copyright$, regardless of the presence of equality.

Let $\mathcal{L}_{\exists,\triangleright}$ denote the (static) ambient logic with existential quantification and the composition adjunct, and let $\mathcal{L}_{\exists,=}$ denote the corresponding logic without the composition adjunct and with equality. We have shown that the parity of trees is not definable in $\mathcal{L}_{\exists,=}$ (and, hence, neither in $\mathcal{L}_{\exists}$), using a standard game inexpressivity argument which we give in the full paper (Theorem 3). It is however definable in $\mathcal{L}_{\exists,\triangleright}$, a result due to Hongseok Yang and reported here (Theorem 4).

Theorem 3 (No Parity in $\mathcal{L}_{\exists,=}$). *No sentence A in $\mathcal{L}_{\exists,=}$ expresses the property that T is flat,*[1] *differently-labelled, and has an even number of edges.*

[1] A 'flat' tree looks like $n_1[] \mid \ldots \mid n_j[]$; 'differently labelled' means $n_i \neq n_j$ for $i \neq j$.

The $\mathcal{L}_{\exists,\triangleright}$ sentence used in Theorem 4 to describe parity in $\mathcal{L}_{\exists,\triangleright}$ is based on the following sentences:

$$
\begin{array}{ll}
\mathit{EachEdge}(\mathrm{A}) & \stackrel{\text{def}}{=} \neg(((\exists y.\, y[\mathsf{T}]) \wedge \neg A) \mid \mathsf{T}) \\
\mathit{Flat} & \stackrel{\text{def}}{=} \mathit{EachEdge}(\exists x.\, x[\mathbf{0}]) \\
\mathit{Diff} & \stackrel{\text{def}}{=} \neg(\exists x.\, x[\mathbf{0}] \mid x[\mathbf{0}] \mid \mathsf{T}) \\
\mathit{Pairs} & \stackrel{\text{def}}{=} \mathit{EachEdge}(\exists x, y.\, c[x[\mathbf{0}] \mid y[\mathbf{0}]]) \\
\mathit{DiffP} & \stackrel{\text{def}}{=} \neg\exists x.\, (c[x[\mathbf{0}] \mid x[\mathbf{0}]] \mid \mathsf{T}) \vee (c[x[\mathbf{0}] \mid \mathsf{T}] \mid c[x[\mathbf{0}] \mid \mathsf{T}] \mid \mathsf{T}) \\
A \propto B & \stackrel{\text{def}}{=} \neg(A \triangleright \neg B)
\end{array}
$$

$T \vDash \mathit{EachEdge}(A)$ denotes that every top-level edge of T satisfies A. Hence, $T \vDash \mathit{Flat}$ states that T is a flat-tree, and $\mathit{Flat} \wedge \mathit{Diff}$ means that its edges have different labels. Similarly, $T \vDash \mathit{Pairs}$ means that T is composed of $c[n[\mathbf{0}] \mid m[\mathbf{0}]]$ edges, while $\mathit{Pairs} \wedge \mathit{DiffP}$ means that all second-level labels are mutually different. Finally, $T \vDash A \propto B$ iff there exists U such that $U \vDash A$ and $T \mid U \vDash B$.

Theorem 4 (Yang: Parity in $\mathcal{L}_{\exists,\triangleright}$). *The sentence*

$$\mathit{Even} \stackrel{\text{def}}{=} (\mathit{Flat} \wedge \mathit{Diff}) \quad \wedge \quad ((\mathit{Pairs} \wedge \mathit{DiffP}) \propto (\forall x.\ x[\mathbf{0}] \mid \mathsf{T} \Leftrightarrow c[x[\mathbf{0}] \mid \mathsf{T}]))$$

defines the set of flat, differently-labelled trees with an even number of edges.

Proof. $T \vDash \mathit{Even}$ iff T is a flat tree where all the labels are different (expressed formally by $T \vDash \mathit{Flat} \wedge \mathit{Diff}$), and there exists U such that $U \vDash \mathit{Pairs} \wedge \mathit{DiffP}$ and $T \mid U \vDash \forall x.\ x[\mathbf{0}] \mid \mathsf{T} \Leftrightarrow c[x[\mathbf{0}] \mid \mathsf{T}]$. Hence, U has a shape

$$c[n_1[\mathbf{0}] \mid n_2[\mathbf{0}]] \mid \ldots \mid c[n_{2k-1}[\mathbf{0}] \mid n_{2k}[\mathbf{0}]],$$

all the n_i's are different, and U contains an even number of them. Finally, $T \mid U \vDash \forall x.\ x[\mathbf{0}] \mid \mathsf{T} \Leftrightarrow c[x[\mathbf{0}] \mid \mathsf{T}]$ says that the labels of T are exactly the same as the second-level labels of U, hence T has an even number of edges.

Games offer an explanation why $\mathcal{L}_{\exists,\triangleright}$ is more expressive than $\mathcal{L}_{\exists,=}$. Consider a $\mathcal{L}_{\exists,\triangleright}$ strategy that corresponds to Yang's sentence. Spoiler must distinguish between even board $T = n_1[] \mid \ldots \mid n_{2k}[]$ and odd board $U = m_1[] \mid \ldots \mid m_{2k+1}[]$. Spoiler adds the context $V = c[n_1[\mathbf{0}] \mid n_2[\mathbf{0}]] \mid \ldots \mid c[n_{2k-1}[\mathbf{0}] \mid n_{2k}[\mathbf{0}]]$ to the even board. Now Duplicator is lost. He may add $c[m_1[\mathbf{0}] \mid m_2[\mathbf{0}]] \mid \ldots \mid c[m_{2k-1}[\mathbf{0}] \mid m_{2k}[\mathbf{0}]]$ to the other board, but in this case there will be a name m_{2k+1} which appears once in $U \mid V$, while every name (but c) appears exactly twice in $T \mid V$. Now Spoiler can use $\exists$ to pebble that name and win.

In a game for $\mathcal{L}$ (with hiding and appears), such a strategy is not available to Spoiler because only hidden names can be pebbled in that game, and no hidden name can be shared between T and V above. Indeed, the key is that a counterpart to Theorem 1(2) does not hold for $\mathcal{L}_{\exists,\triangleright}$ games. It is possible for Duplicator to have a winning strategy on each of (T, U) and (T', U') while Spoiler wins on $(T \mid T', U \mid U')$ because of names shared between T and T'.

6 Conclusions

We have investigated adjunct elimination results for spatial logics, by introducing game techniques for such logics. Our work provides a modular proof of adjunct elimination which helps our understanding of why some combinations of operators admit adjunct elimination while others do not. In particular, we show the adjunct elimination results hold for a logic with hiding quantification and appears (for reasoning about private and public names), and do not hold for the analogous logic with existential quantification (for analysing shared names). Another consequence of our proof is a rank preservation result that shows that the elimination of adjuncts does not increase the rank of a sentence, which is surprising as adjuncts cannot be computably eliminated.