Hypergame Semantics: Full Completeness for System F

Dominic J.D. Hughes
Magdalen college and St. Hugh's college

Thesis submitted for the degree of Doctor of Philosophy at the University of Oxford, Trinity Term, 1999

Abstract

This is a thesis in an area of theoretical computer science and mathematical logic known as game semantics. The idea behind game semantics is to model a program or proof interactively as a strategy in a game, taking the form of a stimulus-response behaviour for Program against Environment, or for Prover against Refuter. By capturing the meaning of programs and proofs in an abstract, syntax-free form, the hope is to further our understanding of programming languages and logics.

The calculus of concern in this thesis is system F, also known as the polymorphic or second-order lambda calculus. From a computer science perspective, system F embodies (parametric) polymorphism, the idea that a function is so schematic that it works on all datatypes. For example, the reversal of a list is independent of the types of its elements. From a logical point of view, the calculus is (propositional) second-order intuitionistic logic, which can express second-order quantification. For example, the statement "every non-empty set of positive integers has a least element" quantifies over sets rather than mere elements of sets.

This thesis introduces a game semantics of system F called the hypergame model, the novelty being the introduction of *second-order moves*. A type is interpreted as a board on which to play a game, and a hypergame is an 'interleaving' of games on one or more boards. A second-order move consists in the introduction of of a new board. Computationally, the introduction of a board models the instantiation of a polymorphic function at a type, and logically, it models the instantiation of a universally quantified proof at a proposition.

The uniformity of system F polymorphism is captured by the fact that the boards of second-order moves by the opposing player are 'hidden' from view. The only way to play on a hidden board is by using a 'copycat' strategy. Thus syntactic uniformity, 'works the same way whatever the type', corresponds to game-theoretic uniformity, 'plays the same way whatever the hidden board'. Back at the level of the syntax, during interaction of strategies this copycat can be seen as η-expansion after type variable instantiation.

The main theorem is that the hypergame model is fully complete. Informally, full completeness means that the model matches up with the syntax 'perfectly'. Technically, every strategy in the model is the interpretation of a term of system F.

The hypergame model first appeared in preliminary form in *Logic in Computer Science 1997* [Hug97].

Acknowledgements

First and foremost, my supervisor Luke Ong, for taking such keen interest in my ideas, and for being so supportive in the final stages of preparation of the thesis.

Martin Hyland, Ralph Loader, and Guy McCusker, without whose personal support and inspiration I would have thrown in research to go and work in corporate finance.

Vaughan Pratt, for showing me that there's more to life than game semantics!

Samson Abramsky and Jean-Yves Girard, for stimulating discussions, particularly at Marktoberdorf.

Peter O'Hearn and Andy Pitts.

Funding from the Engineering and Physical Sciences Research Council, and the Searle Scholarship at St. Hugh's College, Oxford. Magdalen College, Oxford.

The diagrams in this thesis were made with John Hobby's MetaPost. A great tool — I wholeheartedly recommend it.

Everybody in room 002. Countless hours wasted playing interactive tetris with 'the Mexican' and the rest of the lads.

Mum, Dad, Ben, Fuzzball and Nan.

Tonnes of pasta, pesto, tea, and junk food.

Most of all: Jules Abdey, Adèle Carney, Suzie Landray, Rebekah Lee, Joel 'the Wakster' Ouaknine, Zach Shore, and Ben 'James' Worrell.

Contents

Chapter 1

Introduction

Denotational semantics is a programme of research in theoretical computer science pioneered by Scott and Strachey around 1970 [Sco72, Sto79, Gun92]. It is aimed at providing abstract, syntax-independent characterisations of program behaviour. A long-term goal of the ongoing research in denotational semantics is to abstract and formalise the basic structures and concepts of computation—procedures, loops, recursion, objects, sequentiality, concurrency, and so on—to provide a firm foundation for software engineering and programming language design and implementation. Analogously, by successfully abstracting and formalising the basic concepts of mechanics such as velocity, gravity, and momentum, physics provided a firm foundation for civil and mechanical engineering that has been used for centuries.

The programming concept of concern in this thesis is polymorphism. A function or program is *polymorphic* if it can take arguments of a variety of types. An example is the reversal of a list, which is defined irrespective of whether the elements are integers, characters, or of any other type. More specifically, we concentrate on parametric polymorphism, as distinguished from ad hoc polymorphism by Strachey [Str67]. A polymorphic function is *parametric* if it behaves uniformly across all types, as with list reversal, which works the same way regardless of the type of the elements. Other examples include templates in the programming language *C++* and polymorphism in functional languages such as *ML*. An *ad hoc* polymorphic operation works differently at different types, for example, the print functions of *C* or method overriding and overloading in object-oriented languages such as Java.

System F is a canonical calculus of parametric polymorphic functions. It is an extension of the simply typed lambda calculus with an abstraction operation on types, and is also known as the polymorphic or second-order lambda calculus. Abstraction on types is extremely powerful, and in particular any inductive datatype (natural numbers, lists, etc.) can be defined in system F. The system was introduced in the context of proof theory by Girard [Gir71], but was discovered independently in computer science by Reynolds [Rey74]. The terms and types of system F correspond respectively to the proofs and propositions of propositional second-order intuitionistic logic, via a Curry-Howard isomorphism [How, GLT89b]. This thesis, therefore, can be placed in either of two fields: theoretical computer science or mathematical logic.

Models of system F. The abstract, syntax-independent characterisations of programming languages and logics sought in denotational semantics are commonly referred to as models. Models of system F do not come about easily due to a circularity in the types known as impredicativity: the type T of a polymorphic function that works 'for all types

α' is semantically some kind of indexed product $T = \Pi_{\alpha \in \mathit{Types}}\, \alpha$, and since $T \in \mathit{Types}$, T is in its own indexing collection. In particular, Reynolds proved that no classical set-theoretic models could exist [Rey84], though Pitts showed how to relax this in constructive set theory [Pit87].

In nearly three decades of research only two classes of syntax-independent models of system F have been found, domain models [McC79, Gir86, CGW89], and realisability models [LM84, Hyl88]. Domain models adapt Scott's techniques of approximation and continuity originally used to model the lambda calculus [Sco72]. Research on realisability dates back decades, to well before the discovery of system F (see [Hyl88] for a survey). In the light of impredicativity, these two classes of models represent a substantial achievement in denotational semantics. However, none of the models manage to capture system F accurately, because of additional undesirable elements not corresponding to any term of the calculus. In particular, the domain models contain elements which contradict the spirit of parametricity [O'H96].

Full completeness. In 1992 Abramsky and Jagadeesan [AJ94] introduced the term *fully complete* to describe a model of a logic free from undesirable elements. Full completeness refines ordinary completeness with respect to provability to a completeness with respect to proofs: every element (morphism) of the model corresponds to a proof of the logic. In Lawvere-Lambek categorical logic [Law69, Lam68] a model is a (structure-preserving) functor, and the terminology "full completeness" comes from requiring the functor to be full. Fully complete models provide accurate, syntax-independent characterisations of proofs and normalisation. The related concept in the realm of programming languages is known as *full abstraction*. Abramsky and Jagadeesan's full completeness result for multiplicative linear logic [AJ94] stimulated the search for full completeness results for a wide variety of calculi, using a wide variety of techniques. Perhaps the earliest result of this kind, preceding even the terminology "full completeness", is for the simply typed λ-calculus [Plo80].

Game semantics. Recently a style of denotational semantics known as game semantics has emerged, with an impressive track-record of fully complete models of logics and fully abstract models of programming languages [HO93, AJ94, Nic94, AJM94, HO94, AM95, McC96b, Ong96, AM96, HY97, Lai97, AM98, AHM98, MO99, HM99, AM99b]. The idea is to model a program or proof interactively as a strategy in a game, taking the form of a stimulus-response behaviour for Program against Environment or for Prover against Refuter.

Game semantics represents a shift in perspective from the *static* to the *dynamic*. For example, in a model based on sets and functions (including domain and realisability models), a function $F : (X \to X) \to X$ has instant access to the whole (possibly infinite) input-output graph of an argument function $g : X \to X$. By contrast, in game semantics, F can obtain information about g only by repeatedly testing its input-output behaviour as a 'black box'. In fact, game semantics is even more interactive than this example may suggest: strategies are like processes, and composition is a form of 'parallel composition with hiding', in the sense of concurrency theory [Hoa85].

In 1989, before the emergence of game semantics, Girard [GLT89b] wrote[1]:

> ... *denotational* semantics of programs abound: for this kind of semantics nothing changes throughout the execution of program. On the other hand, there is hardly any civilised *operational* semantics of programs (we exclude *ad hoc* semantics

[1] In this passage, read *denotational* as *static* and *operational* as *dynamic*, a pattern which was set up by the author in the preceding section.

> which crudely paraphrase the steps toward normalisation). The establishment of a truly operational semantics of algorithms is perhaps the most important problem in computer science.

Game semantics can be seen as a significant step towards such a semantics of algorithms.

A celebrated achievement of game semantics is the furthering of our understanding of higher-type extensional sequential computation, with the fully abstract *PCF* models of Abramsky/Jagadeesan/Malacaria [AJM94], Hyland/Ong [HO94], and Nickau [Nic94]. *PCF* is an idealised functional programming language with if-then-else, basic arithmetic, and recursion.

The first compositional games models appeared in the early 1990s. In computer science, the origins of ideas can be traced back to the concrete data structures of Kahn and Plotkin [KP93] (which first appeared in 1978) and the sequential algorithms of Berry and Curien [BC82]. In logic, ideas can be traced back to Lorenz and Lorenzen [LL78]. Also influential were the games of Blass [Bla72, Bla92], Joyal's categorical presentation of Conway's games [Con76], and Gandy's dialogues [Gan93]. There are links with Girard's geometry of interaction in linear logic [Gir87, Gir89], and the interaction in abstract machines [DHR96]. For more history, see the introductions of [AJ94, Fel86, Hyl97, HO94, McC96b], and for accessible introductions to game semantics, see [Hyl97, AM99a]. Current research includes the abstraction of strategies and interaction [AM99b, HS99], addressing the issue that mathematical representations in terms of sequences and parallel composition with hiding can be complicated and unwieldy.

Contribution: hypergame semantics. This thesis introduces[2] a fully complete game semantics of system F called the hypergame model, a key novelty being the introduction of a form of second-order move.

A type is interpreted as a board on which to play a game, and a hypergame is an 'interleaving' of games on one or more boards. Picture a chess-board, a backgammon-board, and several others, each with games running on them. A second-order move consists in the introduction of of a new board into the hypergame, for example, a monopoly board, upon which play may begin at some point in the future. More precisely, let B be the board interpreting the type T. Then the application of a polymorphic term to a type argument T is interpreted by a second-order move introducing B into the hypergame. Thus the hypergames model is captured by the slogan *type arguments are second-order moves.*

The uniformity of system F polymorphism is captured by the fact that the boards of second-order moves by the opposing player are hidden from view. The only way to play on a hidden board is by using a 'copycat' strategy. Thus syntactic uniformity, 'works the same way whatever the type', corresponds to game-theoretic uniformity, 'plays the same way whatever the hidden board'. Back at the level of the syntax, during interaction of strategies this copycat can be seen as η-expansion after type variable instantiation. For example, $\Lambda X.\lambda f^X.f\,(Y \to Y) \rightsquigarrow \lambda f^{Y\to Y}.f \rightsquigarrow \lambda x^{Y\to Y}.\lambda y^Y.fy$.

The first-order fragment of the model is based on (the λ-calculus fragment of) Hyland/Ong [HO94] and Nickau [Nic94] *PCF* games. The Hyland/Ong and Nickau games models are widely accepted as being one and the same. However, we highlight a subtle and hitherto neglected difference between Hyland/Ong interaction and Nickau interaction, which turned out to be critical for the construction of the hypergames model. In some sense Hyland/Ong interaction is 'richer' than Nickau interaction; we elaborate at the end of Chapter 3. For technical reasons, we chose Nickau's approach as the first-order basis of interaction in the hypergame model.

[2] The model first appeared in preliminary form as the extended abstract [Hug97].

Contribution: full completeness for system F. We work with system F with products, and the hypergames model $\mathbb{H}$ is presented categorically as a $2\lambda\times$-hyperdoctrine. The main theorem is:

> THEOREM (FULL COMPLETENESS) *Every morphism (winning strategy) σ of $\mathbb{H}$ defines an η-long, β-normal term $\widehat{\sigma}$, whose interpretation is σ.*

Thus the hypergames model gives a precise, syntax-independent characterisation of (constant-free) parametric polymorphic functions.

One way of thinking about full completeness is that the model is isomorphic to a quotient of the syntax. The quotient induced by $\mathbb{H}$ on the product- and unit-free fragment system F includes $\beta\eta$ together with the isomorphism induced on terms (because terms contain types) by identifying every type with its prenex normal form. So the equational theory of the model is very close to initial.

By full completeness, $\mathbb{H}$ is parametric, in the informal Strachey sense that every function acts 'uniformly'. The model is not Reynolds relationally parametric, using a result of [PA93]: $\forall X.X \to X$ is not terminal. Note that the term model of system F is not relationally parametric for the same reason, so this is not a symptom of non-uniformity.

As an application of the theorem, we use full completeness to reason about properties of the system F encodings of products and sums, and simple inductive datatypes such as booleans, natural numbers, and lists.

Related work. The hypergames model is not the first games model of a polymorphic calculus. Abramsky [Abr97] obtained a model of the multiplicative linear cousin of system F, though unfortunately it was not fully complete. The first-order fragment of the model is based on Abramsky/Jagadeesan linear logic games [AJ94]. The new idea was to model polymorphism via the expansion of the 'playing area' as a game proceeds. Relative to this, the novelty of the hypergame approach in this thesis is the *method* by which the shape of the playing area changes, namely, by second-order moves.

At second-order Abramsky uses domain-theoretic techniques, in the form of functions between games that are continuous with respect to an inclusion ordering on the games. With regard to the static/dynamic discussion in the section earlier on game semantics, because of the domain-theoretic underpinning one could say that at second order the model is static. By contrast, the hypergames model, with second-order moves, is *dynamic* at second order. Consequently (and informally), in the context of this related work, the essential contribution of this thesis is as follows:

> Full completeness for system F can be obtained by a semantic shift in perspective from static to dynamic at *second order*, with the notion of a second-order move interpreting type application.

This idea of 'type arguments as second-order moves' first appeared in [Hug97]. Nickau and Ong have adopted the idea and are hoping to develop an alternative model based on it [NO], aiming to be closer to the original Hyland/Ong and Nickau PCF games. Upon the completion of their work, it will be interesting to study of the relationship between the two models.

Currently, Abramsky and Lenisa are working on fully complete games models for ML polymorphsism, which may one day turn out to be a stepping stone towards a different fully complete model for system F.

Hugo Herbelin is working on an abstract machine for system F [Her]. On the basis of recent informal discussions, the interaction of his abstract machine may turn out to be the same as the interaction of the hypergames model presented here (restricted to the product-free case).

1.1 The ideas in a nutshell

This section covers the key ideas of the hypergames model in an informal and accessible way, hopefully making the rest of the thesis redundant for casual readers (apart from, perhaps, the conclusion, Chapter 9).

Due to its second order nature, system F is rather hard to understand. It was not at all obvious where to begin the search for a games model. So the thought was this: given the multitude of existing games models of first-order calculi, can one abstract a *methodology* for moving from syntax to game semantics? If so, one could simply 'plug in' the syntax of system F, and a games model of system F would pop out for free.

In section 1.1.1 below we outline such a methodology, motivated by analysing terms of the simply typed lambda calculus. This methodology is applied to system F in section 1.1.2, the results of which yield two of the key ideas in the hypergame model: the notion of a second-order move introduced in section 1.1.4, and the copycat rule presented in section 1.1.3. Another new idea is introduced in section 1.1.5, that of a polymorphic arena, used as the 'gameboards' in hypergames. Finally, section 1.1.6 explains copycat between hidden arenas during the interaction of strategies back at the level of the syntax, as type instantiation followed by η-expansion.

1.1.1 The 'top-down term' methodology

We set out a methodology for extracting strategies from terms of the simply typed λ-calculus. The strategies so obtained turn out to be Lorenzen E-strategies, which were a precursor of Hyland/Ong innocent strategies and Nickau sequential strategies. Consequently, this section also serves as an informal introduction to some of the ideas behind (the λ-calculus fragment of) Hyland/Nickau/Ong *PCF* games, including their notion of *arena* as a 'game-board'. We assume nothing more than a basic familiarity with the pure simply typed λ-calculus.

Let $\lambda_1^{\rightarrow}$ denote the simply typed lambda calculus with types generated by arrow $\rightarrow$ from a single base type variable, say X. We shall follow the usual conventions that abstraction extends as far to the right as possible, application associates to the left, and arrow associates to the right. For example,

$$\lambda f^{X\rightarrow X\rightarrow X}.\lambda x^X.\lambda y^X.fxy \quad = \quad \lambda f^{X\rightarrow(X\rightarrow X)}.\lambda x^X.\lambda y^X.((fx)y)$$

Here are the basic properties of simply typed lambda calculus that we shall need, which are covered by any introductory textbook on the subject, e.g., [Bar84, GLT89b]. A term is ***closed*** if it contains no free variables. A term is ***β-normal*** if and only if it is of the form $\lambda x_1^{T_1}.\ldots.\lambda x_k^{T_k}.yu_1\ldots u_n$ where y is a variable and the u_i are also β-normal terms. The variable y is called the ***head variable***. If the term is closed, y is necessarily one of the x_i. Such a term is η-long if and only if the body is of ground type (i.e. X, for $\lambda_1^{\rightarrow}$), and the u_i are also η-long. For example, the η-long variant (η-expansion) of $\lambda f^{X\rightarrow X\rightarrow X}.f$ is $\lambda f^{X\rightarrow X\rightarrow X}.\lambda x^X.\lambda y^X.fxy$, with 'all arguments explicit'.

Suppose I ask you to write down a closed, η-long, β-normal term of type T of $\lambda_1^{\rightarrow}$. How do you go about it? Since $T = X_1 \to X_2 \to \ldots \to X_k \to X$ for some k and types[3] X_i, and since t is η-long, t starts with k corresponding abstractions:

$$t = \lambda x_1^{X_1}.\lambda x_2^{X_2}...\lambda x_k^{X_k}.?$$

Now you have a choice. Which of the x_i do you take as head variable? Suppose you choose x_3. Then t is

$$t = \lambda x_1^{X_1}.\lambda x_2^{X_2}...\lambda x_k^{X_k}.x_3\ ?$$

Because x_3 is of type X_3, say $X_3 = A_1 \to A_2 \to \ldots \to A_l \to X$, and t is η-long,

$$t = \lambda x_1^{X_1}.\lambda x_2^{X_2}...\lambda x_k^{X_k}.x_3 a_1 a_2 \ldots a_l$$

for terms a_i of type A_i. You are not finished yet: the arguments a_i are as yet unspecified. Suppose I ask "what is a_5?" Since a_5 is of type A_5, say $A_5 = Y_1 \to Y_2 \to \ldots \to Y_m \to X$, a_5 starts with m corresponding abstractions:

$$a_5 = \lambda y_1^{Y_1}.\lambda y_2^{Y_2}...\lambda y_m^{Y_m}.?$$

Now it is your turn again to choose a head variable, this time either one of the fresh y_i or one of the old x_i. Suppose you choose y_2. Then a_5 is of the form:

$$a_5 = \lambda y_1^{Y_1}.\lambda y_2^{Y_2}...\lambda y_m^{Y_m}.y_2\ ?$$

Because y_2 is of type Y_2, say $Y_2 = B_1 \to B_2 \to \ldots \to B_n \to X$,

$$a_5 = \lambda y_1^{Y_1}.\lambda y_2^{Y_2}...\lambda y_m^{Y_m}.y_2 b_1 b_2 \ldots b_n$$

for terms b_i of type B_i. My turn again: "what is b_4?" Since b_4 is of type B_4, say $B_4 = Z_1 \to Z_2 \to \ldots \to Z_p \to X$, you know that it starts with p corresponding abstractions:

$$b_4 = \lambda z_1^{Z_1}.\lambda z_2^{Z_2}...\lambda z_p^{Z_p}.?$$

It is your turn to choose a head variable again: one of the old x_i or y_i, or one of the newly abstracted z_i. Suppose you choose x_2. Then b_4 is of the form

$$b_4 = \lambda z_1^{Z_1}.\lambda z_2^{Z_2}...\lambda z_p^{Z_p}.x_2?$$

Because x_2 is of type X_2, say $X_2 = C_1 \to C_2 \to \ldots \to C_q \to X$,

$$b_4 = \lambda z_1^{Z_1}.\lambda z_2^{Z_2}...\lambda z_p^{Z_p}.x_2 c_1 c_2 \ldots c_q$$

for terms c_i is of type C_i. Now I ask "what is c_5?" ... and so on.

Our dialogue or 'game'

$$\begin{array}{rccllcl}
\text{I choose } t & t & = & \lambda x_1^{X_1}...\lambda x_k^{X_k}.? & t & : & X_1 \to \ldots \to X_k \to X \\
\text{You choose } x_3 & t & = & \lambda x_1^{X_1}...\lambda x_k^{X_k}.x_3 a_1 \ldots a_l & x_3 & : & A_1 \to \ldots \to A_l \to X \\
\text{I choose } a_5 & a_5 & = & \lambda y_1^{Y_1}...\lambda y_m^{Y_m}.? & a_5 & : & Y_1 \to \ldots \to Y_m \to X \\
\text{You choose } y_2 & a_5 & = & \lambda y_1^{Y_1}...\lambda y_m^{Y_m}.y_2 b_1 \ldots b_n & y_2 & : & B_1 \to \ldots \to B_n \to X \\
\text{I choose } b_4 & b_4 & = & \lambda z_1^{Z_1}...\lambda z_p^{Z_p}.? & b_4 & : & Z_1 \to \ldots \to Z_p \to X \\
\text{You choose } x_2 & b_4 & = & \lambda z_1^{Z_1}...\lambda z_p^{Z_p}.x_2 c_1 \ldots c_q & x_2 & : & C_1 \to \ldots \to C_q \to X \\
\text{I choose } c_5 & c_5 & = & \ldots\ldots\ldots\ldots? & c_5 & : & \ldots \\
\vdots & \vdots & & & \vdots & &
\end{array}$$

[3] Do not confuse the X_i, which are types, with X, which is a type *variable*. The intent is to pattern-match term variables with their capitalised counterparts.

will continue until you choose a head variable of ground type X, upon which I have no responses available, because a variable of ground type requires no arguments: "game over."

The dialogue could have followed a different course. For example, I may have chosen to ask about the third argument b_3 of y_2 instead of the fourth argument b_4, or early on I may have asked about the first argument a_1 of x_3 instead of the fifth. So a closed, η-long, β-normal term is equivalent to a prepared response to every query ("what is a_5?", "what is b_4?", etc.) I could have raised, that eventually 'has the last word' ('wins') by choosing a head-variable of ground type (denying me a response). In other words, a closed, η-long, β-normal term is equivalent to the 'strategy' for writing it down in 'top-down' or 'demand-driven' fashion.

Note how the choice of moves available to either of us at any point in the 'game' was completely determined by T. For example, the k possibilities $x_1, \ldots, x_k$ for your first move corresponded to the fact that T decomposes as $X_1 \to \ldots \to X_k \to X$. The same goes for all subsequent moves for either of us. You were allowed to go back and choose from an earlier batch of abstracted variables—for example, in your last move you chose x_2 rather than one of the fresh z_i—but nonetheless all your options were determined by the structure of T. So we think of T as some kind of 'arena' or 'game-board' in which to play, specifying the available moves at every stage.

Summary of ideas

1. We 'played a game' in the 'arena' T, by 'discussing a term' of type T in 'top-down' or 'demand-driven' fashion.

2. Each of your 'moves' was a choice of head-variable for a sub-term.

3. Each of my 'moves' was a request to inspect an argument sub-term of the head variable you just chose.

4. A 'winning strategy' corresponds to a closed, η-long, β-normal form t.

The 'strategies' so-derived were essentially the E-strategies of Lorenzen [LL78], which are a precursor of Hyland/Ong innocent strategies and Nickau sequential strategies. This takes us part-way to a game semantics of $\lambda_1^{\to}$: Hyland, Nickau and Ong provided a formal definition of an arena together with rules for playing games along the lines of the intuition above (though for *PCF* they were not concerned with any notion of winning). An essential (and highly non-trivial) contribution of Hyland, Ong and Nickau was to provide a notion of composition of such strategies, interpreting computation/evaluation as a form of interaction. We shall cover interaction for $\lambda_1^{\to}$ in Chapter 3, *Simple games.*

1.1.2 From 'top-down' system F terms towards hypergames

We apply the 'top-down term' methodology to system F. The analysis is informal, but illustrates how the seeds of ideas for a hypergames model already 'live in the syntax', so long as we look at syntax in the right way. Two 'rules' in particular become apparent: the *copycat rule* and *types as second-order moves.* For simplicity, in this section we do not consider product types.

Let us discuss a closed, η-long, β-normal term of system F in top-down fashion. To start with, consider a term[4] of type $\texttt{Bool} = \forall X. X \to X \to X$, the standard system F encoding

[4] Following the usual convention, the scope of a quantifier extends far to the right as possible, so $\forall X. X \to X \to X = \forall X.(X \to X \to X)$.

of the Booleans [GLT89b]. The type specifies three 'inputs', one type input corresponding to "$\forall X$", followed by a term input corresponding to the first "$X \rightarrow$", and then a term input corresponding to the second "$X \rightarrow$". So since t is η-long, we have the associated abstractions:

$$t = \Lambda X.\, \lambda x^X.\lambda y^X.?$$

Just as for $\lambda_1^{\rightarrow}$, the body of a β-normal term of system F begins with a head (term-)variable. So you can choose between x and y. Suppose you pick x. Then

$$t = \Lambda X.\, \lambda x^X.\lambda y^X.x?$$

Since x of of type X, a ground type, it requires no arguments:

$$t = \Lambda X.\, \lambda x^X.\lambda y^X.x$$

So I have no moves to play (i.e., no arguments to choose from), and the game is already over. In summary, the dialogue was:

$$\begin{array}{rrclrcl} \text{I choose } t & t & = & \Lambda X.\, \lambda x^X.\lambda y^X.? & t & : & \forall X.\, X \to X \to X \\ \text{You choose } x & t & = & \Lambda X.\, \lambda x^X.\lambda y^X.x & x & : & X \end{array}$$

and your strategy 'pick x' defined the closed, η-long, β-normal term

$$\texttt{true} = \Lambda X.\, \lambda x^X.\lambda y^X.x.$$

Had you instead adopted the strategy 'pick y', the dialogue would have been

$$\begin{array}{rrclrcl} \text{I choose } t & t & = & \Lambda X.\, \lambda x^X.\lambda y^X.? & t & : & \forall X.\, X \to X \to X \\ \text{You choose } y & t & = & \Lambda X.\, \lambda x^X.\lambda y^X.y & y & : & X \end{array}$$

and you would have defined the closed, η-long, β-normal term

$$\texttt{false} \;\; = \;\; \Lambda X.\, \lambda x^X.\lambda y^X.y$$

Since x and y are the only head variables to choose from, these are the only two (winning) strategies available to you. This corresponds to the fact that `Bool` has two (closed, η-expanded) β-normal inhabitants, `true` and `false`.

Now consider the type $\texttt{Nat} = \forall X.\, (X \to X) \to X \to X$. This type specifies three 'inputs': a type input "$\forall X$", a term input "$(X \to X) \to$", and a term input "$X \to$". Since t is η-long, we have corresponding abstractions:

$$t \;\; = \;\; \Lambda X.\, \lambda f^{X\to X}.\lambda x^X.?$$

For your first move, you must pick a head variable, either f or x. Suppose you choose f. Then

$$t \;\; = \;\; \Lambda X.\, \lambda f^{X\to X}.\lambda x^X.f?$$

Since f is of function type $X \to X$, it requires an argument in order for the body of t to be of ground type, and hence for t to be η-expanded:

$$t \;\; = \;\; \Lambda X.\, \lambda f^{X\to X}.\lambda x^X.fa_1$$

For my turn, as was the case for $\lambda_1^{\rightarrow}$, I have to pick an argument of the head-variable. Since a_1 is the only one, I have no choice:

$$a_1 \;\; = \;\; ?$$

For your turn, you need to pick a head variable for a_1. Since a_1 is of ground type (i.e., of type X), it introduces no new abstracted variables, so all you can do is choose between f and x again. Suppose you choose f:

$$a_1 \quad = \quad f?$$

Since f is of function type $X \to X$, it requires an argument, say a_2:

$$a_1 \quad = \quad fa_2$$

As with my last move, I have no choice but to pick the one and only argument of f:

$$a_2 \quad = \quad ?$$

Again your choice of head variable is restricted to f or x, because a_2 introduces no new abstractions. Suppose you pick f:

$$a_2 \quad = \quad f?$$

Since f is of function type $X \to X$, it requires an argument, say a_3:

$$a_2 \quad = \quad fa_3$$

I have to pick the only argument a_3:

$$a_3 \quad = \quad ?$$

This time, suppose you pick x as head-variable:

$$a_3 \quad = \quad x?$$

Since x is of ground type (i.e., of type X), it takes no arguments:

$$a_3 \quad = \quad x$$

The game is over, because I have no moves available (i.e., no arguments to choose from). In summary, our discussion was:

$$\begin{array}{rrcl@{\qquad}rcl}
\text{I choose } t & t & = & \Lambda X.\,\lambda f^{X\to X}.\lambda x^X.? & t & : & \forall X.\, X \to X \to X \\
\text{You choose } f & t & = & \Lambda X.\,\lambda f^{X\to X}.\lambda x^X.fa_1 & f & : & X \to X \\
\text{I choose } a_1 & a_1 & = & ? & a_1 & : & X \\
\text{You choose } f & a_1 & = & fa_2 & f & : & X \to X \\
\text{I choose } a_2 & a_2 & = & ? & a_2 & : & X \\
\text{You choose } f & a_3 & = & fa_3 & f & : & X \to X \\
\text{I choose } a_3 & a_3 & = & ? & a_3 & : & X \\
\text{You choose } x & a_3 & = & x & x & : & X
\end{array}$$

so your strategy defined the Church numeral

$$\underline{3} = \Lambda X.\,\lambda f^{X\to X}.\lambda x^X.f(f(fx)).$$

1.1.3 The copycat rule

So far there has been no significant deviation from $\lambda_1^{\rightarrow}$. Following the same pattern as the `Bool` and `Nat` examples above, here is a possible discussion of a term of type $T = \forall X. \forall Y. X \to Y \to X$. There are four abstractions, corresponding to the inputs "$\forall X$", "$\forall Y$", "$X \to$", and "$Y \to$".

$$\begin{array}{rcclccl} \text{I choose } t & t & = & \Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda y^Y.? & t & : & \forall X. \forall Y. X \to Y \to X \\ \text{You choose } x & t & = & \Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda y^Y.x & x & : & X \end{array}$$

Your strategy here, 'pick x', defines the closed, η-long, β-normal term

$$\Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda y^Y.x$$

Of course you could have adopted the strategy 'pick y' instead:

$$\begin{array}{rcclccl} \text{I choose } t & t & = & \Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda y^Y.? & t & : & \forall X. \forall Y. X \to Y \to X \\ \text{You choose } y & t & = & \Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda y^Y.y & y & : & Y \end{array}$$

But then, since the body of the term is y, of type Y, the overall type of t is

$$\forall X. \forall Y. X \to Y \to Y$$

rather than the desired

$$\forall X. \forall Y. X \to Y \to X$$

So for 'type-checking' reasons, we shall have to 'disallow' moves (choices of head-variable) like this one.

Here is another simple example, with a similar type:

$$T = \forall X. \forall Y. X \to (X \to Y) \to X.$$

First, a discussion which does not violate any 'type-checking', that is essentially the same as the first discussion of the previous example:

$$\begin{array}{rcclccl} \text{I choose } t & t & = & \Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda f^{X \to Y}.? & t & : & \forall X. \forall Y. X \to (X \to Y) \to X \\ \text{You choose } x & t & = & \Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda f^{X \to Y}.x & x & : & X \end{array}$$

If you were to choose f, however:

$$\begin{array}{rcclccl} \text{I choose } t & t & = & \Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda f^{X \to Y}.? & t & : & \forall X. \forall Y. X \to (X \to Y) \to X \\ \text{You choose } f & t & = & \Lambda X.\,\Lambda Y.\,\lambda x^X.\lambda f^{X \to Y}.fa & f & : & X \to Y \end{array}$$

The result is another type mismatch. More specifically, since T is a type of the form "$\cdots X$", the body "?" of t must turn out to be of type X. But in this example the body fa is of type Y, rather than X.

'Type-mismatch' such is in the two examples above suggests the following:

COPYCAT RULE *Whenever I choose a move (argument) whose type is of the form "$\cdots X$", for some type variable X, the body resulting from your next move (after supplying the necessary dummy arguments in order to reach ground type) must be of type X.*

Appendix A gives another example of the copycat rule at work. The example also illustrates (informally) an application of the full completeness of the model to counting the inhabitants of system F types.

1.1.4 Second-order moves

The examples so far have essentially just been $\lambda_1^{\rightarrow}$ dialogues, with $\forall X$, $\forall Y$, ΛX, and ΛY not playing any real role. The quantified type variables in the examples above were in positive position[5]; things get more interesting when type variables occur in negative position.

Recall that $\mathtt{Bool} = \forall X.X \to X \to X$. Let us discuss a closed, η-long, β-expanded term t of type $T = \forall Y.\, Y \to \mathtt{Bool} \to Y$. Since T specifies three inputs, a type input "$\forall Y$", a term input "$Y \to$", and a term input "$\mathtt{Bool} \to$", t starts with the corresponding abstractions:

$$t \;=\; \Lambda Y.\, \lambda y^{Y}.\lambda x^{\mathtt{Bool}}.? \qquad\qquad T \;=\; \forall Y.\, Y \to \mathtt{Bool} \to Y$$

For your first move, you can choose between y and x. Suppose you choose x:

$$t \;=\; \Lambda Y.\, \lambda y^{Y}.\lambda x^{\mathtt{Bool}}.x? \qquad\qquad T \;=\; \forall Y.\, Y \to \mathtt{Bool} \to Y$$

Since x is of type $\mathtt{Bool} = \forall X.X \to X \to X$, it requires three dummy arguments: one type argument for "$\forall X$", one term argument for the first "$X \to$" and another term argument for the second "$X \to$":

$$t \;=\; \Lambda Y.\, \lambda y^{Y}.\lambda x^{\mathtt{Bool}}.\underbrace{xAa_1a_2}_{\text{of type } A} \qquad\qquad T \;=\; \forall Y.\, Y \to \mathtt{Bool} \to Y$$

The body xAa_1a_2 is of type A. But since A is as yet unspecified, we have no way of telling whether or not the body is of ground type, so t may not be η-long. For example, suppose you were to set $A = Y \to Y$. Then the body xAa_1a_2 is of type $Y \to Y$, and requires another dummy term argument of type Y, say b, in order to reach ground type:

$$\begin{aligned} t \;&=\; \Lambda Y.\, \lambda y^{Y}.\lambda x^{\mathtt{Bool}}.\underbrace{xAa_1a_2}_{\text{of type } T\to T}b \qquad\qquad & T \;&=\; \forall Y.\, Y \to \mathtt{Bool} \to Y \\ & & A \;&=\; Y \to Y \end{aligned}$$

Similarly, if you set $A = Y \to (Y \to Y) \to Y$, then the body xAa_1a_2 is of type

$$Y \to (Y \to Y) \to Y,$$

and needs two more dummy arguments to reach ground type, say b_1 corresponding to "$Y \to$" and b_2 corresponding to "$(Y \to Y) \to$":

$$\begin{aligned} t \;&=\; \Lambda Y.\, \lambda y^{Y}.\lambda x^{\mathtt{Bool}}.\underbrace{xAa_1a_2}_{Y\to(Y\to Y)\to Y}b_1b_2 \qquad\qquad & T \;&=\; \forall Y.\, Y \to \mathtt{Bool} \to Y \\ & & A \;&=\; Y \to (Y \to Y) \to Y \end{aligned}$$

So only *after* you specify A do I know the full range of possibilities for my next move: in the first case, I choose from $\{a_1, a_2, b\}$, and in the second case, from $\{a_1, a_2, b_1, b_2\}$. This is symptomatic of the fact that in general, in system F, *terms depend on types.* It motivates our second 'rule':

SECOND-ORDER MOVES *Whenever a type argument appears, you have to specify it in full immediately.*

[5]See [GLT89b] for a definition. Roughly: $\forall X$ is positive in $\forall X.T$; if $\forall X$ is positive (resp. negative) in T, then it is negative (resp. positive) in $T \to T'$; if $\forall X$ is positive (resp. negative) in T' then it is also positive (resp. negative) in $T \to T'$.

One second-order move may not be enough. For example, going back to the point just before you specified A,

$$t \;=\; \Lambda Y.\, \lambda y^{Y}.\lambda x^{\texttt{Bool}}.\, \underbrace{xAa_1a_2}_{\text{of type } A} \qquad T \;=\; \forall Y.\, Y \to \texttt{Bool} \to Y,$$

suppose you choose

$$A = Y \to \forall Z.\, Z \to Z$$

Then the body xAa_1a_2 is of type $Y \to \forall Z.\, Z \to Z$, so to reach ground type, it requires a dummy term argument corresponding to "$Y \to$" (say b_1), a dummy type argument corresponding to "$\forall Z$" (say B), and another dummy term argument corresponding to "$Z \to$" (say b_2):

$$\begin{array}{rclcrcl} t & = & \Lambda Y.\, \lambda y^{Y}.\lambda x^{\texttt{Bool}}.\, \underbrace{xAa_1a_2}_{Y \to \forall Z. Z \to Z} b_1Bb_2 & \qquad & T & = & \forall Y.\, Y \to \texttt{Bool} \to Y \\ & & & & A & = & Y \to \forall Z.\, Z \to Z \end{array}$$

Following the *second-order moves* rule, you have to specify B immediately. Suppose you choose $B = \forall P.\, P \to \texttt{I} \to Y$, where $\texttt{I} = \forall Q.\, Q \to Q$. Then the body of t is of type $\forall P.\, P \to \texttt{I} \to P$, and requires a type argument (say C) corresponding to "$\forall P$", a term argument (say c_1) corresponding to "$P \to$", and a term argument (say c_2) corresponding to "$\texttt{I} \to$":

$$\begin{array}{rclcrcl} t & = & \Lambda Y.\, \lambda y^{Y}.\lambda x^{\texttt{Bool}}.\, \underbrace{xAa_1a_2b_1Bb_2}_{\forall P. P \to \texttt{I} \to P} Cc_1c_2 & \qquad & T & = & \forall Y.\, Y \to \texttt{Bool} \to Y \\ & & & & A & = & Y \to \forall Z.\, Z \to Z \\ & & & & B & = & \forall P.\, P \to \texttt{I} \to P \end{array}$$

By the *second-order moves* rule, you must specify C immediately. Suppose you choose $C = Y \to Y$. Then the body is of type $Y \to Y$, and we need another dummy argument (say d) corresponding to "$Y \to$":

$$\begin{array}{rclcrcl} t & = & \Lambda Y.\, \lambda y^{Y}.\lambda x^{\texttt{Bool}}.\, \underbrace{xAa_1a_2b_1Bb_2Cc_1c_2}_{Y \to Y}\, d & \qquad & T & = & \forall Y.\, Y \to \texttt{Bool} \to Y \\ & & & & A & = & Y \to \forall Z.\, Z \to Z \\ & & & & B & = & \forall P.\, P \to \texttt{I} \to P \\ & & & & C & = & Y \to Y \end{array}$$

At last the body is of ground type (namely Y), so t is η-long, and your turn is complete. The range from which I can choose my next move is $\{a_1, a_2, b_1, b_2, c_1, c_2, d\}$.

Recall the Hyland/Nickau/Ong intuition that a type is an 'arena' or 'game-board'. Since you have played three second-order moves (i.e., types), there are now a total of four 'arenas' present in the 'hypergame': T, A, B, and C, as displayed next to the term. Observe that each dummy argument during our discussion arose in correspondence with a particular 'input', for example, b_1 corresponded to "$Y \to$" in A, c_2 corresponded to "$\texttt{I} \to$" in C, and so on. Tagging the 'inputs' accordingly gives:

$$\begin{array}{lcrcl} \Lambda Y.\, \lambda y^{Y}.\lambda x^{\texttt{Bool}}.xAa_1a_2b_1Bb_2Cc_1c_2d & \qquad & T & = & \forall Y.\, Y \to (\forall X. X^{a_1} \to X^{a_2} \to X) \to Y \\ & & A & = & Y^{b_1} \to \forall Z.\, Z^{b_2} \to Z \\ & & B & = & \forall P.\, P^{c_1} \to \texttt{I}^{c_2} \to P \\ & & C & = & Y^{d} \to Y \end{array}$$

So depending on which of the arguments $\{a_1, a_2, b_1, b_2, c_1, c_2, d\}$ I choose to inspect with my next move, the 'hypergame' could 'continue in' any of the four 'arenas' present. For example, if I ask

$$c_1 =?$$

then I am 'playing in B'. Now since c_1 is of type P, which you instantiated with $C = Y \to Y$, it has an abstraction corresponding to "$Y \to$":

$$c_1 = \lambda z^Y.?$$

Now you pick a head variable for c_1 ... and so on.

Summary of ideas In this section we have come accross the following ideas, on which the hypergames model is based:

- The *copycat rule*. For 'type-checking' reasons, some of your moves had to be outlawed.
- *Second-order moves*. Corresponding to type arguments in the body of a term, you play 'types as moves'. With the Hyland/Nickau/Ong intuition that 'types are arenas', you are playing 'arenas as moves'.
- Later moves in the hypergame can be located in any of the arenas brought in by your second-order moves.

When it comes to interacting strategies (Chapter 6), the various arenas (e.g. T, A, B, C above) will not be left separate, but will be bundled together in a big 'expanded' arena, the one corresponding to making the obvious substitutions (e.g. in the above, $[A/Y]$, $[B/Z]$, $[C/P]$).

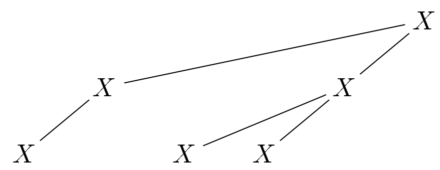
1.1.5 From Hyland/Nickau/Ong arenas to polymorphic arenas

In order to implement the programme for a hypergame model, as suggested by our analysis in the previous section, we require a formal interpretation of a polymorphic type as some kind of 'game board', analogous to the Hyland/Nickau/Ong notion of arena for a *PCF* type. Our solution is a *polymorphic arena*, and we give an informal sketch of the definition. In the product-free case, polymorphic arenas are Böhm trees of some kind, and turn out to correspond to prenex types.

Hyland/Nickau/Ong arenas

Towards the end of section 1.1.1, we sketched how a $\lambda_1^{\to}$ type can be viewed as an 'arena' in which to play a game. Recall that at every stage of the $\lambda_1^{\to}$ dialogue, the choice for each of our moves was determined by decompositions of the form $T = T_1 \to \ldots \to T_n \to X$, in which the type T 'branches like a tree', into an array of n options for moves. A Hyland/Nickau/Ong *PCF* arena,when restricted to $\lambda_1^{\to}$, is simply the tree obtained from a type by iterating this decomposition. For example:

$$(X \to X) \to (X \to X \to X) \to X$$



(See Chapter 3 for more details.) Hyland, Nickau and Ong then go on to define a run of the game as a sequence of vertices, with some additional structure, and satisfying appropriate rules.

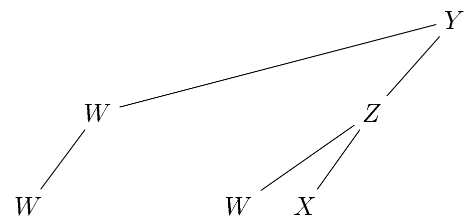
Polymorphic arenas ('Böhm trees for polymorphic types')

We introduce a 'second-order' extension of Hyland/Nickau/Ong arenas, which will play the role of 'game-boards' in our hypergames. In order to obtain the polymorphic arena of a (product-free) system F type, say

$$T \quad = \quad \forall X.\,\forall Y.\,(W \to W) \to (\forall Z.\,W \to X \to Z) \to Y$$

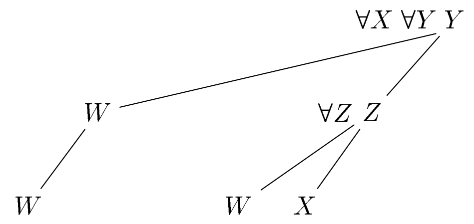
first draw the underlying Hyland/Nickau/Ong arena:

$$\forall X.\,\forall Y.\;(W \to W) \;\to\; (\forall Z.\,W \to X \to Z) \;\to\; Y$$



Then attach quantified type variables next to the root of the appropriate subtree:

$$\forall X.\,\forall Y.\;(W \to W) \;\to\; (\forall Z.\,W \to X \to Z) \;\to\; Y$$



Each quantified variable $\forall V$ binds all occurrences of V in the subtree of the root to which $\forall V$ was attached[6]. Hence a polymorhic arena is a form of Böhm tree [Bar84].

Note that the type

$$T' \quad = \quad \forall X.\,\forall Y.\,(W \to W) \to (W \to \forall Z.\,X \to Z) \to Y$$

(only the position of $\forall Z$ has changed) has the same polymorphic arena, because the root of the subtree of $W \to X \to Z$ is the same as the root of the subtree of $X \to Z$. Likewise,

$$T'' \quad = \quad (W \to W) \to \forall X.\,(W \to X \to \forall Z.Z) \to \forall Y.Y$$

also has the same polymorphic arena.

The prenex normal form[7] of a (product-free) type of system F is obtained by performing all possible conversions of the form $T \to \forall X.T' \leadsto \forall X.T \to T'$ whenever X is not free in T. Hence (in the product-free case) polymorphic arenas are in bijection with prenex normal forms (also known as prenex types).

[6] Avoid any unwanted capture by renaming bound variables if necessary.

[7] A standard notion with respect to quantifiers in first order logic: 'pull all the quantifiers as far to the front of the formula as possible'. Refer to any introductory textbook.

1.1.6 Interaction: the copycat rule corresponds to η-expansion

Games models interpret normalisation as the interaction of strategies. Up to first-order, interaction in the hypergames model is as in the Nickau games model of *PCF* [Nic94][8]. This short section gives an informal overview of the crucial role played by the copycat rule in interaction at second order.

Recall that types are interpreted as polymorphic arenas, and that a type argument T of a polymorphic term t is interpreted as a second-order move, i.e., the introduction into the hypergame of the polymorphic arena A_T interpreting T. During interaction in the hypergame model the copycat rule will apply to the hidden arena A_T. Back at the level of the syntax this can be viewed as a form of η-expansion, as suggested by the example below.

Consider the following normalisation:

$$\begin{array}{rcl}(\Lambda X.\lambda f^X.f)(Y \to Y)\lambda y^Y.y & \rightsquigarrow & (\lambda f^{Y\to Y}.f)\lambda y^Y.y \\ & \rightsquigarrow & (\lambda f^{Y\to Y}.\lambda x^Y.fx)\lambda y^Y.y \\ & \rightsquigarrow & \lambda x^Y.(\lambda y^Y.y)x \\ & \rightsquigarrow & \lambda x^Y.x\end{array}$$

Think of this as the interaction of the 'strategy' $\Lambda X.\lambda f^X.f$ with the arguments $Y \to Y$ and $\lambda y^Y.y$. The first step in the normalisation instantiates the type variable X to $Y \to Y$. In the hypergame this instantiation corresponds to a second-order move, the importation of a hidden 'polymorphic arena' $Y \to Y$. The next step is the η-expansion of $\lambda f^{Y\to Y}.f$ to $\lambda f^{Y\to Y}.\lambda x^Y.fx$. This new aspect of the term $(\ldots\lambda x^Y\ldots x)$ corresponds to copycat on the hidden arena $Y \to Y$. The remaining steps, being first-order, are of less interest.

This example is informal, and serves only as a vague impression to have in the back of the mind when reading Chapter 6 on interaction.

1.2 Overview of chapters

The remaining chapters of the thesis are as follows.

2 Preliminaries
Basic mathematical notation, and categorical semantics for system F.

3 Simple games
We informally sketch a games model for the simply typed λ-calculus, derived from the 'top-down term' methodology introduced in section 1.1.1. We use the model to set out our method of interacting strategies, based on Nickau interaction [Nic94], for later extension to hypergames. We finish by describing a hitherto neglected distinction between Hyland/Ong interaction [HO94] and Nickau interaction.

4 Polymorphic arenas
We formalise the idea of a 'game-board' interpretation of polymorphic type, as outlined above in section 1.1.5.

5 Hypergames
Hyland, Nickau and Ong define a game in an arena using an enriched notion of sequence

[8] Chapter 3 highlights an important and hitherto overlooked difference between Nickau's interaction and Hyland/Ong interaction [HO94].

called a justified sequence. We define a hypersequence as the analogue in our hypergame setting. Hypersequences include polymorphic arenas as 'second-order moves', and 'first-order moves' can be located inside previous 'second-order moves'.

6 Interaction
We define the interaction of strategies in the hypergame model, based on the method set out for $\lambda_1^{\rightarrow}$ in Chapter 3.

7 The categorical model
Using the material developed in chapters 4, 5, and 6, we define a $2\lambda\times$-hyperdoctrine, and hence a model of system F.

8 Full completeness
We prove that the model defined in the previous chapter is fully complete.

9 Conclusions
Concluding remarks.

Chapter 2

Preliminaries

This chapter is structured as follows:

2.1 Basic notation

Notation	**Explanation**
$:=$	definitional equality
$\mathbb{N}$	the set of natural numbers $\{0,1,2,\ldots\}$
$\mathbb{N}^+$	the set of non-zero natural numbers $\{1,2,3,\ldots\}$
$f : X \to Y$	f is a function from X to Y
$f : X \rightharpoonup Y$	f is a partial function from X to Y
$f(a)\Downarrow$	f is defined at a
$f(a)\Uparrow$	f is undefined at a
$A \times B$	product of A and B
$A + B$	sum/coproduct of A and B; if A and B are sets, $+$ is disjoint union
$\pi_1 : A \times B \to A$	first projection associated with a product
$\pi_2 : A \times B \to B$	second projection associated with a product
$\mathsf{in_l} : A \to A + B$	left inclusion associated with a coproduct
$\mathsf{in_r} : B \to A + B$	right inclusion associated with a coproduct
$\mathcal{C}(A,B)$	set of morphisms from A to B in a locally small category $\mathcal{C}$
Set	the category of sets and functions
Cat	the category of small categories and functors

CCCat	the category of small cartesian closed categories and strict cartesian closed functors
$s \cdot t$	the concatenation of the sequences s and t
st	shorthand for concatenation
ϵ	the empty sequence
X^*	the set of finite sequences of elements from the set X
$\|s\|$	the length of the sequence s
$s \sqsubseteq t$	s is a prefix of t, i.e., $t = su$ for some sequence u
$s \sqsubseteq_1 t$	$t = sa$ for some element (singleton sequence) a
$u[v/x]$	the substitution of v for all free occurrences of the variable x in an expression u

2.2 Moves and move-occurrences

The first and last moves of the sequence mnm are 'the same' because they are both "m", yet 'not the same' because they are distinct occurrences of "m". This distinction between "move" and "move-occurrence" can lead to awkwardness in formalising game semantics. In this section we fix conventions and notation designed to deal with this problem.

The problem of occurrences has been familiar to proof theorists for years. Here is a passage from Troelstra and Schwichtenberg's book *Basic Proof Theory* [TS96], Section 1.1.3 entitled *Formulas and formula-occurrences*:

> Formula-occurrences (f.o.'s) will play an even more important role than the formulas themselves. A f.o. is nothing but a formula with a position in another structure (prooftree, sequent, a larger formula etc.). If no confusion is to be feared, we shall permit ourselves a certain 'abus de langage' and talk about formulas when really f.o.'s are meant.

In this thesis we adopt a similar philosophy with respect to moves and move-occurrences.

Equality

If m_1 and m_2 are move-occurrences, equality $m_1 = m_2$ ("m_1 and m_2 are 'the same move'") can be interpreted in one of two ways. Consider once again the sequence $s = mnm$ over the set of moves $M = \{m, n\}$. Let m_1, m_2, m_3, and m_l denote respectively the first, second, third, and last moves of s. Then as move-occurrences, $m_1 \neq m_2$ and $m_2 \neq m_3 = m_l$, but as the underlying elements of M, $m_1 = m_3 = m_l \neq m_2$.

When not stated explicity, we shall leave the type of equality to be inferred from the context. As evidence that such matters are best left to common sense whenever possible, consider the fact that, given the statements

(1) Bill and Ted have the same teacher.
(2) Bill and Ted have the same tie.

it is obvious even to a young child that a different type of equality ('sameness') applies in each case.

Notation

We write $m \in s$ for "m is a move-occurrence of the sequence s". If $m, n \in s$ write $m < n$ if m is strictly before n in s. Write $f : s \rightharpoonup s$ if f is a partial function from the set of move-occurrences of s to the set of move-occurrences of s. Similarly, by identifying a sequence s

with its set of m.o.'s, we can use notation such as $f : s \to t$ for a function assigning m.o.'s of t to m.o.'s of s, or $g : s + t \rightharpoonup u$ for a partial function assigning m.o.'s of u to m.o.'s of s and m.o.'s of t.

More generally, we adopt similar notation for other types of sequences, for example sequences of integers, hypermoves, actions, etc.

2.3 System F

The first three pages of Chapter 5 of Crole's book *Categories for Types* are an intuitive and highly accessible introduction to system F [Cro94]. For a more logical perspective, see *Proofs and Types* [GLT89b], Chapter 11, which includes a nice presentation of how to represent inductive datatypes (natural numbers, lists, etc.) in the calculus, and illustrates the Curry-Howard isomorphism with propositional second-order intuitionistic logic.

We work with the version of system F [Gir71, Rey74] with arrow $(T \to T')$, universal quantification $(\forall X.T)$, and product $(T \times T')$ types. We adopt the notation of Chapter 5 of *Categories for Types*, but without constants. Thus types are generated by the BNF

$$T \;::=\; X \mid \mathsf{Unit} \mid T \times T \mid T \to T \mid \forall X.T$$

where X ranges over a countably infinite set of type variables. Terms are generated by the BNF

$$s \;::=\; x \mid \langle\rangle \mid \langle s, s\rangle \mid \mathsf{fst}(s) \mid \mathsf{snd}(s) \mid \lambda x^T.s \mid ss \mid \Lambda X.s \mid sT$$

where x is any term variable, and T is any type. Well-typed terms-in-context are as given in *Categories for Types*, pages 212–3.

The conversion rules are as follows:

$$\begin{array}{lllll} \beta_1 & (\lambda x^U.v)u & \rightsquigarrow & v[u/x] \\ \beta_2 & (\Lambda X.v)U & \rightsquigarrow & v[U/X] \\ \eta_1 & \lambda x^U.(tx) & \rightsquigarrow & t \\ \eta_2 & \Lambda X.(tX) & \rightsquigarrow & t \end{array}$$

where in η_1 the term variable x does not occur free in t, and in η_2 the type variable X does not occur free in t. We write β for "β_1 and β_2" and likewise η for "η_1 and η_2". System F is strongly normalising with these rules [GLT89b].

Conventions We adopt the following conventions for variables and metavariables:

$$\begin{array}{rl} x, y, z & \text{term variables} \\ X, Y, Z & \text{type variables} \\ t, u, v & \text{terms} \\ T, U, V & \text{types} \end{array}$$

On types $\times$ binds more strongly than $\to$, which in turn binds more strongly than $\forall X$. Both $\to$ and $\times$ associate to the right. For example:

$$\begin{array}{rcl} \forall X.\, U \to V & \equiv & \forall X.\,(U \to V) \\ \forall X.\, U \times V & \equiv & \forall X.\,(U \times V) \\ U \times V \to T & \equiv & (U \times V) \to T \\ U_1 \to U_2 \to \ldots \to U_k \to V & \equiv & U_1 \to (U_2 \to (\ldots \to (U_k \to V)\ldots)) \\ U_1 \times U_2 \times \ldots \times U_k \times V & \equiv & U_1 \times (U_2 \times (\ldots \times (U_k \times V)\ldots)) \end{array}$$

Application st or sT binds more strongly than abstraction $\lambda x^T.$ or $\Lambda X.$, and application associates to the left. For example:

$$\begin{array}{rcl} tuv & \equiv & (tu)v \\ tuV & \equiv & (tu)V \\ \lambda x^T.tu & \equiv & \lambda x^T.(tu) \\ \Lambda X.\,tu & \equiv & \Lambda X.\,(tu) \\ \lambda x^T.tU & \equiv & \lambda x^T.(tU) \\ \Lambda X.\,tU & \equiv & \Lambda X.\,(tU) \end{array}$$

2.4 Categorical semantics of system F

The hypergame model will be presented as a $2\lambda\times$-hyperdoctrine. The thesis has been structured so that the ideas behind the model can be absorbed without any knowledge of category theory; for those without a background in categories think of a $2\lambda\times$-hyperdoctrine as a structure required of the hypergames that assures that one does indeed have a model of system F.

The following definition is from *Categories for Types* [Cro94], with minor changes in notation. A $2\lambda\times$***-hyperdoctrine*** consists of the following data:

1. ***Base category***. A category $\mathbb{B}$ with finite products, whose objects are generated by finite product from a distinguished object $U \in \mathbb{B}$. In other words, for all objects $I \in \mathbb{B}$,

$$I = U^n = \underbrace{U \times U \times \ldots \times U}_{n \text{ occurrences}}$$

   for some $n \geq 0$. In particular, U^0 is terminal.

2. ***Fibres***. A $\mathbb{B}$-indexed cartesian closed category $\mathbb{E} : \mathbb{B}^{op} \to \mathsf{CCCat}$. The cartesian closed category $\mathbb{E}I$ is called the fibre over I, often written $\mathbb{E}_I$, and the set of objects of $\mathbb{E}_I$ is exactly the homset $\mathbb{B}(I, U)$. Given a morphism $\alpha : I \to J$ in $\mathbb{B}$, the strict cartesian closed functor $\mathbb{E}\alpha : \mathbb{E}_J \to \mathbb{E}_I$ is often written $\alpha^* : \mathbb{E}_J \to \mathbb{E}_I$, and is called the reindexing functor of α.

3. ***Indexed products***. For each object $I \in \mathbb{B}$ we are given a functor $\Pi_I : \mathbb{E}_{I\times U} \to \mathbb{E}_I$ which is right adjoint to the reindexing functor $\pi_I^* : \mathbb{E}_I \to \mathbb{E}_{I\times U}$ of the product projection $\pi_I : I \times U \to I$. Furthermore, for all morphisms $\alpha : I \to J$ in $\mathbb{B}$, the following diagram of functors

$$\begin{tikzcd} \mathbb{E}_{J\times U} \arrow[r, "\Pi_J"] \arrow[d, "(\alpha \times \mathit{id}_U)^*"'] & \mathbb{E}_J \arrow[d, "\alpha^*"] \\ \mathbb{E}_{I\times U} \arrow[r, "\Pi_I"'] & \mathbb{E}_I \end{tikzcd}$$

   commutes, and the canonical natural transformation $\alpha^* \circ \Pi_J \longrightarrow \Pi_I \circ (\alpha \times \mathit{id}_U)^*$ is an identity. The commutation of this diagram, together with the specification on the canonical natural tansformation, is called the Beck-Chevalley condition for $2\lambda\times$-hyperdoctrines.

Intuitively, the Beck-Chevalley condition means *quantification commutes with substitution* (when the about-to-be-quantified variable is left untouched). Refer to *Categories for Types* for formal details of how a $2\lambda\times$-hyperdoctrine is a model of system F.

Remark The original categorical presentation of the hypergame model in the extended abstract [Hug97] was in terms of fibrations rather than $2\lambda\times$-hyperdoctrines. After canvassing opinions from a number of researchers, it turned out that most found the hyperdoctrine presentation of a model of system F to be more intuitive, because substitution (reindexing) functors are more concrete than the universal property of cartesian maps.

Chapter 3

Simple games

We illustrate the key ideas behind Hyland/Nickau/Ong *PCF* games [HO94, Nic94] by stripping them down to the bare minimum: a model of the pure simply typed lambda calculus $\lambda_1^{\rightarrow}$ with types generated by arrow $\rightarrow$ from a single base type variable X. This elaborates on the ideas sketched in section 1.1.1. The chapter is relatively informal, because the material is a special case of the second-order games presented formally in Chapters 4, 5, and 6.

It is now folklore in game semantics that the Hyland/Ong and Nickau models are the same. At the end of the chapter we highlight a hitherto neglected difference between the interaction in the two models. For technical reasons, we took Nickau's interaction as the first-order basis of hypergame interaction.

3.1 Basic ingredients

There are two players, O and P. Think of P as System, Program, or Us, and think of O as Environment, Context, or Them. In our 'discussion of a term' in section 1.1.1 (page 6), I was O, and you were P.

The other basic ingredients are *arenas*, *positions*, *winning strategies*, and the *interaction* of strategies. Before giving the details, we reiterate the following associations outlined in section 1.1.1:

Syntax	Game
type	arena
term	winning strategy (for P)
computation/normalisation	interaction

Arenas

The ***arena*** of a type is the finite tree defined as follows. We start with an example. Given the type

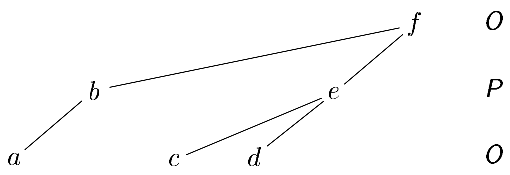
$$(X \rightarrow X) \rightarrow (X \rightarrow X \rightarrow X) \rightarrow X,$$

first 'tag' the type variables in order to distinguish between them:

$$(X^a \rightarrow X^b) \rightarrow (X^c \rightarrow X^d \rightarrow X^e) \rightarrow X^f$$

and then the arena is

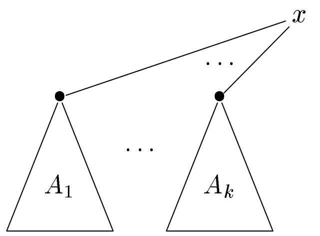
$$(X^a \to X^b) \to (X^c \to X^d \to X^e) \to X^f$$



The vertices of an arena are called ***moves***, and if v is the parent of w, we say that v ***enables*** w. As indicated in the picture, moves at odd depth are associated with O, and moves at even depth are associated with P.

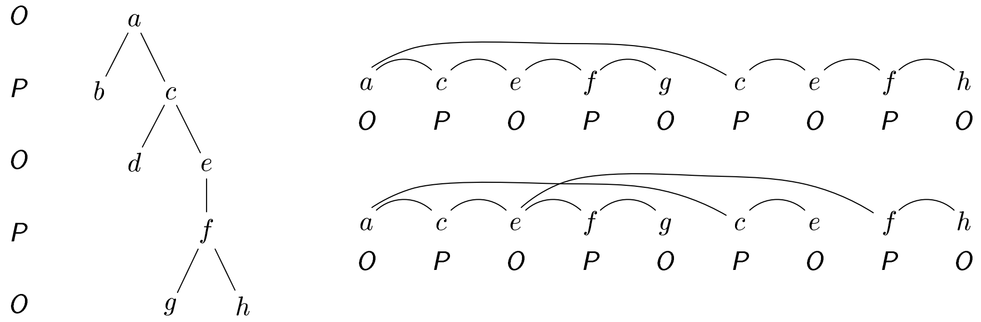
The general rule for constructing an arena is suggested by the decompositions $T = T_1 \to \ldots \to T_k \to X$ in our 'discussions of a term of type T' of section 1.1.1. Given arenas $A_1, \ldots, A_k$ for types $T_1, \ldots, T_k$, the arena of $T_1 \to \ldots \to T_k \to X$ is obtained by grafting the A_i underneath a new move x corresponding to X:

$$T_1 \;\to\; \ldots \;\to\; T_k \;\to\; X$$



Positions

A ***position*** in an arena is a sequence of moves each of which (apart from the first) is equipped with a ***justification pointer*** to an earlier enabling move by the opposite player. The first move is by O, and the pointers from the subsequent O moves are forced to target the preceding move. For example:



So a position is an 'interlacing' of paths down the arena. Note that although the underlying sequences of moves are the same, the two positions above differ because the second occurrence of f is justified by a different occurrence of e in each case.

Intuition As motivated by our 'discussion of a term of type T', think roughly in terms of P-moves as indicating a choice of head-variable, and O-moves as indicating a choice of an argument. For example, as a position, our ‘discussion of a term’ on page 6 looks (informally) like this:

$$\begin{array}{ccccccc} O & P & O & P & O & P & O \\ t & x_3 & a_5 & y_2 & b_4 & x_2 & c_5 \end{array}$$

A justification pointer from a P-move (head-variable) indicates the argument (O-move) from which the head-variable is chosen, i.e., the argument containing the ‘batch’ of λ-abstractions from which the head-variable is chosen. O’s justification pointers always target the previous P-move, because O is always asks about an argument of the most recent head-variable. So O pointers are somewhat redundant.

Remark Hyland, Nickau and Ong originally defined a notion of justified sequence which was symmetric in the sense that O was not subject to our constraint on pointers. For technical reasons, in moving to hypergames it was necessary to adopt the asymmetric viewpoint.

Winning strategies

A winning strategy (for P) is a pre-determined move in response to every move played by O, such that no matter how O decides to play, he will inevitably be ‘run out of moves’. We clarify the definition with a few examples. A formal definition is provided in chapter 5.

Examples

We consider some ‘discussions’ of terms, and then show the corresponding winning strategies in arenas.

Booleans Let `Bool` be the type $X \to X \to X$, so-called because there are two normal forms of this type:

$$\begin{array}{rcl} \texttt{true} & = & \lambda t^X.\lambda f^X.t \\ \texttt{false} & = & \lambda t^X.\lambda f^X.f \end{array}$$

These two η-long, β-normal terms correspond to the following ‘strategies’ in the ‘top-down term-discussion’ of a term b of type `Bool`. The ‘strategy’ of `true` is to select t as head variable:

$$\begin{array}{lrclrcl} O \text{ chooses } b & b & = & \lambda t^X.\lambda f^X.? & b & : & X \to X \to X \\ P \text{ chooses } t & b & = & \lambda t^X.\lambda f^X.t & t & : & X \end{array}$$

The ‘strategy’ of `false` is to select f as head variable:

$$\begin{array}{lrclrcl} O \text{ chooses } b & b & = & \lambda t^X.\lambda f^X.? & b & : & X \to X \to X \\ P \text{ chooses } f & b & = & \lambda t^X.\lambda f^X.f & f & : & X \end{array}$$

In either case, the game is over, since t and f are of ground type X: O has been run out of moves, since t and f take no arguments.

In terms of arenas and winning strategies, the corresponding picture is as follows. The arena of `Bool`, say via the ‘tagging’ $X^t \to X^f \to X^b$, is

$$\begin{array}{lccc} O & & b & \\ P & t & & f \end{array}$$

The winning strategies are:

1. `true` plays t in response to b: $\quad b \quad t$

2. `false` plays f in response to b: $\quad b \quad f$

O is 'run out of moves' after either strategy because there are no moves below t or f in the arena.

Boolean `not` function Consider the term

$$\texttt{not} = \lambda b_1^{\texttt{Bool}}.\lambda t^X.\lambda f^X.b_1 f t = \lambda b_1^{X\to X\to X}.\lambda t^X.\lambda f^X.b_1 f t$$

of type

$$\texttt{Bool} \to \texttt{Bool} = (X \to X \to X) \to (X \to X \to X)$$

which has β-reductions

$$\begin{aligned} \texttt{not}(\texttt{true}) &\rightsquigarrow_\beta^* \texttt{false} \\ \texttt{not}(\texttt{false}) &\rightsquigarrow_\beta^* \texttt{true} \end{aligned}$$

Here is a discussion of `not`:

O chooses `not`	`not`	$=$	$\lambda b_1^{\texttt{Bool}}.\lambda t^X.\lambda f^X.?$	`not`	:	$\texttt{Bool} \to X \to X \to X$
P chooses b_1	`not`	$=$	$\lambda b_1^{\texttt{Bool}}.\lambda t^X.\lambda f^X.b_1 u v$	b_1	:	$X \to X \to X$
O chooses u	u	$=$	$?$	u	:	X
P chooses f	u	$=$	f	f	:	X

Had O decided to look at the second argument v of b_1 instead, the discussion would have been:

O chooses `not`	`not`	$=$	$\lambda b_1^{\texttt{Bool}}.\lambda t^X.\lambda f^X.?$	`not`	:	$\texttt{Bool} \to X \to X \to X$
P chooses b_1	`not`	$=$	$\lambda b_1^{\texttt{Bool}}.\lambda t^X.\lambda f^X.b_1 u v$	b_1	:	$X \to X \to X$
O chooses v	v	$=$	$?$	v	:	X
P chooses t	v	$=$	t	t	:	X

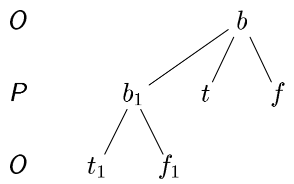
In terms of arenas and winning strategies, the corresponding picture is as follows. The arena of

$$\texttt{Bool} \to \texttt{Bool} = (X \to X \to X) \to (X \to X \to X)$$

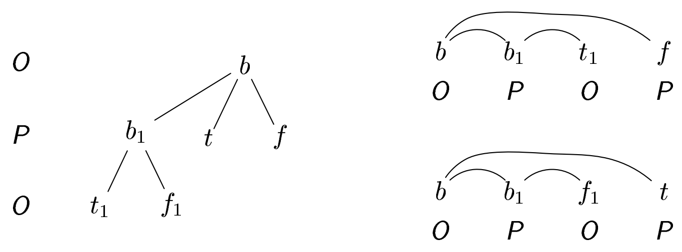
via the tagging

$$(X^{t_1} \to X^{f_1} \to X^{b_1}) \to (X^t \to X^f \to X^b)$$

is



The two discussions of `not` correspond to the following positions:



Naturals Let `Nat` be the type

$$(X \to X) \to X \to X$$

whose closed, η-long, β-normal forms are the Church numerals:

$$\underline{m} = \lambda s^{X \to X}.\, \lambda z^{X}.\, \underbrace{s\,(s\,(s\,(\ldots(s}_{m \text{ occurrences}}\, z)\ldots))).$$

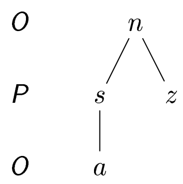
Think of s as ‘successor’ and z as ‘zero’. Here is a discussion of the Church numeral $\underline{m}$:

O chooses n	n	$=$	$\lambda s^{X\to X}.\lambda z^{X}.?$	n	:	$(X \to X) \to X \to X$
P chooses s	n	$=$	$\lambda s^{X\to X}.\lambda z^{X}.sa_1$	s	:	$X \to X$
O chooses a_1	a_1	$=$	$?$	a_1	:	X
P chooses s	a_1	$=$	sa_2	s	:	$X \to X$
O chooses a_2	a_2	$=$	$?$	a_2	:	X
P chooses s	a_2	$=$	sa_3	s	:	$X \to X$
$\vdots$			$\vdots$			$\vdots$
P chooses s	a_{m-1}	$=$	sa_m	s	:	$X \to X$
O chooses a_m	a_m	$=$	$?$	a_m	:	X
P chooses z	a_m	$=$	z	z	:	X

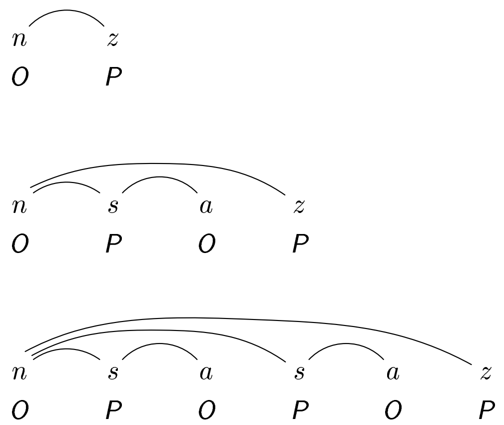
In terms of arenas and winning strategies, the corresponding picture is as follows. The arena of `Nat` $= (X \to X) \to X \to X$ via the tagging

$$(X^a \to X^s) \to X^z \to X^n$$

is



Think of O’s move a as ‘applied to’ or ‘argument=?’. The discussions of $\underline{0}$, $\underline{1}$, and $\underline{2}$ correspond to the following positions:



The strategy for $\underline{m}$ plays s m times before playing z.

Successor Consider the term[1]

$$
\begin{array}{rcl}
\mathtt{suc} & = & \lambda n_1^{\mathtt{Nat}}.\lambda s^{X\to X}.\lambda z^X.n_1(\lambda a_1^X.sa_1)(sz) \\
 & = & \lambda n_1^{(X\to X)\to X\to X}.\lambda s^{X\to X}.\lambda z^X.n_1(\lambda a_1^X.sa_1)(sz)
\end{array}
$$

of type

$$
\mathtt{Nat}\to\mathtt{Nat} \quad = \quad ((X\to X)\to X\to X)\to((X\to X)\to X\to X)
$$

which has β-reductions

$$
\mathtt{suc}(\underline{m}) \quad \rightsquigarrow_\beta^* \quad \underline{m+1}
$$

for each church numeral $\underline{m}$.

Here is the discussion of suc when O chooses to look at the first argument of n_1 (we abbreviate "chooses" to "ch."):

$$
\begin{array}{llcllcl}
O \text{ ch. } n & n & = & \lambda n_1^{\mathtt{Nat}}.\lambda s^{X\to X}.\lambda z^X.? & n & : & \mathtt{Nat}\to(X\to X)\to X\to X \\
P \text{ ch. } n_1 & n & = & \lambda n_1^{\mathtt{Nat}}.\lambda s^{X\to X}.\lambda z^X.n_1 s_1 z_1 & n_1 & : & (X\to X)\to X\to X \\
O \text{ ch. } s_1 & s_1 & = & \lambda a_1^X.? & s_1 & : & X\to X \\
P \text{ ch. } s & s_1 & = & \lambda a_1^X.sa & s & : & X\to X \\
O \text{ ch. } a & a & = & ? & a & : & X \\
P \text{ ch. } a_1 & a & = & a_1 & a_1 & : & X
\end{array}
$$

Had O decided to look at the second argument of n_1 instead, the discussion would have been:

$$
\begin{array}{llcllcl}
O \text{ ch. } n & n & = & \lambda n_1^{\mathtt{Nat}}.\lambda s^{X\to X}.\lambda z^X.? & n & : & \mathtt{Nat}\to(X\to X)\to X\to X \\
P \text{ ch. } n_1 & n & = & \lambda n_1^{\mathtt{Nat}}.\lambda s^{X\to X}.\lambda z^X.n_1 s_1 z_1 & n_1 & : & (X\to X)\to X\to X \\
O \text{ ch. } z_1 & z_1 & = & ? & z_1 & : & X \\
P \text{ ch. } s & z_1 & = & sa & s & : & X\to X \\
O \text{ ch. } a & a & = & ? & a & : & X \\
P \text{ ch. } z & a & = & z & z & : & X
\end{array}
$$

[1] Without η-expansion suc would be $\lambda n_1^{\mathtt{Nat}}.\lambda s^{X\to X}.\lambda z^X.n_1 s(sz)$.

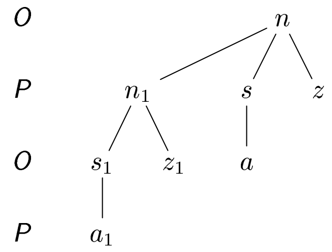
In terms of arenas and winning strategies, the corresponding picture is as follows. The arena of

$$\mathtt{Nat} \to \mathtt{Nat} = ((X \to X) \to X \to X) \to ((X \to X) \to X \to X)$$

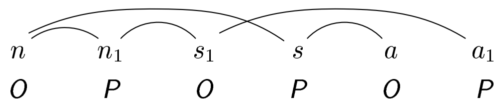
via the tagging

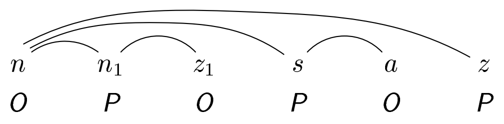
$$((X^{a_1} \to X^{s_1}) \to X^{z_1} \to X^{n_1}) \to ((X^a \to X^s) \to X^z \to X^n)$$

is



The two discussions of `suc` correspond to the following positions:





3.2 Interaction

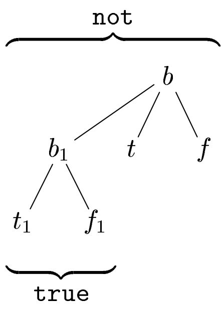
We interpret computation (i.e. normalisation or reduction) as the interaction of strategies. Think of a strategy as a kind of process, in the sense of concurrency theory (e.g. *CCS* [Mil80] or *CSP* [Hoa85]), namely as the transition graph (tree) consisting of its set of stimulus-response traces (sequences). Then interaction is a form of 'parallel composition with hiding'. We shall explore this idea with three examples of interaction, corresponding to the reductions:

$$\begin{array}{rcl} \mathtt{not(true)} & \rightsquigarrow_\beta^* & \mathtt{false} \\ \mathtt{suc}(\underline{0}) & \rightsquigarrow_\beta^* & \underline{1} \\ \mathtt{suc}(\underline{1}) & \rightsquigarrow_\beta^* & \underline{2} \end{array}$$

The third example requires a concept introduced by Hyland, Nickau, and Ong, which is critical for making interaction succeed: the *view* of a sequence with justification pointers.

Example 1

We consider the interaction of the `not` strategy on $\mathtt{Bool}_1 \to \mathtt{Bool}$ with the strategy `true` on $\mathtt{Bool}_1$. (We subscript the input copy of `Bool` for distinguishability.)



By playing `not` and `true` in ‘parallel’, and then ‘hiding’ their interaction on `Bool`, we shall obtain a strategy ‘?’ on `Bool`. We anticipate, of course, that ‘?’ will turn out to be the winning strategy `false`.

There are three points of view during the interaction: (i) the play so far in `Bool`, (ii) the play so far in $\texttt{Bool}_1 \to \texttt{Bool}$, and (iii) the play so far in $\texttt{Bool}_1$. The first move is by O in `Bool`:

$\boxed{\overset{?}{\texttt{Bool}}}$	$\underset{O}{b}$
$\boxed{\texttt{Bool}_1 \overset{\texttt{not}}{\to} \texttt{Bool}}$	
$\boxed{\overset{\texttt{true}}{\texttt{Bool}_1}}$	

This is ‘transmitted’ to $\texttt{Bool}_1 \to \texttt{Bool}$ immediately, as the opening O-move of that arena:

$\boxed{\overset{?}{\texttt{Bool}}}$	$\underset{O}{b}$
$\boxed{\texttt{Bool}_1 \overset{\texttt{not}}{\to} \texttt{Bool}}$	$\underset{O}{b}$
$\boxed{\overset{\texttt{true}}{\texttt{Bool}_1}}$	

This move acts as a stimulus for `not`. Looking at its ‘crib-sheet’, consisting of the two positions on page 26, we see that the response of `not` is as follows:

$$\boxed{\overset{?}{\texttt{Bool}}} \qquad \begin{array}{c} b \\ O \end{array}$$

$$\boxed{\texttt{Bool}_1 \overset{\texttt{not}}{\to} \texttt{Bool}} \qquad \begin{array}{cc} b & b_1 \\ O & P \end{array}$$

$$\boxed{\overset{\texttt{true}}{\texttt{Bool}_1}}$$

The response is transmitted to `true` in $\texttt{Bool}_1$ to arrive as an opening O-stimulus:

$$\boxed{\overset{?}{\texttt{Bool}}} \qquad \begin{array}{c} b \\ O \end{array}$$

$$\boxed{\texttt{Bool}_1 \overset{\texttt{not}}{\to} \texttt{Bool}} \qquad \begin{array}{cc} b & b_1 \\ O & P \end{array}$$

$$\boxed{\overset{\texttt{true}}{\texttt{Bool}_1}} \qquad \begin{array}{cc} & b_1 \\ & O \end{array}$$

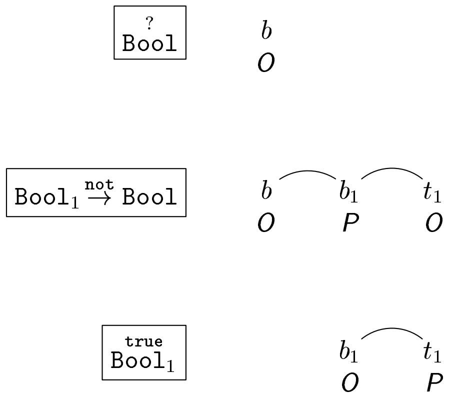
The response of `true` is:

$$\boxed{\overset{?}{\texttt{Bool}}} \qquad \begin{array}{c} b \\ O \end{array}$$

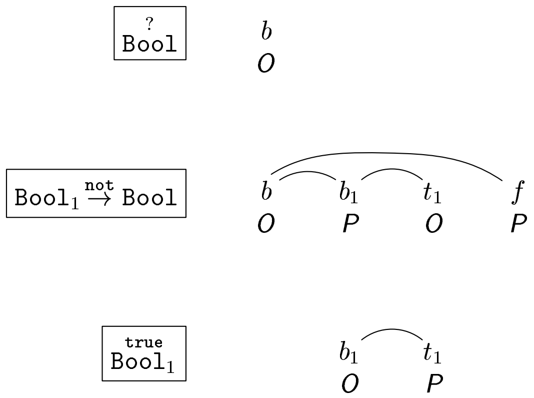
$$\boxed{\texttt{Bool}_1 \overset{\texttt{not}}{\to} \texttt{Bool}} \qquad \begin{array}{cc} b & b_1 \\ O & P \end{array}$$

$$\boxed{\overset{\texttt{true}}{\texttt{Bool}_1}} \qquad \begin{array}{ccc} & b_1 & t_1 \\ & O & P \end{array}$$

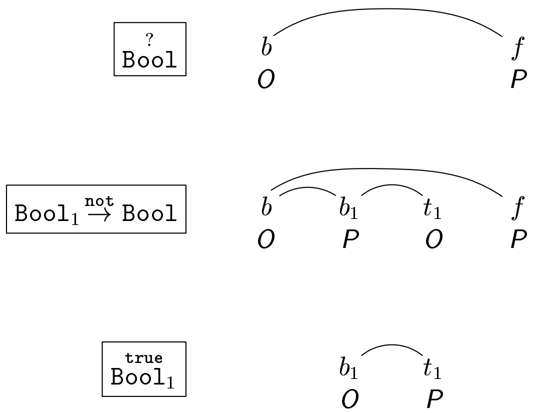
which is duly sent to $\texttt{Bool}_1 \to \texttt{Bool}$, inside which it appears as an O-move:



Again, looking at `not`'s cribsheet, its response is:



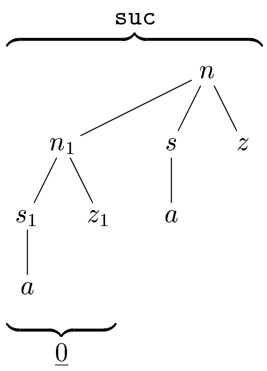
which, since it is in the `Bool` sub-arena of $\texttt{Bool}_1 \to \texttt{Bool}$, is sent as the final 'team-response' in `Bool`:



Now 'hiding' the bottom two rows, and just looking at what happened on `Bool`, we see that the composite '?' is indeed the strategy `false`.

Example 2

We consider the interaction of the `suc` strategy on $\texttt{Nat}_1 \to \texttt{Nat}$ with the strategy $\underline{0}$ on $\texttt{Nat}_1$. (We subscript the input copy of `Nat` for distinguishability.)



By playing `suc` and $\underline{0}$ in 'parallel', and then 'hiding' their interaction on `Nat`, we hope to observe the winning strategy $\underline{1}$.

There are three points of view during the interaction: (i) the play so far in `Nat`, (ii) the play so far in $\mathtt{Nat}_1 \to \mathtt{Nat}$, and (iii) the play so far in $\mathtt{Nat}_1$. The first move is by O in `Nat`:

$$\boxed{\overset{?}{\mathtt{Nat}}} \qquad \begin{matrix} n \\ O \end{matrix}$$

$$\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}}$$

$$\boxed{\overset{\underline{0}}{\mathtt{Nat}_1}}$$

This is 'transmitted' to $\mathtt{Nat}_1 \to \mathtt{Nat}$ immediately, as the opening O-move of that arena:

$$\boxed{\overset{?}{\mathtt{Nat}}} \qquad \begin{matrix} n \\ O \end{matrix}$$

$$\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} \qquad \begin{matrix} n \\ O \end{matrix}$$

$$\boxed{\overset{\underline{0}}{\mathtt{Nat}_1}}$$

This move acts as a stimulus for `suc`. Looking at its 'crib-sheet', consisting of the two positions on page 28, we see that the response of `suc` is to play:

$$\begin{array}{lccc}
\boxed{\overset{?}{\mathtt{Nat}}} & \begin{array}{c} n \\ O \end{array} & & \\[2em]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & \begin{array}{c} n \\ O \end{array} & \begin{array}{c} n_1 \\ P \end{array} & \\[2em]
\boxed{\overset{\underline{0}}{\mathtt{Nat}_1}} & & &
\end{array}$$

The response is transmitted to $\underline{0}$ in $\mathtt{Nat}_1$ to arrive as an opening O-stimulus:

$$\begin{array}{lccc}
\boxed{\overset{?}{\mathtt{Nat}}} & \begin{array}{c} n \\ O \end{array} & & \\[2em]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & \begin{array}{c} n \\ O \end{array} & \begin{array}{c} n_1 \\ P \end{array} & \\[2em]
\boxed{\overset{\underline{0}}{\mathtt{Nat}_1}} & & \begin{array}{c} n_1 \\ O \end{array} &
\end{array}$$

The response of $\underline{0}$ is:

$$\begin{array}{lccc}
\boxed{\overset{?}{\mathtt{Nat}}} & \begin{array}{c} n \\ O \end{array} & & \\[2em]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & \begin{array}{c} n \\ O \end{array} & \begin{array}{c} n_1 \\ P \end{array} & \\[2em]
\boxed{\overset{\underline{0}}{\mathtt{Nat}_1}} & & \begin{array}{c} n_1 \\ O \end{array} & \begin{array}{c} z_1 \\ P \end{array}
\end{array}$$

which is duly sent to $\mathtt{Nat}_1 \to \mathtt{Nat}$, inside which it appears as an O-move:

$$\begin{array}{llllll}
\boxed{\overset{?}{\mathtt{Nat}}} & & n & & & \\
 & & O & & & \\[2em]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & & n & n_1 & z_1 & \\
 & & O & P & O & \\[2em]
\boxed{\overset{\underline{0}}{\mathtt{Nat}_1}} & & & n_1 & z_1 & \\
 & & & O & P &
\end{array}$$

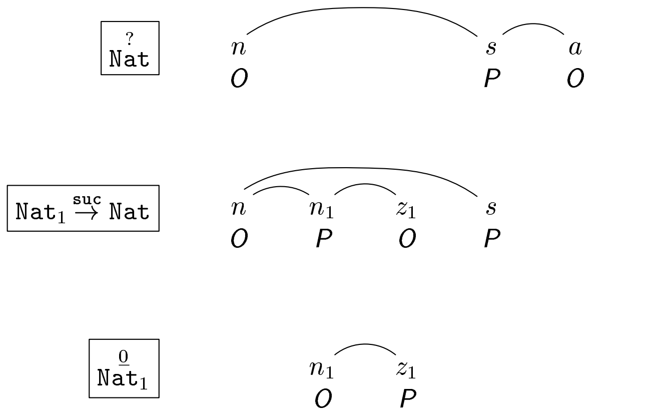
Again, looking at `suc`'s cribsheet, its response is:

$$\begin{array}{llllll}
\boxed{\overset{?}{\mathtt{Nat}}} & & n & & & \\
 & & O & & & \\[2em]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & & n & n_1 & z_1 & s \\
 & & O & P & O & P \\[2em]
\boxed{\overset{\underline{0}}{\mathtt{Nat}_1}} & & & n_1 & z_1 & \\
 & & & O & P &
\end{array}$$

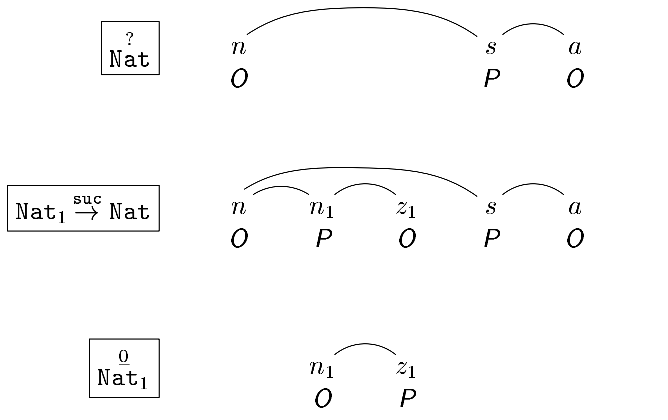
which, since it is in the `Nat` sub-arena of $\mathtt{Nat}_1 \to \mathtt{Nat}$, is sent as the first 'team-response' in `Nat`:

$$\begin{array}{llllll}
\boxed{\overset{?}{\mathtt{Nat}}} & & n & & & s \\
 & & O & & & P \\[2em]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & & n & n_1 & z_1 & s \\
 & & O & P & O & P \\[2em]
\boxed{\overset{\underline{0}}{\mathtt{Nat}_1}} & & & n_1 & z_1 & \\
 & & & O & P &
\end{array}$$

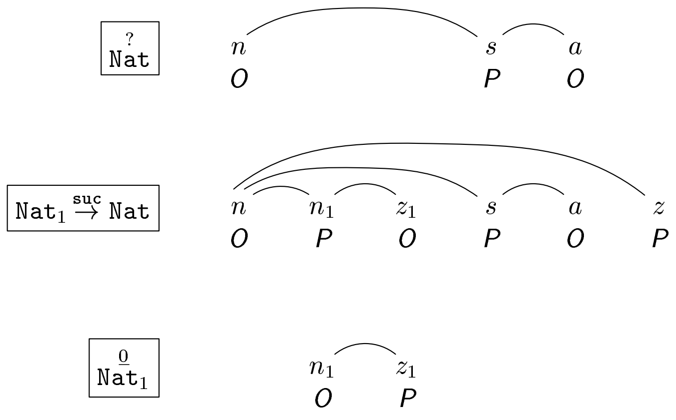
Now it is O to play in `Nat`. Since s has a unique child a, there is no choice but to play it:



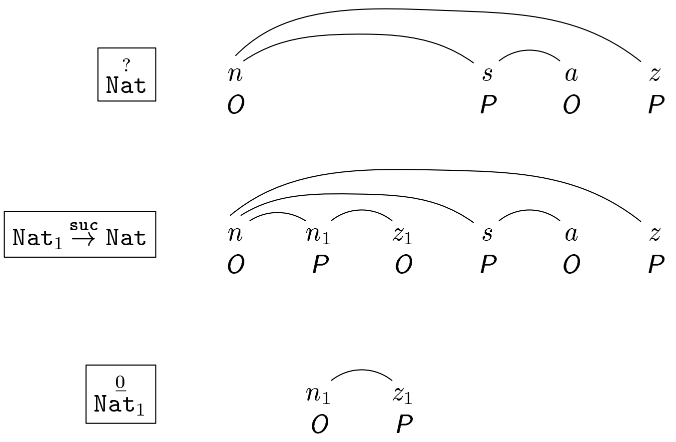
This move is in the Nat sub-arena of $\mathtt{Nat}_1 \to \mathtt{Nat}$, so is transmitted as another stimulus for `suc`:



Looking at the 'crib-sheet' (page 28), we see that `suc` determines to play:



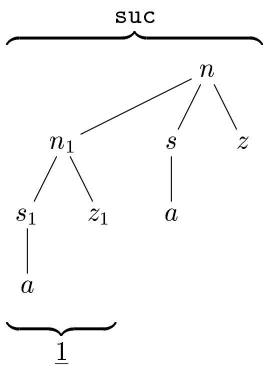
This is transmitted back up to `Nat` as the final move of the interaction:



As we can see, the end result of the interaction is the strategy $\underline{1}$ in `Nat`.

Example 3

This example will introduce an important concept in the Hyland/Ong/Nickau approach to game semantics: the ***view*** of a sequence with justification pointers. We consider the interaction of `suc` on $\texttt{Nat}_1 \to \texttt{Nat}$ with $\underline{1}$ on $\texttt{Nat}_1$. (Again, we subscript the input copy of `Nat` for distinguishability.)



By playing `suc` and $\underline{1}$ in 'parallel', and then 'hiding' their interaction on `Nat`, we hope to observe the winning strategy $\underline{2}$.

The first four events are exactly as in the last interaction, (i) send n from `Nat`, (ii) receive n in $\texttt{Nat}_1 \to \texttt{Nat}$ (iii) send n_1 from $\texttt{Nat}_1 \to \texttt{Nat}$; (iv) receive n_1 in $\texttt{Nat}_1$:

$$\begin{array}{lccc}
\boxed{\overset{?}{\mathtt{Nat}}} & n & & \\
 & O & & \\
 & & & \\
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & n & n_1 & \\
 & O & P & \\
 & & & \\
\boxed{\overset{\underline{1}}{\mathtt{Nat}_1}} & & n_1 & \\
 & & O &
\end{array}$$

At this point, $\underline{1}$ deviates from the strategy $\underline{0}$ by responding to n_1 with:

$$\begin{array}{lccc}
\boxed{\overset{?}{\mathtt{Nat}}} & n & & \\
 & O & & \\
 & & & \\
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & n & n_1 & \\
 & O & P & \\
 & & & \\
\boxed{\overset{\underline{1}}{\mathtt{Nat}_1}} & & n_1 & s_1 \\
 & & O & P
\end{array}$$

This is received as an O-move by `suc` in $\mathtt{Nat}_1 \to \mathtt{Nat}$:

$$\begin{array}{lccc}
\boxed{\overset{?}{\mathtt{Nat}}} & n & & \\
 & O & & \\
 & & & \\
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & n & n_1 & s_1 \\
 & O & P & O \\
 & & & \\
\boxed{\overset{\underline{1}}{\mathtt{Nat}_1}} & & n_1 & s_1 \\
 & & O & P
\end{array}$$

As indicated by the first of the two positions of the cribsheet (page 28), `suc`'s response is:

$$\boxed{\overset{?}{\texttt{Nat}}} \qquad \begin{array}{cccc} n & & & \\ O & & & \end{array}$$

$$\boxed{\texttt{Nat}_1 \overset{\texttt{suc}}{\to} \texttt{Nat}} \qquad \begin{array}{cccc} n & n_1 & s_1 & s \\ O & P & O & P \end{array}$$

$$\boxed{\overset{1}{\texttt{Nat}_1}} \qquad \begin{array}{cccc} & n_1 & s_1 & \\ & O & P & \end{array}$$

Sending this up to `Nat` provides the response to O's original move in `Nat`:

$$\boxed{\overset{?}{\texttt{Nat}}} \qquad \begin{array}{cccc} n & & & s \\ O & & & P \end{array}$$

$$\boxed{\texttt{Nat}_1 \overset{\texttt{suc}}{\to} \texttt{Nat}} \qquad \begin{array}{cccc} n & n_1 & s_1 & s \\ O & P & O & P \end{array}$$

$$\boxed{\overset{1}{\texttt{Nat}_1}} \qquad \begin{array}{cccc} & n_1 & s_1 & \\ & O & P & \end{array}$$

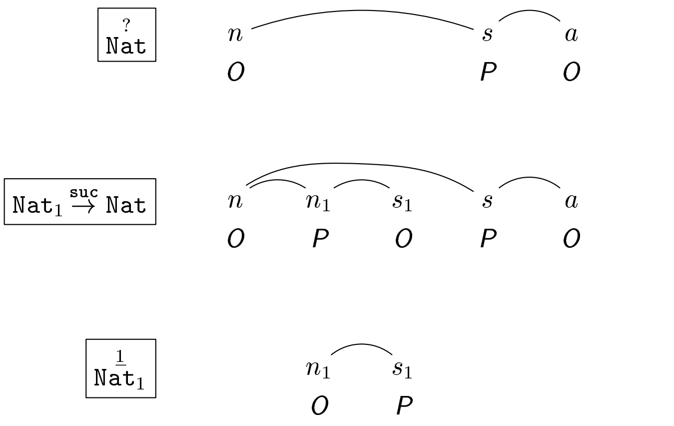
Since s has but one child a in `Nat`, O has no choice but to play it:

$$\boxed{\overset{?}{\texttt{Nat}}} \qquad \begin{array}{ccccc} n & & & s & a \\ O & & & P & O \end{array}$$

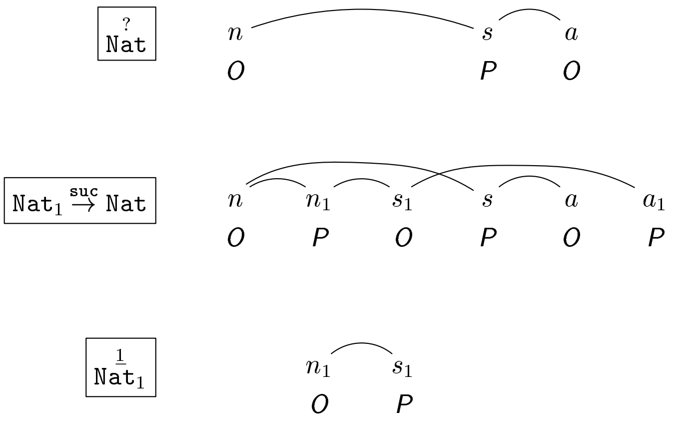
$$\boxed{\texttt{Nat}_1 \overset{\texttt{suc}}{\to} \texttt{Nat}} \qquad \begin{array}{cccc} n & n_1 & s_1 & s \\ O & P & O & P \end{array}$$

$$\boxed{\overset{1}{\texttt{Nat}_1}} \qquad \begin{array}{cccc} & n_1 & s_1 & \\ & O & P & \end{array}$$

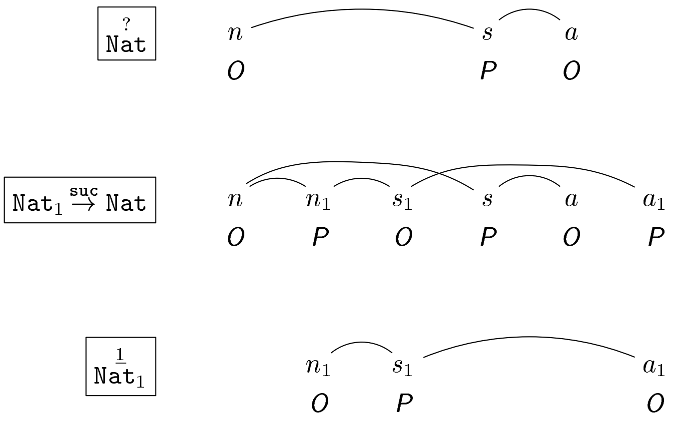
This is received in $\texttt{Nat}_1 \to \texttt{Nat}$:



As determined by the first of the two positions of its ‘crib-sheet’, `suc` is programmed to play:



Copying this move accross to $\texttt{Nat}_1$, where it is seen as an O-move, we have:



The response of $\underline{1}$, having already played s_1 once, is to play z_1, ‘winning’ the game in $\texttt{Nat}_1$:

$$
\begin{array}{c|cccccccc}
\boxed{\overset{?}{\mathtt{Nat}}} & n & & & s & a & & & \\
 & O & & & P & O & & & \\[2ex]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & n & n_1 & s_1 & s & a & a_1 & & \\
 & O & P & O & P & O & P & & \\[2ex]
\boxed{\overset{1}{\mathtt{Nat}_1}} & & n_1 & s_1 & & & a_1 & z_1 & \\
 & & O & P & & & O & P &
\end{array}
$$

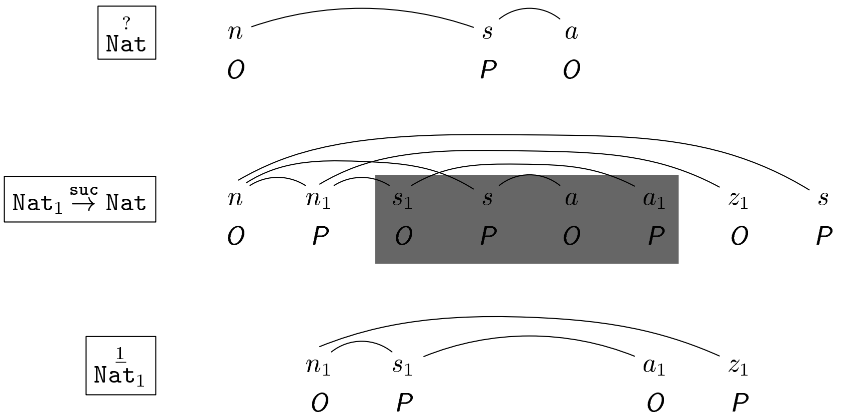
We copy this back up to $\mathtt{Nat}_1 \to \mathtt{Nat}$, where it appears as an O-move:

$$
\begin{array}{c|cccccccc}
\boxed{\overset{?}{\mathtt{Nat}}} & n & & & s & a & & & \\
 & O & & & P & O & & & \\[2ex]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & n & n_1 & s_1 & s & a & a_1 & z_1 & \\
 & O & P & O & P & O & P & O & \\[2ex]
\boxed{\overset{1}{\mathtt{Nat}_1}} & & n_1 & s_1 & & & a_1 & z_1 & \\
 & & O & P & & & O & P &
\end{array}
$$

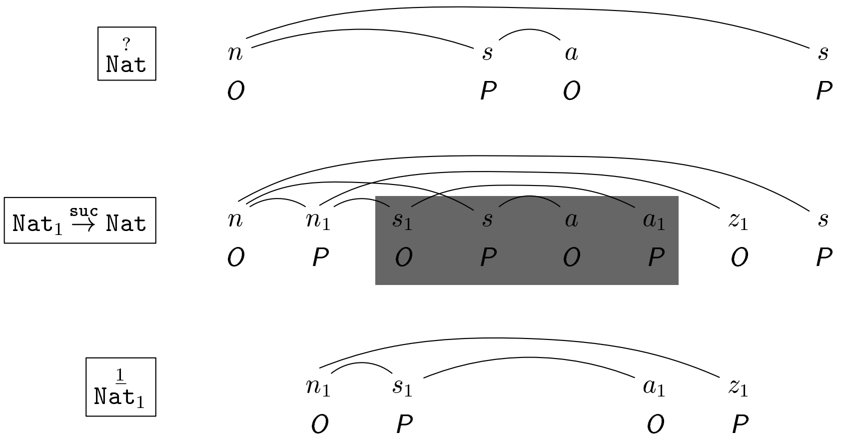
At this point, all of a sudden there is a problem: the current state in $\mathtt{Nat}_1 \to \mathtt{Nat}$ is *not* a position, because the justifier of z_1 is not the immediately preceding P-move. (Recall our definition of position on page 23.) The solution of Hyland, Nickau and Ong was to ***hide*** everything between the source z_1 and the target n_1 of the O-pointer, so that the ***view***, as far as `suc` is concerned, is a well-defined position:

$$
\begin{array}{c|cccccccc}
\boxed{\overset{?}{\mathtt{Nat}}} & n & & & s & a & & & \\
 & O & & & P & O & & & \\[2ex]
\boxed{\mathtt{Nat}_1 \overset{\mathtt{suc}}{\to} \mathtt{Nat}} & n & n_1 & s_1 & s & a & a_1 & z_1 & \\
 & O & P & O & P & O & P & O & P \\[2ex]
\boxed{\overset{1}{\mathtt{Nat}_1}} & & n_1 & s_1 & & & a_1 & z_1 & \\
 & & O & P & & & O & P &
\end{array}
$$

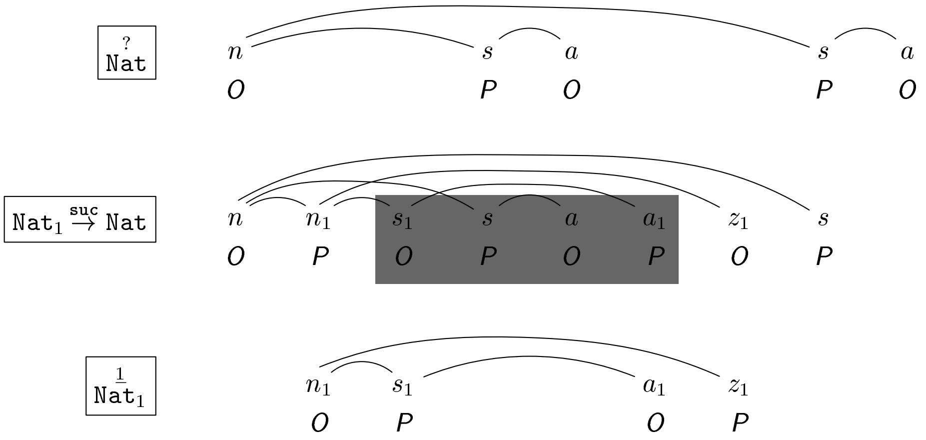
The view is an initial segment of the second position of `suc`'s 'crib-sheet' (page 28), so we obtain the following response:



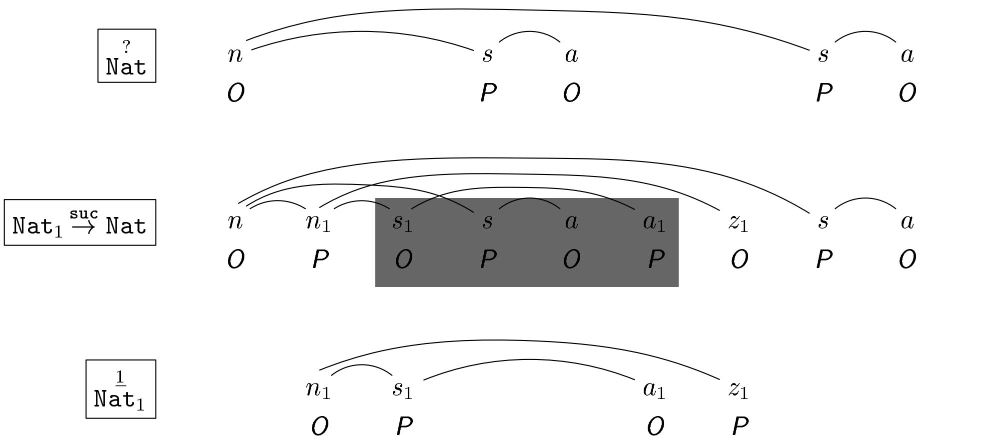
The move s is in the `Nat` subarena, so is transmitted upstairs to form the second 'team-response':



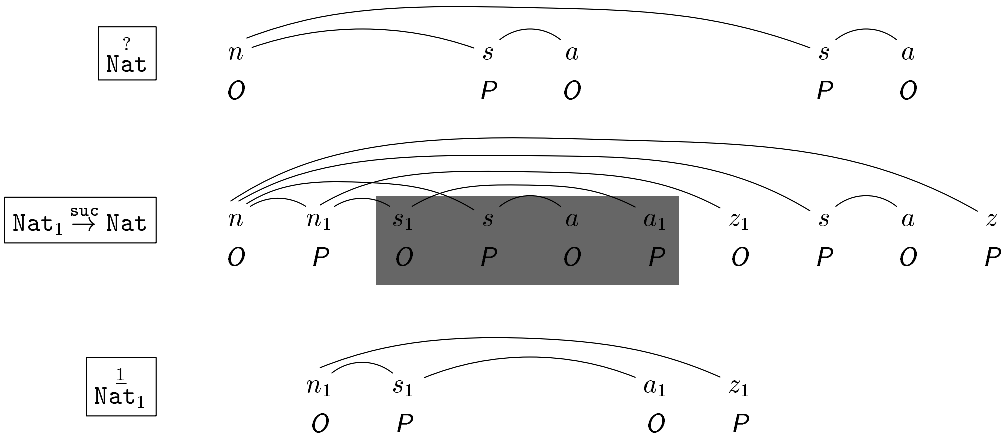
O in `Nat` has no choice but to play the only child a of s:



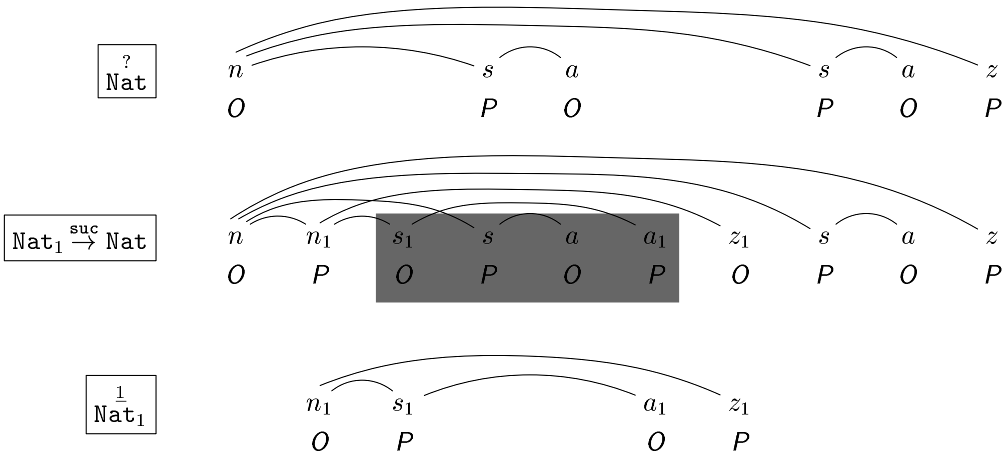
This is transmitted back down to `suc`:



Looking once more at `suc`'s crib-sheet, we get:



This is sent to `Nat` as the final winning move:



As anticipated, the result of the interaction is the strategy $\underline{2}$ on `Nat`.

This section was intended to be informal and accessible. A formal definition of interaction for hypergames is given in Chapter 6, the first-order fragment of which is exactly the interaction introduced intuitively above.

3.3 Hyland/Ong interaction differs from Nickau interaction

It is now folklore in game semantics that the Hyland/Ong and Nickau models are the same. In this section we highlight a hitherto neglected difference between the interaction in the two models. For technical reasons, we took Nickau's interaction as the first-order basis of hypergame interaction.

The notion of strategy introduced in this chapter corresponds to Hyland and Ong's notion of innocent strategy, and to Nickau's notion of sequential strategy (also called a decision tree). Hyland and Ong define a more general notion of strategy in an arena, of which an innocent strategy is a specialisation. They then define composition of general strategies, and prove that innocence is preserved by composition. Nickau, however (see pages 22–27 of [Nic96]) defines the composition of sequential strategies *directly*, without reference to a larger space of more general strategies.

We highlight two consequences of the difference. The first is an advantage of the Nickau approach, and the second is an advantage of the Hyland/Ong approach.

1. In dealing with only the space of strategies of ultimate interest, the proof of compositionality is much easier in the Nickau approach.

2. Hyland and Ong's definition of composition for a larger class of strategies enabled one to think of innocence as an additional 'rule' or 'constraint' in a larger notion of game. This line of thinking leads one to ask what can be modelled by relaxing the rule. This idea (as well as the relaxation of various other constraints) is at the heart of Abramsky's programme of using game semantics to explore the space of programming languages [AM99a].

We elaborate on the distinction between the two models. Our definition of position for $\lambda_1^{\rightarrow}$, asymmetric in the sense that O's pointers are forced to target the previous move, coincides exactly with a path in a Nickau decision tree[2]. For example, take a look once again at the top sequence of the last example of interaction, the `Nat` component: since this is an asymmetric position, we see a path of the decision tree emerging *directly*. O in `Nat`, because of asymmetry, can target only the previous move with a pointer. If this were the corresponding picture of interaction in the Hyland/Ong model, things would be complicated by the fact that O would not be subject to this constraint in `Nat`.

The hypergame model uses Nickau-style composition for two reasons. The pragmatic reason is that the move to second order brings with it sufficient combinatorial complications as it is, without one also having to cope with defining composition for some 'larger' class of strategies. The technical reason is that there seemed to be no immediately obvious notion of a 'larger space' in which both players could be treated symmetrically.

3.3.1 Games, interaction, and abstract machines

We finish the chapter with a remark about the relationship between the interaction of strategies and abstract machines. Danos, Herbelin, and Regnier [DHR96], have indicated that the interaction in the Hyland/Ong model is the hyper-lazy linear head reduction of the Pointer Abstract Machine, a variant of Krivine's Abstract Machine. In the light of the distinction between Hyland/Ong and Nickau interaction elicited above, it would be more accurate to

[2] Once stripped of the *PCF* apparatus, such as questions, answers, and bracketing. All our moves are effectively questions.

say that it is the interaction in the *Nickau* model that corresponds to linear head reduction. The justification is that a decision tree is the same thing as a Böhm tree presentation of a head normal form. The abstract machine produces the output term *directly*, in other words, it produces paths of a decision tree directly, exactly as in the Nickau model. There is no 'larger space of terms' computed by the machine, of which the 'innocent' ones are a specialisation.

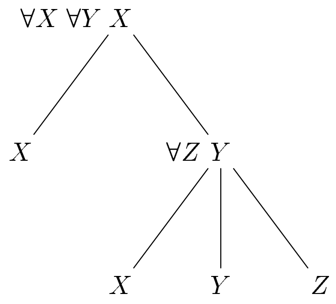
Chapter 4

Polymorphic arenas

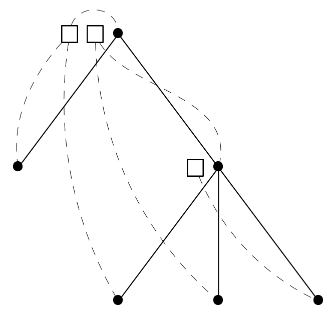
A type of system F will be interpreted as a kind of 'abstract Böhm forest' called a *polymorphic arena*. As an example, given the type

$$T \quad = \quad \forall X.\forall Y.(X \to (\forall Z.X \to Y \to Z \to Y) \to X)$$

we first draw its Böhm tree



(the process for which will be explained below) and then "abstract" to form the polymorphic arena interpreting T:



The six bound occurrences of type variables become the six vertices $\bullet$, the three binding occurrences of type variables become the three vertices $\square$, the five occurrences of the arrow $\to$ in the type become the five straight edges ——, and binding is represented by the arcs - - -. In this case, the forest consists of only a single tree.

4.1 Böhm forests

Our formal game semantics will be in terms of polymorphic arenas, but since Böhm forests (our terminology for "forest of Böhm trees") offer a useful intuitive stepping stone, we present an informal interpretation of types as Böhm forests. We begin with product- and unit-free types of system F, for which a Böhm forest will always be a single tree.

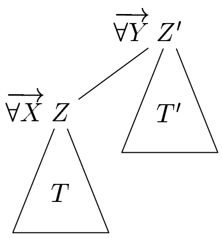
Böhm trees for system F types

- *Type variables.* A type variable X is interpreted as the Böhm tree consisting of a single vertex labelled X.

- *Arrow.* If T and T' have Böhm trees

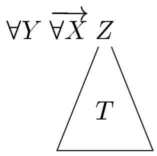
  

  respectively, where $\overrightarrow{\forall X} = \forall X_1\ \ldots\ \forall X_m$ and $\overrightarrow{\forall Y} = \forall Y_1\ \ldots\ \forall Y_n$, then $T \to T'$ has Böhm tree

  formed by connecting the root variables Z of T and Z' of T' with an edge, where we rename the Y_i if necessary in order to avoid capture of free variables below it in the Böhm tree of T.

- *Universal quantification.* With T as above, the type $\forall Y.T$ has Böhm tree

Note that, after renaming X if necessary to avoid capture of free variables in T, the types $T \to \forall X.T'$ and $\forall X.T \to T'$ have the same Böhm tree: the graphical operations "inserting an edge between root variables" and "slotting in a new binding variable next to a root variable" commute. For example, $Y \to \forall X.X$ and $\forall X.Y \to X$ have the same Böhm tree. Interpreting $Y \to \forall X.X$ we "slot in a binding variable" then "insert an edge",

$$X \quad \mapsto \quad \forall X\ X \quad \mapsto \quad \begin{array}{c} \forall X\ X \\ \quad\ \ | \\ \quad\ \ Y \end{array}$$

and interpreting $\forall X.Y \to X$ we "insert and edge" and then "slot in a binding variable"

$$X \quad \mapsto \quad \begin{array}{c} X \\ | \\ Y \end{array} \quad \mapsto \quad \begin{array}{cc} \forall X & X \\ & | \\ & Y \end{array}$$

By induction, since $T \to \forall X.T' \rightsquigarrow \forall X.T \to T'$ is the conversion scheme that takes a type of system F to its prenex form, modulo α-conversion two types of system F have the same prenex form if and only if they have the same Böhm tree. In particular, modulo α-conversion the prenex types of system F are in 1-1 correspondence with Böhm trees.

We now generalise to include product and unit types.

Böhm forests for system *F* types

- *Unit.* The Böhm forest of Unit is the empty forest.
- *Type variables.* A type variable X is interpreted as the single-vertex Böhm forest X.
- *Product.* If the Böhm forest of T consists of n trees

  $$\mathcal{T}_1 \quad \cdots \quad \mathcal{T}_n$$

  and the Böhm forest of T' consists of m trees

  $$\mathcal{T}'_1 \quad \cdots \quad \mathcal{T}'_m$$

  then the Böhm forest of $T \times T'$ is

  $$\mathcal{T}_1 \quad \cdots \quad \mathcal{T}_n \quad \mathcal{T}'_1 \quad \cdots \quad \mathcal{T}'_m$$

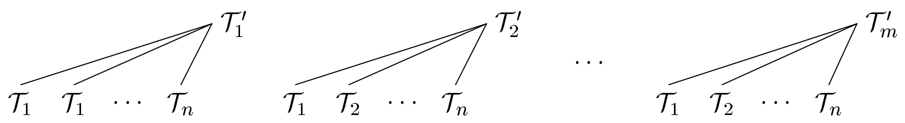
  obtained by laying the forests side by side.
- *Arrow.* If the Böhm forest of T consists of n trees

  $$\mathcal{T}_1 \quad \mathcal{T}_2 \quad \cdots \quad \mathcal{T}_n$$

  and the Böhm forest of T' consists of m trees

  $$\mathcal{T}'_1 \quad \mathcal{T}'_2 \quad \cdots \quad \mathcal{T}'_m$$

  then the forest of $T \to T'$ is

  where $\begin{smallmatrix} & \mathcal{T}' \\ \diagup & \\ \mathcal{T} & \end{smallmatrix}$ denotes the "graft" of $\mathcal{T}$ onto the root of $\mathcal{T}'$, as in the interpretation of the arrow $\to$ of system F above. Thus a copy of the forest of T is made for each tree of T'.

- *Universal quantification.* If T has Böhm forest

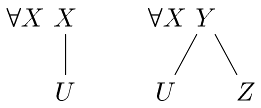
$$\mathcal{T}_1 \quad \cdots \quad \mathcal{T}_n$$

then $\forall Y.T$ has Böhm forest

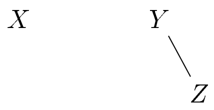
$$\forall Y\,\mathcal{T}_1 \quad \cdots \quad \forall Y\,\mathcal{T}_n$$

obtained by placing $\forall Y$ next to the root of each tree, as in the interpretation of universal quantification $\forall Y$ for system F above.

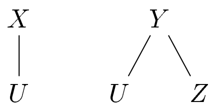
As an example here is the step-by-step interpretation of $\forall X.U \to (X \times (Z \to Y))$ as the Böhm forest



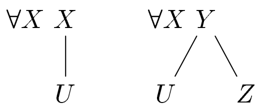
First, $X \times (Z \to Y)$ has the forest



Interpreting the arrow $U \to \cdots$ grafts a copy of U onto each of the two trees:



Finally, interpreting $\forall X. \cdots$ places a quantifier next to the root of each of the trees:



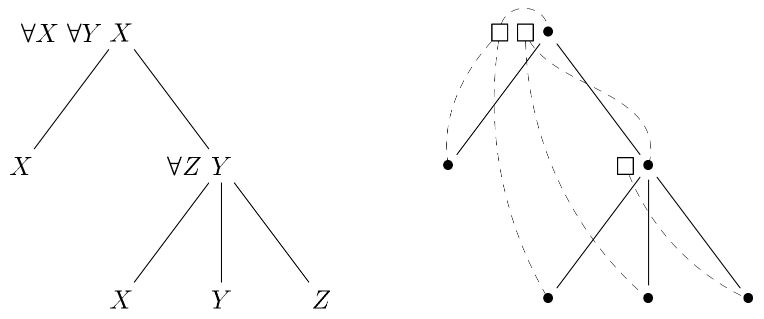
We note various equalities immediate from the graphical nature of our interpretation. Write $T =_{\mathsf{BF}} T'$ for the assertion that modulo α-conversion types T and T' have the same Böhm forest. Then

$$\begin{array}{rcl@{\qquad}rcll}
\mathsf{Unit} \to T & =_{\mathsf{BF}} & T & T_1 \to T_2 \to T & =_{\mathsf{BF}} & T_1 \times T_2 \to T & \\
T \to \mathsf{Unit} & =_{\mathsf{BF}} & \mathsf{Unit} & T \to T_1 \times T_2 & =_{\mathsf{BF}} & (T \to T_1) \times (T \to T_2) & \\
\mathsf{Unit} \times T & =_{\mathsf{BF}} & T & \forall X.T \times T' & =_{\mathsf{BF}} & (\forall X.T) \times (\forall X.T') & \\
T \times \mathsf{Unit} & =_{\mathsf{BF}} & T & T \to \forall X.T' & =_{\mathsf{BF}} & \forall X.T \to T' & (X \text{ not free in } T) \\
\forall X.\mathsf{Unit} & =_{\mathsf{BF}} & \mathsf{Unit} & & & &
\end{array}$$

We have already seen the "scope extrusion" $T \to \forall X.T' =_{\mathsf{BF}} \forall X.T \to T'$, in the case of Böhm trees for product and unit-free types.

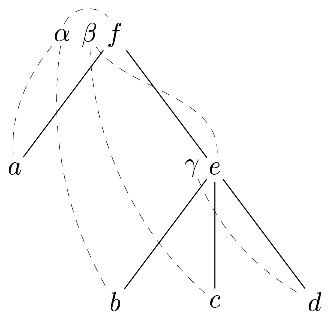
4.2 Polymorphic arenas

Recall our first example of a polymorphic arena obtained via the Böhm tree of the type $\forall X.\forall Y.(Y \to \forall Z.(X \to Y \to Z \to Y) \to X)$:



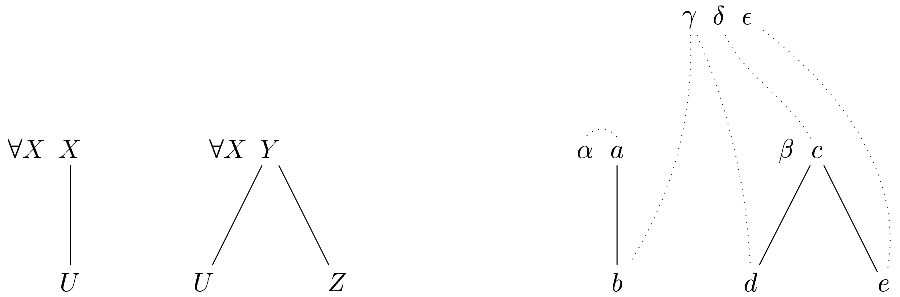
The vertices □ are called ***holes***, and the vertices • are called ***actions***, so-called because of their future roles in a hypergame. We choose the name "action" for a vertex in order to reserve "move" for later.

A hypergame will involve the players tracing paths down polymorphic arenas. For example, taking the previous picture of a polymorphic arena, and naming the vertices in order to distinguish them from one another,



one player starts by playing the opening action f, which enables the other player to play either of the actions a or e. The choice of e then enables the original player to play b, c, or d. The role of the holes α, β, γ and the reference arcs will be explained in the next chapter.

Before embarking on the formal definition of a polymorphic arena, we show the polymorphic arena abstracting the Böhm forest of $\forall X.U \to (X \times (Z \to Y))$, the example from page 48:



This demonstrates the way in which we will handle free variables as "global holes" floating above the forest. Here, γ, δ and ϵ correspond to U, Y and Z respectively.

Let $G = (V, \to)$ be a graph consisting of a set of vertices V and an edge relation $\to \subseteq V \times V$. A (finite) ***path*** in G is a finite sequence $v_1 \ldots v_n$ of vertices such that $v_i \to v_{i+1}$ for $i = 1, \ldots, n-1$. A ***root*** in G is a vertex with no incoming edge, i.e. a vertex $v \in V$ such that the set $\{ w \in V : w \to v \}$ is empty. A ***root-path*** is a path beginning with a root. A graph $G = (V, \to)$ is a ***forest*** if for every vertex $v \in V$ there is exactly one root-path in G ending with v.

DEFINITION 4.1 *A **polymorphic arena** $(\mathsf{Act}_A, \mathsf{Hol}_A, \vdash_A, \mathsf{act}_A, \mathsf{ref}_A)$ consists of:*

1. *A finite set of **actions** Act_A.*
2. *A finite set of **holes** Hol_A.*
3. *An **enabling relation** $\vdash_A \subseteq \mathsf{Act}_A \times \mathsf{Act}_A$ on actions, such that the graph $(\mathsf{Act}_A, \vdash_A)$ is a forest. If $a \vdash_A b$, i.e. b is a child of a in the forest, we say that a **enables** b. If a is a root we write $\vdash_A a$, and say that a is an **opening** action; otherwise $b \vdash_A a$ for some b, and a is called a **continuation** action. The set of opening actions of A is written Act_A^{op}, and the set of continuation actions of A is Act_A^{cont}.*
4. *A partial function $\mathsf{act}_A : \mathsf{Hol}_A \rightharpoonup \mathsf{Act}_A$ attaching holes to actions. If $\mathsf{act}_A(\alpha) \in \mathsf{Act}_A$ we say that α is a **local hole**; if $\mathsf{act}_A(\alpha)\!\uparrow$ we say that α is a **global hole**.*
5. *A function $\mathsf{ref}_A : \mathsf{Act}_A \to \mathsf{Hol}_A$ assigning to every action a **reference** hole further up in the forest, i.e. such that if $\mathsf{ref}_A(a) = \alpha$ and α is local, then $\mathsf{act}_A(\alpha) \vdash_A^* a$, where $\vdash_A^*$ is the transitive, reflexive closure of $\vdash_A$. (We picture global holes "above the whole forest".)*

The set of global holes of A is written $\mathsf{Hol}_A^{\uparrow}$, the set of local holes is written $\mathsf{Hol}_A^{\downarrow}$, and the set of holes $\mathsf{act}_A^{-1}(a)$ attached to an action a (i.e. the holes next to a in a picture) is written $\mathsf{Hol}_A(a)$. Thus $\mathsf{Hol}_A = \mathsf{Hol}_A^{\uparrow} \cup \mathsf{Hol}_A^{\downarrow}$, and $\mathsf{Hol}_A^{\downarrow} = \bigcup_{a \in \mathsf{Hol}_A} \mathsf{Hol}_A(a)$. The ***level*** $\mathsf{level}(a)$ of an action $a \in \mathsf{Act}_A$ is how far down it is in the forest. Formally, $\mathsf{level}(a) = 1$ if $\vdash_A a$, and $\mathsf{level}(a) = \mathsf{level}(b) + 1$ if $b \vdash_A a$. When the polymorphic arena A can be deduced easily from the context we will sometimes omit subscripts, writing $\vdash$ for $\vdash_A$ or Hol for Hol_A for example.

If H is a set, a ***polymorphic arena over*** H is a polymorphic arena A with global holes $\mathsf{Hol}_A^{\uparrow} = H$. We write PA for the class of polymorphic arenas, and PA_H for the class of polymorphic arenas over H.

All of the operations and constructions we shall define on polymorphic arenas in the next section work equally well for infinite polymorphic arenas, as do hypergames and winning strategies in the next chapter. But since we need finiteness in order to obtain full completeness, we might as well take finiteness as a basic part of the definition. Intuitively the reason finiteness is necessary for full completeness is that system F terms can contain types, and types are finite. So if we want to make sure there are not too many strategies in the model, we have to discard infinite polymorphic arenas.

4.3 Operations on polymorphic arenas

We define product $A \times B$, function space $A \Rightarrow B$, and quantification $\forall\alpha(A)$ of a global hole α, which will be used to interpret syntactic $\times$, $\to$, and $\forall$. They can be thought of as the "abstract" versions of the interpretations of $\times$, $\to$ and $\forall$ on Böhm trees that we saw earlier. Below A and B are polymorphic arenas over a common set of global holes H.

Product The product $A \times B$ is obtained by laying the graphs A and B side by side.

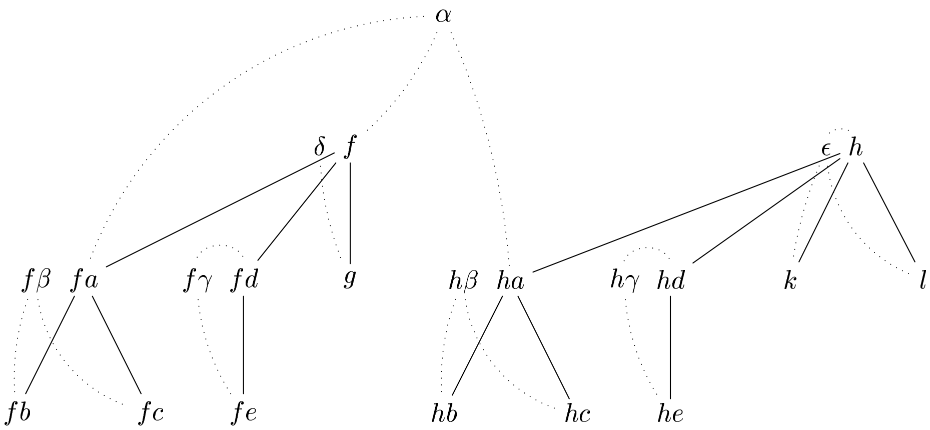
$$\begin{array}{rcl}
\mathit{Act}_{A\times B} & = & \mathit{Act}_A + \mathit{Act}_B \\
\mathit{Hol}^{\uparrow}_{A\times B} & = & H \quad (= \mathit{Hol}^{\uparrow}_A = \mathit{Hol}^{\uparrow}_B) \\
\mathit{Hol}^{\downarrow}_{A\times B} & = & \mathit{Hol}^{\downarrow}_A + \mathit{Hol}^{\downarrow}_B \\
\vdash_{A\times B} & = & \vdash_A + \vdash_B \\
\mathit{act}_{A\times B} & = & \mathit{act}_A + \mathit{act}_B \\
\mathit{ref}_{A\times B} & = & \mathit{ref}_A + \mathit{ref}_B
\end{array}$$

Function space The polymorphic arena $A \Rightarrow B$ consists of B together with a copy bA of A for each opening action b of B, with in addition b enabling every opening action of bA.

$$\begin{array}{rcl}
\mathit{Act}_{A\Rightarrow B} & = & \mathit{Act}^{op}_B \times \mathit{Act}_A \;+\; \mathit{Act}_B \\
\mathit{Hol}^{\uparrow}_{A\Rightarrow B} & = & H \quad (= \mathit{Hol}^{\uparrow}_A = \mathit{Hol}^{\uparrow}_B) \\
\mathit{Hol}^{\downarrow}_{A\Rightarrow B} & = & \mathit{Act}^{op}_B \times \mathit{Hol}^{\downarrow}_A \;+\; \mathit{Hol}^{\downarrow}_B \\
\vdash_{A\Rightarrow B} & = & \mathit{Act}^{op}_B \mathbin{\dot{\times}} \vdash_A \;+\; \vdash_B \;+\; \vdash_{\mathsf{graft}} \\
\mathit{act}_{A\Rightarrow B} & = & \mathit{Act}^{op}_B \mathbin{\dot{\times}} \mathit{act}_A \;+\; \mathit{act}_B \\
\mathit{ref}_{A\Rightarrow B} & = & \mathit{Act}^{op}_B \mathbin{\dot{\times}} \mathit{ref}_A \;+\; \mathit{ref}_B
\end{array}$$

where for a set X and a binary relation $R \subseteq Y \times Z$, the relation $X \mathbin{\dot{\times}} R \subseteq (X \times Y) \times (X \times Z)$ is given by $(x, y) X \mathbin{\dot{\times}} R (x, z)$ whenever yRz, and the relation $\vdash_{\mathsf{graft}}$ is given by $b \vdash_{\mathsf{graft}} (b, a)$ whenever b is an opening action of B, and a is an opening action of A. Here is an example:

is

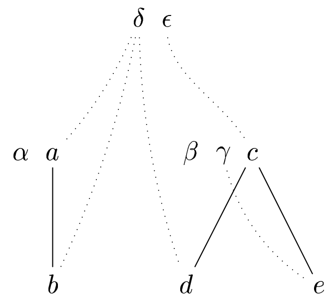


where we have written pairs as two-element sequences. On the underlying trees of actions, ignoring holes and references, this reduces to the function space construction of Hyland and Ong.

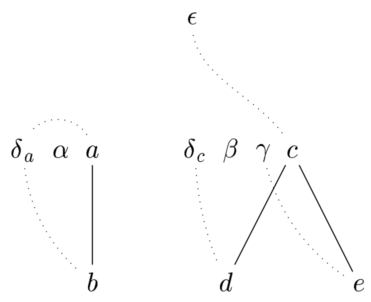
Quantification Given a global hole $\alpha \in \mathit{Hol}^{\uparrow}_{A}$, the quantification $\forall\alpha(A)$ of α in A is obtained by replacing α with a local copy α_a attached to each opening action a of A. an action b below a that used to reference α is set to reference α_a. The forest $(\mathit{Act}_{\forall\alpha(A)}, \vdash_{\forall\alpha(A)})$ of $\forall\alpha(A)$ is that of A, and, with $\tilde{b}$ denoting the opening action above an action b,

$$\begin{aligned}
\mathit{Hol}^{\uparrow}_{\forall\alpha(A)} &= \mathit{Hol}^{\uparrow}_{A} \setminus \{\alpha\} \\
\mathit{Hol}^{\downarrow}_{\forall\alpha(A)} &= \mathit{Hol}^{\downarrow}_{A} + \{\, \alpha_a \,:\, a \in \mathit{Act}^{op}_{A} \,\} \\
\mathit{Act}_{\forall\alpha(A)} &= \mathit{Act}_{A} \;+\; \{\, \langle \alpha_a, a \rangle \,:\, a \in \mathit{Act}^{op}_{A} \,\} \\
\mathit{ref}_{\forall\alpha(A)}(b) &= \begin{cases} \alpha_{\tilde{b}} & \text{if } \mathit{ref}_A(b) = \alpha \\ \mathit{ref}_A(b) & \text{if } \mathit{ref}_A(b) \neq \alpha \end{cases}
\end{aligned}$$

Here is an example. Quantifying δ in the polymorphic arena



gives the polymorphic arena



4.4 Expansion

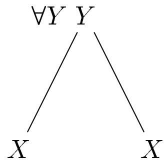
We define a form of substitution of polymorphic arenas for global holes of a polymorphic arena, called *expansion*, modelled on the substitution of types for the free variables of a type. Expansion will be crucial for the interaction of strategies in Chapter 6. Since it will not be used until then, the reader can safely skip the rest of this Chapter for the time being.

Let A be a polymorphic arena over H. An ***assignment*** of polymorphic arenas to the global holes of A is a partial function $\phi : H \rightharpoonup \mathbb{PA}$. On page 59 we define the ***expansion*** $\phi^*(A)$ of A along ϕ. Rather than launch cold into the definition, we warm up with a few motivating examples.

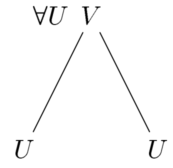
Example 1 Consider the types

$$\begin{aligned} T &= X \to X \to Y \\ T' &= \forall U.U \to V \to U \\ T[T'/X] &= (\forall U.U \to V \to U) \to (\forall U.U \to V \to U) \to Y \end{aligned}$$

where $T[T'/X]$ denotes the result of substituting T' for X in T. Looking at the Böhm trees of the three types, we observe the substitution of trees for vertices:

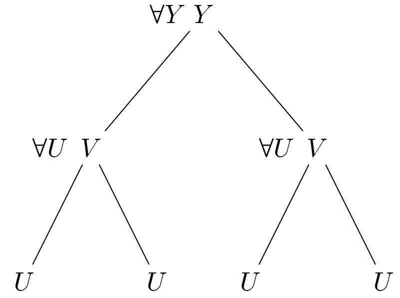


with

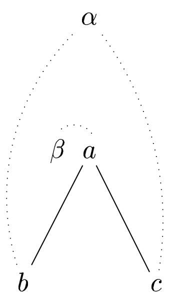


for X

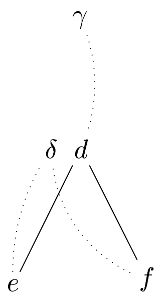
is



Mimicking this process with polymorphic arenas, we have our first example of *expansion*:

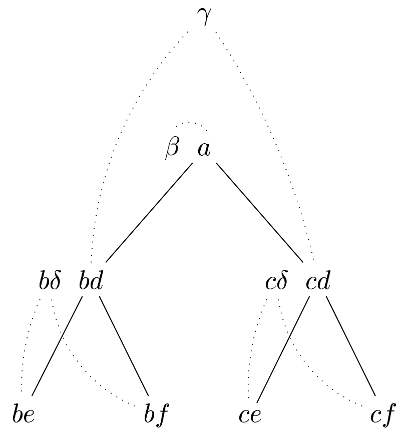


with



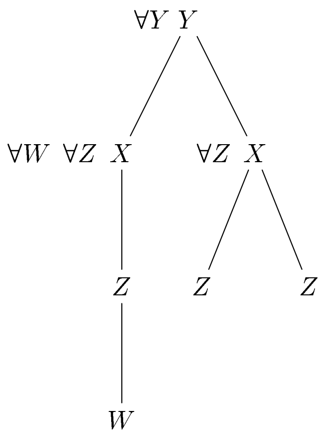
in α

is

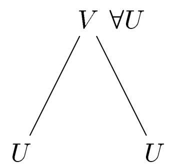


In order to obtain two distinct copies of the incoming polymorphic arena, we have 'tagged' its vertices by prefixing them with the action onto which they are 'hooked'. Here for example, there are two copies $b\delta$ and $c\delta$ of the local hole δ, and two copies bd and cd of the action d.

Example 2 We add subtrees below the two occurrences of X in the previous example. To aid pattern matching, we switch $\forall U$ to the right of V.

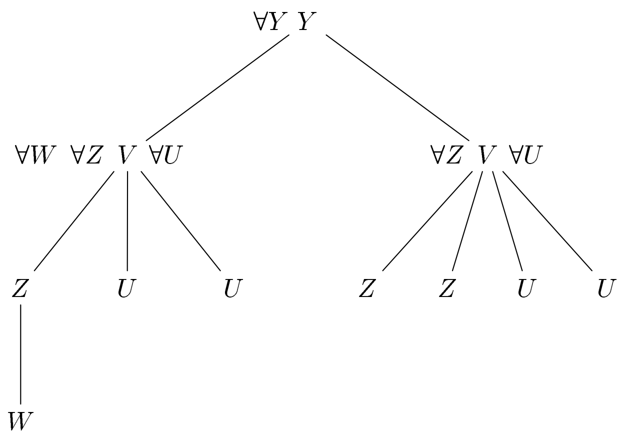


with

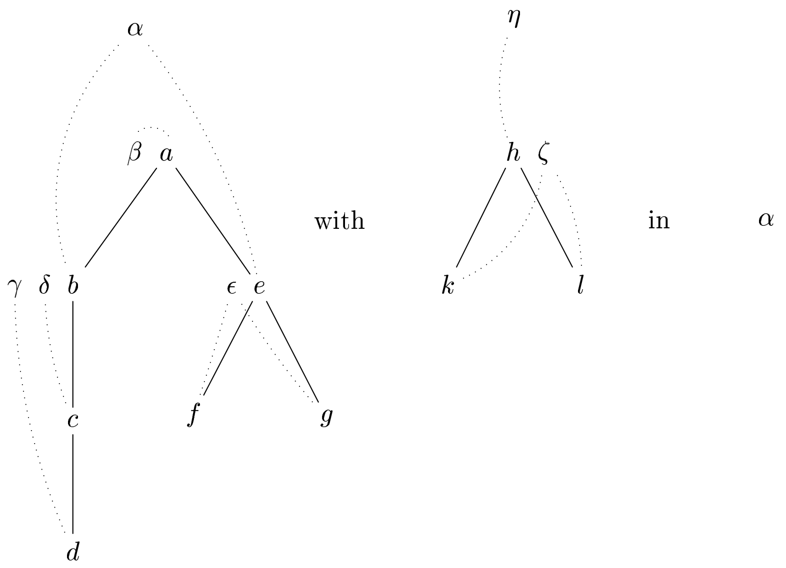


for X

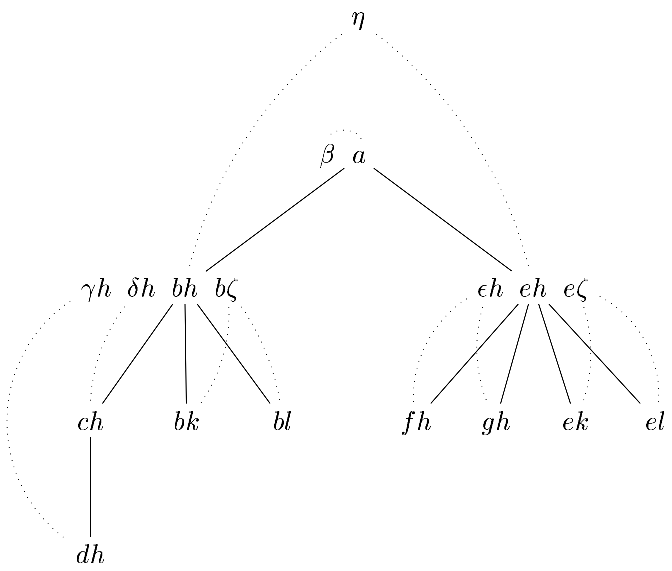
is



The expansion parallelling this substitution with polymorphic arenas is:



is

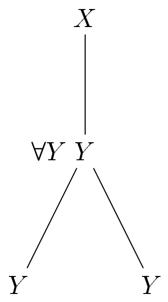


The result of Example 1 is a subtree. The vertices γ, δ, c, d, ϵ, f, and g have been postfixed with an h 'tag' for reasons that will become apparent later.

Example 3 In the previous examples we saw duplication due to multiple occurrences of X. The following example shows another form of duplication, caused when the incoming forest has more than one component.

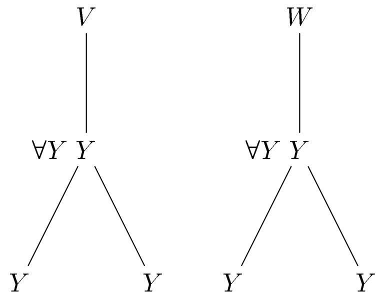
$$\begin{aligned} T &= (\forall Y.Y \to Y \to Y) \to X \\ T' &= V \times W \\ T[T'/X] &= (\forall Y.Y \to Y \to Y) \to V \times W \end{aligned}$$

Interpreting each of these three types as Böhm forests gives:

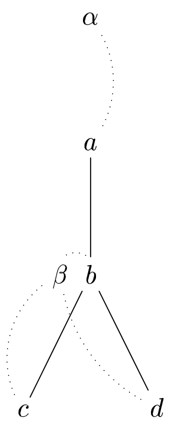


with V W for X

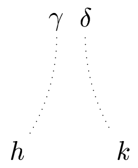
is



Note the duplication that occurs when we interpret $T[T'/X] = (\forall Y.Y \to Y \to Y) \to V \times W$ as a Böhm forest. This is because of the way arrow $\to$ is interpreted on Böhm forests. For polymorphic arenas we will track such duplications by postfixing:

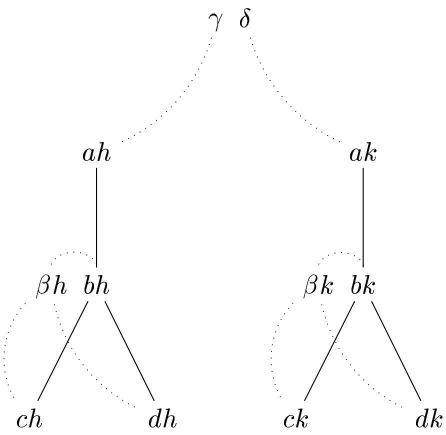


with



in α

is

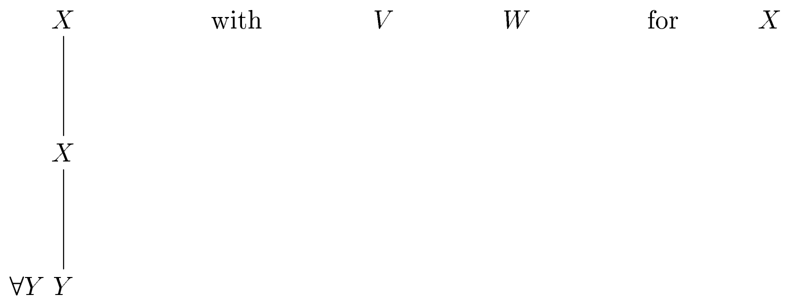


Now we see the origin of the h "postfix tags" in the previous example, which is the degenerate case when the incoming polymorphic arena has just one component.

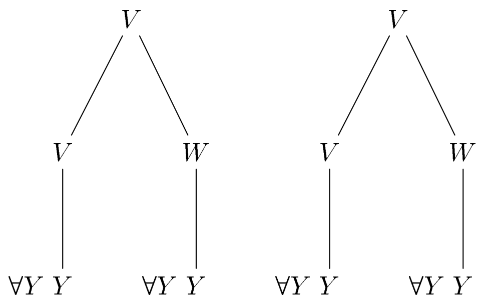
Example 4 Duplication such as in the last example can occur repeatedly. For example

$$\begin{aligned} T &= ((\forall Y.Y) \to X) \to X \\ T' &= V \times W \\ T[T'/X] &= ((\forall Y.Y) \to V \times W) \to V \times W \end{aligned}$$

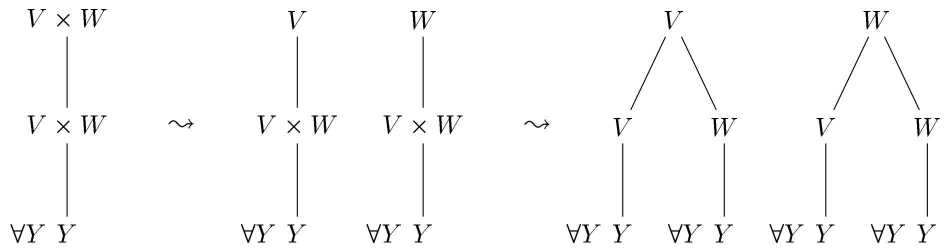
Interpreting the three types yields:



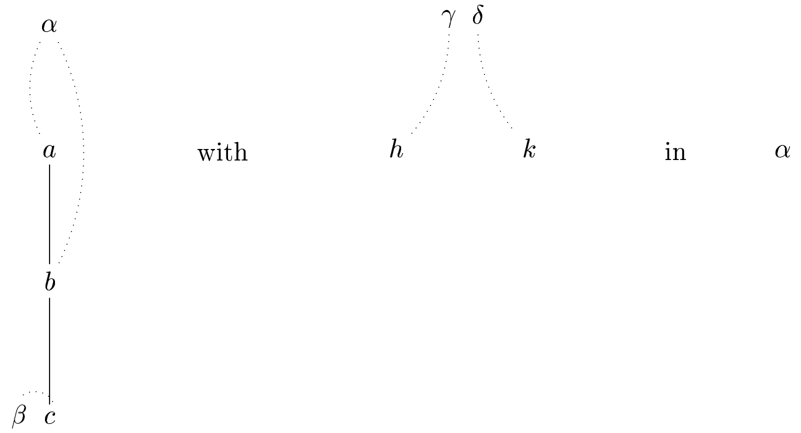
is



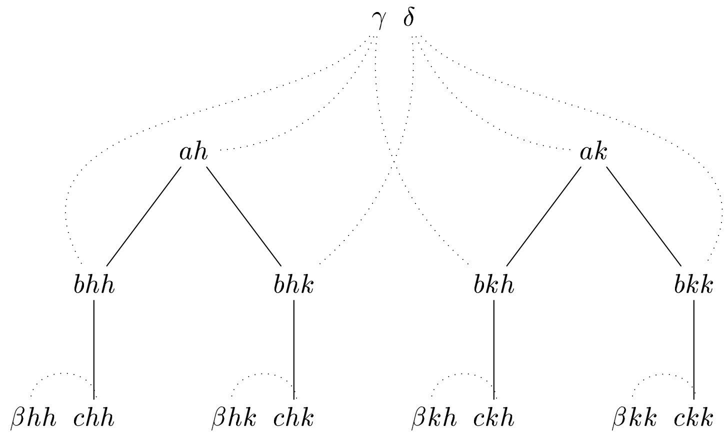
We can visualise the duplication something like this:



Only the last is actually a Böhm forest; the first two are informal graphs that live somewhere between types and Böhm forests. Here is how we use postfixing to track the duplication:



is



There are two copies ah, ak of a, four copies bhh, bhk, bkh, bkk of b, four copies chh, chk, ckh, ckk of c, and four copies $\beta hh, \beta hk, \beta kh, \beta kk$ of the local hole β.

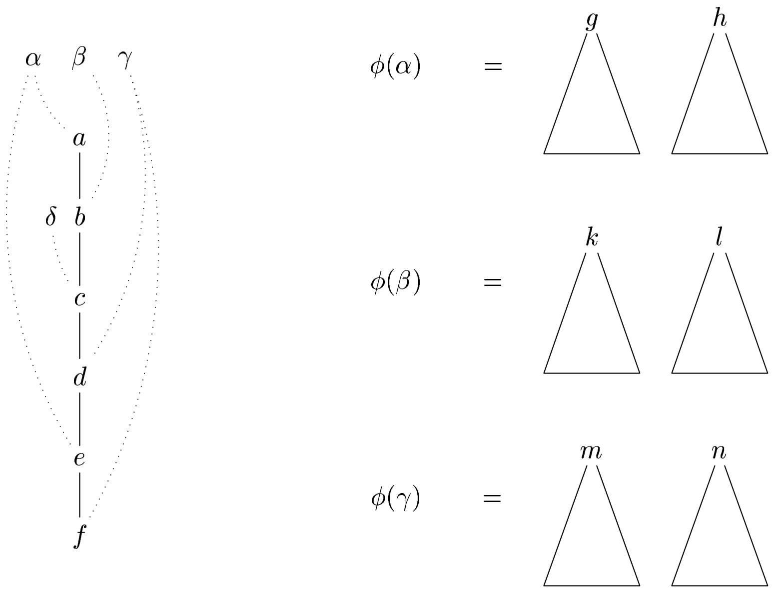
Having considered these examples, there are two key points giving rise to the formal definition of expansion:

1. Duplication of an incoming forest is tracked by *pre*fixing.

2. Duplication of sub-forests of the receiving polymorphic arena is tracked by *post*fixing.

4.4.1 Formalisation

Let A be a polymorphic arena over H. Recall from the beginning of section 4.4 that an assignment of polymorphic arenas to the global holes of A is a partial function[1] $\phi : H \rightharpoonup \mathbb{PA}$. We say that an action a of A references the polymorphic arena B via ϕ if a references a global hole, say $\alpha \in H$, and $\phi(\alpha) = B$. Let a be an action of A, and let b be the root above a. A ***trail of opening actions*** above a via ϕ is a choice of opening action from each of the polymorphic arenas referenced via ϕ as we travel down the path from b to the enabling action (parent) of a. For example, if A is the polymorphic arena below, and ϕ is as shown to the right,



then typical trails of opening actions above f are $glnh$, $hlmg$, and $gkmg$. Actions in A referencing local holes, such as c above, do not contribute. Likewise actions referencing global holes $\alpha \in H$ that are not in the domain of ϕ do not contribute. Trails will be the 'postfix tags' used to keep track of duplication, for example the four trails hh, hk, kh, kk of c in Example 4 above. We write $\mathsf{trails}_\phi(a)$ for the set of trails above a via ϕ.

DEFINITION 4.2 *Given a polymorphic arena A over H and an assignment $\phi : H \rightharpoonup \mathbb{PA}$, the* ***expansion*** *$\phi^*(A)$ of A by ϕ is the polymorphic arena defined as follows.*

- **Actions.** *The actions of $\phi^*(A)$ are given by*

$$\begin{aligned} \mathsf{Act}_{\phi^*(A)} \;=\; & \{\, a\eta \;:\; \eta \in \mathsf{trails}_\phi(a) \text{ and } \mathsf{ref}_A(a) \notin \mathsf{dom}(\phi) \,\} \quad \cup \\ & \{\, a\eta b \;:\; \eta \in \mathsf{trails}_\phi(a),\ \mathsf{ref}_A(a) \in \mathsf{dom}(\phi) \text{ and } b \in \mathsf{Act}_{\phi(\mathsf{ref}_A(a))} \,\} \end{aligned}$$

  The actions of $\phi^*(A)$ are called ***compound*** *actions, and the original actions of A and of the $\phi(\alpha)$ for $\alpha \in \mathsf{dom}(\phi)$ (i.e. a, b, and the elements of η) are called* ***atomic*** *actions.*

- **Enabling.** *In the definition below η and η' range over trails above a and a' respectively. The four cases arise from the two different kinds of action in $\phi^*(A)$, either of the form*

[1] It does not matter that the codomain $\mathbb{PA}_H$ is a proper class, since the domain is a set.

$a\eta$ or of the form $a\eta b$.

$$
\begin{array}{llcl}
a\eta & \vdash_{\phi^*(A)} & a'\eta' & \Leftrightarrow & a \vdash_A a' \text{ and } \eta = \eta' \\
a\eta & \vdash_{\phi^*(A)} & a'\eta'b' & \Leftrightarrow & a \vdash_A a' \text{ and } \eta = \eta' \text{ and } \vdash b' \\
a\eta b & \vdash_{\phi^*(A)} & a'\eta' & \Leftrightarrow & a \vdash_A a' \text{ and } \eta b = \eta' \text{ and } \vdash b \\
a\eta b & \vdash_{\phi^*(A)} & a'\eta'b' & \Leftrightarrow & \begin{cases} a \vdash_A a' \text{ and } \eta b = \eta' \text{ and } \vdash b \text{ and } \vdash b', & \text{or} \\ b \vdash b' \text{ and } a\eta = a'\eta' \end{cases}
\end{array}
$$

Thus a path down the forest of $\phi^*(A)$ from a root will be of the form

$$
\begin{array}{l}
a_1[c_1] \\
a_2[c_1][c_2] \\
a_3[c_1][c_2][c_3] \\
\quad\vdots \\
a_m[c_1]\ldots[c_m] \\
a_{m+1}[c_1]\ldots[c_m]b_1 \\
a_{m+1}[c_1]\ldots[c_m]b_2 \\
\quad\vdots \\
a_{m+1}[c_1]\ldots[c_m]b_n
\end{array}
$$

where $a_1a_2\ldots a_m a_{m+1}$ is a path down A starting from a root a_1 and $b_1b_2\ldots b_n$ is a path down $\phi(\mathsf{ref}_A(a_{m+1}))$ starting from a root b_1. Here $[c_i]$ is an opening action c_i of $\phi(\mathsf{ref}_A(a_i))$ if $\mathsf{ref}_A(a_i) \in \mathsf{dom}(\phi)$ and is empty otherwise. Thus $[c_1]\ldots[c_{j-1}] \in \mathsf{trails}_\phi(a_j)$ for $j = 1, \ldots, m+1$.

Essentially a path in $\phi^*(A)$ is a path in A followed by a path in one of the polymorphic arenas in the image of ϕ, modulo 'duplication tags' $[c_i]$. Observe this in the paths of Examples 1 to 4, most notably those of Example 2.

- **Holes.** The holes of $\phi^*(A)$ are given by

$$
\begin{array}{rcl}
\mathsf{Hol}^{\uparrow}_{\phi^*(A)} & = & H \setminus \mathsf{dom}(\phi) \quad \cup \displaystyle\bigcup_{\alpha \in \mathsf{dom}(\phi)} \mathsf{Hol}^{\uparrow}_{\phi(\alpha)} \\
\mathsf{Hol}_{\phi^*(A)}(a\eta) & = & \{\, \alpha\eta \;:\; \alpha \in \mathsf{Hol}_A(a) \,\} \\
\mathsf{Hol}_{\phi^*(A)}(a\eta b) & = & \begin{cases} \{\, \alpha\eta b \;:\; \alpha \in \mathsf{Hol}_A(a) \,\} \cup \{\, a\eta\beta \;:\; \beta \in \mathsf{Hol}(b) \,\} & \text{if } \vdash b \\ \{\, a\eta\beta \;:\; \beta \in \mathsf{Hol}(b) \,\} & \text{if } \nvdash b \end{cases}
\end{array}
$$

  Again, see Examples 1 to 4. Recall that given an action c of an arena C, the set $\mathsf{Hol}_C(c)$ is $\mathsf{act}_C^{-1}c$, the holes of C attached to c. So the above not only defines the local holes of $\phi^*(A)$ by $\mathsf{Hol}^{\downarrow}_{\phi^*(A)} = \bigcup_{c \in \mathsf{Act}_{\phi^*(A)}} \mathsf{Hol}_{\phi^*(A)}(c)$, but defines $\mathsf{act}_{\phi^*(A)}$ also.

- **References.** Note that by the consideration of the shape of paths in $\phi^*(A)$ above, given an action $a\zeta$ of $\phi^*(A)$, for every action a' above a in A there is a unique action above $a\zeta$ in $\phi^*(A)$ of the form $a'\zeta'$, and furthermore ζ' will be a prefix of ζ. Write $[\zeta, a', a]$ for the unique ζ' such that $a'\zeta' \vdash^*_{\phi^*(A)} a\zeta$.

  Recall that an action of $\phi^*(A)$ is either $a\eta$ or $a\eta b$, where η is a trail above a. Define

$$\mathit{ref}_{\phi^*(A)}(a\eta) \quad = \quad \begin{cases} \mathit{ref}(a)\,\eta' & \text{if } \mathit{ref}(a) \text{ is local} \\ \mathit{ref}(a) & \text{if } \mathit{ref}(a) \text{ is global} \end{cases}$$

$$\mathit{ref}_{\phi^*(A)}(a\eta b) \quad = \quad \begin{cases} a\,\eta\,\mathit{ref}(b) & \text{if } \mathit{ref}(b) \text{ is local} \\ \mathit{ref}(b) & \text{if } \mathit{ref}(b) \text{ is global} \end{cases}$$

where $\eta' = [\eta, \mathit{act}_A(\mathit{ref}(a)), a]$.

4.4.2 Live atomic enabling

The following definition will be important for the interaction of strategies. Given an action d of $\phi^*(A)$ define its atomic actions to be ***live*** or ***dead*** according to whether they 'play a role' in the enabling relation of $\phi^*(A)$, as follows. Let b^0 range over opening actions, and let b^1 range over non-opening actions. Then there are three cases, where $\eta \in \mathit{trails}_\phi(a)$:

1. $d = a\eta$: a is live, and the elements of η are dead.
2. $d = a\eta b^0$: a and b^0 are live, and the elements of η are dead.
3. $d = a\eta b^1$: a is dead, b^1 is live, and the elements of η are dead.

So every enabling $d \vdash_{\phi^*(A)} d'$ involves exactly one live atomic action $c \in d$ and one live atomic action $c' \in d'$, with $c \vdash c'$ in the appropriate location. We call this $c \vdash c'$ the ***live atomic enabling***. Observe this with the analysis of paths of $\phi^*(A)$ back on page 60, where the live atomic enablings involved were

$$a_1 \vdash a_2, \quad a_2 \vdash a_3, \quad \ldots, \quad a_m \vdash a_{m+1}, \quad b_1 \vdash b_2, \quad b_2 \vdash b_3, \quad \ldots, \quad b_{n-1} \vdash b_n.$$

Also witness it concretely with Examples 1 to 4 earlier. Note that the elements of η are always dead, so when $d \vdash d'$ with live atomic enabling $c \vdash c'$, either c and c' are the first elements of d and d' respectively, or they are the last elements.

4.4.3 Repeated expansion

Suppose

$$A_0 \overset{\phi_1}{\mapsto} A_1 \overset{\phi_2}{\mapsto} \ldots A_{k-1} \overset{\phi_k}{\mapsto} A_k$$

is a sequence of expansions, i.e., $A_i = \phi_i^*(A_{i-1})$ for assignments ϕ_i, $1 \le i \le k$. Then by definition the actions d of $A_k = \phi_k^*(\ldots(\phi_1^*(A_0))\ldots)$ are of the form:

$$d = b_0\eta_1[b_1]\eta_2[b_2]\cdots\eta_k[b_k]$$

where

$$\begin{aligned} & b_0 \in \mathit{Act}_{A_0} \\ & \eta_i \in \mathit{trails}_{\phi_i}(b_0\eta_1[b_1]\ldots\eta_{i-1}[b_{i-1}]) \\ & [b_i] = \begin{cases} \epsilon & \text{if } \mathit{ref}_{A_{i-1}}(b_0\eta_1[b_1]\ldots\eta_{i-1}[b_{i-1}]) \notin \mathit{dom}(\phi_i) \\ b_i \in \mathit{Act}_{B_i} & \text{otherwise, where } B_i := \phi_i(\mathit{ref}_{A_{i-1}}(b_0\eta_1[b_1]\ldots\eta_{i-1}[b_{i-1}])) \end{cases} \end{aligned}$$

(Note that in the expression for d we have erased a number of brackets, e.g., writing $b_0\eta_1[b_1]\eta_2[b_2]$ for $(b_0\eta_1[b_1])\eta_2[b_2]$.)

Analogous to the definition for the case of a single expansion, define the actions of d (in its unbracketed form, as above) to be ***live*** or ***dead*** as follows.

First, all the elements of the η_i are dead. This leaves b_0 and the non-empty $[b_i]$. Of the b_i (i.e. b_0 or non-empty $[b_i]$) define the last non-opening b_i and subsequent opening b_i to be live, and define all other b_i to be dead. In the special case that all b_i are opening actions, take all the b_i to be live.

Formally, let $l \geq 0$ be least with b_i (i.e. b_0 or non-empty $[b_i]$) an opening action for all $i \geq l$. For $j \geq 0$ define b_j as dead if $j < l-1$ and live if $j \geq l-1$, and define each of the elements of the η_i as dead.

Thus live atomic enabling generalises to the k-ary case: every enabling $d \vdash_{A_k} d'$ involves exactly one live atomic action $c \in d$ and one live atomic action $c' \in d'$ with $c \vdash c'$ (in the appropriate location).

Note that with $k = 1$ this coincides with the definition of section 4.4.2.

4.5 Equivalence

We say that polymorphic arenas A and B over H are ***equivalent***, denoted $A \approx B$, if one can be obtained from the other by renaming of actions and local holes. Formally, $A \approx B$ if and only if there exist isomorphisms $\theta : \mathsf{Act}_A \cong \mathsf{Act}_B$ and $\phi : \mathsf{Hol}_A \cong \mathsf{Hol}_B$ preserving and reflecting the three relations $\vdash$, act, and ref, and with $\phi \restriction H$ the identity on H.

Define $\mathsf{PA}^{\approx}$ to be the set of $\approx$-equivalence classes of PA, and define $\mathsf{PA}_H^{\approx}$ to be the set of $\approx$-equivalence classes of PA_H. From now on we shall work almost exclusively with $\mathsf{PA}^{\approx}$ and $\mathsf{PA}_H^{\approx}$, i.e. with polymorphic arenas modulo $\approx$-equivalence. Note that the operations of product, function space, and universal quantification of polymorphic arenas respect equivalence, so they are well-defined on $\mathsf{PA}_H^{\approx}$. Furthermore, expansion respects equivalence: if $A \in \mathsf{PA}_H^{\approx}$ and $\phi : H \rightharpoonup \mathsf{PA}^{\approx}$ then $\phi^*(A) \in \mathsf{PA}^{\approx}$ is well-defined.

Define the set $\mathbb{PA} \cong \mathsf{PA}^{\approx}$ to be a particular choice of representative for each $\approx$-class of $\mathsf{PA}^{\approx}$, and define $\mathbb{PA}_H \subseteq \mathbb{PA}$ to be the polymorphic arenas of $\mathbb{PA}$ with set of global holes H. Via the isomorphism $\mathbb{PA} \cong \mathsf{PA}^{\approx}$, product, function space, universal quantification and expansion are defined on $\mathbb{PA}$ and on $\mathbb{PA}_H$.

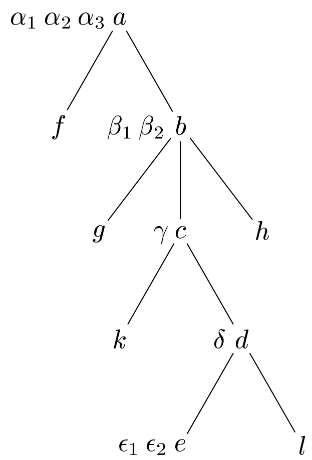
Chapter 5

Hypergames

A polymorphic arena is a board on which to play a game, just as an 8-by-8 grid is a board on which to play chess. A hypergame is an 'interlacing' of threads of games running on a variety of different boards. Initially there is just one board, but as a hypergame progresses the two players O and P import fresh boards, on which new threads may open in the future. When we come to interpret terms of system F as strategies for hypergames, importing a new board will interpret the application sT of a polymorphic subterm s to a type T. More specifically, P imports the polymorphic arena interpreting T into the 'playing area'.

5.1 Informal overview

Threads A *thread* in a polymorphic arena is sequence of moves tracing a path of actions down its forest, storing fresh polymorphic arenas into the holes encountered along the path. Here is an example of a thread:



$$\overset{\alpha_1}{A_1}\ \overset{\alpha_2}{A_2}\ \overset{\alpha_3}{A_3}\ a \qquad \overset{\beta_1}{B_1}\ \overset{\beta_2}{B_2}\ b \qquad \overset{\gamma}{C}\ c \qquad \overset{\delta}{D}\ d \qquad \overset{\epsilon_1}{E_1}\ \overset{\epsilon_2}{E_2}\ e$$

(The references of the arena are omitted.) The first move consists in the action a together with the storage of polymorphic arenas A_1, A_2, and A_3 into the holes α_1, α_2 and α_3 attached to a. The second move consists in the action b, enabled by a, together with the storage of polymorphic arenas B_1 and B_2 into the holes β_1 and β_2 attached to b. The third move consists in the action c, together with the storage of the polymorphic arena C in the hole γ attached to c. And so on.

Hiding of O-imports A crucial feature of hypergames is an asymmetry between the two players O and P: imports by O will remain 'hidden' from P. So strictly speaking, depending on whether O or P opened, the thread above ought to have been one of

$$\begin{smallmatrix}\alpha_1 & \alpha_2 & \alpha_3\\ * & * & *\end{smallmatrix}\ a \qquad \begin{smallmatrix}\beta_1 & \beta_2\\ B_1 & B_2\end{smallmatrix}\ b \qquad \begin{smallmatrix}\gamma\\ *\end{smallmatrix}\ c \qquad \begin{smallmatrix}\delta\\ D\end{smallmatrix}\ d \qquad \begin{smallmatrix}\epsilon_1 & \epsilon_2\\ * & *\end{smallmatrix}\ e$$

$$\begin{smallmatrix}\alpha_1 & \alpha_2 & \alpha_3\\ A_1 & A_2 & A_3\end{smallmatrix}\ a \qquad \begin{smallmatrix}\beta_1 & \beta_2\\ * & *\end{smallmatrix}\ b \qquad \begin{smallmatrix}\gamma\\ C\end{smallmatrix}\ c \qquad \begin{smallmatrix}\delta\\ *\end{smallmatrix}\ d \qquad \begin{smallmatrix}\epsilon_1 & \epsilon_2\\ E_1 & E_2\end{smallmatrix}\ e$$

where we think of $*$ as "some unknown polymorphic arena imported by O."

The hiding of O-imported arenas from P corresponds to uniformity in system F. Intuitively, in a term such as $\Lambda X.\lambda x^X.x$ of type $\forall X.X \to X$, which is uniform in the sense that it 'works the same way for all types X', the type variable X denotes a hole containing an arena $*$ forever hidden from P. In a hypergame 'works the same way for all types X' becomes 'plays the same way whatever the mystery arena $*$ turns out to be'. And 'plays the same way' will be the *copycat* strategy ubiquitous in game semantics, in which P simply copies moves to and fro between threads. Copycat will happen between two threads on the hidden arena with opposite polarity: one opened by P and one opened at some earlier stage in the hypergame by O. (We shall not witness the uniformity/copycat idea until the next chapter, when we define the interaction of strategies.)

Moves A *move* is an action located in a particular arena, together with the storage of polymorphic arenas in its holes. For example, $\begin{smallmatrix}\alpha_1 & \alpha_2 & \alpha_3\\ A_1 & A_2 & A_3\end{smallmatrix}\ a$ of the thread depicted above. So a thread is a sequence of moves, each in the same location.

Hypermoves A hypermove consists of one or more moves in different locations, the first of which continues or opens a thread in a polymorphic arena, and the remainder of which open threads in other polymorphic arenas. Figure 5.1 is an example of a run of a hypergame in which eight threads open up on imported arenas. Each hypermove is displayed as a column of moves, and the arcs keep track of which thread is continued by which move.

The polymorphic arena A is assumed as given before the start of the hypergame. The hypermoves numbered 1 to 10 are as follows.

1. The first hypermove, by O, consists in a single move: he opens a thread in A with its opening action a_1, and stores a hidden polymorphic arena in the hole α attached to a_1.

2. The hypermove response of P also consists in a single move: the action a_2 enabled by a_1 in A, together with the storage of arenas B and D into the holes α_2 and α_2' of a_2.

3. An O-hypermove consisting in two moves:

   (a) The first move continues the thread in A, with the action a_3.

   (b) The second move opens a new thread in B, one of the arenas just stored by P, with its opening action b_1. The opening of new threads will always be governed by the references of polymorphic arenas. In this case, since the action a_3 referenced the hole α_2, which stores B, O is obliged to open a thread on B.

4. A P-hypermove consisting in two moves:

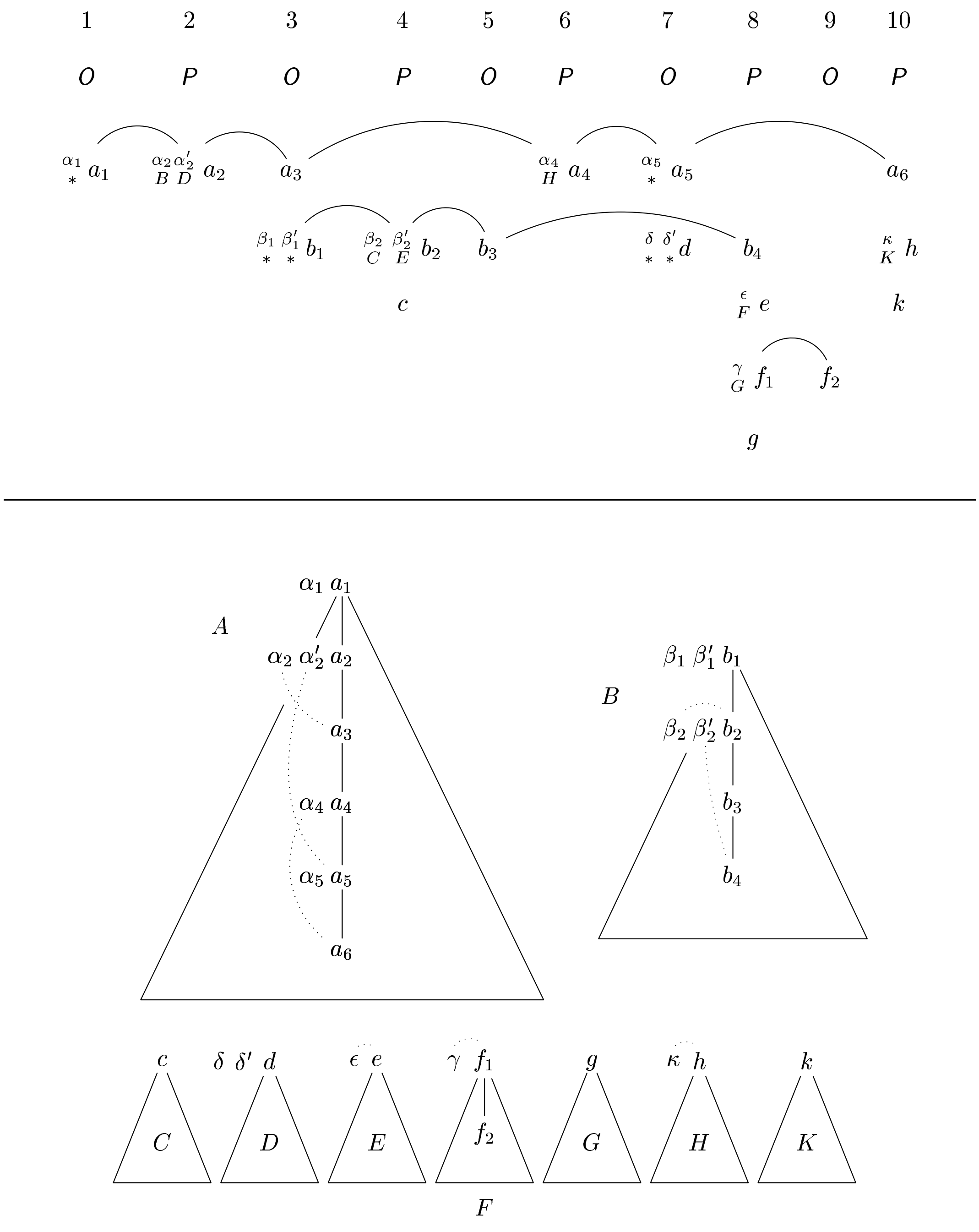


Figure 5.1: An example of a hypersequence. The polymorphic arenas involved are shown schematically underneath.

   (a) Two arenas A and B are open, and P is free to continue in either. He decides to pursue the thread in B by playing b_2, and storing arenas C and E in its holes β_2 and β_2' respectively. Now, because b_2 references β_2, which stores C, P must open a thread on C.
   (b) P opens a thread on C with its opening action c.

5. O continues the thread in B with b_3.
6. P switches back to the old thread in A, playing a_4 and storing H in its hole α_4.
7. (a) O continues the thread in A by playing a_5, and stores a 'hidden' polymorphic arena $*$ in the hole α_5 next to a_5. Since a_5 references the hole α_2' in A, O is obliged to open a new thread in its contents D, which was stored in α_2' by P on hypermove 2.
   (b) O opens a new thread on D with the action d, storing hidden arenas in its holes δ and δ'.
8. (a) P returns to the old thread in B with b_4, ignoring the thread just opened by O in D. Since b_4 references β_2' in B, whose contents is E, P must open a thread on E.
   (b) P plays opening action e in E, storing F in its hole ϵ. Since e references ϵ in E, P must open a thread on its contents F.
   (c) P plays the opening action f_1 in F, storing G in its hole γ. Since f_1 references γ in E, P must open a thread on its contents G.
   (d) P plays the opening action g in G.
9. O plays f_2 continuing the thread on F.
10. (a) P returns to the original thread in A, playing a_6. Since a_6 references α_4 containing H, stored by P on hypermove 6, P must open a thread on H.
   (b) P plays the opening action h in H, storing K in the hole κ next to h. Because h references κ, P must open a thread on K.
   (c) P plays the opening action k on K.

The next two sections introduce formal definitions of move, thread, and hypersequence.

5.2 Moves and threads

DEFINITION 5.1 *A* ***move*** $m = (A, \phi, a)$ *consists of:*

- *A polymorphic arena* $A \in \mathbb{PA}$, *called the* ***location*** *of* m.
- *An action* $a \in \mathsf{Act}_A$.
- *A function* $\phi : \mathsf{Hol}_A(a) \to \mathbb{PA} + \{*\}$ *assigning polymorphic arenas to the holes of* a, *called the* ***store*** *of* m.

Intuitively, we think of the symbol $*$ as denoting a 'hidden' polymorphic arena. Write Mov_A for the set of moves located in A, and write $\mathsf{Loc}(n) \in \mathbb{PA}$, $\mathsf{act}(n) \in \mathsf{Act}_{\mathsf{Loc}(n)}$ and $\mathsf{store}_n : \mathsf{Act}_{\mathsf{Loc}(n)} \to \mathbb{PA} + \{*\}$ respectively for the location, action and store of a move n. Given a move $m = (A, \phi, a)$, write $\mathsf{Hol}(m)$ for $\mathsf{Hol}_A(a)$, and define m to be an ***opening move*** or ***continuation move*** according as a is an opening action or continuation action. In other words, $m = (A, \phi, a)$ is an opening move if $\vdash_A a$, and is a continuation move otherwise.

When displaying a store ϕ, we often stack the ordered pairs of the graph of ϕ, for example writing $\begin{smallmatrix}\beta & \gamma & \delta \\ B & C & D\end{smallmatrix}$ for $\phi = \{ \langle\beta, B\rangle, \langle\gamma, C\rangle, \langle\delta, D\rangle \}$. When picturing a move within a thread (as with the examples earlier), we tend to leave the location implicit. For example, we may write $\begin{smallmatrix}\beta & \gamma & \delta \\ B & C & D\end{smallmatrix}\, a$ for $(A, \{ \langle\beta, B\rangle, \langle\gamma, C\rangle, \langle\delta, D\rangle \}, a)$, when the location of the move can be inferred easily from the context.

DEFINITION 5.2 *A* ***thread*** θ *is a non-empty sequence*[1] *of moves* $\theta = m_1 \ldots m_k$, $m_i = (A, \phi_i, a_i)$ *such that* $\vdash_A a_1$ *and* $a_i \vdash_A a_{i+1}$, $1 \leq i < k$. *The* ***location*** *of* θ *is the common location* A *of its moves* m_i. *The thread is a* ***P-thread*** *(resp.* ***O-thread****) if for all* $1 \leq i \leq k$ *and holes* $\alpha \in \mathsf{Hol}_A(a_i)$, $\phi_i(\alpha) = *$ *iff* i *is even (resp. odd). In other words, a* Q*-thread is a thread opened by* Q.

Here once again are the threads we saw in the informal introductory section:

$$\begin{smallmatrix}\alpha_1 & \alpha_2 & \alpha_3 \\ * & * & *\end{smallmatrix}\, a \qquad \begin{smallmatrix}\beta_1 & \beta_2 \\ B_1 & B_2\end{smallmatrix}\, b \qquad \begin{smallmatrix}\gamma \\ *\end{smallmatrix}\, c \qquad \begin{smallmatrix}\delta \\ D\end{smallmatrix}\, d \qquad \begin{smallmatrix}\epsilon_1 & \epsilon_2 \\ * & *\end{smallmatrix}\, e$$

$$\begin{smallmatrix}\alpha_1 & \alpha_2 & \alpha_3 \\ A_1 & A_2 & A_3\end{smallmatrix}\, a \qquad \begin{smallmatrix}\beta_1 & \beta_2 \\ * & *\end{smallmatrix}\, b \qquad \begin{smallmatrix}\gamma \\ C\end{smallmatrix}\, c \qquad \begin{smallmatrix}\delta \\ *\end{smallmatrix}\, d \qquad \begin{smallmatrix}\epsilon_1 & \epsilon_2 \\ E_1 & E_2\end{smallmatrix}\, e$$

The first is an O-thread, and the second is a P-thread. As will be common, the locations of the moves are left implicit.

Given a sequence of moves $s = m_1 \ldots m_k$, $m_i = (A_i, \phi_i, a_i)$, define

$$\mathsf{Hol}(s) = \sum_{1 \leq i < k} \mathsf{Hol}_{A_i}(a_i)$$

and define

$$\mathsf{store}_s : \mathsf{Hol}(s) \to \mathbb{PA} + \{ * \}$$

by

$$\mathsf{store}_s = [\phi_1, \phi_2, \ldots, \phi_k]$$

where the source tupling $f = [f_1, \ldots, f_k] : \sum_{1 \leq i \leq k} X_i \to Y$ of functions $f_i : X_i \to Y$ is defined on $a \in X_i \hookrightarrow X$ by $f(a) = f_i(a)$. For example, if s is either of the two threads depicted above, then $\mathsf{Hol}(s) = \{ \alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \gamma, \delta, \epsilon_1, \epsilon_2 \}$, and if s is the second of the two threads above, then store_s is

$$\{ \langle\alpha_1, A_1\rangle, \langle\alpha_2, A_2\rangle, \langle\alpha_3, A_3\rangle, \langle\beta_1, *\rangle, \langle\beta_2, *\rangle, \langle\gamma, C\rangle, \langle\delta, *\rangle, \langle\epsilon_1, E_1\rangle, \langle\epsilon_2, E_2\rangle \}$$

Note that $\mathsf{Hol}(s)$ and store_s are defined for arbitrary sequences of moves s, not just threads.

5.3 Hypermoves and hypersequences

DEFINITION 5.3 *A* ***hypermove*** μ *is a non-empty sequence of moves* $\mu = m_1 \ldots m_k$, $m_i = (A_i, \phi_i, a_i)$ *such that* $\vdash_{A_i} a_i$ *for* $2 \leq i \leq k$.

Typically we draw a hypermove in column format, as in Figure 5.1. There, as usual, the locations of the moves are not shown explicitly.

[1] All sequences in this thesis are finite, unless stated otherwise.

5.3.1 Sequences of hypermoves

Throughout the rest of this section fix $s = \mu_1 \dots \mu_l$ to be a sequence of hypermoves, $\mu_i = m_{i1} m_{i2} \dots m_{ik_i}$, and fix t to be the sequence of hypermoves of Figure 5.1 (ignoring the additional arcs for now).

Write $\mathsf{mov}(s)$ for the underlying sequence of moves of s. So

$$\mathsf{mov}(s) = m_{11} m_{12} \dots m_{1k_1} m_{21} \dots\dots m_{l-1,k_{l-1}} m_{l1} \dots m_{lk_l}$$

and

$$\mathsf{mov}(t) = \overset{\alpha_1}{\underset{*}{}} a_1 \cdot \overset{\alpha_2\ \alpha_2'}{\underset{B\ \ D}{}} a_2 \cdot a_3 \cdot \overset{\beta_1\ \beta_1'}{\underset{*\ \ *}{}} b_1 \cdot \overset{\beta_2\ \beta_2'}{\underset{C\ \ E}{}} b_2 \cdot \dots \cdot \overset{\kappa}{\underset{K}{}} h \cdot k$$

Write $\mathsf{mov}^{op}(s)$ for the subsequence of $\mathsf{mov}(s)$ consisting of the opening moves, and $\mathsf{mov}^{cont}(s)$ for the complementary subsequence of $\mathsf{mov}(s)$ consisting of the continuation moves. Since every move is either an opening move or a continuation move, $\mathsf{mov}(s)$ is an interleaving of $\mathsf{mov}^{op}(s)$ and $\mathsf{mov}^{cont}(s)$.

Given moves[2] $m, n \in \mathsf{mov}(s)$, write $m < n$ if m strictly precedes n. So if $m = m_{ij}$ and $n = m_{pq}$ then $n < m$ if and only if $i < p$ or $[i = p$ and $j < q]$. Given a hypermove[3] $\mu \in s$, define $\mathsf{pl}(\mu) = \mathsf{O}$ or P according as μ occurs in odd or even position. In other words, $\mathsf{pl}(\mu_i) = \mathsf{O}$ if i is odd, and $\mathsf{pl}(\mu_i) = \mathsf{P}$ if i is even. For each move $m \in \mu$, define $\mathsf{pl}(m) = \mathsf{pl}(\mu)$. Thus $\mathsf{pl}(m_{ij}) = \mathsf{O}$ or P according as i is odd or even.

Write $\mathsf{Hol}(s)$ as shorthand for $\mathsf{Hol}(\mathsf{mov}(s))$, and store_s as shorthand for $\mathsf{store}_{\mathsf{mov}(s)}$. Define $\mathsf{act}(s)$ to be the underlying sequence of actions of s. So with $m_{ij} = (A_{ij}, \phi_{ij}, a_{ij})$,

$$\mathsf{act}(s) = a_{11} a_{12} \dots a_{1k_1} a_{21} \dots\dots a_{l-1,k_{l-1}} a_{l1} \dots a_{lk_l}$$

and for the hypersequence t of Figure 5.1,

$$\mathsf{act}(t) = a_1 a_2 a_3 b_1 b_2 c b_3 a_4 a_5 d b_4 e f_1 g f_2 a_6 h k.$$

By taking actions and holes to inherit properties from their containing move, we obtain useful shorthand notation. For example, suppose $m = (A, \phi, a)$ and $n = (B, \psi, b)$ are moves in s (i.e., $m, n \in \mathsf{mov}(s)$), $\alpha \in \mathsf{Hol}(m) \hookrightarrow \mathsf{Hol}(s)$, and $\beta \in \mathsf{Hol}(n) \hookrightarrow \mathsf{Hol}(s)$. Then $\mathsf{pl}(\alpha)$ and $\mathsf{pl}(a)$ are each equivalent to $\mathsf{pl}(m)$, and $a < b$, $\alpha < \beta$, $a < n$, $m < \beta$, $\alpha < b$ are each equivalent to $m < n$.

Define

$$\begin{aligned} \mathsf{OHol}(s) &= \{\, \alpha \in \mathsf{Hol}(s) \;:\; \mathsf{pl}(\alpha) = \mathsf{O} \,\} \\ \mathsf{PHol}(s) &= \{\, \alpha \in \mathsf{Hol}(s) \;:\; \mathsf{pl}(\alpha) = \mathsf{P} \,\} \end{aligned}$$

For example, for t of Figure 5.1,

$$\begin{aligned} \mathsf{OHol}(t) &= \{\, \alpha_1, \beta_1, \beta_1', \alpha_5, \delta, \delta' \,\} \\ \mathsf{PHol}(t) &= \{\, \alpha_2, \alpha_2', \beta_2, \beta_2', \alpha_4, \epsilon, \gamma, \kappa \,\} \end{aligned}$$

5.3.2 Hypersequences

DEFINITION 5.4 *Given a polymorphic arena A over a set of global holes H, a* ***hypersequence*** $(s, \curvearrowleft)$ *of A consists of:*

[2] Strictly speaking, ‘move-occurrences’.

[3] Strictly speaking, ‘hypermove-occurrence’.

- *A sequence of hypermoves s.*
- *A function $\curvearrowleft : \mathsf{mov}^{cont}(s) \to \mathsf{mov}(s)$ assigning a **justifier** to every continuation move of s. (Recall that $m = (B, \phi, b)$ is an opening move if $\vdash_B b$, and is a continuation move otherwise.) We write $m \curvearrowleft n$ for $\curvearrowleft(n) = m$, saying that m **justifies** n. If $m \curvearrowleft n$, where $m = (B, \phi, b)$ and $n = (C, \psi, c)$, then we require that:*
  1. *m strictly precedes n: $m < n$.*
  2. *m and n are played by opposite players: $\mathsf{pl}(m) \neq \mathsf{pl}(n)$. In other words, if m and n are in the i^{th} and j^{th} hypermove of s respectively, then $i - j$ is odd.*
  3. *m and n are in the same location: $B = C$.*
  4. *m 'enables' n: $b \vdash_B c$.*

We require the following conditions to be satisfied:

1. ***Origin**. The first move of s is located in A (unless s is empty).*
2. ***Hiding**. The contents of O-holes in s are 'hidden': for all holes $\alpha \in \mathsf{Hol}(s)$, $\mathsf{store}_s(\alpha) = *$ if and only if $\mathsf{pl}(\alpha) = \mathsf{O}$.*
3. ***Locations**. Every move m is located either in A or in a P-stored arena strictly preceding m. Formally, for all moves $m = (B, \phi, b)$ in s,*

$$B \quad \in \quad \{A\} \cup \{\, \mathsf{store}_s(\alpha) \ : \ \alpha \in \mathsf{Hol}(s), \alpha < m, \text{ and } \mathsf{pl}(\alpha) = \mathsf{P} \,\}$$

4. ***Scope**. Every global hole of a polymorphic arena B stored by P is either an element of H or is an occurrence of an O-hole preceding B. Formally, for all $m \in \mathsf{mov}(s)$ and $\alpha \in \mathsf{Hol}(m)$ with $\mathsf{pl}(m) = \mathsf{P}$,*

$$\mathsf{Hol}^{\uparrow}_{\mathsf{store}_s(\alpha)} \quad = \quad H \ + \ \{\, \beta \in \mathsf{OHol}(s) \ : \ \beta < m \,\}.$$

Conditions 1, 2 and 3 can be seen to hold in Figure 5.1. Intuitively, the scope condition can be thought of as follows. Given a polymorphic arena A interepreting a type T, a hypersequence of A is a 'journey' down into the syntax tree of a term of type T, in the sense of the 'top-down terms' of Chapter 1. Every bound variable ΛX encountered on the way corresponds to an O-hole α, and type arguments correspond to arenas imported by P. Just as a type argument lies in the 'scope' of a bound variable ΛX above it in the syntax tree, and can contain X, an imported arena lies in the 'scope' of a hole α before it in the hypersequence, and can reference α.

Extending the convention that actions inherit properties from their containing move, if $m, n \in \mathsf{mov}(s)$, $m = (B, \phi, b)$ and $n = (B, \psi, c)$, then $b \curvearrowleft c$, $b \curvearrowleft n$, and $m \curvearrowleft c$ are all equivalent to $m \curvearrowleft n$.

Depth

Given a hypersequence $h = \mu_1 \ldots \mu_l$, $\mu_i = m_{i1} \ldots m_{ik_i}$, the ***depth*** $\mathsf{depth}(m_{ij})$ of a move m_{ij} is defined recursively as follows:

- If m_{ij} is a justified move, say $m_{gh} \curvearrowleft m_{ij}$, then $\mathsf{depth}(m_{ij}) = \mathsf{depth}(m_{gh})$.
- If m_{ij} is an opening move, then

– if $j = 1$ then $\mathit{depth}(m_{ij}) = 1$.
– otherwise $j > 1$, and set $\mathit{depth}(m_{ij}) = \mathit{depth}(m_{i,j-1}) + 1$.

When we display hypersequences, such as in Figure 5.1, moves of the same depth are all in the same row, and the top row is depth 1.

Prefix

Given hypersequences $h = (s, \curvearrowleft)$ and $h' = (s', \curvearrowleft')$ of a polymorphic arena A, we say that h is a prefix of h', denoted $h \sqsubseteq h'$, if s is a prefix of s' and the restriction of $\curvearrowleft'$ to s is exactly $\curvearrowleft$. The ***prefix-closure*** h^{pref} of a hypersequence h is the set

$$\{\, h' \ : \ h' \sqsubseteq h \,\}$$

of prefixes of h. The prefix-closure S^{pref} of a set S of hypersequences is $S^{\mathsf{pref}} = \{\, h' \ : \ h' \sqsubseteq h \in S \,\}$.

We write $h \sqsubset h'$ if h is a strict prefix of h', i.e., if $h \sqsubseteq h'$ and $h \neq h'$.

Threads in hypersequences

Write $\curvearrowleft^*$ for the reflexive, transitive closure of $\curvearrowleft$, and if $m \curvearrowleft^* n$ say that m ***hereditarily justifies*** n. For example, in Figure 5.1 the move $\overset{\alpha_2}{B}\,\overset{\alpha_2'}{D}\,a_2$ hereditarily justifies the move $\overset{\alpha_5}{*}\,a_4$. Given a move $m = (B, \phi, b) \in \mathit{mov}(s)$, define $\mathit{thread}(m)$ to be the subsequence of s consisting of the moves hereditarily justifying m, and write $\mathring{m}$ for the the first move of $\mathit{thread}(m)$. Note that because of conditions 1–4 in the definition of hypersequence, $\mathit{thread}(m)$ will be a thread located in B, as per the original definition of thread (Definition 5.2, page 67), and $\mathring{m}$ will be an opening move, i.e., if $\mathring{m} = (B, \psi, c)$, then $\vdash_B c$. Furthermore, $\mathit{thread}(m)$ is a P-thread iff $\mathring{m}$ is a P-move, and an O-thread iff $\mathring{m}$ is an O-move.

Relationship with Hyland/Ong and Nickau

In the special case that every hypermove consists of a single move, so that all moves are at depth 1, and after disregarding locations, holes, and storage of arenas, a hypersequence reduces to what Hyland and Ong call a *justified sequence*. Nickau independently introduced the same concept as a *play* (not to be confused with a different notion of play introduced later in this thesis, in Chapter 6).

5.3.3 Well-formed hypersequences

In the commentary of Figure 5.1 we saw that the opening of new threads was governed by references. In this subsection we formally define such a hypersequence to be *well-formed*.

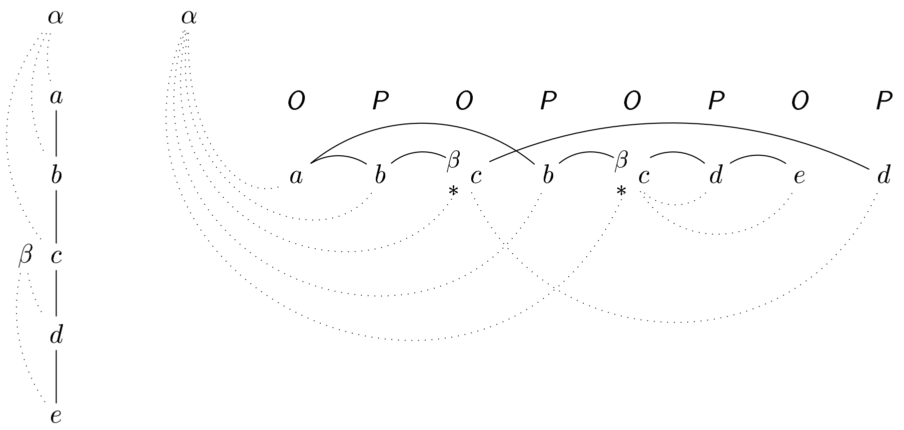
References in hypersequences

Given a hypersequence $h = (s, \curvearrowleft)$, define $\mathit{ref}_h : \mathit{act}(s) \to H + \mathit{Hol}(s)$ to mimic the references of actions in the location polymorphic arenas of s. For each move $m = (B, \phi, a) \in \mathit{mov}(s)$ define $\mathit{ref}_h(a)$ as follows:

1. If $\mathit{ref}_B(a)$ is global (i.e., if $\mathit{ref}_B(a) \in H$), simply set $\mathit{ref}_h(a) = \mathit{ref}_B(a)$.

2. Otherwise $\mathit{ref}_B(a)$ is local, say $\mathit{ref}_B(a) = \beta \in \mathit{Hol}_B(b)$, for some action $b \in \mathit{Act}_B$ with $b \vdash_B^* a$. Define $\mathit{ref}_h(a)$ to be the unique occurrence of β in $\mathit{thread}(a)$ (necessarily a hole of the unique occurrence of b in $\mathit{thread}(a)$).

For example, given the following hypersequence $h = (s, \curvearrowleft)$ of the polymorphic arena A shown to the left,



(in which each hypermove happens to consist of a single move) the references in the hypersequences are indicated with dotted lines (always oriented from right to left). Notice how the distinct occurrences of d reference distinct occurrences of β; the referenced β is the one in the appropriate thread.

The image of ref_h is well-defined in $H + \mathit{Hol}(s)$ because of conditions 3 (Locations) and 4 (Scope) of Definition 5.4 of hypersequence.

Well-formed hypersequences

Definition 5.5 *Let A be a polymorphic arena over a set of global holes H. A hypersequence $h = (s, \curvearrowleft)$ of A is **well-formed** if the opening of new threads and the change of turn between O and P is governed by $\mathit{ref}_h : \mathit{act}(s) \to H + \mathit{Hol}(s)$ as follows. Let μ be a hypermove of s, let $m = (B, \phi, a)$ be a move of μ, and let $\alpha = \mathit{ref}_h(a)$.*

1. ***Thread-opening**. If $\alpha \in \mathit{PHol}(s)$, then whoever played m must open a new thread on the arena stored in α. Formally, $\mu = m_1 \dots m_k m n n_1 \dots n_l$ for some move $n = (\phi(\alpha), \psi, b)$ and $\vdash_{\phi(\alpha)} b$.*

2. ***Change of turn**. Otherwise $\alpha \notin \mathit{PHol}(s)$ (i.e., if $\alpha \in H + \mathit{OHol}(s)$), and the other player goes next. Formally, m is the last move of μ, i.e. $\mu = m_1 \dots m_k m$, $k \geq 0$.*

We saw the thread-opening condition at work in Figure 5.1, but not the change of turn condition, because the arenas were too schematic. The two conditions together mean that from the point of view of a player, a hypermove works like this: "keep opening threads in the arenas of referenced holes until you reference a hole whose contents is 'inaccessible', upon which your turn is over." Here 'inaccessible' means either a global hole of H or an O-hole containing a hidden arena $*$.

5.3.4 Legal hypersequences

Recall the copycat rule motivated intuitively in section 1.1.2 by 'type-checking' considerations.

DEFINITION 5.6 *A well-formed hypersequence h is **legal** if it satisfies the following condition:*

- ***Copycat rule.*** *The reference of the last move of every P-hypermove is the same as the reference of the last move of the previous O-hypermove. In other words, if $h = h_1 \mu \nu h_2$, $\mu = \mu' m$, $\nu = \nu' n$, and $\mathit{pl}(\nu) = \mathit{pl}(n) = P$, then $\mathit{ref}_h(n) = \mathit{ref}_h(m)$.*

The hypersequence of Figure 5.1 is too schematic for us to be able to observe the copycat rule at work.

The copycat *rule* should not be confused with the copycat *strategy*, though when we define composition in the next chapter we will see a strong relationship between the two. The copycat rule will ensure that, on a hidden arena, there will exist two threads between which P can play copycat.

Convention We take m, n to range over moves and move-occurrences, μ, ν to range over hypermoves and hypermove-occurrences, and h to range over hypersequences.

5.4 Positions and winning

Our final refinement of the notion of hypersequence is that of a *position*. We think of the collection of positions of a polymorphic arena A as defining a game, namely the hypergame associated with A.

DEFINITION 5.7 *Let A be a polymorphic arena over a set of global holes H. A **position** of A is a legal hypersequence $p = (s, \curvearrowleft)$ of A satisfying the following conditions.*

1. ***Full justification.*** *Apart from the first hypermove of p, the first move of every hypermove of p is a continuation move. (Hence the first move of every hypermove of p (apart from the first hypermove) is justified, and in a picture of p 'every column is justified'.)*
2. ***O-trivial justification.*** *Justification pointers from O-moves point into the previous hypermove. In other words, if $m \in \mu \in s$, $n \in \nu \in s$, $m \curvearrowleft n$ and $\mathit{pl}(n) = O$, then μ and ν are consecutive in p, i.e., $s = s_1 \mu \nu s_2$.*

We write $\mathit{Pos}(A)$ for the (non-empty, prefix-closed) set of positions of A.

The hypersequence in Figure 5.1 satisfies conditions 1 and 2.

Intuitively, we think of the collection $\mathit{Pos}(A)$ of positions of a polymorphic arena A as defining the hypergame associated with A. The 'opening position' is the empty sequence ϵ, and p is reachable from q if $q \sqsubseteq p$. Note that $\mathit{Pos}(A)$ is non-empty (containing ϵ) and prefix-closed.

5.4.1 Winning

In order to define the notion of winning strategy later, which is the class of strategy for which our full completeness result is obtained, we require the following definition.

DEFINITION 5.8 *A **winning position**[4] is a position in which P has 'run O out of moves'. Formally, a winning position is a prefix-maximal position in which O is to move next, in other words, a position $p \in \mathit{Pos}(A)$ such that $|p|$ is even and for all $q \in \mathit{Pos}(A)$, if $p \sqsubseteq q$ then $p = q$.*

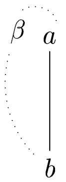
[4]We adhere to the usual bias towards the point of view of P.

5.4.2 Examples

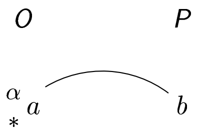
In this section we consider some examples of positions on various polymorphic arenas. In particular, we look at the arena interpreting the type $\forall X.X \to X$ of polymorphic identity, and the arenas interpreting the system F encodings of the boolean and natural number datatypes, as given (for example) in book *Proofs and Types* [GLT89a]. These examples are very simple in that each of the hypermoves consists of only a single move, because the types have no universal quantifiers in negative position. More interesting examples will be considered in a later section.

Polymorphic identity

The polymorphic arena interpreting[5] the type $\forall X.X \to X$ of the polymorphic identity $\Lambda X.\lambda x^X.x$ is:



The only positions on this arena are the prefixes of the following winning position:

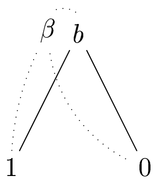


After O's first hypermove consisting of the action a, storing a 'hidden' arena in the hole α, P has no choice but to play b justified by a. Formally, the set of positions of this arena is $\{\epsilon, \overset{\alpha}{\underset{*}{}} a, p\}$, where p is the two-move position displayed above. Note that the copycat rule is satisfied, as both b and a reference α in the position.

The storage of the hidden arena by O into β plays no real role. We shall see the influence of holes such as β in the examples of a later section.

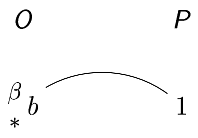
Booleans

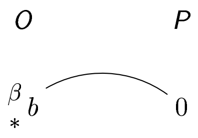
Let $\mathit{Bool} \in \mathbb{PA}$ be the polymorphic arena



interpreting `Bool` $= \forall X.X \to X \to X$, the system F encoding of the Boolean datatype. Then the set of positions $\mathit{Pos}(\mathit{Bool})$ is the prefix-closure of the following pair of winning positions:

[5] The formal definition of the interpretation of types as polymorphic arenas comes in Chapter 8, as a side-effect of showing that the model has the structure of a $2\lambda\times$-hyperdoctrine.

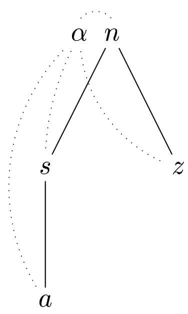




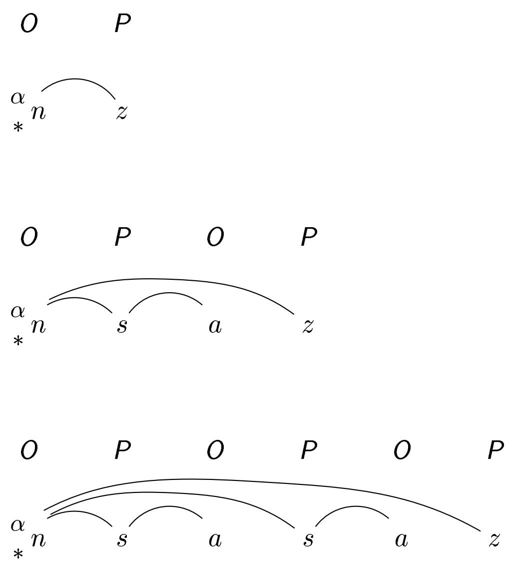
After O's first hypermove consisting of the action b 'requesting Boolean data', P has a choice: either 'supply Boolean data 1' or 'supply Boolean data 0'. The hypergame is then finished, because O has no moves available. Note that the copycat rule is satisfied in both positions.

Natural numbers

Let $\mathit{Nat} \in \mathbb{PA}$ be the polymorphic arena



interpreting $\texttt{Nat} = \forall X.(X \to X) \to X \to X$, the system F encoding of the datatype of natural numbers. Then $\mathit{Pos}(\mathit{Nat})$ is the prefix-closure of the following scheme of winning positions:



⋮

After O's first hypermove with action n 'requesting a number', P has a choice: either 'successor' s or 'zero' z. After z the game is over because there are no O-moves available (as in the first position above); after s, O has one continuation a available, 'request for argument' or "successor of what?" (as in the second and third positions). The request for an argument can be satisfied either by supplying 'zero' z (as in the second position), or by making another call to 'successor' s (as in the third position). And so on. Thus there winning positions p_i for each $i \geq 0$, where i is the number of times P plays s before playing z.

5.4.3 Various technical lemmas

The following lemmas will not be needed for the time being, but are collected together here since they concern positions.

LEMMA 5.9 *Positions are finitely branching at O-hypermoves. In other words, if $p \in \mathit{Pos}(A)$ and $|p|$ is even, then $\{\ p\mu\ :\ p\mu \in \mathit{Pos}(A)\ \}$ is finite.*

Proof O does not import arenas, so every O-move corresponds merely to the choice of an action in a location. The result is then immediate from the fact that polymorphic arenas are by definition finite, and since O can only target justification pointers into the previous P-hypermove, which consists of a finite number of moves, there are only a finite number of locations to choose from. □

LEMMA 5.10 *Let p be a position of A. Then all depth 1 moves of p are located in A.*

Proof By definition of depth, every depth 1 move m is necessarily the first move of the hypermove containing it. Hence, by the full justification condition on positions, every such move (apart from the very first move of p) is a justified move, and all depth 1 moves are hereditarily justified by the very first move m_1 of p. Since the location of m_1 is A (by condition 1 of Definition 5.4 of hypersequence), and since justification respects enabling (condition 3 of the definition of $\curvearrowleft$ of Definition 5.4), m is located in A. □

LEMMA 5.11 *In any position p,*

1. *If $m \curvearrowleft n$ then $\mathsf{depth}(m) = \mathsf{depth}(n)$.*
2. *If m is an opening move, and is not the very first move of p, then $\mathsf{depth}(m) \geq 2$.*

Proof Immediate from the inductive definition of depth.

LEMMA 5.12 *Let p be a position of a polymorphic arena A. If A has no local holes at even level*[6] *then:*

1. *Every hypermove in p consists of a single move.*
2. *Every move of p is at depth 1 and is located in A.*
3. *No polymorphic arenas are stored during p, i.e., $\bigcup_{m \in p} \mathsf{store}_p(m) = \epsilon$ or $\{*\}$.*

Proof Immediate from the definition of position. □

LEMMA 5.13 *Let p be a position of a polymorphic arena A.*

1. *Every O-hypermove of p consists in at most 2 moves.*
2. *The first O-hypermove of p (if p is non-empty) is a single move.*

Proof (1) The second move $m = (B, \phi, a)$ of a hypermove is an opening move. Since m is an opening move played by O, there are no P-moves hereditarily justifiying it, so it cannot reference a hole storing an arena.

(2) is trivial. □

5.5 Winning strategies

A strategy is a 'cribsheet' telling P how to respond every time O plays a hypermove. We work with Hyland/Ong's formalisation [HO94], adapted to a hypersequence/hypermove context. Following the conventions of McCusker [McC96b], we leave justification pointers implicit whenever possible. For example, given a position p and a hypermove μ, when we write "$p\mu$" we mean "a hypersequence extending p by μ, together with a pointer from the first move of μ back to some move in p." So a statement such as "$p\mu = p\nu$" asserts the equality of the positions $p\mu$ and $p\nu$ (including their 'silent' justification pointers), rather than the mere equality of their underlying sequences of hypermoves.

DEFINITION 5.14 *A **strategy** σ for A is a non-empty prefix-closed subset of $\mathsf{Pos}(A)$ satisfying*

1. P-determinism. *If $p\mu, p\nu \in \sigma$ and $\mathsf{pl}(\mu) = \mathsf{P}$, then $p\mu = p\nu$.*

[6] Recall that opening actions have level 1.

2. O-contingent completeness. *If $p \in \sigma$ and $p\mu \in Pos(A)$ with $pl(\mu) = O$, then $p\mu \in \sigma$.*

A winning strategy is one that always reaches a winning position, no matter how hard O tries to avoid it. We formalise the idea below. Recall that a winning position is one in which P has just 'run O out of moves', i.e., a winning position is an even-length prefix-maximal position.

DEFINITION 5.15 *A strategy σ for A is **winning** if it is the prefix-closure of a finite set of winning positions. We write $wins(\sigma)$ for the finite set of winning positions generating σ by prefix-closure.*

Finiteness is part of the definition because it is no good to 'wait forever' to reach a winning position. For example, the strategy corresponding to the infinite 'top-down term' $\lambda f^{(X\to X)\to X\to X}.fi(fi(fi(\ldots)))$ of type $((X \to X) \to X \to X) \to X$, where $i = \lambda x^X.x$, is the prefix-closure of a set of winning positions.

A property related to winning is the following.

DEFINITION 5.16 *A strategy σ is **total** if it always manages to produce a response to an O-hypermove. Formally, if $p\mu \in \sigma$ with $pl(\mu) = O$ then $p\mu\nu \in \sigma$ for some hypermove ν.*

LEMMA 5.17 *If a strategy is the prefix-closure of a set of winning positions, then it is total.*

Proof Every position p in which P has to find a move is the *strict* prefix-closure of one of the wins q, since $|p|$ is odd and $|q|$ is even. $\square$

The converse is not true. The strategy corresponding to the infinite 'top-down term' $\lambda f^{X\to X}.f(f(f(\ldots)))$ of type $(X \to X) \to X$ ('trying hard' to be the fixpoint combinator) is total, but is not the prefix-closure of a set of winning positions.

PROPOSITION 5.18 *A strategy is winning if and only if it is total and finite.*

Proof If σ is winning, then it is finite by definition, and is total by Lemma 5.17.

Suppose σ is total and finite. Then take $wins(\sigma)$ to be the set of prefix-maximal positions of σ. By finiteness of sigma, the prefix-closure of $wins(\sigma)$ is σ. Furthermore, every $p \in wins(\sigma)$ is of even length, otherwise σ would not be total at p, hence every $p \in wins(\sigma)$ is a winning position. $\square$

5.5.1 Examples

We consider some examples of winning strategies on various polymorphic arenas. In particular, we look at the arena interpreting the type $\forall X.X \to X$ of polymorphic identity, and the arenas interpreting the standard system F encodings of booleans, natural numbers, sums, products, and lists, as given in *Proofs and Types* [GLT89a].

Polymorphic identity

The unique winning position of the polymorphic arena interpreting $\forall X.X \to X$ was shown on page 73. Since there is only move available to P after O's opening move, there is only one winning strategy σ, given by $wins(\sigma) = \{p\}$, where p is the (unique) winning position of the arena displayed on page 73. Formally, the winning strategy is $\{\epsilon, \overset{\alpha}{\underset{*}{}}\, a, p\}$.

Booleans

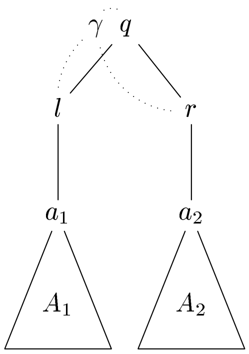
Let p be the first winning position of *Bool* shown on page 74, and let q be the position displayed underneath p. There are two winning strategies, ***true*** and ***false***, given by *wins*(***true***) $= \{p\}$ and *wins*(***false***) $= \{q\}$. The former picks “1” in response to *O*'s opening hypermove, and the latter picks “0”.

Natural numbers

Refer to the positions p_i of *Nat* displayed on page 75. Each $i \geq 0$ corresponds to a winning strategy $\underline{i}$, which plays s (‘successor’) i times, then plays z (‘zero’). Formally, *wins*$(\underline{i}) = \{p_i\}$.

Sum

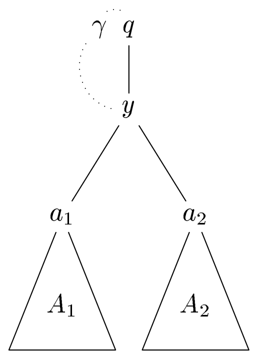
Given types T_1 and T_1, the system F encoding of the sum of T_1 and T_2 is $\mathtt{Sum}(T_1,T_2) = \forall X.(T_1 \to X) \to (T_2 \to X) \to X$, for X chosen not free in T_1 or T_2. Let A_1 and A_2 be the arenas interpreting T_1 and T_2 respectively, and for simplicity assume that A_1 and A_2 are trees, with opening actions a_1 and a_2 respectively. Then the polymorphic arena interpreting $\mathtt{Sum}(T_1,T_2)$ has the following shape:



Any winning strategy σ on A_1 gives rise to a winning strategy for this arena as follows. When *O* plays the ‘question’ q asking “in which component would you like to play?”, reply l for “left”. Then proceed to play σ in the subarena A_1. Similarly, any winning strategy τ on A_2 gives rise to a winning strategy on $A_1 + A_2$ by replying to q with r for “right”, then playing τ in A_2. Interestingly, this “left or right?” idea is similar to the interpretation of FPC sums in Guy McCusker's Ph.D. thesis [McC98].

Product

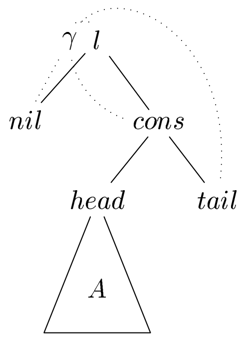
Given types T_1 and T_1, the system F encoding of the product of T_1 and T_2 is $\mathtt{Prod}(T_1,T_2) = \forall X.(T_1 \to T_2 \to X) \to X$, for X chosen not free in T_1 or T_2. Let A_1 and A_2 be the arenas interpreting T_1 and T_2 respectively, and for simplicity assume that A_1 and A_2 are trees, with opening actions a_1 and a_2 respectively. Then the polymorphic arena interpreting $\mathtt{Prod}(T_1,T_2)$ has the following shape:



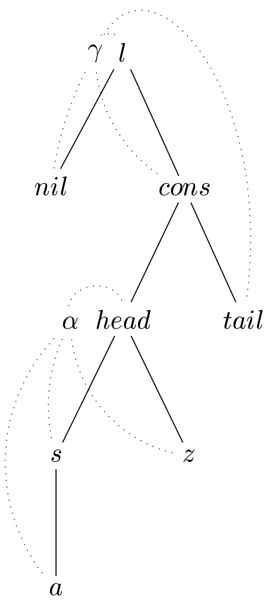
Any pair (σ, τ) of winning strategies σ on A_1 and τ on A_2 defines a winning strategy for this arena as follows. When O plays the 'question' q asking "are you ready to play?", reply y for "yes". Then proceed to play σ in the subarena A_1 if O's next move is a_1, and play τ in the subarena A_2 if O's next move is a_2.

Lists

Given a type T, the system F encoding of the type of lists of elements of type T is $\mathtt{List}T = \forall X.X \to (T \to X \to X) \to X$, for X chosen not free in T. Let A be the arena interpreting T, and for simplicity assume that A is a tree. Without loss of generality, rename the opening action of A to be *head*. The polymorphic arena interpreting $\mathtt{List}T$ has the following shape:



Every list $\sigma_1 \ldots \sigma_k$ of winning strategies σ_i for A defines a distinct winning strategy $\langle \sigma_1, \ldots, \sigma_k \rangle$ for this arena as follows. In order to simplify matters, we fix a specific type T, namely $T = \mathtt{Nat}$, and focus on type $\mathtt{ListNat} = \forall X.X \to (\mathtt{Nat} \to X \to X) \to X$ of lists of natural numbers. The following arena *ListNat* is the interpretation of this type:



(Notice that the *Nat* subarena has had its opening move n renamed to *head*). Since we already know (from page 78) that natural numbers are in bijection with winning strategies on *Nat*, we work with lists of natural numbers such as 132 rather than actual lists of winning strategies $\underline{1}\,\underline{3}\,\underline{2}$.

The empty sequence of natural numbers corresponds to the winning strategy with the following winning position:

$$\begin{array}{cc} O & P \\ \overset{\gamma}{*}\, l & nil \end{array}$$

Upon O asking to 'look at the list' with the action l, P immediately responds with *nil* because the list is empty.

The winning strategy corresponding to the singleton list 1 is given by the following pair of winning positions:

$$\begin{array}{cccccc} O & P & O & P & O & P \\ \overset{\gamma}{*}\, l & cons & \overset{\alpha}{*}\, head & s & a & z \\ \overset{\gamma}{*}\, l & cons & tail & nil & & \end{array}$$

Upon O's opening request l to 'look at the list', P responds with *cons* because, unlike the example of the empty sequence above, in which P responded with *nil*, there is an element in the list this time. Now O has a choice between (a) *head*, looking at the 'head' of the list (as in the first position above), or (b) *tail*, skipping the first element and asking to look at the next element (as in the second position). In case (a), *head* is the opening move of the *Nat* subarena (remember that the opening action n of *Nat* was renamed to *head*), and P

continues with the strategy $\underline{1}$ on *Nat*. In case (b), since there is only one element in the list, the tail is empty, so *P* responds with nil.

The winning strategy corresponding to the three-element list 101 is given by the following winning positions:

O	*P*	*O*	*P*	*O*	*P*	*O*	*P*	*O*	*P*
$\overset{\gamma}{\underset{*}{}}\, l$	$cons$	$\overset{\alpha}{\underset{*}{}}\, head$	s	a	z				
$\overset{\gamma}{\underset{*}{}}\, l$	$cons$	$tail$	$cons$	$\overset{\alpha}{\underset{*}{}}\, head$	z				
$\overset{\gamma}{\underset{*}{}}\, l$	$cons$	$tail$	$cons$	$tail$	$cons$	$\overset{\alpha}{\underset{*}{}}\, head$	s	a	z
$\overset{\gamma}{\underset{*}{}}\, l$	$cons$	$tail$	$cons$	$tail$	$cons$	$tail$	nil		
O	*P*	*O*	*P*	*O*	*P*	*O*	*P*	*O*	*P*

The first position is *O* inspecting the first element 1, and is the same as the first position of the singleton sequence 1 above. The second position is *O* inspecting the second element 0, having played $tail$ to skip the first element. The third position is *O* inspecting the third element 1, skipping the first two elements by playing $tail$ twice. The fourth position is an attempt by *O* to inspect a fourth element, by playing $tail$ three times; *P* returns nil because there is no fourth element.

In general a list $m_1 \dots m_k$ of k numbers defines the winning strategy with $k+1$ winning positions $p_1, p_2, \dots, p_{k+1}$ as follows. Each p_i begins with $\overset{\gamma}{\underset{*}{}}\, l$, *O*'s opening request to extract data from the list. This is followed by $i-1$ pairs of $cons \frown tail$, corresponding to *O* 'skipping' the first $i-1$ elements of the list in order to reach the i^{th}. If $i \leq k$ this is followed by a $cons$ from *P* to say "yes, there is an i^{th} element"; otherwise $i = k+1$ and it is followed by a nil from *P* to say "there are no more elements". For $i \leq k$ the rest of p_i consists of the strategy for the natural number m_i (with the opening move n of *Nat* renamed to $head$).

Thus every list of natural numbers defines a distinct winning strategy on *ListNat*.

5.6 A suite of examples

We present more examples of winning strategies for polymorphic arenas with holes occurring at even level. These strategies are more interesting than those of the previous sections, because they involve the storage of arenas and then the subsequent migration of play to the stored arenas. The winning strategies presented here (specifically *even* : *Nat* $\rightarrow$ *Bool* and *inc* : *Nat* $\rightarrow$ *Nat*) will be used in the motivating example of interaction at the beginning of Chapter 6.

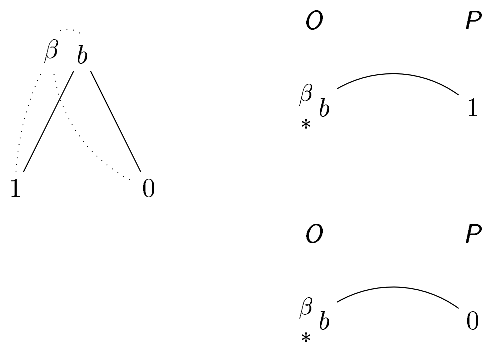


Figure 5.2: The winning strategies *true* and *false* interpreting the terms `true` $= \Lambda X.\lambda x^X.\lambda y^X.x$ and `false` $= \Lambda X.\lambda x^X.\lambda y^X.y$ respectively. The top position is the winning position of *true* and the bottom position is the winning position of *false*.

For the sake of completeness, we also recall the boolean and natural number examples of the previous section. There are six figures, as follows:

- Figure 5.2: *true* and *false* on *Bool*.
- Figure 5.3: *not* on *Bool* $\rightarrow$ *Bool*.
- Figure 5.4: $\underline{0}$, $\underline{1}$ and $\underline{2}$ on *Nat*.
- Figure 5.5: *inc* (increment by one) on *Nat* $\rightarrow$ *Nat*.
- Figure 5.6: *even* on *Nat* $\rightarrow$ *Bool*, the interpretation of a term returning `true`/`false` according to whether its input is even/odd.
- Figure 5.7: *odd* on *Nat* $\rightarrow$ *Bool*, the interpretation of a term returning `false`/`true` according to whether its input is even/odd.

Below are commentaries to the figures.

Commentary to Figure 5.2 (*true*, *false* : *Bool*.) The strategies *true* and *false* were discussed in section 5.5.1.

Commentary to Figure 5.3 (*not* : *Bool* $\rightarrow$ *Bool*.) Before detailing the relationship between the structure of the strategy and the structure of the term, we give an intuitive reading of the strategy in pure game-theoretic terms.

In the arena *Bool*, think of the action b as "request for boolean input", and think of 1 and 0 as "supply of boolean data". Then, focusing on the underlying sequences of actions $bb'1'0$ and $bb'0'1$ in the top rows of the two positions, the strategy reads as follows. *O* starts with a request b for boolean data. Since *not* is a constant function *P* cannot output a value until he has received input. So he goes to the 'input *Bool* component', *Bool*$'$, and requests boolean data from *O* by playing b'. If *O* supplies data 1' on the input component, *P* responds with 0 to *O*'s original request for boolean data b (hence the justification pointer back to b); if *O* suplies data $0'$ on the input component, *P* responds with 1 to *O*'s original request b.

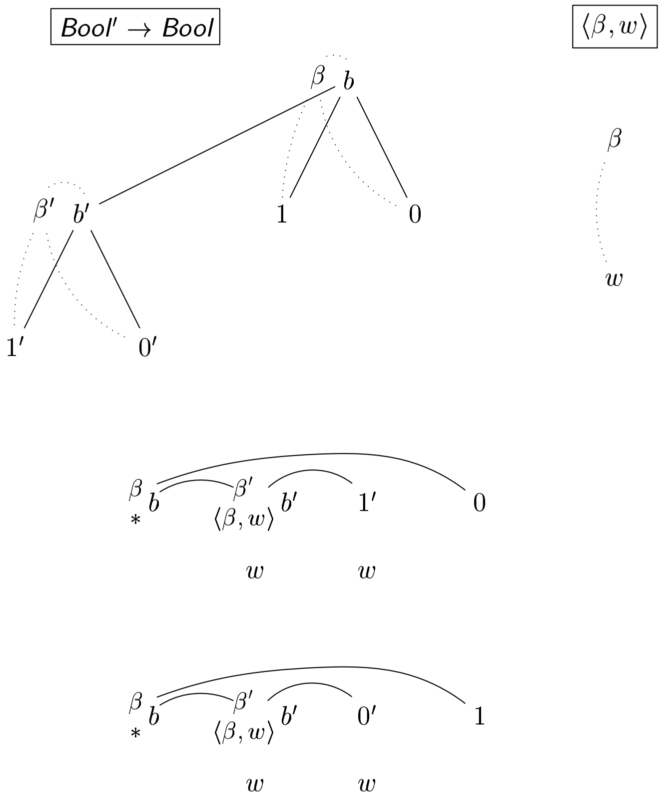


Figure 5.3: The winning positions of the winning strategy *not* interpreting the term `not` $= \lambda f^{\texttt{Bool}}.\Lambda X.\lambda x^X.\lambda y^X.fXyx$. A commentary is given in the main body of the text, on page 82.

The relationship of the structure of the strategy with the structure of the interpreted term

$$\mathtt{not} = \lambda f^{\mathtt{Bool}}.\Lambda X.\lambda x^X.\lambda y^X.fXyx$$

is as follows. After *O*'s first move $\overset{\beta}{*}\, b$, the three children b', 0 and 1 correspond respectively to the three λ-abstractions λf, λx, and λy in the term. The hole β of *O*'s first move $\overset{\beta}{*}\, b$ corresponds to the Λ-abstraction ΛX. The fact that subsequent arenas imported by *P* can reference β corresponds to the fact that subsequent type arguments can have X as a free variable.

P chooses the action b', corresponding to the choice of f as headvariable for the body of the term. Since b' has a hole β', *P* must store a polymorphic arena in β'. This corresponds to the fact that f is of polymorphic type, and must be supplied a type argument. *P* stores the arena $\langle \beta, w\rangle$ in β', the singleton arena consisting of the action w referencing the hole β of *O*'s first move. This corresponds to the argument X supplied to f in the term.

Since b' references β, *P* must open a thread on its contents $\langle \beta, w\rangle$. Since the arena is a singleton, there is no choice put to play its only action w. Since w does not reference a hole containing a stored arena, there is no need to play another opening move, and *P*'s first hypermove is complete. Note that the copycat rule is satisfied, because this last move w references β, which is the reference of *O*'s action b.

For his second move, *O* can choose between two locations: he can either continue in the arena *Bool*$'\to$ *Bool* by justifying from b', or continue in the arena $\langle \beta, w\rangle$ by justifying from w. He is forced to do the former, since w has no children. So *O*'s options are (i) pick the child $1'$ of b' in *Bool*$'\to$ *Bool*, as in the top winning position, or (ii) pick the child $0'$ of b' in *Bool*$'\to$ *Bool*, as in the bottom winning position.

(i) In the term this corresponds to 'inspecting the first argument of fX'. Since $1'$ references β', *O* must follow up immediately with an opening move on the arena $\langle \beta, w\rangle$ stored in β'. Since $\langle \beta, w\rangle$ is a singleton, there is no choice but to play its only action w. This completes a hypermove for *O*, since w references β, which does not store an arena.

P's response is to play 0 in *Bool*$'\to$ *Bool*, justified by *O*'s first action b. This corresponds to the fact that the first argument of fX is y, which, as discussed earlier, is the abstracted variable corresponding to the action 1. This completes a hypermove for *P*, because 1 references β, which does not store an arena. The copycat rule is satisfied, because β was the reference of *O*'s last move w.

We have reached a winning position, since 0 is a leaf, leaving no actions available to *O*.

(ii) In the term this corresponds to 'inspecting the second argument of fX'. It is analogous to (i).

Commentary to Figure 5.4 ($\underline{0}$, $\underline{1}$, and $\underline{2}$ on *Nat*.) Thinking of n as "What's the number?", s as "successor", z as "zero", and a as "of", the third sequence $nsasaz$ reads as: "What's the number? Successor of successor of zero." In more detail, *O* starts with "What's the number?". *P* replies with "It's the successor." *O* says "The successor of what?" *P* responds with "(The successor of) the successor". *O* replies "(The successor of) the successor of what?". *P* finishes with "((The successor of) the successor of) zero". In other words, the number 2.

Each action s by *P* corresponds to picking the variable f. Each action a by *O* corresponds to 'inspecting the argument of f'. The action z corresponds to picking the variable x.

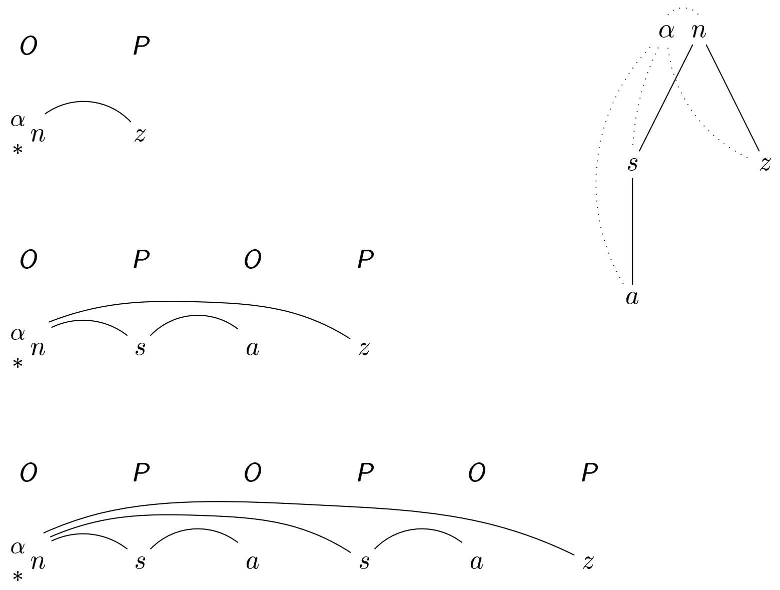


Figure 5.4: The winning strategies $\underline{\mathit{0}}$, $\underline{\mathit{1}}$, and $\underline{\mathit{2}}$ interpreting the terms $\Lambda X.\lambda f^{X\to X}.\lambda x^X.x$, $\Lambda X.\lambda f^{X\to X}.\lambda x^X.fx$, and $\Lambda X.\lambda f^{X\to X}.\lambda x^X.f(fx)$ respectively. The top position is the winning position of $\underline{\mathit{0}}$, the middle position is the winning position of $\underline{\mathit{1}}$, and the bottom position is the winning position of $\underline{\mathit{2}}$. A commentary is given in the main body of the text, on page 84.

Commentary to Figure 5.5 ($\mathit{inc} : \mathit{Nat} \to \mathit{Nat}$.) Before detailing the relationship between the structure of the increment strategy and the structure of the term, we give an intuitive reading of the strategy in pure game-theoretic terms.

In the arena *Nat*, think of the action n as "request for a number", think of s as "successor", think of a (standing for "argument") as "successor of what?", and think of z as "zero". Then, focusing on the underlying sequences of actions in the top rows of the two positions (ignoring the storage of the arena $\langle \alpha, w\rangle$ and the moves w located on it), the strategy reads as follows.

O starts with n asking "What's the number?" Since *inc* is not a constant function, *P* must inspect the input. So he goes to the 'input *Nat* component', *Nat*$'$, and plays n', asking "What's the input number?" *O* responds with z', "the input number is zero." Now *P* proceeds with the strategy for successor of zero back on the output component *Nat*, i.e., *P* proceeds with the strategy $\underline{\mathit{1}}$.

The intuition for the second winning position only makes sense when one observes the interaction of *inc* with a number.

The relationship of the structure of the strategy with the structure of the η-long variant

$$\lambda m^{\mathit{Nat}}.\Lambda X.\lambda f^{X\to X}.\lambda x^X.mX(\lambda e^X.fe)(fx)$$

of the interpreted term

$$\lambda m^{\mathtt{Nat}}\Lambda X.\lambda f^{X\to X}.\lambda x^X.mXf(fx)$$

is is as follows.

After *O*'s first move $\overset{\alpha}{\underset{*}{}} n$, the three children n', s and z correspond respectively to the three λ-abstractions λg, λf, and λx. The hole α of *O*'s first move $\overset{\alpha}{\underset{*}{}} n$ corresponds to the Λ-

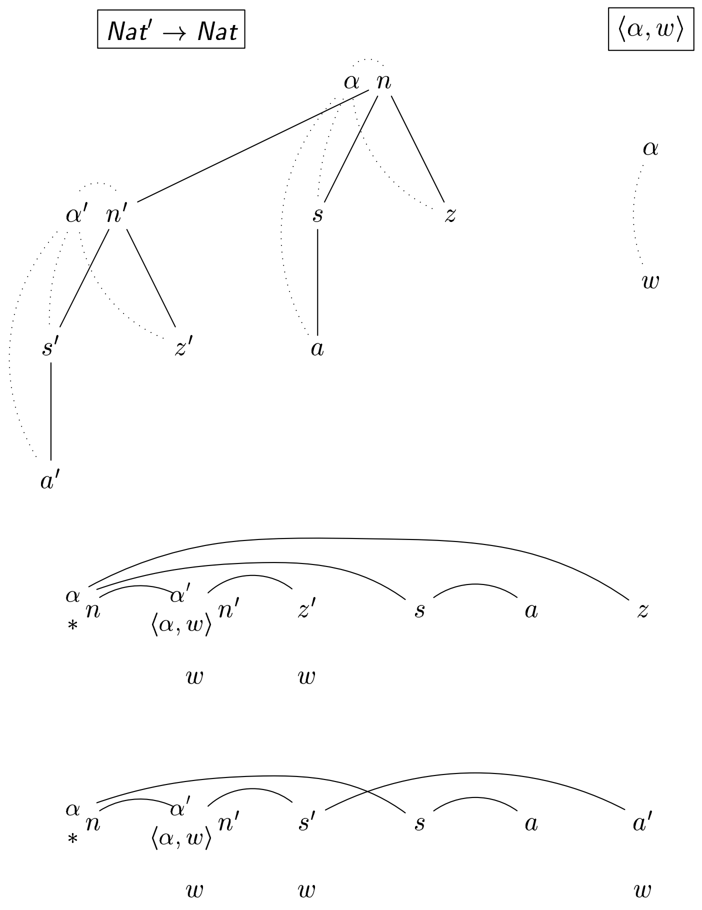


Figure 5.5: The two winning positions of the winning strategy *inc* : $\mathsf{Nat}' \to \mathsf{Nat}$ interpreting the term $\lambda g^{\mathtt{Nat}}\Lambda X.\lambda f^{X\to X}.\lambda x^X.gXf(fx)$, which is $\lambda g^{\mathsf{Nat}}.\Lambda X.\lambda f^{X\to X}.\lambda x^X.gX(\lambda e^X.fe)(fx)$ in η-long form. A commentary is given in the main body of the text, on page 85.

abstraction ΛX. The fact that subsequent arenas imported by *P* can reference α corresponds to the fact that subsequent type arguments can have X as a free variable.

P chooses to play the action n', corresponding to the choice of g as headvariable for the body of the term. Since n' has a hole α', *P* must store a polymorphic arena in α'. This corresponds to the fact that g is of polymorphic type, and must be supplied a type argument. *P* stores the arena $\langle\alpha, w\rangle$ in α', the singleton arena consisting of the action w referencing the hole α of *O*'s first move. This corresponds to the argument X supplied to g in the term.

Since n' references α, *P* must open a thread on its contents $\langle\alpha, w\rangle$. Since the arena is a singleton, there is no choice put to play its only action w. Since w does not reference a hole containing a stored arena, there is no need to play another opening move, and *P*'s first hypermove is complete. Note that the copycat rule is satisfied, because this last move w references α, which is the reference of *O*'s action n.

For his second move, *O* can choose between two locations: he can either continue in the arena $\mathit{Nat}' \to \mathit{Nat}$ by justifying from n', or continue in the arena $\langle\alpha, w\rangle$ by justifying from w. He is forced to do the former, since w has no children. So *O*'s options are (i) pick the second child z' of n' in $\mathit{Nat}' \to \mathit{Nat}$, as in the top winning position, or (ii) pick the first child s' of n' in $\mathit{Nat}' \to \mathit{Nat}$, as in the bottom winning position.

(i) In the term this corresponds to 'inspecting the second argument of gX'. Since z' references α', *O* must follow up immediately with an opening move on the stored arena $\langle\alpha, w\rangle$. Since $\langle\alpha, w\rangle$ is a singleton, there is no choice but to play its only action w. This completes a hypermove for *O*, since w references α, which does not store an arena.

P's response is to play s back on $\mathit{Nat}' \to \mathit{Nat}$, justified by *O*'s first action n. This corresponds to the fact that the head-variable of the second argument of gX is f, which, as discussed earlier, is the abstracted variable corresponding to the action s. This completes a hypermove for *P*, because s references α, which does not store an arena. The copycat rule is satisfied, because α was the reference of *O*'s last move w.

O has no choice but to play the only child a of s. This corresponds to the fact that f, being of type $X \to X$, has only one argument. This immediately completes a hypermove for *O*, since a references α, which does not store an arena.

P has the following choices for his next action: n' justified by n, s justified by n, or z justified by n. These correspond to the lambda-abstractions λg, λf, and λx respectively, from which *P* chooses the headvariable of the argument to f. The fact that the argument is x corresponds to the fact that *P*'s next action is z.

(ii) In the term this corresponds to 'inspecting the first argument of gX'. Since s' references α', *O* must follow up immediately with an opening move on the stored arena $\langle\alpha, w\rangle$. Since $\langle\alpha, w\rangle$ is a singleton, there is no choice but to play its only action w. This completes a hypermove for *O*, since w references α, which does not store an arena.

The first argument of gX is of type $X \to X$. This is not a base type, so in η-long form, the argument must begin with an abstraction of type X, namely λe^X.

P has the following choices for his next action: n' justified by n, s justified by n, z justified by n, or a' justified by s'. These correspond to the lambda-abstractions λg, λf, λx, and λe, respectively, from which *P* chooses the headvariable of the argument subterm $\lambda e^X.fe$. The fact that the headvariable is f corresponds to the fact that *P*'s next action is s.

O is forced to pick the only child a of s, corresponding to the fact that he is forced to inspect the one and only argument of f.

P's last action is a' in $\mathit{Nat}' \to \mathit{Nat}$, justified by *O*'s action s'. This corresponds to the fact that the argument of f is e, which, as discussed above, corresponds to the action a'.

This is a winning position because a' is a leaf (corresponding to the fact that the body of $\lambda e^X.fe$ has reached ground type).

Commentary to Figure 5.6 ($\mathit{even} : \mathit{Nat} \to \mathit{Bool}$.) The idea of the interpreted term

$$\lambda g^{\mathtt{Nat}}.g\,\mathtt{Bool}\,\mathtt{not}\,\mathtt{true}$$

is to apply `not` to `true` g times, hence (after normalisation) yielding `true` if g is even and `false` if g is odd.

The structure of the strategy relates to the structure of the η-long variant

$$\lambda g^{\mathtt{Nat}}.\Lambda X.\lambda x^X.\lambda y^X.g\,\mathtt{Bool}\,\mathtt{not}\,\mathtt{true}\,X\,x\,y$$

of the term as follows. First we disentangle the different occurrences of the quantified type variable $\forall X$. Let $\mathtt{Bool}' = \forall X'.X' \to X' \to X'$, i.e., a copy of $\mathtt{Bool} = \forall X.X \to X \to X$ with its bound variable renamed to X'. Write $\mathtt{not}' : \mathit{Bool}' \to \mathit{Bool}'$ for

$$\lambda f^{\mathtt{Bool}'}.\Lambda X'.\lambda x^{X'}.\lambda y^{X'}.fX'yx$$

and write $\mathtt{true}' : \mathtt{Bool}'$ for

$$\Lambda X'.\lambda x^{X'}.\lambda y^{X'}.x$$

Then the η-long form is

$$\lambda g^{\mathtt{Nat}}.\Lambda X.\lambda x^X.\lambda y^X.g\,\mathtt{Bool}'\,\mathtt{not}'\,\mathtt{true}'\,X\,x\,y$$

i.e.,

$$\lambda g^{\mathtt{Nat}}.\Lambda X.\lambda x^X.\lambda y^X.g(\forall X'.X' \to X' \to X')(\lambda f^{\mathtt{Bool}'}.\Lambda X'.\lambda x^{X'}.\lambda y^{X'}.fX'yx)(\Lambda X'.\lambda x^{X'}.\lambda y^{X'}.x)Xxy$$

After O's first move $\overset{\beta}{*}\,b$, the three children n, 1 and 0 correspond respectively to the three λ-abstractions λg, λx, and λy. The hole β of O's first move $\overset{\beta}{*}\,b$ corresponds to the Λ-abstraction ΛX. The fact that subsequent arenas imported by P can reference β corresponds to the fact that subsequent type arguments can have X as a free variable.

P chooses to play the action n, corresponding to the choice of g as headvariable for the body of the term. Since n has a hole α, P must store a polymorphic arena in α. This corresponds to the fact that g is of polymorphic type, and must be supplied a type argument. P stores the arena Bool' in α, a tagged copy of Bool, as displayed top-right in the figure. In the term this corresponds to the type argument $\mathtt{Bool}'$ supplied to g.

Since n references α, P must open a thread on its contents Bool'. Since Bool' is a tree, with only one opening action b', there is no choice but to play b'.

Since b' has a hole β', P must store a polymorphic arena in β'. This correspondst to the fact that $g\mathtt{Bool}'$ is of type $(\mathtt{Bool} \to \mathtt{Bool}) \to \mathtt{Bool} \to \mathtt{Bool}$, i.e., $(\mathtt{Bool} \to \mathtt{Bool}) \to \mathtt{Bool} \to \forall X'.X' \to X' \to X'$, the prenex ('quantifiers outermost') form of which is $\forall X'.(\mathtt{Bool} \to \mathtt{Bool}) \to \mathtt{Bool} \to X' \to X' \to X'$, a polymorphic type requiring an argument. Another way of looking at this is a follows. Whatever arguments u and v are supplied in the future, $g\mathtt{Bool}'uv$ will be of polymorphic type, so it will require a type argument; we are being 'eager' and supplying the type argument (X) as early as possible.

P elects to store the arena $\langle \beta, c\rangle$ in β', the singleton arena with action c referencing the hole β of O's first move $\overset{\beta}{*}\,b$. This corresponds to the type argument X.

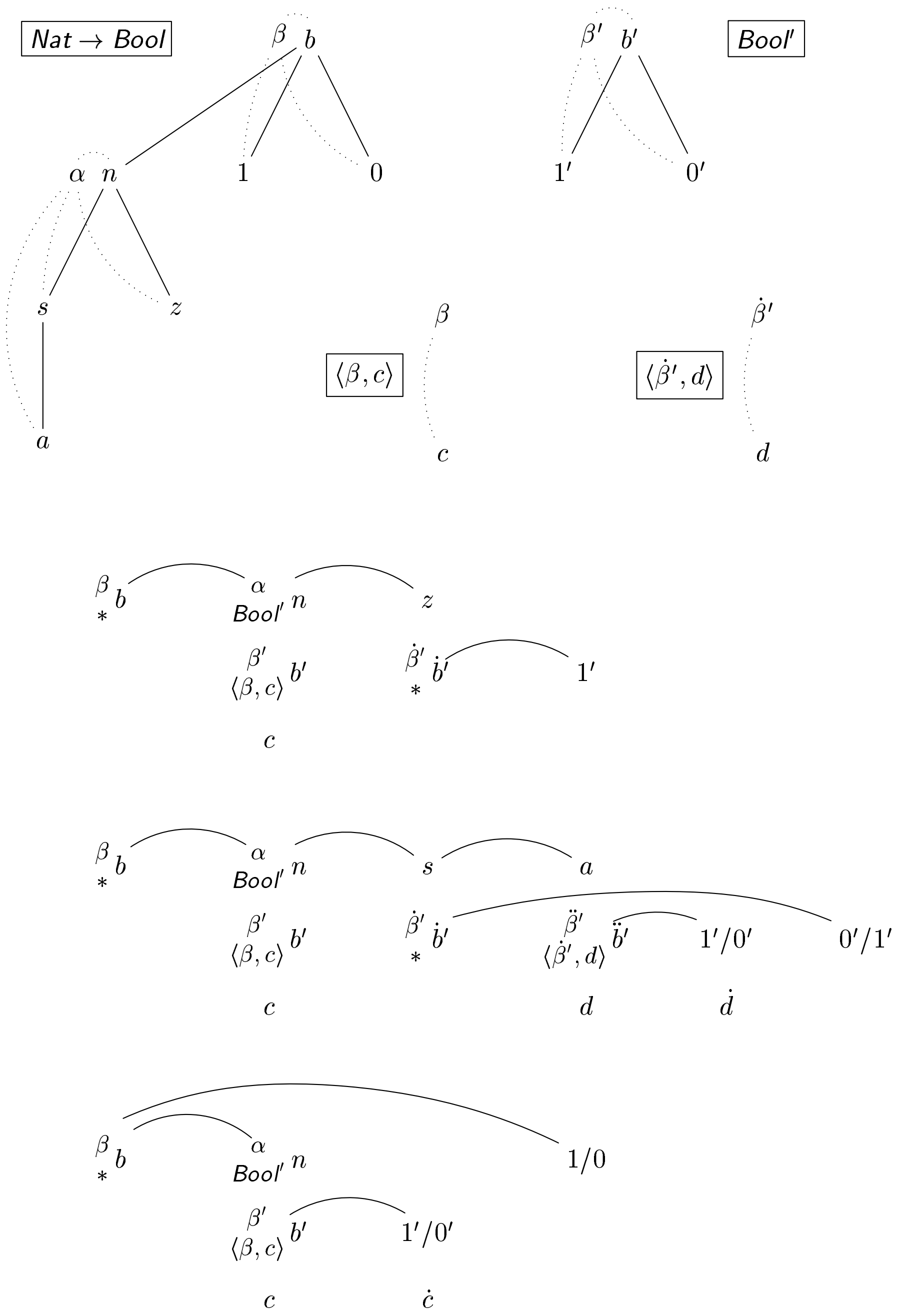


Figure 5.6: The winning strategy *even* : *Nat* → *Nat* interpreting the term $\lambda g^{\mathtt{Nat}}.g\,\mathtt{Bool}'\,\mathtt{not}'\,\mathtt{true}'$, which is $\lambda g^{\mathtt{Nat}}.\Lambda X.\lambda x^{X}.\lambda y^{X}.g\,\mathtt{Bool}'\,\mathtt{not}'\,\mathtt{true}'\,X\,x\,y$ in η-long form. The five winning positions are represented compactly: the second and third are 'two-in-one'. Read a_1 in each a_1/a_2 to extract one position, and read a_2 to extract the other. A commentary is given in the main body of the text, on page 88.

Since b' references β', *P* must open a thread on its contents $\langle\beta, c\rangle$. Since the arena is a singleton, there is no choice put to play its only action c. Since c does not reference a hole containing a stored arena, *P*'s first hypermove is complete. Note that the copycat rule is satisfied, because c references β, which is the reference of *O*'s action b.

For his second move, *O* can choose between three locations: (i) he can continue in *Nat* $\rightarrow$ *Bool* by justifying from n, (ii) he can continue in *Bool'* by justifying from b', or (iii), he can continue in $\langle\beta, c\rangle$ by justifying from c. The last is not possible, since c has no children, so we need consider (i) and (ii) only.

(i) *O* continues in *Nat* $\rightarrow$ *Bool* by justifying from n. Since n has two children, s and z, (a) *O* can choose z, as in the top winning position, or (b) *O* can choose s, as in the middle winning position(s).

(i)(a) *O* plays z, the second child of n. This corresponds to inspecting the second argument of g`Bool'`. Since z references β, *O* must open a thread on its stored arena *Bool'*. Because *Bool'* is a tree, *O* has no choice but to pick its root b', and 'store a hidden arena' $*$ in β'. (In order to be able to distinguish the new occurrences of the action b' and hole β' in the third hypermove from the previous occurrences in the second hypermove, we add an overhead 'dot'. This is a convention we shall employ in general, whenever it is useful to be able to distinguish different occurrences of actions, moves, holes, or hypermoves.) Since the second argument of g`Bool'` is `true'`, *P* finishes by playing the strategy for `true'` inside *Bool'*, i.e., by playing $1'$ justified by $\dot{b}'$.

(i)(b) *O* plays the first child s of n. This corresponds to inspecting the first argument of g`Bool'`. Since s references β, *O* must open a thread on its stored arena *Bool'*. Because *Bool'* is a tree, *O* has no choice but to pick its root b', and 'store a hidden arena' $*$ in β'. (As in (i)(a), we 'dot' the fresh occurrences of b' and β'.) Rather than going into the nitty gritty details of the remaining moves, it is more informative to observe that the rest of the position[7] consists in *P* playing the strategy for `not'` on *Bool'*. (To see this, pattern-match $\dot{b}'\ddot{b}'1'0'$ and $\dot{b}'\ddot{b}'0'1'$ with the picture of *not* in Figure 5.3.)

(ii) *O* continues in *Bool'* by justifying from b'. This sequence is 'two-in-one'; without loss of generality choose $1'$ and 1 from $1'/0'$ and $1/0$. So *O* plays the action $1'$ in *Bool'*, justified by b'. This corresponds to 'inspecting the first argument of g`Bool'``true'``not'`X'. Since $1'$ references β', containing the singleton arena $\langle\beta, c\rangle$, *O* has no choice but to play c.

P replies with 1 back up on *Nat* $\rightarrow$ *Bool*, corresponding to the fact that the inspected argument (i.e., the first argument of g`Bool'``true'``not'`X) is the variable x. (Recall that, as discussed at the beginning of the commentary, 1 and 0 correspond to λx and λy respectively.)

This is a winning position, because 1 is a leaf.

Commentary to Figure 5.7 (*odd* : *Nat* $\rightarrow$ *Bool*.) The strategy is the same as *even*, apart from $0'$ in place of $1'$ in the top position. This corresponds syntactically to the fact that the interpreted term has `false'` in place of `true'`.

5.7 Application: counting inhabitants of types

In this section we look ahead to the full completeness of the hypergames model, and demonstrate how it can be used to reason about system F. The reader interested in getting to the details of the hypergame model can safely skip to the next chapter without loss of continuity. The syntactic 'top-down term' argument corresponding to the game-theoretic proof of the proposition below is given in Appendix A.

[7] Remember that this is really two positions, depending on whether one reads the a_1/a_2 as a_1 or a_2.

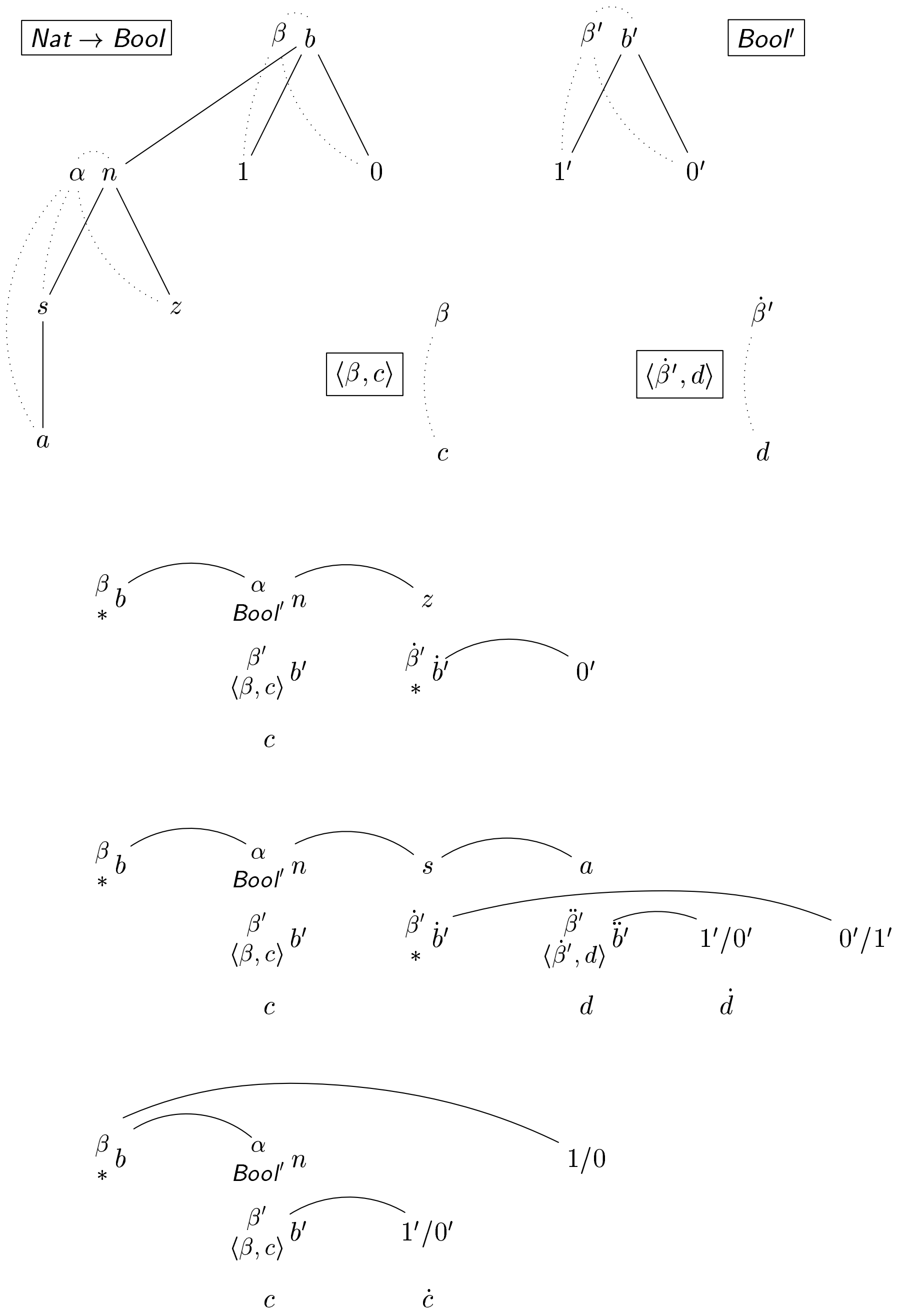


Figure 5.7: The winning strategy *odd* : *Nat* → *Nat* interpreting the term $\lambda g^{\mathtt{Nat}}.g\,\mathtt{Bool}'\,\mathtt{not}'\,\mathtt{false}'$, which is $\lambda g^{\mathtt{Nat}}.\Lambda X.\lambda x^{X}.\lambda y^{X}.g\,\mathtt{Bool}'\,\mathtt{not}'\,\mathtt{false}'\,X\,x\,y$ in η-long form. The five winning positions are represented compactly: the second and third are 'two-in-one'. Read a_1 in each a_1/a_2 to extract one position, and read a_2 to extract the other. The strategy is identical to *even* of Figure 5.6, but for $0'$ in place of $1'$ in the top position. A commentary is given in the main body of the text, on page 90.

We use the copycat rule as leverage to prove:

PROPOSITION 5.19 *Let U and V be types without negative occurrences of quantifiers. Then*

1. *$\beta\eta$-normal forms of type $\mathtt{Prod}(U,V) = \forall X.(U \to V \to X) \to X$ are in bijection with pairs consisting of a $\beta\eta$-normal form of type U and a $\beta\eta$-normal form of type V.*

2. *$\beta\eta$-normal forms of type $\mathtt{List}U = \forall X.X \to (U \to X \to X) \to X$ are in bijection with finite lists of $\beta\eta$-normal forms of type U.*

3. *The set of $\beta\eta$-normal forms of type $\mathtt{Sum}(U,V) = \forall X.(X \to U) \to (X \to V) \to X$ is in bijection with the disjoint union of the set of $\beta\eta$-normal forms of type U and the set of $\beta\eta$-normal forms of type V.*

These properties are not in general true for arbitrary types U and V: system F encodings of inductive datatypes are well-known not to be universal. For example, $\mathtt{Prod}(U,V)$ does not always satisfy surjective pairing. Although every pair of closed $\beta\eta$-normal forms u and v defines a distinct closed $\beta\eta$-normal form $\langle u,v\rangle$ of type $\mathtt{Prod}(U,V)$, by

$$\langle u,v\rangle = \Lambda X.\lambda x^{U\to V\to X}.xuv,$$

for some types U and V not every $\beta\eta$-normal inhabitant of $\mathtt{Prod}(U,V)$ is of this form. Similarly, $\mathtt{List}U$ may have more $\beta\eta$-normal inhabitants than those arising as lists of inhabitants of $\beta\eta$-normal forms of type U, and $\mathtt{Sum}(U,V)$ may have more $\beta\eta$-normal inhabitants than the union of the $\beta\eta$-normal inhabitants of U and V.

The stepping-stone to the proof of Proposition 5.19 is:

PROPOSITION 5.20 *Let U and V be types without negative occurrences of quantifiers, and let A and B be the polymorphic arenas interpreting U and V.*

1. *Winning strategies for the polymorphic arena interpreting $\mathtt{Prod}(U,V)$ are in bijection with pairs consisting of a winning strategy for A and winning strategy for B.*

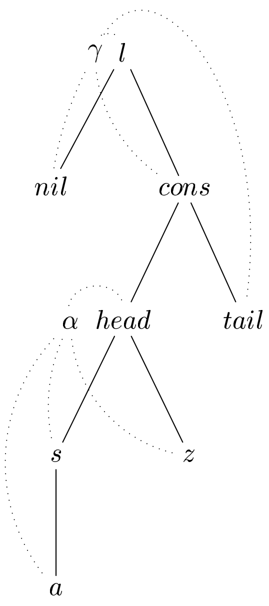
2. *Winning strategies for the polymorphic arena interpreting $\mathtt{List}U$ are in bijection with finite lists of winning strategies for A.*

3. *The set of winning strategies for the polymorphic arena interpreting $\mathtt{Sum}(U,V)$ is in bijection with the disjoint union of the set of winning strategies for A and the set of winning strategies for B.*

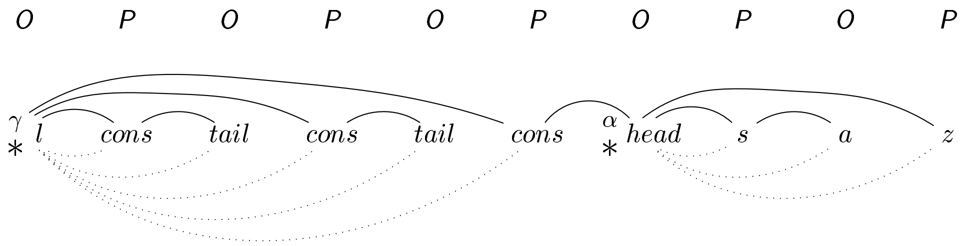
The original proposition then follows immediately from the following corollary of the main full completeness theorem of Chapter 8:

COROLLARY *If T is a type with no negative quantifiers then every winning strategy σ for the polymorphic arena interpreting T defines a distinct η-long β-normal form t_σ whose interpretation is σ. So in particular, $\beta\eta$-normal forms of type T are in bijection with winning strategies for the polymorphic arena interpreting T.*

So all that remains is to prove Proposition 5.20. We prove part 2 of the proposition. Parts 1 and 3 follow by similar reasoning. The key to the proof is the copycat rule. We illustrate the reasoning with the arena *ListNat* from page 80.



Notice that in each of the positions displayed underneath the original picture of the arena, the copycat rule is satisfied. For example, here is the third position of the list 101, with its reference arcs drawn in, so we can verify the copycat rule at a glance:



The copycat rule means that as soon as the game goes into the *Nat* subarena of *ListNat*, *P* is 'stuck' there, in the following sense. Both of the *O* actions *head* and a in *Nat* reference α, so there is no way *P* can 'get back out' by playing *cons* or *nil*: they both reference γ, which would break the copycat rule. Thus every prefix-maximal position p of *ListNat* starts with the *O*-move $\overset{\gamma}{*}\, l$, is followed by $i \geq 0$ pairs *cons tail*, and ends with either *nil* or a prefix-maximal position of *Nat* (with its opening action n renamed to *head*). In other words, p is of the form

$$\overset{\gamma}{*}\, l \quad (\mathit{cons} \quad \mathit{tail})^i \quad q$$

where q is either *nil* or a prefix-maximal position of *Nat* with its opening move n renamed to *head*. From this and the fact that every prefix-maximal position q of *Nat* defines a winning strategy of *Nat* by taking its prefix-closure, we are done.

This idea of *P* being 'stuck' in the *Nat* component generalises to arbitrary types U with no negative quantifiers. Since `List`U has no negative quantifiers (because U has no negative quantifiers), by Lemma 5.12, part (iii), no arenas get stored during the hypergame.

The same argument of getting 'stuck' applies to `Prod`(U, V) and `Sum`(U, V). □

Chapter 6

Interaction

Games models interpret normalisation as the interaction of strategies. Given strategies $\sigma : A \to B$ and $\tau : B \to C$, the composite $\sigma;\tau : A \to C$ is obtained by allowing σ and τ to interact through the 'interface' B, and then deleting the 'chit-chat' moves made in B. This is reminiscent of "parallel composition plus hiding" in concurrency theory.

Up to first-order, interaction in the hypergames model is as in the Nickau games model of PCF [Nic94][1]. Before presenting the definition of interaction in section 6.1, we first work through a motivating example.

The strategies $\mathit{inc} : \mathit{Nat}' \to \mathit{Nat}$ and $\mathit{even} : \mathit{Nat} \to \mathit{Bool}$, increment and a test for evenness of a natural number, were presented in the previous chapter in Figures 5.5 and 5.6. We demonstrate that their composite $\mathit{Nat}' \xrightarrow{\mathit{inc}} \mathit{Nat} \xrightarrow{\mathit{even}} \mathit{Bool}$ by interaction through Nat is exactly $\mathit{odd} : \mathit{Nat}' \to \mathit{Bool}$, the strategy depicted in Figure 5.7. The interaction will involve three 'agents', inc on $\mathit{Nat} \to \mathit{Bool}$, even on $\mathit{Nat}' \to \mathit{Nat}$, and O on $\mathit{Nat}' \to \mathit{Bool}$. The three arenas $\mathit{Nat}' \to \mathit{Bool}$, $\mathit{Nat} \to \mathit{Bool}$, and $\mathit{Nat}' \to \mathit{Nat}$ are shown together in Figure 6.1.

The first-order (λ-calculus) fragment of interaction was set up in Chapter 3, pages 29–42. Now, as then, we keep track of the events in each of the three arenas ($\mathit{Nat}' \to \mathit{Bool}$, $\mathit{Nat} \to \mathit{Bool}$, and $\mathit{Nat}' \to \mathit{Nat}$) in parallel. At this point, it may be useful to flip quickly through the next 30 pages or so for an impressionistic preview of how interaction will work.

The interaction begins with the opening move $\overset{\beta}{\underset{*}{}} b$ by O in $\mathit{Nat}' \to \mathit{Bool}$:

$$\boxed{\mathit{Nat}' \to \mathit{Bool}}$$

$$\overset{\beta}{\underset{*}{}} b$$

This is 'transmitted' through Bool, the common component of $\mathit{Nat}' \to \mathit{Bool}$ and $\mathit{Nat} \to \mathit{Bool}$, to arrive as an opening O-move in $\mathit{Nat} \to \mathit{Bool}$:

[1] Chapter 3 highlighted an important and hitherto overlooked difference between Nickau's interaction and Hyland/Ong interaction [HO94].

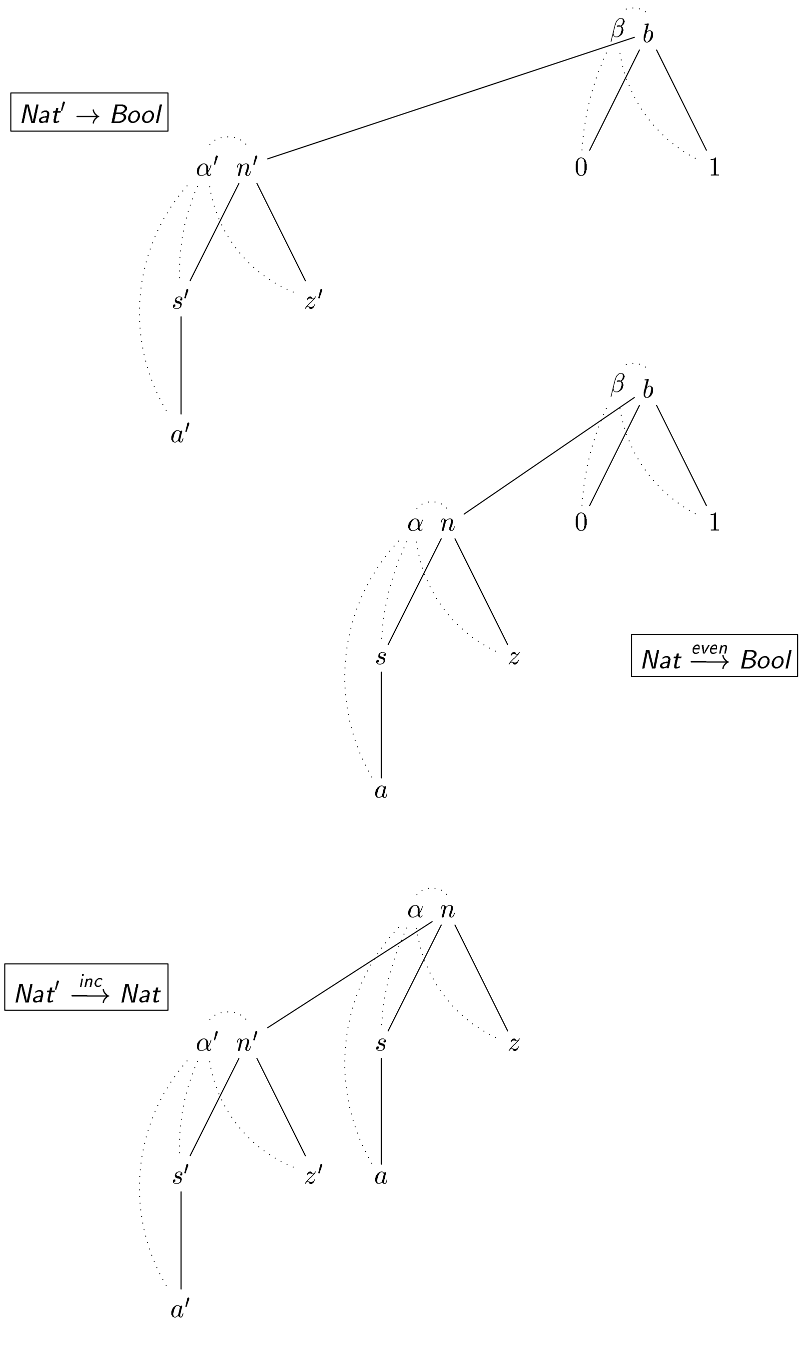


Figure 6.1: The polymorphic arenas involved in the interaction $\mathsf{Nat'} \xrightarrow{\mathsf{inc}} \mathsf{Nat} \xrightarrow{\mathsf{even}} \mathsf{Bool}$.

$\boxed{\mathsf{Nat}' \to \mathsf{Bool}}$

$\overset{\beta}{*}\, b$

$\boxed{\mathsf{Nat} \xrightarrow{\mathsf{even}} \mathsf{Bool}}$

$\overset{\beta}{*}\, b$

In first-order interaction (pages 29–42), actions were transmitted up and down between the three positions. In anticipation of the fact that here, in second-order interaction, actions will be copied around in more complex patterns, we shall keep a record of the origin of copied actions.

CONVENTION 6.1 *Given any action a (resp. hole α) appearing in the interaction, use a subscript 1, 2, or 3 in order to indicate an occurrence of a (resp. α) in the top, middle, or bottom hypersequence. For example, in the diagram above, b_1 denotes the occurrence of b played by O in $\mathsf{Nat}' \to \mathsf{Bool}$, b_2 denotes the occurrence of b just played by O in $\mathsf{Nat} \to \mathsf{Bool}$, β_1 denotes the occurrence of β next to b_1, and β_2 denotes the occurrence of β next to b_2.*

We record the fact that b_2 was transmitted from b_1 in a table underneath the interaction state:

$\boxed{\mathsf{Nat}' \to \mathsf{Bool}}$

$\overset{\beta}{*}\, b$

$\boxed{\mathsf{Nat} \xrightarrow{\mathsf{even}} \mathsf{Bool}}$

$\overset{\beta}{*}\, b$

action	b_2
source	b_1

The next step of the interaction is just as at first-order. The action b_2 that just arrived in $\mathsf{Nat} \to \mathsf{Bool}$ acts as a stimulus for *even*. Consulting the 'chart' of winning positions of *even* (page 89), we see that *even* determines the following response in $\mathsf{Nat} \to \mathsf{Bool}$, a hypermove consisting in three moves:

$$\boxed{Nat' \to Bool}$$

$$^{\beta}_{*}\, b$$

$$\boxed{Nat \xrightarrow{even} Bool}$$

$$^{\beta}_{*}\, b \qquad {}^{\alpha}_{Bool'}\, n$$

$$^{\beta'}_{\langle \beta, c \rangle}\, b'$$

$$c$$

action	b_2
source	b_1

Now the first-order interaction algorithm breaks down. We would like to transmit this hypermove to $Nat' \to Nat$, in a similar manner to the way in which, on page 30 (for example), b_1 was transmitted down to $Bool_1$. But P's three-move hypermove would not be legal on $Nat' \to Nat$, so it does not make sense to transmit it. In particular, it would arrive as an O-hypermove rather than a P-move, so the storage of arenas would be anomalous.

Instead, motivated by a 'uniformity' intuition, that arenas stored by *even* should be 'hidden' from *inc*, and vice versa, we shall transmit only the first action of *even*'s hypermove down to $Nat' \to Nat$:

$$\boxed{Nat' \to Bool}$$

$$^{\beta}_{*}\, b$$

$$\boxed{Nat \xrightarrow{even} Bool}$$

$$^{\beta}_{*}\, b \qquad {}^{\alpha}_{Bool'}\, n$$

$$^{\beta'}_{\langle \beta, c \rangle}\, b'$$

$$c$$

$$\boxed{Nat' \xrightarrow{inc} Nat}$$

$$n$$

action	b_2	n_3
source	b_1	n_2

Since the copy of n in $\mathit{Nat}' \to \mathit{Nat}$ has had its parity reversed to that of being an action of O, we fill its hole with $*$, 'hiding' from *inc* the arena Bool' originally stored in α.

$$\boxed{\mathit{Nat}' \to \mathit{Bool}}$$

$${}^{\beta}_{*}b$$

$$\boxed{\mathit{Nat} \xrightarrow{\mathit{even}} \mathit{Bool}}$$

$${}^{\beta}_{*}b \qquad {}^{\alpha}_{\mathit{Bool}'}n$$

$${}^{\beta'}_{\langle \beta, c\rangle}b'$$

$$c$$

$$\boxed{\mathit{Nat}' \xrightarrow{\mathit{inc}} \mathit{Nat}}$$

$${}^{\alpha}_{*}n$$

action	b_2	n_3
source	b_1	n_2

This completes a hypermove for O in $\mathit{Nat}' \to \mathit{Nat}$, so (for now, at least), nothing need be done with the remaining actions b' and c of *even*'s hypermove.

With the arrival of a stimulus O-move ${}^{\beta}_{*}n$, now $\mathit{Nat}' \to \mathit{Nat}$ is 'active'. Consulting the winning positions of *inc* (page 86), we see that *inc* determines the following response in $\mathit{Nat}' \to \mathit{Nat}$, a two-move hypermove:

$$\boxed{\mathit{Nat}' \to \mathit{Bool}}$$

$$\overset{\beta}{*}\, b$$

$$\boxed{\mathit{Nat} \xrightarrow{\mathit{even}} \mathit{Bool}}$$

$$\overset{\beta}{*}\, b \qquad \overset{\alpha}{\mathit{Bool}'}\, n$$

$$\overset{\beta'}{\langle \beta, c \rangle}\, b'$$

$$c$$

$$\boxed{\mathit{Nat}' \xrightarrow{\mathit{inc}} \mathit{Nat}}$$

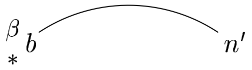
$$\overset{\alpha}{*}\, n \qquad \overset{\alpha'}{\langle \alpha, w \rangle}\, n'$$

$$w$$

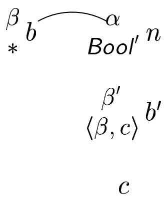
action	b_2	n_3
source	b_1	n_2

Now how should we proceed? Since *inc* just played an action n' in *Nat*$'$, a component shared with *Nat*$'$ $\to$ *Bool*, we can certainly transmit n' to the top:

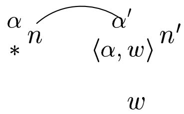
$\boxed{\mathit{Nat}' \to \mathit{Bool}}$



$\boxed{\mathit{Nat} \xrightarrow{\mathit{even}} \mathit{Bool}}$

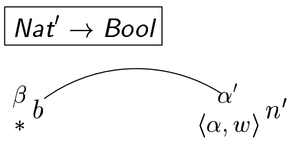


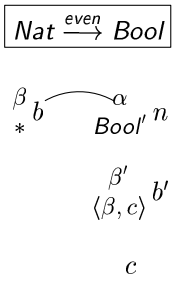
$\boxed{\mathit{Nat}' \xrightarrow{\mathit{inc}} \mathit{Nat}}$

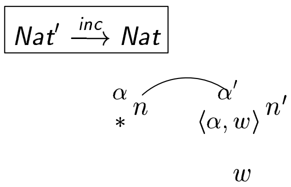


action	b_2	n_3	n'_1
source	b_1	n_2	n'_3

To turn the action n' into a move, we have to store an arena in the hole α' of n' in *Nat'* $\to$ *Bool*. The simplest thing to do would be to directly copy accross *inc*'s storage of the singleton arena $\langle \alpha, w \rangle$ in α':







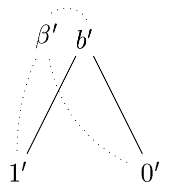
action	b_2	n_3	n'_1
source	b_1	n_2	n'_3

But this is ill-defined, because the stored arena $\langle\alpha, w\rangle$ has a global hole α which is not a preceding *O*-hole in the top position, breaking the scope condition of Definition 5.4 of hypersequence (page 68). The only global hole permitted in a stored arena is β, the *O*-hole at the beginning of the sequence.

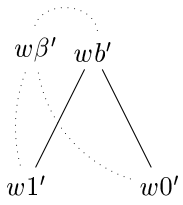
To get around this problem, we use the fact that α 'secretly' contains the arena *Bool'* stored by ***even*** on his first hypermove, although down in $Nat' \to Nat$ this fact remains 'hidden' from *inc*. More precisely, since the global hole α of the arena $\langle\alpha, w\rangle$ stored by *inc* in $Nat' \to Nat$ is the *O*-hole next to n_3, we consult the 'source' copy of α next to the source n_2 of n_3, which stores the arena *Bool'*. Then we 'substitute *Bool'* for α' in $\langle\alpha, w\rangle$, i.e., we expand the arena $\langle\alpha, w\rangle$ along the assignment $\alpha \mapsto$ *Bool'*. (Expansion, our game-theoretic analogue of the substitution of types for type variables, was defined on page 59.)

The singleton arena $\langle\alpha, w\rangle$ is:

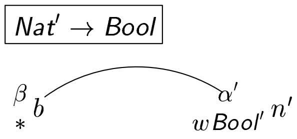
and the arena *Bool'*, to be 'substituted' for α, is:

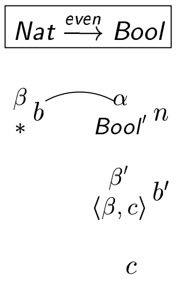


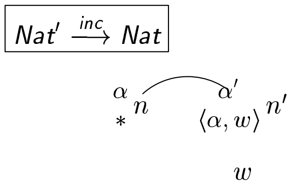
The result of the expansion is a 'w-prefixed' copy of Bool':



which we shall refer to as $w\mathit{Bool}'$. We store $w\mathit{Bool}'$ instead of $\langle\alpha, w\rangle$ in the hole α' of n' in $\mathit{Nat}' \to \mathit{Bool}$:





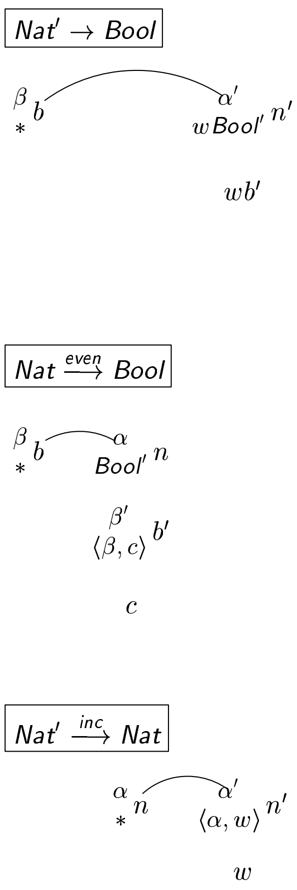


action	b_2	n_3	n'_1
source	b_1	n_2	n'_3

This time the scope condition is satisfied, because $w\mathit{Bool}'$ does not have any global holes at all, let alone a global hole distinct from the preceding O-hole β.

Since n' references α' in $\mathsf{Nat}' \to \mathsf{Bool}$, we have to play an opening move on the contents $w\mathsf{Bool}'$ of α'. From where do we obtain such a move?

The polymorphic arena $w\mathsf{Bool}'$ is an 'amalgamation' of the two arenas $\langle\alpha, w\rangle$ and Bool'. In $\mathsf{Nat}' \to \mathsf{Nat}$, inc has already played an opening action w on $\langle\alpha, w\rangle$, and in $\mathsf{Nat} \to \mathsf{Bool}$, even has already played an opening action b' on Bool'. So we 'amalgamate' the two actions w and b' and play the opening compound action wb' of $w\mathsf{Bool}'$:

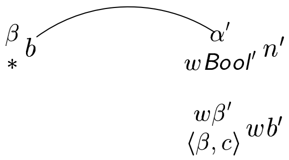


action	b_2	n_3	n'_1	w_1	b'_1
source	b_1	n_2	n'_3	w_3	b'_2

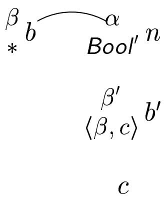
As usual, we record the source of each transmitted action in the table underneath the interaction state, this time indicating that the occurrences of w and b' in the top position came from the occurrences of w and b' in the bottom and middle positions respectively.

The action wb' has a hole $w\beta'$ in $w\mathsf{Bool}'$, so we must find an arena to store in $w\beta'$. From the definition of expansion, the reason that wb' has the hole $w\beta'$ in $w\mathsf{Bool}'$ is because b' has the hole β' in Bool' (because we just 'prefixed w onto Bool'' to get $w\mathsf{Bool}'$). So since even stored the singleton arena $\langle\beta, c\rangle$ in β', we copy this and store $\langle\beta, c\rangle$ in $w\beta'$:

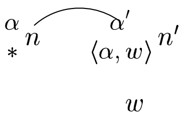
$\boxed{\mathit{Nat}' \to \mathit{Bool}}$



$\boxed{\mathit{Nat} \xrightarrow{\mathit{even}} \mathit{Bool}}$



$\boxed{\mathit{Nat}' \xrightarrow{\mathit{inc}} \mathit{Nat}}$



action	b_2	n_3	n'_1	w_1	b'_1
source	b_1	n_2	n'_3	w_3	b'_2

Still this does not complete a hypermove, because wb' references the hole $w\beta'$, which contains the arena $\langle\beta, c\rangle$ just stored. In the corresponding situation in *Nat* $\to$ *Bool* of b' referencing β' after *even*'s move $\overset{\beta'}{\langle\beta, c\rangle}\, b'$, *even* played the opening move c of $\langle\beta, c\rangle$. So we copy this behaviour up to *Nat*$'$ $\to$ *Bool*:

$\boxed{\mathit{Nat}' \to \mathit{Bool}}$

$$\begin{array}{ll} {}^{\beta}_{*}b & {}^{\alpha'}_{w\mathit{Bool}'}n' \\ & {}^{w\beta'}_{\langle\beta, c\rangle}wb' \\ & c \end{array}$$

$\boxed{\mathit{Nat} \xrightarrow{\mathit{even}} \mathit{Bool}}$

$$\begin{array}{ll} {}^{\beta}_{*}b & {}^{\alpha}_{\mathit{Bool}'}n \\ & {}^{\beta'}_{\langle\beta, c\rangle}b' \\ & c \end{array}$$

$\boxed{\mathit{Nat}' \xrightarrow{\mathit{inc}} \mathit{Nat}}$

$$\begin{array}{ll} {}^{\alpha}_{*}n & {}^{\alpha'}_{\langle\alpha, w\rangle}n' \\ & w \end{array}$$

action	b_2	n_3	n'_1	w_1	b'_1	c_1
source	b_1	n_2	n'_3	w_3	b'_2	c_2

As usual, the source of the copied action is recorded in the table.

Suppose O responds in $\mathit{Nat}' \to \mathit{Bool}$ with the following two-move hypermove:

$\boxed{\mathsf{Nat}' \to \mathsf{Bool}}$

$$\begin{array}{lll}
{}^{\beta}_{*}b & {}^{\alpha'}_{w\mathsf{Bool}'}n' & z' \\
 & {}^{w\beta'}_{\langle\beta,c\rangle}wb' & {}^{\dot{w}\dot{\beta}'}_{*}\dot{w}\dot{b}' \\
 & c &
\end{array}$$

$\boxed{\mathsf{Nat} \xrightarrow{\mathsf{even}} \mathsf{Bool}}$

$$\begin{array}{ll}
{}^{\beta}_{*}b & {}^{\alpha}_{\mathsf{Bool}'}n \\
 & {}^{\beta'}_{\langle\beta,c\rangle}b' \\
 & c
\end{array}$$

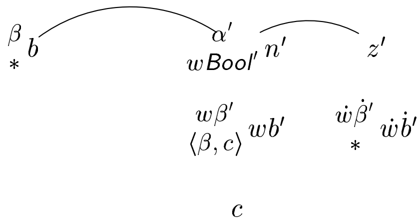
$\boxed{\mathsf{Nat}' \xrightarrow{\mathsf{inc}} \mathsf{Nat}}$

$$\begin{array}{ll}
{}^{\alpha}_{*}n & {}^{\alpha'}_{\langle\alpha,w\rangle}n' \\
 & w
\end{array}$$

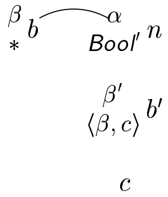
action	b_2	n_3	n'_1	w_1	b'_1	c_1
source	b_1	n_2	n'_3	w_3	b'_2	c_2

(In order to distinguish the new occurrences of actions and holes from those in the previous hypermove, we have placed ‘dots’ over them.) Following the standard first-order interaction protocol, we immediately transmit z' to $\mathsf{Nat}' \to \mathsf{Nat}$ through the shared component Nat':

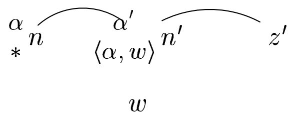
$\boxed{\mathit{Nat}' \to \mathit{Bool}}$



$\boxed{\mathit{Nat} \xrightarrow{\mathit{even}} \mathit{Bool}}$

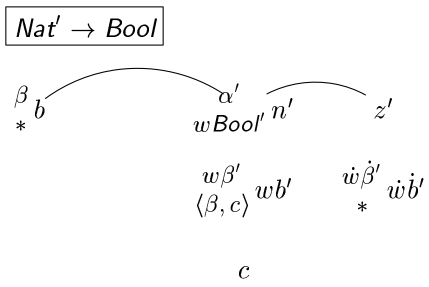


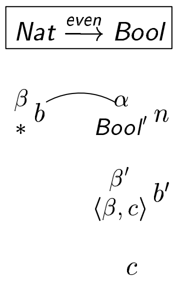
$\boxed{\mathit{Nat}' \xrightarrow{\mathit{inc}} \mathit{Nat}}$

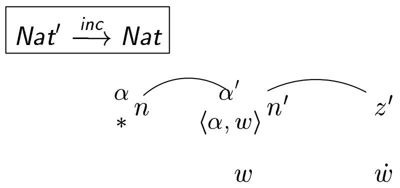


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1

Now z' in the bottom position references the P-hole α', which stores the arena $\langle\alpha, w\rangle$, so we must find an opening move of $\langle\alpha, w\rangle$. The original z' in the top position analogously referenced α' in the top position, and O played $\dot{w}\dot{b}'$ on $w\mathit{Bool}'$. This compound action $\dot{w}\dot{b}'$ includes (a copy $\dot{w}$) of the action w of $\langle\alpha, w\rangle$, so we throw away the $\dot{b}'$ part and use $\dot{w}$ as our opening action in the bottom position:



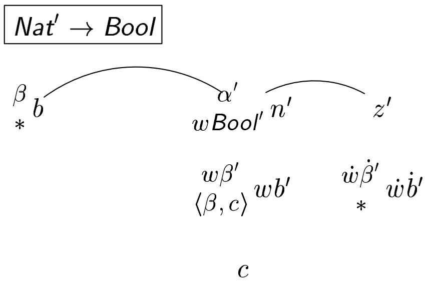


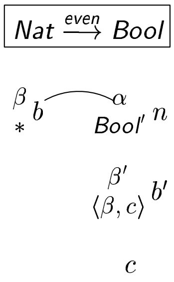


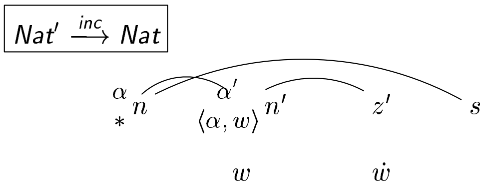
action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$

This completes a hypermove for O, since $\dot{w}b'$ references an O-hole (namely $\dot{w}\dot{\beta}'$ in $w\mathit{Bool}'$).

Referring to the winning positions of *inc* (page 86), we see that *inc* responds in $\mathit{Nat}' \to \mathit{Nat}$ as follows:

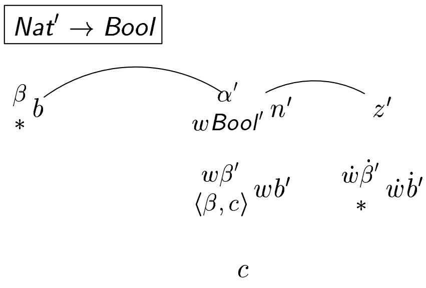


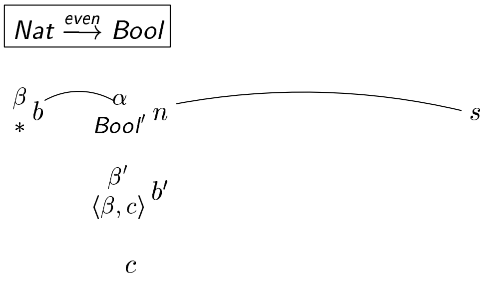


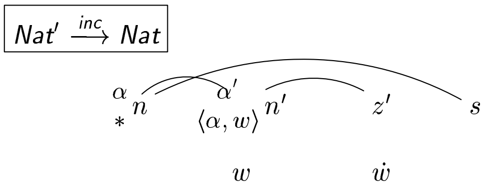


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$

The standard first-order protocol applies, transmitting s accross to $\mathit{Nat} \to \mathit{Bool}$ through the shared component Nat:

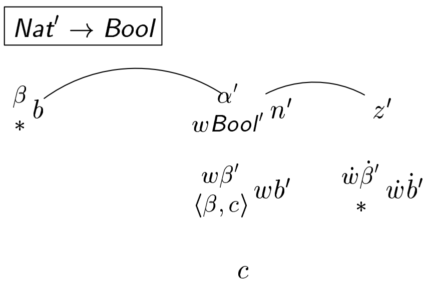


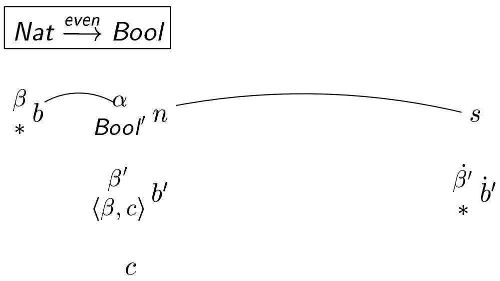


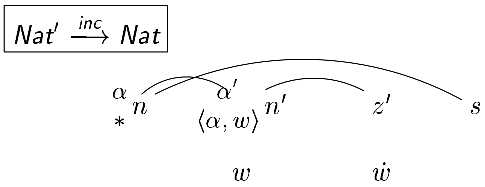


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3

However, unlike its source, the transmitted copy of s does not complete a hypermove for O on $\mathsf{Nat} \to \mathsf{Bool}$, since in $\mathsf{Nat} \to \mathsf{Bool}$, s references the earlier P-hole α, which contains Bool'. We copy accross the occurrence $\dot{b}'$ of b' just played by O in the top position as part of the compound action $\dot{w}\dot{b}'$ in $w\mathsf{Bool}'$, and 'hide' the stored arena $\langle \beta, c\rangle$:





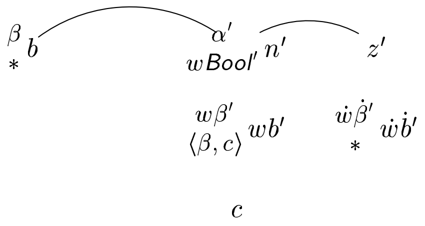


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$

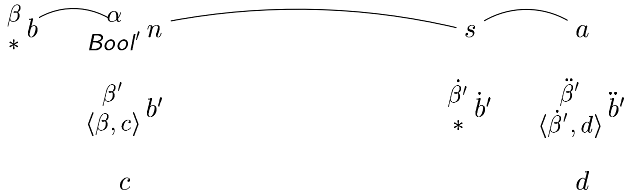
This completes a hypermove for O, since $\dot{b}'$ references an O-hole, namely $\dot{\beta}'$ in $Bool'$.

Now it is *even* to move. Consulting *even*'s table of winning positions, we obtain the following response:

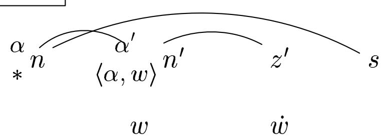
$\boxed{\mathsf{Nat}' \to \mathsf{Bool}}$



$\boxed{\mathsf{Nat} \xrightarrow{\mathsf{even}} \mathsf{Bool}}$

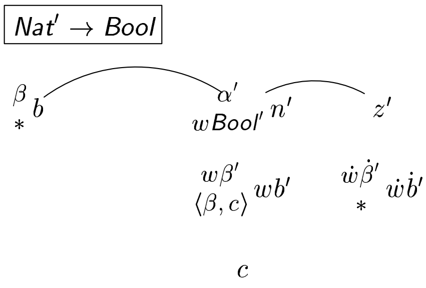


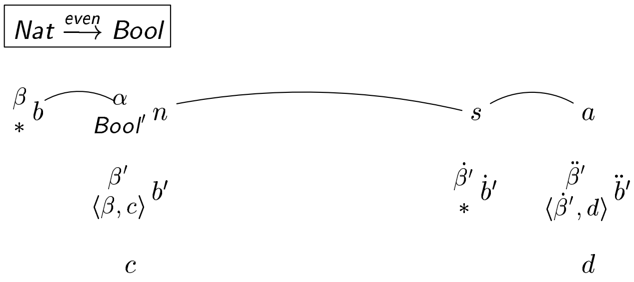
$\boxed{\mathsf{Nat}' \xrightarrow{\mathsf{inc}} \mathsf{Nat}}$

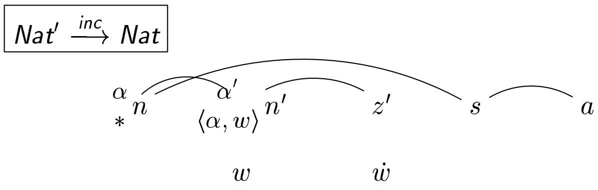


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$

The standard first-order protocol transmits the first action a through the component Nat shared with $\mathsf{Nat}' \to \mathsf{Nat}$:





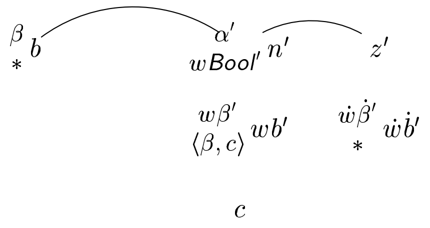


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2

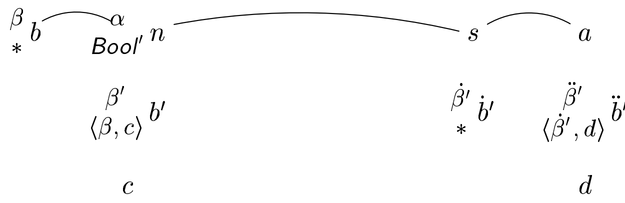
This immediately completes a hypermove for O against *inc*, because in the bottom position a references an O-hole, namely the O-hole α of the first move.

The response of *inc*, as determined by its 'cribsheet', is:

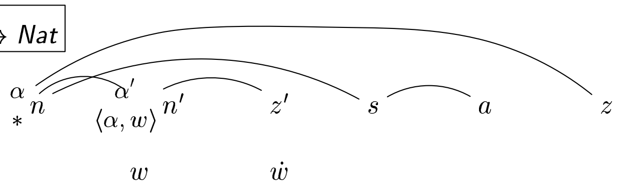
$\boxed{Nat' \to Bool}$



$\boxed{Nat \xrightarrow{even} Bool}$

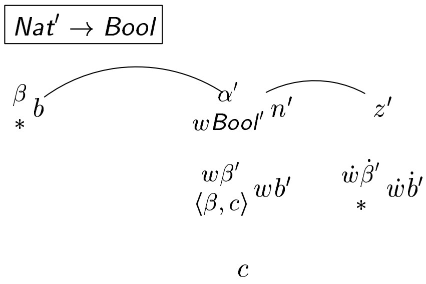


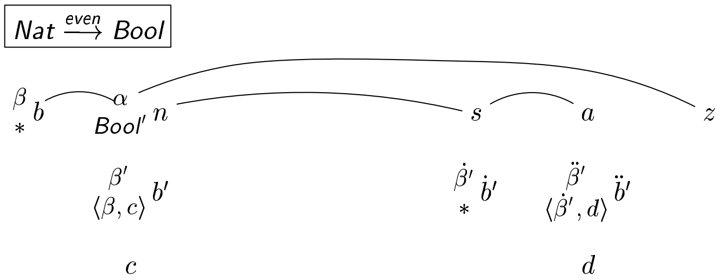
$\boxed{Nat' \xrightarrow{inc} Nat}$

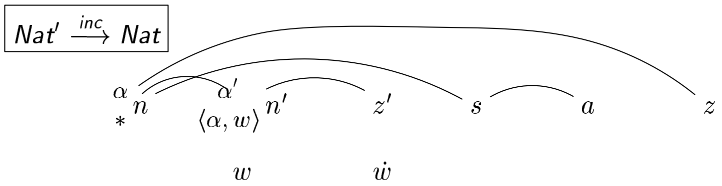


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2

The standard first-order protocol transmits the P-action z through the component *Nat* shared with *Nat* $\to$ *Bool*, to arrive there as an O-action:

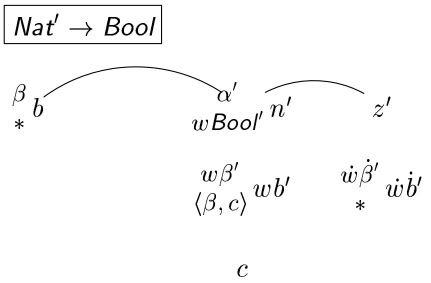


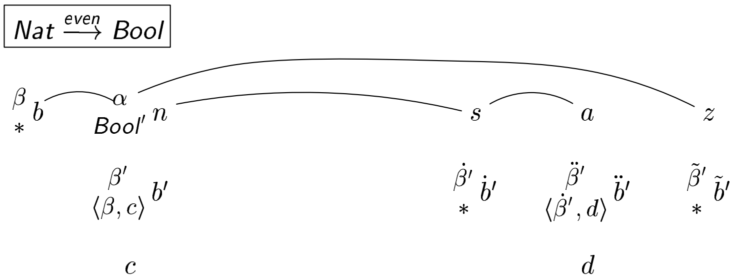


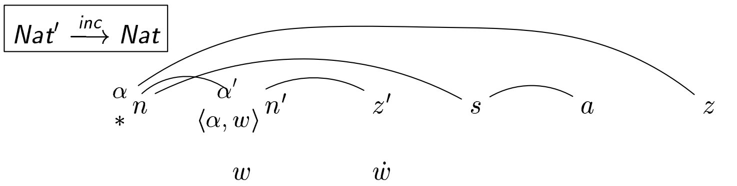


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3

Since z references α, which stores Bool', we have to find an opening move of Bool'. The interaction algorithm plays this move by *copycat*, as follows. Since *inc* played the original z in the bottom position subject to the copycat constraint, we know that in the bottom position a references the same hole as z, namely the hole α of the first move n. So correspondingly, because the moves n, s, a, z were transmitted up and down between the middle and bottom positions through the shared component Nat, we know that the same is true of the middle position: a references the same hole α of z. Hence *even* must just have played an opening action of Bool' underneath a. We take this opening action $\ddot{b}'$ and open a fresh 'copycat thread' inside Bool' by copying it as an O-action $\tilde{b}'$ underneath z, and we 'hide' the stored arena $\langle\dot{\beta}', d\rangle$ (For distinguishability, we have 'tilde'-ed the new occurrences of β' and b'.)



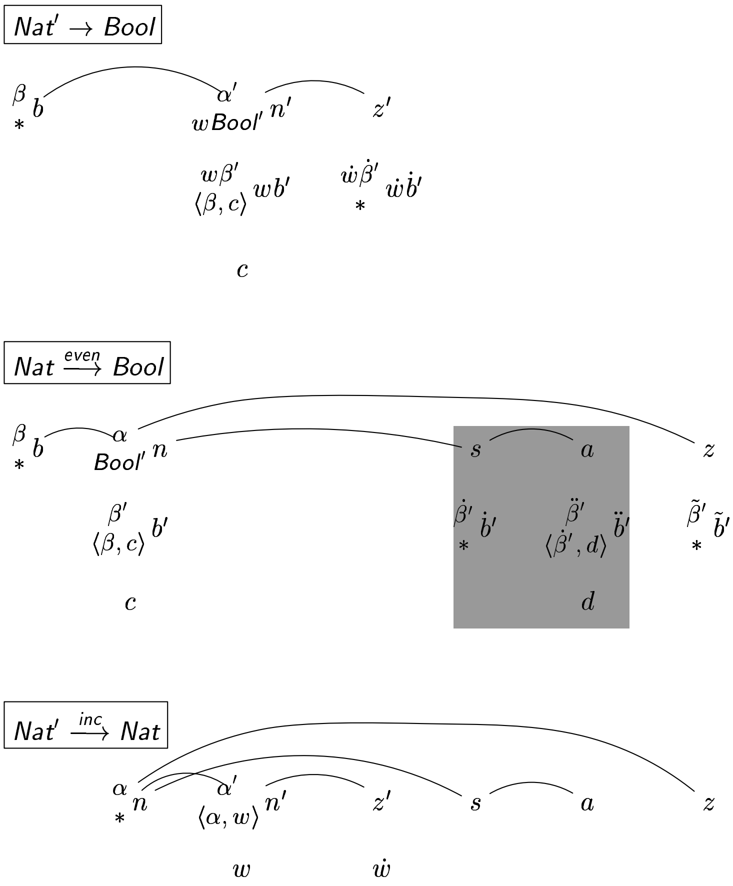




action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2	$\tilde{b}'_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3	$\ddot{b}'_2$

This completes a hypermove for O, since $\tilde{b}'$ references an O-hole, namely its own hole $\tilde{\beta}'$. This is the first instance of an action that arose as a copy of a move from *within the same position*, rather than as a copy transmitted from one of the other positions. Nothing resembling this happens in first-order/λ-calculus interaction. We call such an action a *copycat action*, and think of *even* as having been made to 'play a move against himself'. As usual, we record the source of $\tilde{b}'$ in the table.

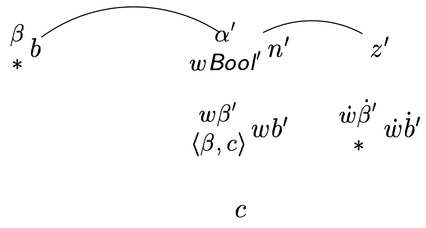
Now it is *even*'s turn to play in $Nat \to Bool$. However, the hypersequence is not a position, because z's justification pointer does not target the previous O-move, violating condition 2 of the definition of position (page 72). In order to obtain a response from *even*, we precede exactly as we did in the first-order case (page 40), and 'force' the hypersequence to be a position by 'hiding' all the hypermoves between the source and target of the latest pointer:



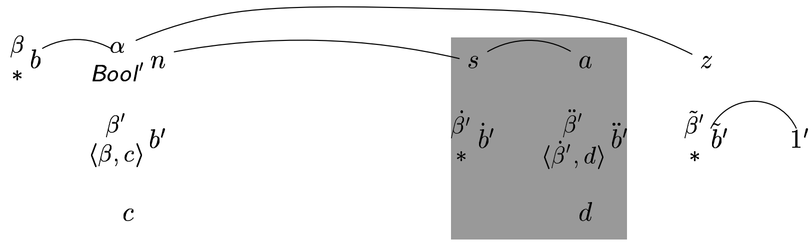
action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2	$\tilde{b}'_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3	$\ddot{b}'_2$

Looking up this ‘view’ in *even*’s set of winning positions, we obtain the following response:

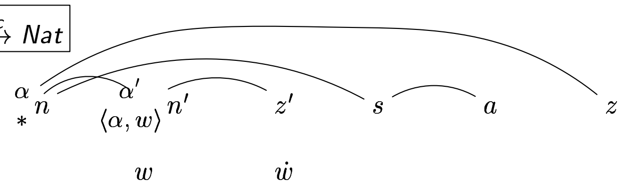
$\boxed{Nat' \to Bool}$



$\boxed{Nat \xrightarrow{even} Bool}$

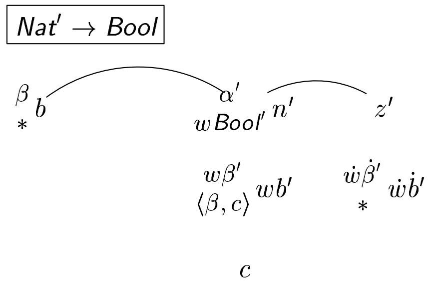


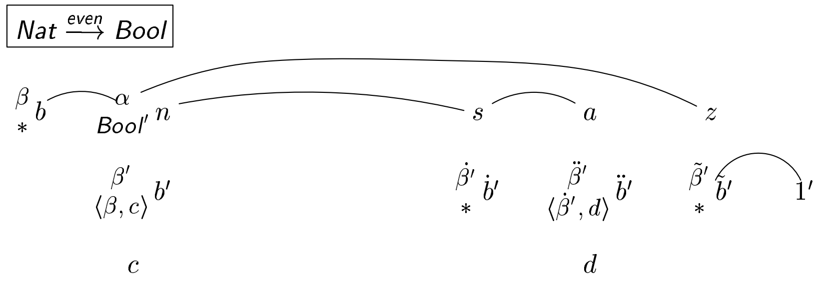
$\boxed{Nat' \xrightarrow{inc} Nat}$

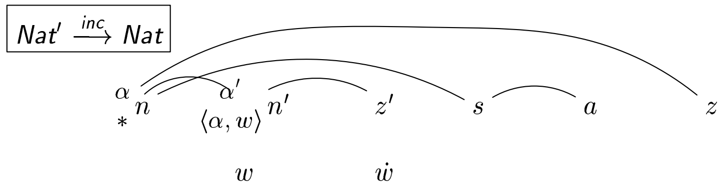


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2	$\tilde{b}'_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3	$\ddot{b}'_2$

which, after restoring the hidden moves, gives

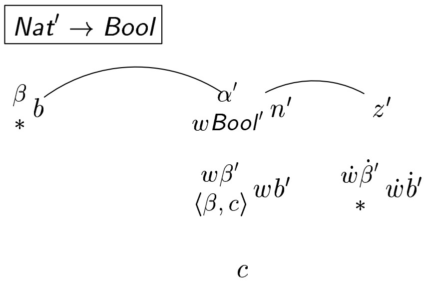


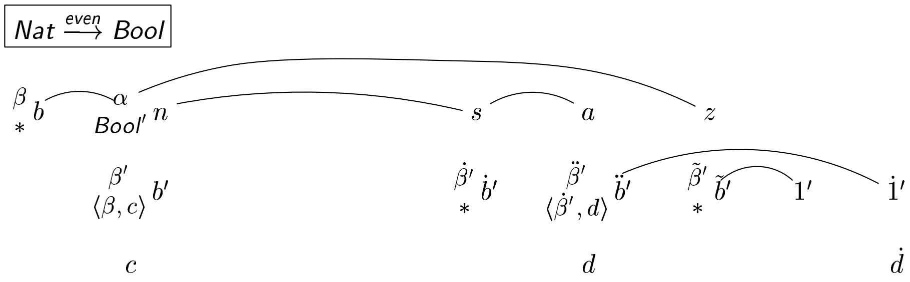


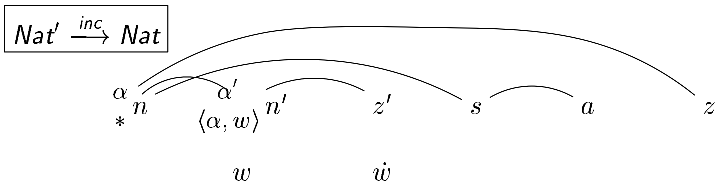


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2	$\tilde{b}'_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3	$\ddot{b}'_2$

The action $1'$ in *Bool′* is justified by $\tilde{b}'$, which was a copycat action. Accordingly, we 'continue to copycat' by playing a copy $\dot{1}'$ of $1'$ justified by the source $\ddot{b}'$ of $\tilde{b}'$, and at the same time we 'copycat' accross the action d:

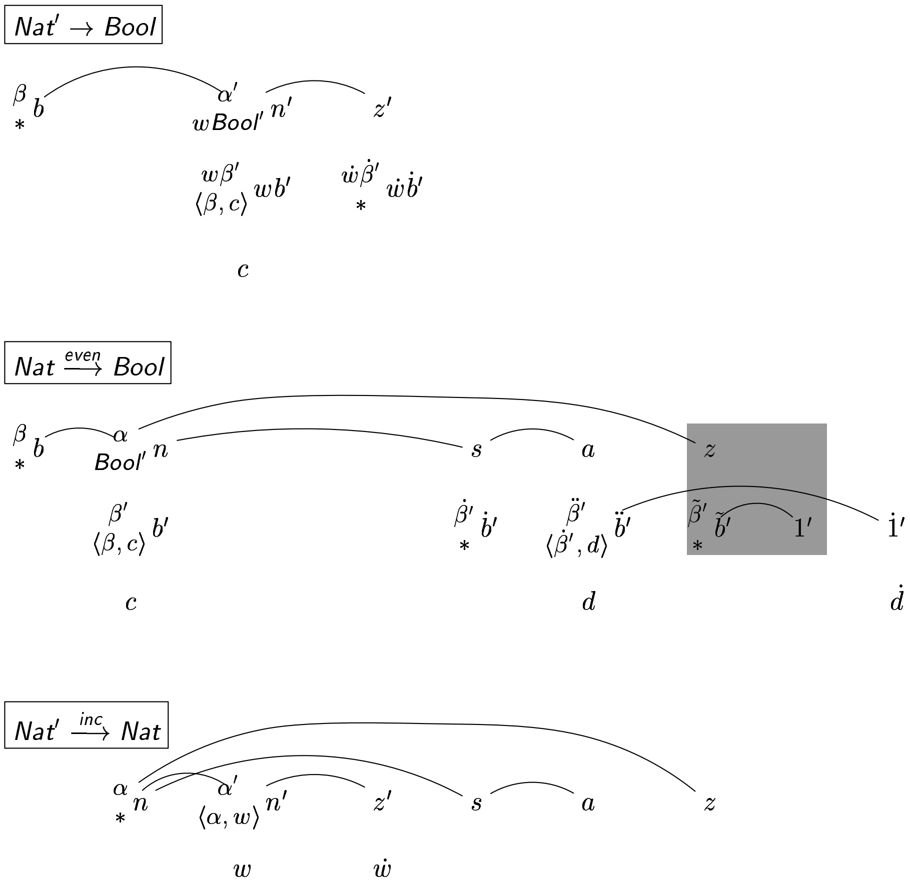






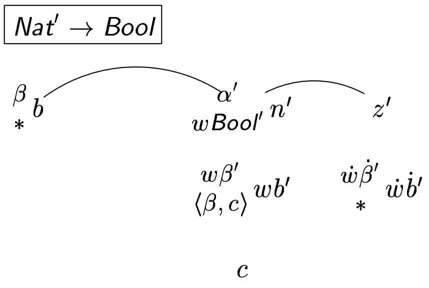
action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2	$\tilde{b}'_2$	$\dot{1}'_2$	$\dot{d}_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3	$\ddot{b}'_2$	$1'_2$	d_2

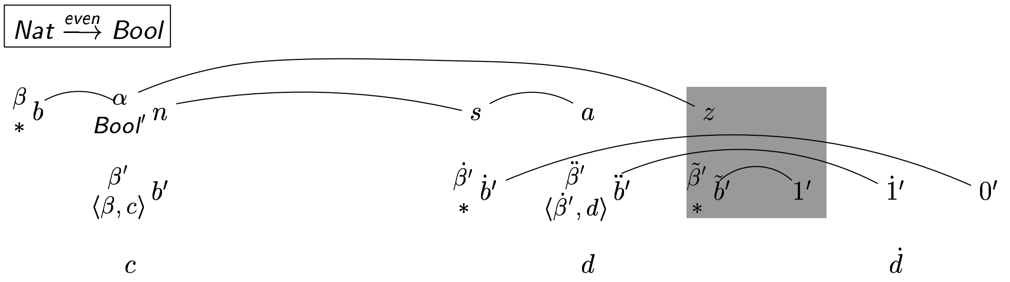
Again, we turn the hypersequence into a position by hiding the moves that lie between the source and target of the latest justification pointer:

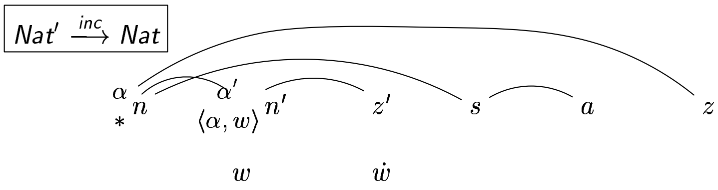


action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2	$\tilde{b}'_2$	$\dot{1}'_2$	$\dot{d}_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3	$\ddot{b}'_2$	$1'_2$	d_2

Looking up the view in *even*'s set of winning positions, we obtain the following response:

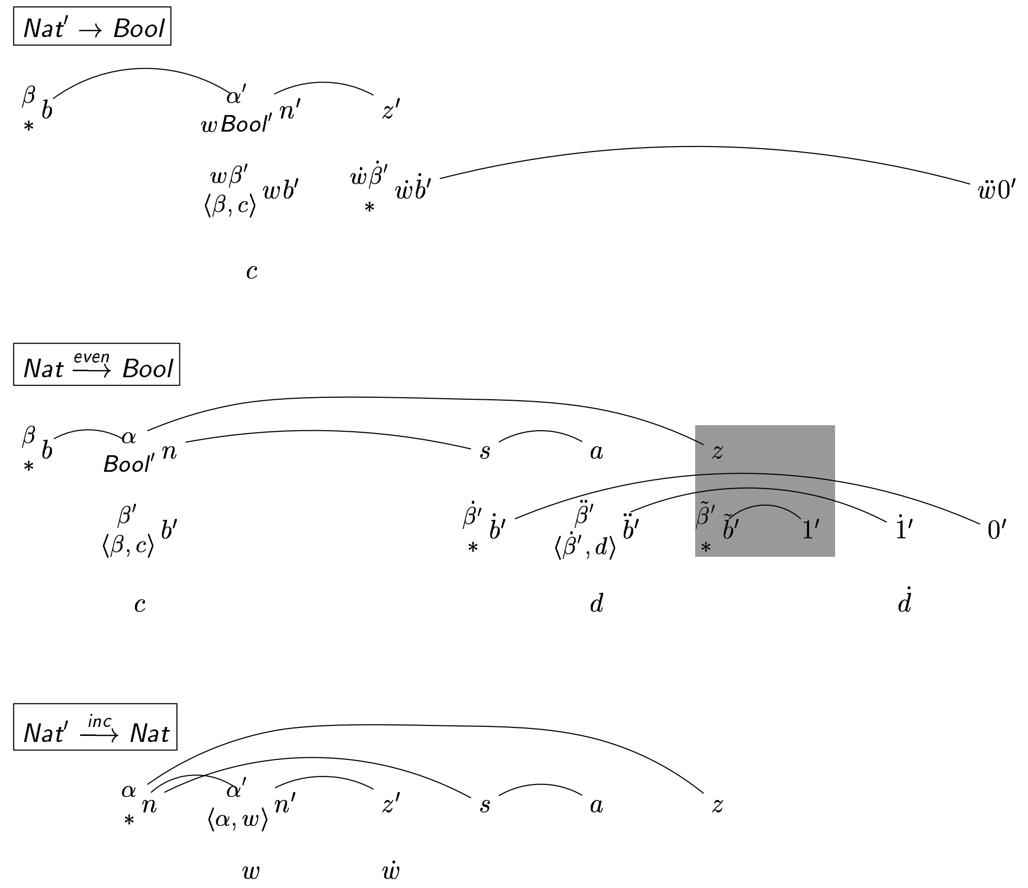






action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2	$\tilde{b}'_2$	$\dot{1}'_2$	$\dot{d}_2$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3	$\ddot{b}'_2$	$1'_2$	d_2

The action $0'$ is justified by $\dot{b}'_2$ whose source is $\dot{b}'_1$, so we transmit $0'$ to the top position, justified by (the compound move $\dot{w}\dot{b}'$ containing) $\dot{b}'_1$:



action	b_2	n_3	n'_1	w_1	b'_1	c_1	z'_3	$\dot{w}_3$	s_2	$\dot{b}'_2$	a_3	z_2	$\tilde{b}'_2$	$\dot{1}'_2$	$\dot{d}_2$	$0'_1$
source	b_1	n_2	n'_3	w_3	b'_2	c_2	z'_1	$\dot{w}_1$	s_3	$\dot{b}'_1$	a_2	z_3	$\ddot{b}'_2$	$1'_2$	d_2	$0'_2$

We record the source of $0'$ in the table[2]. Now the interaction is over, because $\ddot{w}0'$ is a leaf. So the combined efforts of *even* and *inc* have achieved victory in $\mathsf{Nat}' \to \mathsf{Bool}$. Notice that this winning position is the top winning position of the winning strategy $\mathit{odd} : \mathsf{Nat}' \to \mathsf{Bool}$ displayed in Figure 5.7, as one would hope.

6.1 The interaction algorithm

In this section we formalise the algorithm that governed the interaction of the introductory example. The algorithm is presented in subsection 6.1.9. First, in subsections 6.1.1–6.1.8, we introduce some auxiliary definitions: views and plays, interaction states, live atomic actions and live atomic enabling, holes of live atomic actions, match, source, lookup and full expansion, and well-formedness of interaction states.

[2] For technical reasons, it turns out that the 'prefix' $\dot{w}$ is simply 'copied forward' as $\ddot{w}$, and this fact is not recorded as part of the source table. This is because $\dot{w}$, although an atomic action of a compound action, is what we call a *dead* atomic action (defined in section 4.4.2).

6.1.1 Views and plays

The definition below is motivated by the situations on pages 117 and 121, in which anomalous O-pointers forced us to restrict P's 'view' of the hypersequence. Given hypermoves μ, ν of a hypersequence s, write $\mu \curvearrowleft \nu$ if the justifier of the first move of ν is in μ, and say that μ ***justifies*** ν. We recall (McCusker's [McC96a] variant of) Hyland/Ong and Nickau's notion of the view of a justified sequence, adapted to our hypersequence setting.

DEFINITION 6.2 *The **view** of a hypersequence $h = (s, \curvearrowleft)$ is defined by induction on the length of s as follows:*

$$\begin{array}{rcll} \ulcorner \epsilon \urcorner & = & \epsilon & \\ \ulcorner s_1 \mu s_2 \nu \urcorner & = & \ulcorner s_1 \mu \urcorner \nu & \text{if } \mathsf{pl}(\nu) = O \text{ and } \mu \curvearrowleft \nu \\ \ulcorner s_1 \nu \urcorner & = & \nu & \text{if } \mathsf{pl}(\nu) = O \text{ and } \nu \text{ is unjustified} \\ \ulcorner s_1 \nu \urcorner & = & \ulcorner s_1 \urcorner \nu & \text{if } \mathsf{pl}(\nu) = P \end{array}$$

The graph of $\curvearrowleft$ on the view is defined as the restriction of $\curvearrowleft$ from the original hypersequence s.

The justfication graph of the view $\ulcorner h \urcorner$ may turn out to be a partial function rather than a function, because some of the targets of the pointers may become deleted. So $\ulcorner h \urcorner$ need not be a hypersequence. Similarly, some of the O-holes referenced by stored arenas may become deleted. The following definition yields a class of hypersequence whose view will be free of such 'dangling pointers'.

DEFINITION 6.3 *A **play** is a well-formed hypersequence $h = (s, \curvearrowleft)$ of a polymorphic arena A satisfying the following conditions.*

1. *If the first move m of a hypermove μ is an opening move, i.e., if m is unjustified, then m is an O-move located in A.*
2. ***Justifier visibility**. The justifier of every P-move is in the view: if $s = s_1 \mu s_2 \nu s_3$, $\mathsf{pl}(\nu) = P$ and $\mu \curvearrowleft \nu$, then $\mu \in \ulcorner s_1 \mu s_2 \urcorner$.*
3. ***Reference visibility**. The global references of every polymorphic arena stored by P are amongst the O-holes of the view: if $s = s_1 \mu s_2$, $\mathsf{pl}(\mu) = P$, $m \in \mu$, and $\alpha \in \mathsf{Hol}(m)$, then*
$$\mathsf{Hol}^{\uparrow}_{\mathsf{store}_s(\alpha)} \quad = \quad H \;+\; \{\, \beta \in \mathsf{OHol}(\ulcorner s \urcorner) \;:\; \beta < m \,\}.$$
(This is a refinement of the scope condition of Definition 5.5 of well-formed hypersequence.)

We write $\mathsf{Plays}(A)$ for the set of plays of A.

The second condition is adapted from Hyland/Ong. The next two lemmas, though trivial, will turn out to be very useful later.

LEMMA 6.4 *Let h be a hypersequence. If h is a play, then the view $\ulcorner h \urcorner$ of h is a position.*

Proof A simple induction on the length of h. The view $\ulcorner h \urcorner$ satisfies the copycat condition because (unwinding the definition of view) $\ulcorner s \mu \nu \urcorner = s' \mu \nu$ whenever $\mathsf{pl}(\nu) = P$. $\square$

LEMMA 6.5 *Suppose $p \in \mathsf{Plays}(A)$ and μ is a P-hypermove such that $\ulcorner p \urcorner \mu \in \mathsf{Pos}(A)$, with justification pointer targeting some O-move m in $\ulcorner p \urcorner$. Then $p\mu \in \mathsf{Pos}(A)$.*

Proof The target m of the justification pointer is in $\ulcorner p \urcorner$, as are the references, hence $p\mu$ satisfies the visibility conditions. $\square$

6.1.2 Interaction states

We formalise the interaction states of the illustrative example of interaction at the beginning of the chapter. First we formalise the notion of 'hypersequence with gaps' that was used in the example.

DEFINITION 6.6 *A **timestamped** hypersequence $(s, \curvearrowleft, T)$ is a hypersequence $(s, \curvearrowleft)$ together with an assignment $T : s \to \mathbb{N}^+$, such that for all hypermoves $\mu, \nu \in s$, if $\mu < \nu$ then $T(\mu) < T(\nu)$.*

The three hypersequences in the motivating interaction example earlier are each timestamped: the timestamp of a hypermove is the index of its column, counting from left to right. Given two timestamped hypersequences $(s, \curvearrowleft_s, T_s)$ and $(t, \curvearrowleft_t, T_t)$ and hypermoves $\mu, \nu \in s + t$, write $\mu \prec \nu$ for $T(\mu) < T(\nu)$ and $\mu \preccurlyeq \nu$ for $T(\mu) \leq T(\nu)$, where here each occurrence of T is T_s or T_t as appropriate. Given moves m of μ and n of ν, we write $m \prec n$ if and only if $\mu \prec \nu$ and write $m \preccurlyeq n$ if and only if $\mu \preccurlyeq \nu$. So $m \prec n$ means "m is in a column strictly preceding the column containing n". The ***length*** $|(s, \curvearrowleft, T)|$ of a timestamped sequence $(s, \curvearrowleft, T)$ is 0 if s is empty, otherwise the timestamp of the last hypermove of s. Note that the length of $(s, \curvearrowleft, T)$ may be strictly greater than the length $|s|$ of s, because of 'gaps'.

Given an arena D with global holes $H + H_1$ and a function $\mathit{rename} : H_1 \to H_2$ (a 'renaming' of some of the global holes), define the arena $\underline{\mathit{rename}}(D)$ with global holes $H + H_2$ in the obvious way, i.e.,

$$\mathit{ref}_{\underline{\mathit{rename}}(D)}(a) = \begin{cases} \mathit{rename}(\mathit{ref}_D(a)) & \text{if } \mathit{ref}_D(a) \in H_1 \\ \mathit{ref}_D(a) & \text{otherwise} \end{cases}$$

DEFINITION 6.7 *Given polymorphic arenas A, B, C over a set of global holes H, an **interaction state** $\rho = (p, q, r, \mathit{next})$ of A, B and C consists of:*

1. *A timestamped position p of $A \Rightarrow C$;*
2. *A timestamped play q of $B \Rightarrow C$;*
3. *A timestamped play r of $A \Rightarrow B$;*
4. *A 'control' $\mathit{next} \in \{1, 2, 3\}$ such that:*
   - (a) *If $\mathit{next} = 1$ then in p it is O to play next.*
   - (b) *If $\mathit{next} = 2$ then in q it is P to play next.*
   - (c) *If $\mathit{next} = 3$ then in r it is P to play next.*

Every polymorphic arena D stored by P in p is required to be either:

1. *A polymorphic arena stored by P in q or r, expanded repeatedly by polymorphic arenas stored by P in q or r. Formally,*

$$D = \underline{\mathit{rename}}(\phi_k^*(\ldots(\phi_1^*(\psi(\alpha)))\ldots)),$$

for α a hole of a P-move $m = (E, \psi, a)$ of q or r, assignments $\phi_1, \ldots, \phi_k$ with

$$\mathsf{im}(\phi_i) \subseteq \mathsf{im}(\mathit{store}_q) \cup \mathsf{im}(\mathit{store}_r),$$

and

$$\mathit{rename} : \mathit{OHol}(q) + \mathit{OHol}(r) \to \mathit{OHol}(p).$$

(Note that since D is stored in p, it must have global holes $H + \mathit{OHol}(p)$, and by definition of the expansions, $\phi_k^*(\ldots(\phi_1^*(\psi(\alpha)))\ldots)$ has global holes $H + \mathit{OHol}(q) + \mathit{OHol}(r)$, hence the renaming function.)

2. *A singleton arena $\langle \alpha, \bullet \rangle$ with the distinguished action $\bullet$ (called the* ***dummy*** *action) referencing a previous O-hole α of p. (In our motivating inc/even example we did not come accross any dummy actions. For a preview of dummy actions, look to pages 150 or 153.) We assume that $\bullet$ is not an action of A, B or C, so in particular $\bullet$ does not occur as a depth 1 action of p, q, or r. Furthermore, by convention we shall assume that $\bullet$ is not an action of any stored arena other than one of the $\langle \alpha, \bullet \rangle$.*

We write $\mathit{Int}_0(A, B, C)$ for the set of interaction states of A, B, and C. The ***length*** *$|\rho|$ of ρ is $\max\{|p|, |q|, |r|\}$, i.e., zero if each of p, q, and r is empty, otherwise the maximum of $T(\mu)$ as μ ranges over all the hypermoves of ρ.*

Each picture in the interaction of *even* and *inc* earlier is an interaction state, with $A = \mathit{Bool}$, $B = \mathit{Nat}$ and $C = \mathit{Nat}'$, and with p, q, and r as the top, middle and bottom hypersequences respectively. The function of *next* is to indicate at every step of the interaction which of the hypersequences p, q, or r has control, i.e., which of the three hypersequences is about to be extended by a hypermove.

Recall our convention that for any action a (resp. hole α) appearing in the interaction, we use a subscript 1, 2, or 3 in order to indicate an occurrence of a (resp. α) in the top, middle, or bottom hypersequence. The polymorphic arena $w\mathit{Bool}'$ stored by P in p is the expansion of the arena $\langle \alpha_3, w \rangle$ stored in r, along the assignment $\alpha_3 \mapsto \mathit{Bool}'$ ($k = 1$ in condition 1 above), followed by the renaming of β_2 to the corresponding occurrence β_1 of β in the top hypersequence. The polymorphic arena $\langle \beta_1, c \rangle$ stored by P in p is the 'zero-step' expansion of the arena $\langle \beta_2, c \rangle$ stored in the second hypermove of q ($k = 0$ in condition 1 above), again followed by the renaming of β_2 to β_1.

6.1.3 Live atomic actions and live atomic justification

By condition 1 of the definition of interaction state above, given an interaction state $\rho = (p, q, r, \mathit{next})$, the location of every action d of p will be (the renaming of) a repeatedly expanded arena $\phi_k^*(\ldots(\phi_1^*(\psi(\alpha)))\ldots)$. Hence, by definition of repeated expansion (Definition 4.2, and section 4.4.3), every action $d \in \mathit{act}(p)$ is a compound action

$$d = b_0 \eta_1[b_1] \eta_2[b_2] \cdots \eta_k[b_k]$$

where

$$\begin{aligned}
&b_0 \in \mathit{Act}_{\psi(\alpha)} \\
&\eta_i \in \mathit{trails}_{\phi_i}(b_0 \eta_1[b_1] \ldots \eta_{i-1}[b_{i-1}]) \\
&[b_i] = \begin{cases} \epsilon & \text{if } \mathit{ref}_{A_{i-1}}(b_0 \eta_1[b_1] \ldots \eta_{i-1}[b_{i-1}]) \notin \mathit{dom}(\phi_i) \\ b_i \in \mathit{Act}_{B_i} & \text{otherwise, where } B_i := \phi_i(b_0 \eta_1[b_1] \ldots \eta_{i-1}[b_{i-1}]) \end{cases}
\end{aligned}$$

(The above is simply cut-and-pasted from section 4.4.3, with $\psi(\alpha)$ in place of A_0.) In particular every atomic action a of d (i.e., b_0, non-empty $[b_i]$ or an element of one of the η_i) is located in an arena stored by P in q or r, because (by condition 1 of the definition of interaction state) $B_i \in \mathsf{im}(\phi_i)$ is an arena stored by P in q or r. Write $\tilde{\mathit{loc}}(a)$ for this location. In the degenerate case that $d = \bullet$, the distinguished action of a singleton arena $\langle \alpha, \bullet \rangle$, define $\tilde{\mathit{loc}}(\bullet) = \mathit{Loc}(\bullet) = \langle \alpha, \bullet \rangle$.

Write $\tilde{\mathit{act}}(p)$ for the sequence of atomic actions underlying $\mathit{act}(p)$. For example, with p the top hypersequence of the illustrative interaction example on page 123, $\mathit{act}(p) = bn'(wc)z'(wc)(w0')$ and $\tilde{\mathit{act}}(p) = bn'wcz'wcw0'$. Write $O\tilde{\mathit{act}}(p)/P\tilde{\mathit{act}}(p)$ for the subsequences of atomic O/P-actions respectively. By convention, take all actions of q and r to be atomic, and define the ***atomic actions*** of ρ to be $\tilde{\mathit{act}}(\rho) = \tilde{\mathit{act}}(p) + \mathit{act}(q) + \mathit{act}(r)$.

Define $\overline{\mathit{act}}(p)$ to be the subsequence of $\tilde{\mathit{act}}(p)$ consisting of the live atomic actions. By convention take every action of q and r to be live atomic, and define the ***live atomic actions*** of ρ to be $\overline{\mathit{act}}(\rho) = \overline{\mathit{act}}(p) + \mathit{act}(q) + \mathit{act}(r)$. In order to maintain the consistency of the 'overline' notation, given a live atomic action $a \in \overline{\mathit{act}}(\rho)$, we write $\overline{\mathit{loc}}(a)$ for its location $\tilde{\mathit{loc}}(a)$. Write $\overline{\mathit{act}}^{op}(p)$ and $\overline{\mathit{act}}^{cont}(p)$ for the subsequences of $\overline{\mathit{act}}(p)$ consisting of the live atomic opening- and continuation actions respectively, and write $O\overline{\mathit{act}}(p)$ and $P\overline{\mathit{act}}(p)$ for the subsequences of $\overline{\mathit{act}}(p)$ consisting of the O- and P live atomic actions respectively.

Define ***live atomic justification***

$$\overline{\curvearrowleft} : \overline{\mathit{act}}^{cont}(p) \to \overline{\mathit{act}}(p)$$

to correspond to $\curvearrowleft : \mathit{act}^{cont}(p) \to \mathit{act}(p)$ as follows: set $a \overline{\curvearrowleft} b$ whenever a is a live atomic action of d, b is a live atomic action of e, $d \curvearrowleft e$, and $a \vdash b$ is the live atomic enabling behind $d \vdash e$. (Live atomic labelling was defined in section 4.4.2, page 61.) For example, on page 123, in the top hypersequence, $\dot{b}' \overline{\curvearrowleft} 0'$ is the live atomic enabling behind $\dot{w}\dot{b}' \curvearrowleft \ddot{w}0'$. (So w just 'comes along for the ride' as a 'prefix-tag'.) Note that (as sets of pairs, i.e., graphs of functions) $\overline{\curvearrowleft}$ and $\curvearrowleft$ are in bijection, since every enabling of compound actions is due to exactly one live atomic enabling. Define $\overline{\curvearrowleft} : \overline{\mathit{act}}^{cont}(\rho) \to \overline{\mathit{act}}(\rho)$ as the extension of $\overline{\curvearrowleft} : \overline{\mathit{act}}^{cont}(p) \to \overline{\mathit{act}}(p)$ with $\curvearrowleft : \mathit{act}^{cont}(q) \to \mathit{act}(q)$ and $\curvearrowleft : \mathit{act}^{cont}(r) \to \mathit{act}(r)$.

6.1.4 Holes of live atomic actions

For any atomic action $a \in d = b_0\eta_1[b_1]\eta_2[b_2]\cdots\eta_k[b_k]$ of p as in the previous subsection (i.e., with b_0, η_i, and $[b_i]$ page 126) define $\zeta^a_<$ as the subsequence of d strictly preceding a and $\zeta^a_>$ as the subsequence of d strictly following a, i.e., $d = \zeta^a_< a \zeta^a_>$. Then, unwinding the definition of the holes of a compound action d (Definition 4.2, page 59),

$$\mathit{Hol}(d) \quad = \quad \bigcup_{\text{live } a \in d} \{ \ \zeta^a_< \alpha \zeta^a_> \ : \ \alpha \in \mathit{Hol}_{B_i}(a) \ \}$$

where the range "live $a \in d$" means "live atomic actions a of d". For each live atomic action a of d, define

$$\mathit{Hol}(a) \quad = \quad \{ \ \zeta^a_< \alpha \zeta^a_> \ : \ \alpha \in \mathit{Hol}_{B_i}(a) \ \}$$

i.e.,

$$\mathit{Hol}(a) \quad = \quad \{ \ \zeta^a_< \alpha \zeta^a_> \ : \ \alpha \in \mathit{Hol}_{\overline{\mathit{loc}}(a)}(a) \ \}$$

Thus

$$\mathit{Hol}(d) \quad = \quad \bigcup_{\text{live } a \in d} \mathit{Hol}(a),$$

and since the union is disjoint,

$$\mathsf{Hol}(d) \quad = \sum_{\text{live } a \in d} \mathsf{Hol}(a).$$

By definition

$$\mathsf{Hol}(p) \quad = \sum_{d \in \mathsf{act}(p)} \mathsf{Hol}(d),$$

so

$$\mathsf{Hol}(p) \quad = \sum_{a \in \overline{\mathsf{act}}(p)} \mathsf{Hol}(a).$$

For example, in the top hypersequence (position) p of the example at the beginning of the chapter (page 123), $\mathsf{Hol}(b) = \{\beta\}$, $\mathsf{Hol}(n') = \{\alpha'\}$, $\mathsf{Hol}(w) = \emptyset$, $\mathsf{Hol}(b') = \{w\beta'\}$, $\mathsf{Hol}(z') = \emptyset$, $\mathsf{Hol}(\dot{w}) = \emptyset$, $\mathsf{Hol}(\dot{b}') = \{\dot{w}\dot{\beta}'\}$, $\mathsf{Hol}(\ddot{w}) = \emptyset$, $\mathsf{Hol}(0') = \emptyset$, and $\mathsf{Hol}(p) = \{\beta, \alpha', w\beta', \dot{w}\dot{\beta}'\}$.

6.1.5 Match

In the motivating interaction presented at the beginning of the chapter, the 'sent' and 'received' copies of an action were 'the same'. We formalise 'the same' in the definition below. The depth 1 case appears complicated at first sight, but it is just the obvious bookkeeping associated with the fact that in a function space $D \Rightarrow E$ there are multiple copies of D. Write an action $(c, b) \in \mathsf{Act}_C^{\mathrm{op}} \times \mathsf{Act}_B \hookrightarrow \mathsf{Act}_{B \Rightarrow C}$ as b_c, with the idea that (c, b) is a copy of b 'tagged' with c. Given an atomic action a of a compound action d of ρ, define $\mathsf{depth}(a) = \mathsf{depth}(d)$.

Definition 6.8 *Live atomic actions $e \neq \bullet$ and $e' \neq \bullet$ of an interaction state $\rho = (p, q, r, \mathsf{next})$ of $\mathsf{Int}_0(A, B, C)$* **match**, *denoted $e \sim e'$, if either of the following conditions hold:*

1. *$\mathsf{depth}(e) > 1$, $\mathsf{depth}(e') > 1$, $\overline{\mathsf{loc}}(e) = \overline{\mathsf{loc}}(e')$, and $e = e'$ in $\mathsf{Act}_{\overline{\mathsf{loc}}(a)}$. In other words, e and e' are occurrences of the same action.*
2. *$\mathsf{depth}(e) = \mathsf{depth}(e') = 1$ and, with a, b, c ranging over $\mathsf{Act}_A, \mathsf{Act}_B, \mathsf{Act}_C$ respectively,*
   - *$e \in p$, $e' \in q$, $e = c \in \mathsf{Act}_{A \Rightarrow C}$, and $e' = c \in \mathsf{Act}_{B \Rightarrow C}$; or*
   - *$e \in q$, $e' \in r$, $e = b_c \in \mathsf{Act}_{B \Rightarrow C}$, and $e' = b \in \mathsf{Act}_{A \Rightarrow B}$; or*
   - *$e \in p$, $e' \in r$, $e = a_c \in \mathsf{Act}_{A \Rightarrow C}$, and $e' = a_b \in \mathsf{Act}_{A \Rightarrow B}$; or*
   - *any of the above three cases with e and e' exchanged.*

The depth 1 case (case 2) is familiar in first-order (i.e. λ-calculus) interaction. For example in the following typical interaction:

$\boxed{A \Rightarrow C}$ $\quad c \quad\quad a_c$

$\boxed{B \Rightarrow C}$ $\quad c \quad b_c$

$\boxed{A \Rightarrow B}$ $\quad\quad b \quad a_b$

The two copies of c (one in $A \Rightarrow C$ and the other in $B \Rightarrow C$) match; $b_c \in B \Rightarrow C$ and $b \in A \Rightarrow B$ match; $a_b \in A \Rightarrow B$ and $a_c \in A \Rightarrow C$ match.

Note that in the example interaction at the beginning of the chapter we judiciously renamed the actions of $\mathsf{Nat}' \rightarrow \mathsf{Bool}$, $\mathsf{Nat} \rightarrow \mathsf{Bool}$, and $\mathsf{Nat}' \rightarrow \mathsf{Nat}$ in order to avoid 'subscript tags' (such as c in b_c or a_c). So in all cases the matching of a 'sent' and 'received' live atomic action reduced simply to 'having the same name'. Examples of match in the interaction state of page 123 include any of the pairs in the action-source table underneath the picture, e.g. $b_2 \sim b_1$ (an instance of condition 2, modulo the judicious renaming), and $w_1 \sim w_3$ (an instance of condition 1).

An important fact used in the interaction algorithm is that the sets of holes associated with matched live atomic actions are isomorphic. This will allow us to pass stored arenas back and forth between the corresponding holes during interaction (modulo possible expansion of the stored arenas on their way into the top hypersequence).

LEMMA 6.9 *For all live atomic actions $a, a' \in \mathsf{P\overline{act}}(\rho)$, if $a \sim a'$ then $\mathsf{Hol}(a) \cong \mathsf{Hol}(a')$.*

Proof Direct from Definition 6.8 of match and the definition of $\mathsf{Hol}(a)$ in section 6.1.4. Note in particular that by definition of match, for $b_c \in \mathsf{Act}_{A \Rightarrow B}$, $\mathsf{Hol}_{A \Rightarrow B}(b_c) \cong \mathsf{Hol}_B(b) \cong \mathsf{Hol}_{A \Rightarrow B}(b)$, and for $a_c \in \mathsf{Act}_{A \Rightarrow C}$, $\mathsf{Hol}_{A \Rightarrow C}(a_c) \cong \mathsf{Hol}_A(a) \cong \mathsf{Hol}_{A \Rightarrow B}(a_b)$. □

6.1.6 Source

We formalise the source depicted underneath each of the interaction states in the motivating interaction example.

DEFINITION 6.10 *A **sourced interaction state** $(p, q, r, \mathsf{next}, \mathsf{source})$ is an interaction state (p, q, r, next) together with a partial function*

$$\mathsf{source} : \mathsf{P\overline{act}}(p) + \mathsf{Oact}(q) + \mathsf{Oact}(r) \rightharpoonup \mathsf{O\overline{act}}(p) + \mathsf{Pact}(q) + \mathsf{Pact}(r)$$

*assigning a **source** to each of the live atomic actions a in its domain, such that*

- *$\mathsf{source}(a)\Uparrow$ if and only if $a = \bullet$, the distinguished dummy action.*
- *$\mathsf{source}(a) \sim a$, i.e., every action matches its source.*
- *$\mathsf{source}(a) \preccurlyeq a$, i.e., the source of an action is in a (non-strictly) preceding column of the interaction state.*

*If $\mathsf{source}(a)\Uparrow$ then a (where necessarily $a = \bullet$) is a **dummy** action; if $\mathsf{source}(a)$ is in the same hypersequence as a, a is a **copycat** action; otherwise a is a **transmitted** action.*

$\mathsf{Int}_\mathsf{s}(A, B, C)$ denotes the set of sourced interaction states of A, B, and C.

In the example interaction state of page 123, the graph of the source was shown in the table underneath the interaction state. (In that particular example we did not come accross any dummy actions.) Note that, because of the typing of source, if a is a copycat action then the condition $\mathsf{source}(a) \preccurlyeq a$ becomes strict, i.e., $\mathsf{source}(a) \prec a$.

LEMMA 6.11 *For all live atomic actions $a \in \mathsf{P\overline{act}}(\rho)$, $\mathsf{Hol}(a) \cong \mathsf{Hol}(\mathsf{source}(a))$.*

Proof Immediate from Lemma 6.9, since by definition of source, $\mathsf{source}(a) \sim a$. $\square$

Extend

$$\mathsf{source} : \mathsf{P\overline{act}}(p) + \mathsf{Oact}(q) + \mathsf{Oact}(r) \rightharpoonup \mathsf{O\overline{act}}(p) + \mathsf{Pact}(q) + \mathsf{Pact}(r)$$

to holes, by defining

$$\mathsf{source} : \mathsf{PHol}(p) + \mathsf{OHol}(q) + \mathsf{OHol}(r) \to \mathsf{OHol}(p) + \mathsf{PHol}(q) + \mathsf{PHol}(r)$$

via the isomorphism $\mathsf{Hol}(a) \cong \mathsf{Hol}(\mathsf{source}(a))$ of Lemma 6.9 just above. In other words, for every live atomic action a in the domain of source, writing $f_a : \mathsf{Hol}(a) \cong \mathsf{Hol}(\mathsf{source}(a))$ for the isomorphism, define $\mathsf{source}(\alpha) = f(\alpha) \in \mathsf{Hol}(\mathsf{source}(a))$ for each hole $\alpha \in \mathsf{Hol}(a)$.

Following the convention that $\alpha_1, \alpha_2, \alpha_3$ mean respectively occurrences of the hole α in the top, middle, or bottom hypersequence, the *source* table on holes of the *inc/even* example (page 123) is

hole	β_2	α_3	α'_1	$w\beta'_1$	$\dot{\beta}'_2$
source	β_1	α_2	α'_3	β'_2	$\dot{w}\dot{\beta}'_1$

Note that, on holes, *source* is total, because $\mathsf{Hol}(\bullet) = \emptyset$ always.

6.1.7 Lookup and full expansion

We define a form of repeated expansion of arenas stored in the plays q or r of a sourced interaction state $\rho = (p, q, r, \mathsf{next}, \mathsf{source})$ which is crucial for the definition of the interaction algorithm in the next section.

A motivating example is the expansion of the singleton arena $\langle \alpha, w \rangle$ in the illustrative example of interaction at the beginning of the chapter. First we 'looked up the hidden contents' Bool' of O's first hole α of the $\mathsf{Nat}' \to \mathsf{Bool}$ play, and then (page 102) performed the expansion of the singleton arena $\langle \alpha, w \rangle$ by the assignment $\alpha \mapsto \mathsf{Bool}'$.

We begin with a formal notion of 'look up the hidden contents' of an O-hole.

Definition 6.12 *Let $\rho = (p, q, r, \mathsf{next}, \mathsf{source})$ be a sourced interaction state. Define*

$$\mathsf{lookup} : \mathsf{OHol}(q) + \mathsf{OHol}(r) \to \mathsf{OHol}(p) + \mathbb{PA}$$

on $\alpha \in \mathsf{OHol}(q) + \mathsf{OHol}(r)$ by

$$\mathsf{lookup}(\alpha) \quad = \quad \begin{cases} \mathsf{store}(\mathsf{source}(\alpha)) \in \mathbb{PA} & \mathit{if}\ \mathsf{source}(\alpha) \in \mathsf{PHol}(q) + \mathsf{PHol}(r) \\ \mathsf{source}(\alpha) & \mathit{if}\ \mathsf{source}(\alpha) \in \mathsf{OHol}(p) \end{cases}$$

In case 1, we are 'looking up the hidden contents of α' by going back to the source P-move. In the illustrative example earlier, we found Bool' this way when we constructed $w\mathsf{Bool}'$ (page 102). In case 2, there are no 'hidden contents' because the move containing the hole originated from an O-move in p.

On page 102, having 'looked up the hidden contents' Bool' of α, we then expanded the singleton arena $\langle \alpha, w \rangle$ by the assignment $\alpha \mapsto \mathsf{Bool}'$, modifying $\langle \alpha, w \rangle$ in order to make it a viable stored arena of the top sequence p. This idea is formalised in the definition below. Given a set holes G of ρ, denote by $G^{<i}$ the subset of G consisting of holes with timestamp strictly less than i.

Definition 6.13 *Let $\rho = (p, q, r, \mathsf{next}, \mathsf{source})$ be a sourced interaction state. Then given an arena D stored in a P-hole $\alpha \in \mathsf{PHol}(q) + \mathsf{PHol}(r)$, say $\mathsf{store}(\alpha) = D$ for α a hole of $n \in \mathsf{Pmov}(p)$, define the* ***full expansion*** *D^* of D, with global holes*

$$\mathsf{Hol}^{\uparrow}_{D^*} \quad \subseteq \quad H + \mathsf{OHol}(p)^{<\mathsf{T}(n)},$$

(i.e., with global holes amongst H or the O-holes of p preceding n) as follows.

By the scope condition of Definition 6.3 of play,

$$\mathsf{Hol}^{\uparrow}_{D} \quad \subseteq \quad H + \mathsf{OHol}(s)^{<\mathsf{T}(n)}$$

where s is q or r according as $\alpha \in \mathsf{PHol}(q)$ or $\alpha \in \mathsf{PHol}(r)$. In other words, the global holes of D are either global holes of the whole interaction (members of H), or are earlier O-holes of the play (q or r) containing D.

Define

$$\begin{aligned} p\mathsf{OHol}(q) &= \{\, \beta \in \mathsf{OHol}(q) \;:\; \mathsf{source}(\beta) \in \mathsf{OHol}(p)\,\} \\ \not{p}\mathsf{OHol}(q) &= \{\, \beta \in \mathsf{OHol}(q) \;:\; \mathsf{source}(\beta) \notin \mathsf{OHol}(p)\,\} \end{aligned}$$

Since

$$\mathsf{Hol}^{\uparrow}_{D} \quad \subseteq \quad H + \mathsf{OHol}(s)^{<\mathsf{T}(n)}$$

we have

$$\mathsf{Hol}^{\uparrow}_{D} \quad \subseteq \quad H + \mathsf{OHol}(q)^{<\mathsf{T}(n)} + \mathsf{OHol}(r)^{<\mathsf{T}(n)}$$

hence

$$\begin{aligned} \mathsf{Hol}^{\uparrow}_{D} \quad \subseteq \quad & H + p\mathsf{OHol}(q)^{<\mathsf{T}(n)} + p\mathsf{OHol}(q)^{<\mathsf{T}(n)} \\ & + \not{p}\mathsf{OHol}(q)^{<\mathsf{T}(n)} + \not{p}\mathsf{OHol}(r)^{<\mathsf{T}(n)} \end{aligned}$$

Define the partial function

$$\mathsf{l\overline{oo}kup} : \mathsf{OHol}(q) + \mathsf{OHol}(r) \rightharpoonup \mathbb{PA}$$

in the canonical way from the function

$$\mathsf{lookup} : \mathsf{OHol}(q) + \mathsf{OHol}(r) \to \mathsf{OHol}(p) + \mathbb{PA}$$

of Definition 6.12, i.e., $\mathsf{l\overline{oo}kup}(\alpha) = \mathsf{lookup}(\alpha)$ if $\mathsf{lookup}(\alpha) \in \mathbb{PA}$, and $\mathsf{l\overline{oo}kup}(\alpha)\!\uparrow$ if $\mathsf{l\overline{oo}kup}(\alpha) \in \mathsf{OHol}(p)$. By the scope condition of Definition 6.3, which constrains store in Definition 6.12, the result $\mathsf{l\overline{oo}kup}^*(D)$ of expanding D along the assignment $\mathsf{l\overline{oo}kup}$ has global holes

$$\begin{aligned} \mathsf{Hol}^{\uparrow}_{\mathsf{l\overline{oo}kup}^*(D)} \quad \subseteq \quad & H + p\mathsf{OHol}(q)^{<\mathsf{T}(n)} + p\mathsf{OHol}(r)^{<\mathsf{T}(n)} \\ & + \not{p}\mathsf{OHol}(q)^{<\mathsf{T}(n)-1} + \not{p}\mathsf{OHol}(r)^{<\mathsf{T}(n)-1} \end{aligned}$$

Hence the repeated expansion $(\mathsf{l\overline{oo}kup}^*)^k(D)$, k times along $\mathsf{l\overline{oo}kup}$, has global holes

$$\begin{aligned} \mathsf{Hol}^{\uparrow}_{(\mathsf{l\overline{oo}kup}^*)^k(D)} \quad \subseteq \quad & H + p\mathsf{OHol}(q)^{<\mathsf{T}(n)} + p\mathsf{OHol}(r)^{<\mathsf{T}(n)} \\ & + \not{p}\mathsf{OHol}(q)^{<\mathsf{T}(n)-k} + \not{p}\mathsf{OHol}(r)^{<\mathsf{T}(n)-k} \end{aligned}$$

so in particular

$$\mathsf{Hol}^{\uparrow}_{(\mathsf{l\overline{oo}kup}^*)^{\mathsf{T}(n)-1}(D)} \quad \subseteq \quad H + p\mathsf{OHol}(q)^{<\mathsf{T}(n)} + p\mathsf{OHol}(q)^{<\mathsf{T}(n)}$$

Define D^* *to be* $(\overline{\mathsf{loo}}\mathsf{kup}^*)^{\mathsf{T}(n)-1}(D)$ *with each global hole*

$$\beta \in p\mathsf{OHol}(q)^{<\mathsf{T}(n)} + p\mathsf{OHol}(q)^{<\mathsf{T}(n)}$$

renamed to

$$\mathsf{source}(\beta) \in \mathsf{OHol}(p)^{<\mathsf{T}(n)}$$

Thus

$$\mathsf{Hol}^{\uparrow}_{D^*} \quad \subseteq \quad H + \mathsf{OHol}(p)^{<\mathsf{T}(n)}$$

6.1.8 Well-formed (sourced) interaction states

Below we refine our notion of sourced interaction state by imposing various conditions needed in order to define interaction smoothly. In digesting the conditions, it may be helpful to browse the examples following the definition in parallel with the definition itself.

DEFINITION 6.14 *A sourced interaction state* $(p, q, r, \mathsf{next}, \mathsf{source})$ *is* ***well-formed*** *if for all live atomic actions* $e \in P\overline{\mathsf{act}}(p) + \mathsf{Oact}(q) + \mathsf{Oact}(r)$,

1. $\mathsf{depth}(e) = 1$ *if and only if* $\mathsf{depth}(\mathsf{source}(e)) = 1$.

2. *If* $\mathsf{depth}(e) > 1$ *(so* $\mathsf{depth}(\mathsf{source}(e)) > 1$ *also) then*

   $$f \overline{\curvearrowleft} \mathsf{source}(e) \qquad \text{if and only if} \qquad \mathsf{source}(f) \curvearrowleft e$$

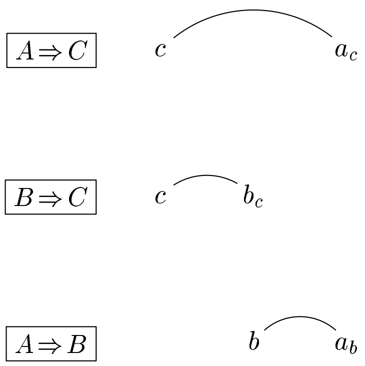
   So in particular e *is (live atomic) unjustified if and only if* $\mathsf{source}(e)$ *is (live atomic) unjustified.*

3. *Suppose* $\alpha \in \mathsf{PHol}(p)$, *hence* $\mathsf{source}(\alpha) \in \mathsf{OHol}(p) + \mathsf{PHol}(q) + \mathsf{PHol}(r)$.

   (a) *If* $\mathsf{source}(\alpha) \in \mathsf{PHol}(q) + \mathsf{PHol}(r)$ *then* $\mathsf{store}(\alpha) = \mathsf{store}(\mathsf{source}(\alpha))^*$, *the full expansion of the contents of* $\mathsf{source}(\alpha)$.

   (b) *If* $\mathsf{source}(\alpha) \in \mathsf{OHol}(p)$ *then* $\mathsf{store}(\alpha) = \langle \mathsf{source}(\alpha), \bullet \rangle$, *the singleton arena with dummy action* $\bullet$ *referencing the* O*-hole* $\mathsf{source}(\alpha)$.

4. *If* $\mathsf{depth}(e) = 1$ *(so* $\mathsf{depth}(\mathsf{source}(e)) = 1$, *by condition 1) then, as motivated by the following typical first-order/*λ*-calculus interaction*

   (a) $\mathsf{T}(e) = \mathsf{T}(\mathsf{source}(e))$, *i.e.,* e *and* $\mathsf{source}(e)$ *are in the same column. (So in particular, by the typing of* source, e *and* $\mathsf{source}(e)$ *are in distinct hypersequences.)*

(b) *if* $e \in \mathsf{act}(r)$, $\mathsf{source}(e) \in \mathsf{act}(q)$ *and* $\mathsf{source}(e) = b_c \in \mathsf{Act}_C^{op} \times \mathsf{Act}_B^{op} \hookrightarrow \mathsf{Act}_{A \Rightarrow B}$, *then* e *is unjustified. (E.g. in the diagram above* $e = b$ *and* $\mathsf{source}(e) = b_c$.*)*

(c) *if* $e \in \overline{\mathsf{act}}(p)$, $\mathsf{source}(e) \in \mathsf{act}(r)$ *and* $\mathsf{source}(e) = a_c \in \mathsf{Act}_C^{op} \times \mathsf{Act}_A^{op} \hookrightarrow \mathsf{Act}_{A \Rightarrow C}$ *then* $\mathsf{source}(f)\overline{\curvearrowleft}e$, *where* $f\overline{\curvearrowleft}\mathsf{source}(g)$ *and* $g\overline{\curvearrowleft}\mathsf{source}(e)$. *(E.g., in the diagram above,* $e = a_c$, $\mathsf{source}(e) = a_b$, $g = b$, $\mathsf{source}(g) = b_c$, f *is the occurrence of* c *at the start of the second row, and* $\mathsf{source}(f)$ *is the occurrence of* c *at the start of the top row.)*

(d) *Otherwise (i.e., neither of (b) or (c) hold)* $f\overline{\curvearrowleft}\mathsf{source}(e)$ *if and only if* $\mathsf{source}(f)\overline{\curvearrowleft}e$. *(This is the depth 1 counterpart of condition 2.)*

We use "well-formed interaction state" as an abbreviation for "well-formed sourced interaction state". Write $\mathsf{Int}_{\mathsf{wf}}(A, B, C)$ *for the set of well-formed interaction states.*

Consider the $\mathsf{Nat}' \to \mathsf{Nat} \to \mathsf{Bool}$ example on page 123. Condition 1 is clearly satisfied. Instances of condition 2 are: $\ddot{b}'_2\overline{\curvearrowleft}\dot{1}'_2$ and $\tilde{b}'_2\overline{\curvearrowleft}1'_2$ (i.e., $\mathsf{source}(\tilde{b}'_2)\overline{\curvearrowleft}1'_2$ and $\tilde{b}'_2\overline{\curvearrowleft}\mathsf{source}(\dot{1}'_2)$); $b'_1\overline{\curvearrowleft}0'_1$ and $\dot{b}'_2\overline{\curvearrowleft}0'_2$ (i.e., $\mathsf{source}(\dot{b}'_2)\overline{\curvearrowleft})'_1$ and $b'_2\overline{\curvearrowleft}\mathsf{source}(\dot{0}'_1)$). Recall that the graph of source on holes is

hole	β_2	α_3	α'_1	$w\beta'_1$	$\dot{\beta}'_2$
source	β_1	α_2	α'_3	β'_2	$\dot{w}\dot{\beta}'_1$

The (one and only) instance of condition 3(a) is

$$\mathsf{store}(w\beta'_1) \quad = \quad \mathsf{store}(\mathsf{source}(w\beta'_1))^* \quad = \quad \mathsf{store}(\beta'_2)^* \quad = \quad \langle \beta_2, c \rangle^* \quad = \quad \langle \beta_1, c \rangle$$

(This example is somewhat degenerate, since the expansion is trivial.) Condition 3(b) does not arise in this example. Conditions 4(a-c) are seen to hold trivially (bear in mind for (b) and (c) that we judiciously renamed the actions of the function spaces, as remarked on page 129). An example of condition 4(d) is that $n'_1\overline{\curvearrowleft}z'_1$ and $n'_3\overline{\curvearrowleft}z'_3$, i.e., $n'_1\overline{\curvearrowleft}\mathsf{source}(z'_3)$ and $\mathsf{source}(n'_1)\overline{\curvearrowleft}z'_3$.

6.1.9 The algorithm

Having armed ourselves with all the necessary auxiliary definitions and concepts, we are ready to present the interaction algorithm. Let $\underline{*}$ denote the constant function with value $*$, with domain to be deduced from the context.

Definition 6.15 *The set of* ***valid*** *interaction states* $\mathsf{Int}(A, B, C) \subseteq \mathsf{Int}_{\mathsf{wf}}(A, B, C)$, *a subset of the set of well-formed interaction states of* A, B, *and* C, *is defined by recursion on length of* $\rho = (p, q, r, \mathsf{next}, \mathsf{source}) \in \mathsf{Int}_{\mathsf{wf}}(A, B, C)$.

Base cases.

1. $|\rho| = 0$: *Set* $(\epsilon, \epsilon, \epsilon, 1, \epsilon) \in \mathsf{Int}(A, B, C)$. *So initially, with* $\mathsf{next} = 1$, O *is about to play in* $p = \epsilon$.

2. $|\rho| = 1$. *For singleton positions* $\mu = (A \Rightarrow C, \phi, c) \in \mathsf{Pos}(A \Rightarrow C)$, *let* $\mu' = (B \Rightarrow C, \phi, c)$ *and set*

$$(\mu, \mu', \epsilon, 2, \mathsf{source}(\mu') = \mu) \in \mathsf{Int}(A, B, C).$$

In other words, given any opening O*-hypermove* μ *of* $A \Rightarrow C$ *(necessarily a single move, by Lemma 5.13, page 76) in* p, *we transmit* μ *across to* q *to appear as an opening* O*-hypermove of* $B \Rightarrow C$.

Induction step. *Suppose $\rho \in \mathsf{Int}(A, B, C)$, with $|\rho| \geq 1$. Subdivide into cases according to $\mathsf{next} \in \{1, 2, 3\}$. We consider the cases in the order $\mathsf{next} = 2$, $\mathsf{next} = 3$, $\mathsf{next} = 1$. The bulk of the material is case $\mathsf{next} = 2$, on pages 134–151; $\mathsf{next} = 3$ is essentially the same as $\mathsf{next} = 2$; $\mathsf{next} = 1$ is on pages 151–154.*

1. *$\mathsf{next} = 2$, i.e. q has control, so in q it is P's turn. Let $\mu = m_1 \ldots m_k$ be a hypermove extending the view of q, in other words such that $\ulcorner q \urcorner \mu \in \mathsf{Pos}(B \Rightarrow C)$. Let $m_1 = (D, \phi, a)$ and let $m = (D, \underline{*}, b)$ be the justifier of m_1. Let $q' = q\mu$, the extension of q by μ, justify m_1 by $m = (D, \underline{*}, b)$, and set $\mathsf{T}(m_1) = |\rho| + 1$:*

$$\begin{array}{ccccc}
 & & \mathsf{O} & & \mathsf{P} \\
\boxed{q'} & \ldots & (D, \underline{*}, b) & \ldots & (D, \phi, a) \\
 & & & & m_2 \\
 & & & & \vdots \\
 & & & & m_k
\end{array}$$

Note that, by Lemma 6.5 (page 124), $q' = q\mu \in \mathsf{Plays}(B \Rightarrow C)$. Let $\dot{b} = \mathsf{source}(b)$. Due to the typing of source, there are three subcases: (a) $\dot{b} \in \mathsf{act}(q)$; (b) $\dot{b} \in \mathsf{act}(r)$; (c) $\dot{b} \in \overline{\mathsf{act}}(p)$. Note that $b = \bullet$ is impossible, since by condition 2 of the definition of interaction state (page 125), $\bullet \in \overline{\mathsf{act}}(p)$ only, and here $b \in \mathsf{act}(q)$.

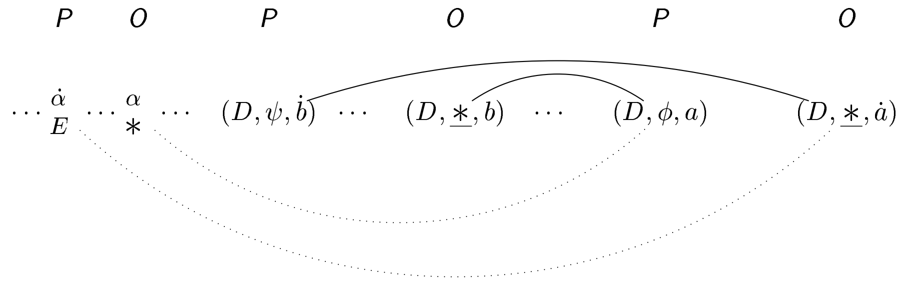
   (a) *$\dot{b} \in \mathsf{act}(q)$. So b is a copycat action, $\mathsf{pl}(\dot{b}) = \mathsf{P}$, $\mathsf{depth}(b) = \mathsf{depth}(a) > 1$, and $\dot{b}$ and b are occurrences of the same action, i.e., $\dot{b} = b$ in Act_D (see diagram below). Let $\dot{m} = (D, \psi, \dot{b})$ be the move containing $\dot{b}$. Let $\dot{m}_1$ be a copy of m with stored arenas 'hidden', i.e. $\dot{m}_1 = (D, \underline{*}, \dot{a})$, where $\dot{a}$ and a are distinct occurrences of the same action, i.e., $\dot{a} = a$ in Act_D. Take $q'' = q'\mu\dot{m}_1$, justify $\dot{m}_1$ by $\dot{m}$, and set $\mathsf{T}(\dot{m}_1) = \mathsf{T}(m) + 1$. So q'' is:*

$$\begin{array}{cccccccc}
 & & \mathsf{P} & & \mathsf{O} & & \mathsf{P} & \mathsf{O} \\
\boxed{q''} & \ldots & (D, \psi, \dot{b}) & \ldots & (D, \underline{*}, b) & \ldots & (D, \phi, a) & (D, \underline{*}, \dot{a}) \\
 & & & & & & m_2 & \\
 & & & & & & \vdots & \\
 & & & & & & m_k &
\end{array}$$

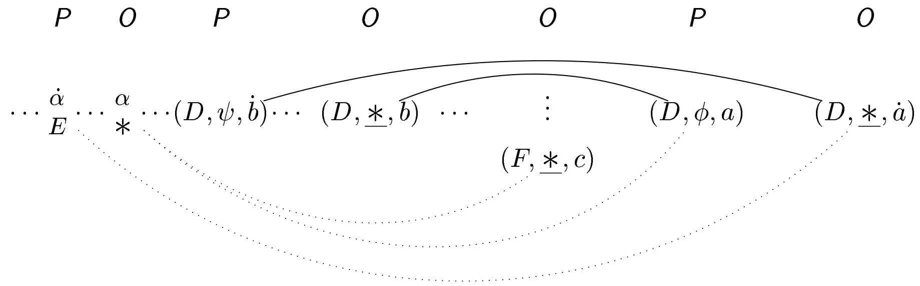
   Due to the typing of $\mathsf{ref}_{q''}$ we consider two subsubcases: i. $\mathsf{ref}_{q''}(\dot{a}) \in H + \mathsf{OHol}(q'')$; ii. $\mathsf{ref}_{q''}(\dot{a}) \in \mathsf{PHol}(p)$.

      i. *$\mathsf{ref}_{q''}(\dot{a}) \in H + \mathsf{OHol}(q'')$. Since $\mathsf{ref}_{q'}(\dot{a})$ has no stored arena, $\dot{m}_1 = (D, \underline{*}, \dot{a})$ completes a hypermove for O. So set $(p, q'', r, 2, \mathsf{source}') \in \mathsf{Int}(A, B, C)$, where source' extends source with $\mathsf{source}'(\dot{a}) = a$.*

      ii. *$\mathsf{ref}_{q''}(\dot{a}) \in \mathsf{PHol}(p)$. Let $\alpha = \mathsf{ref}_{q''}(a)$ (see diagram below). Since $\mathsf{depth}(b) = \mathsf{depth}(a) > 1$, by condition 2 of the definition of interaction state (page 132), a and $\dot{a}$ are in 'copycat threads', so $\mathsf{ref}_{q''}(\dot{a}) = \mathsf{source}(\alpha)$. Let $\dot{\alpha} = \mathsf{source}(\alpha)$ (which happens to be an occurrence of α, i.e., $\dot{\alpha} = \alpha$ in $\mathsf{Hol}^{\downarrow}_D$). Let E be the polymorphic arena stored in $\dot{\alpha}$. So q'' looks like this:*

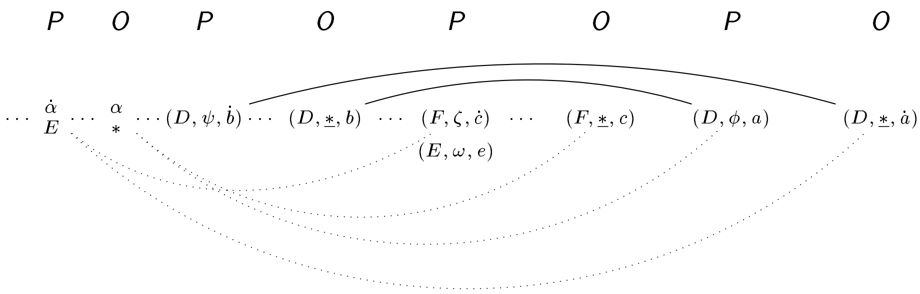


(Reference arcs are shown underneath.) Note that necessarily $k = 1$, i.e., the hypermove μ consists of the single move $m_1 = (D,\phi,a)$, because the reference α of a is an O*-hole. Since the reference $\dot\alpha$ of $\dot a$ stores an arena E, unlike subsubcase i above, $\dot m_1 = (D,\underline{*},\dot a)$ does not complete a hypermove:* O *is required to open a thread on E.*
Since P *played $m_1 = (D,\phi,a)$ subject to the copycat rule, the* O*-move $m_1^- = (F,\underline{*},c)$ immediately preceding m_1 (in other words, the last move of the previous* O*-hypermove) also references α, i.e., $\mathsf{ref}_{q''}(m_1^-) = \alpha$.*

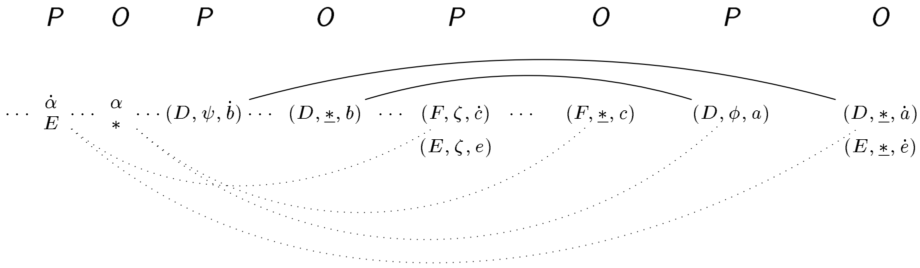


Since c references an O*-hole which is not at depth 1, $\mathsf{depth}(c) > 1$, so $\mathsf{depth}(\mathsf{source}(c)) > 1$ and $\mathsf{source}(c)$ is another occurrence $\dot c$ of c.*
Due to the typing of source*, there are two cases to consider: A. $\dot c \in \mathsf{act}(q)$, and B. $\dot c \in \overline{\mathsf{act}}(p)$. (The case $\dot c \in \mathsf{act}(r)$ is impossible, since that could only happen if $\mathsf{depth}(c) = 1$. The case $\dot c = \mathsf{dummy}$, i.e., $c = \bullet$, is impossible, since $\bullet \in \overline{\mathsf{act}}(p)$ only, and $c \in \mathsf{act}(q)$.) The two cases A and B are very similar: both involve retrieving a move $(E,\underline{*},\dot e)$ to play as a second* O*-move beneath $(D,\underline{*},\dot a)$.*

A. *$\dot c \in \mathsf{act}(q)$. Let $n_1 = (F,\zeta,\dot c)$ be the move containing $\dot c$. By condition 2 of the definition of well-formedness (132), c and $\dot c$ are in 'copycat threads', so $\mathsf{ref}_{q''}(\dot c) = \dot\alpha$, the same as the reference of $\dot a$. Since $\dot\alpha$ stores the arena E, immediately below $n_1 = (F,\zeta,\dot c)$ there must be a move $n_2 = (E,\omega,e)$ which opens a new thread on E.*

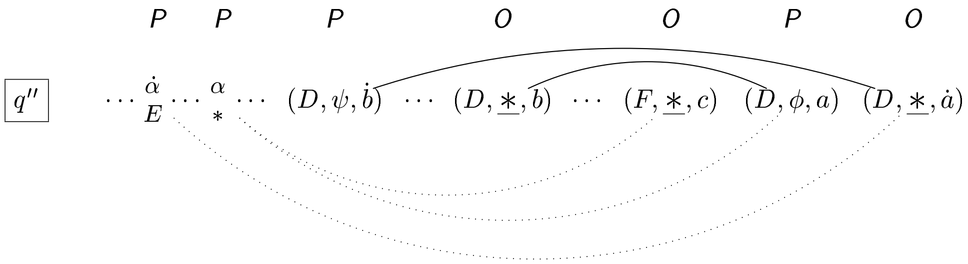


Let $\dot n_2 = (E,\underline{*},\dot e)$, a copy of $n_2 = (E,\omega,e)$ with stored arenas 'hidden', and where $\dot e$ is a fresh occurrence of e (i.e., $\dot e = e$ in Act_E). Define q''' to be q'' with $\dot n_2$ immediately below $\dot m_1 = (D,\underline{*},\dot a)$:



Put $(p, q''', r, 2, \mathsf{source}') \in \mathsf{Int}(A,B,C)$, where source' extends source with $\mathsf{source}'(\dot a) = a$ and $\mathsf{source}'(\dot e) = e$.

B. *$\dot c \in \overline{\mathsf{act}}(p)$. Thus $\dot c$ is part of a compound action $\zeta\dot c\zeta' \in \mathsf{act}(p)$ located in a fully expanded arena G^*:*



Since $\mathsf{ref}_{q''}(c)$ is an O-hole of q'', which was expanded to E in the full expansion G^*, $\dot c$ is not the last atomic action of $\zeta\dot c\zeta'$: the next live atomic action after $\dot c$ is an opening action of E, i.e. $\zeta' = \eta e \zeta''$ for η a trail above ζc and $e \in \mathsf{Act}_E^{op}$. O's action $\dot a$ references $\dot\alpha$, storing E, so O is required to *play an opening move on* E*. We obtain this opening move as* $(E, \underline{*}, \dot{e})$*, for* $\dot{e}$ *a fresh occurrence of* e*, i.e.,* $\dot{e} = e$ *in* Act_E*:*

$$\begin{array}{ccc}
 & & \mathsf{O} \\
\boxed{p} & \cdots & (G^*, \underline{*}, \zeta\dot{c}\eta e\zeta'')
\end{array}$$

$$\begin{array}{cccccccccccccc}
 & & \mathsf{P} & & \mathsf{P} & & \mathsf{P} & & \mathsf{O} & & \mathsf{O} & \mathsf{P} & \mathsf{O} \\
\boxed{q'''} & \cdots & \begin{smallmatrix}\dot{\alpha}\\ E\end{smallmatrix} & \cdots & \begin{smallmatrix}\alpha\\ *\end{smallmatrix} & \cdots & (D, \psi, \dot{b}) & \cdots & (D, \underline{*}, b) & \cdots & (F, \underline{*}, c) & (D, \phi, a) & (D, \underline{*}, \dot{a}) \\
 & & & & & & & & & & & & (E, \underline{*}, \dot{e})
\end{array}$$

This completes a hypermove for O*, since* $\dot{e}$ *references an* O*-hole. Put* $(p, q''', r, 2, \mathsf{source}') \in \mathsf{Int}(A, B, C)$*, where* source' *extends the function* source *with* $\mathsf{source}'(\dot{a}) = a$ *and* $\mathsf{source}'(\dot{e}) = e$*.*

(b) $\dot{b} \in \mathsf{act}(r)$*. So* $m = (D, \underline{*}, b)$ *was a transmission from* P *in* r*.*

$$\begin{array}{ccccc}
 & & \mathsf{O} & & \mathsf{P} \\
\boxed{q'} & \cdots & (D, \underline{*}, b) & \cdots & (D, \phi, a) \\
 & & & & m_2 \\
 & & & & \vdots \\
 & & & & m_k
\end{array}$$

By definition of source (page 129), since $\mathsf{source}(b) = \dot{b}$*, we have* $b \sim \dot{b}$*, and by definition of match (page 128),* $\mathsf{depth}(b) = \mathsf{depth}(\dot{b}) = 1$ *and* $D = B \Rightarrow C$*. Furthermore* $b = d_c \in \mathsf{Act}_{B \Rightarrow C}$ *for* $d \in \mathsf{Act}_B$ *and* $c \in \mathsf{Act}_C^{op}$*, and* $a = e_c \in \mathsf{Act}_{B \Rightarrow C}$ *for* $e \in \mathsf{Act}_B$*, with* $d \vdash_B e$*. So* $m = (B \Rightarrow C, \underline{*}, d_c)$*,* $m_1 = (B \Rightarrow C, \phi, e_c)$*, and* $q' = q\mu$ *looks like this:*

$$\begin{array}{cccc}
 & \mathsf{O} & & \mathsf{P} \\
\cdots & (B \Rightarrow C, \underline{*}, d_c) & \cdots & (B \Rightarrow C, \phi, e_c) \\
 & & & m_2 \\
 & & & \vdots \\
 & & & m_k
\end{array}$$

(Both d_c and e_c are at depth 1.) By definition of match (page 128), $\dot{b} = \mathsf{source}(b) = \mathsf{source}(d_c) \in \mathsf{act}(r)$ is an occurrence $\dot{d}$ of d (i.e., $\dot{d} = d$ in Act_B), and $\mathsf{depth}(\dot{d}) = 1$. Let $\dot{m} = (A \Rightarrow B, \psi, \dot{d}) \in \mathsf{act}(r)$ be the move containing $\dot{d}$. So the interaction state looks like this:

$$\begin{array}{cccccc} & & \mathsf{O} & & & \mathsf{P} \\ \boxed{q'} & \cdots & \cdots\ (B \Rightarrow C, \underline{*}, d_c) & \cdots & \cdots & (B \Rightarrow C, \phi, e_c) \\ & & & & & m_2 \\ & & & & & \vdots \\ & & & & & m_k \\ & & & & & \\ & & \mathsf{P} & & & \\ \boxed{r'} & \cdots & \cdots\ (A \Rightarrow B, \psi, \dot{d}) & \cdots & \cdots & \end{array}$$

We shall transmit $m_1 = (B \Rightarrow C, \phi, e_c)$ accross to r, justified by $\dot{m} = (A \Rightarrow B, \psi, \dot{d})$. Let $\dot{m}_1 = (A \Rightarrow B, \underline{*}, \dot{e})$, a 'copy' of m_1 with stored arenas 'hidden', and where e and $\dot{e}$ are occurrences of the same action, i.e., $e = \dot{e}$ in Act_B. Let $r' = r\dot{m}_1$, justify $\dot{m}_1$ by $\dot{m} = (A \Rightarrow B, \psi, \dot{d})$, and set $\mathsf{T}(\dot{m}_1) = \mathsf{T}(m_1)$, i.e., put $\dot{m}_1 = (A \Rightarrow B, \underline{*}, \dot{e})$ in the same column as $m_1 = (B \Rightarrow C, \phi, e_c)$:

$$\begin{array}{cccccc} & & \mathsf{O} & & & \mathsf{P} \\ \boxed{q'} & \cdots & \cdots\ (B \Rightarrow C, \underline{*}, d_c) & \cdots & \cdots & (B \Rightarrow C, \phi, e_c) \\ & & & & & m_2 \\ & & & & & \vdots \\ & & & & & m_k \\ & & & & & \\ & & \mathsf{P} & & & \mathsf{O} \\ \boxed{r'} & \cdots & \cdots\ (A \Rightarrow B, \psi, \dot{d}) & \cdots & \cdots & (A \Rightarrow B, \underline{*}, \dot{e}) \end{array}$$

Analogous to (a)i and (a)ii earlier, there are two subcases: i. $\mathsf{ref}_{r'}(\dot{e}) \in H + \mathsf{OHol}(r')$, and ii. $\mathsf{ref}_{r'}(\dot{e}) \in \mathsf{Hol}(n)$ for some P-move n in r'.

i. *$\mathsf{ref}_{r'}(\dot{e}) \in H + \mathsf{OHol}(r')$. Since $\mathsf{ref}_{r'}(\dot{e})$ has no stored arena, $\dot{m}_1 = (A \Rightarrow B, \underline{*}, \dot{e})$ completes a hypermove for O. Set $(p, q', r', 3, \mathsf{source}') \in \mathsf{Int}(A, B, C)$, where source' extends source with $\mathsf{source}'(\dot{e}) = e_c$.*

ii. *$\mathsf{ref}_{r'}(\dot{e}) \in \mathsf{Hol}(n)$ for some P-move n in r'. Let $\alpha = \mathsf{ref}_{q'}(e_c)$, and note that, by construction of function space, $\alpha = \beta_c \in \mathsf{Act}_C^{op} \times \mathsf{Hol}_B^{\downarrow} \hookrightarrow \mathsf{Hol}_{B \Rightarrow C}^{\downarrow}$ for some $\beta \in \mathsf{Hol}_B^{\downarrow}$. (Adopting the usual convention, we write β_c for the pair $\langle c, \beta \rangle$.) Since $\dot{e}$ is a transmission of e_c, by condition 4(d) of Definition 6.14 of well-formed interaction state, $\mathsf{ref}_{r'}(\dot{e})$ is a copy $\dot{\beta}$ of β, and $\mathsf{source}(\beta_c) = \dot{\beta}$:*

$$\begin{array}{cccccccc}
 & & \mathsf{O} & & \mathsf{O} & & & \mathsf{P} \\
\boxed{q'} & \cdots & \beta_c & \cdots & (B \Rightarrow C, \underline{*}, d_c) & \cdots & \cdots & (B \Rightarrow C, \phi, e_c) \\
 & & * & & & & &
\end{array}$$

$$\begin{array}{cccccccc}
 & & \mathsf{P} & & \mathsf{P} & & & \mathsf{O} \\
\boxed{r'} & \cdots & \dot{\beta} & \cdots & (A \Rightarrow B, \psi, \dot{d}) & \cdots & \cdots & (A \Rightarrow B, \underline{*}, \dot{e}) \\
 & & E & & & & &
\end{array}$$

Note that $k = 1$, i.e., P's hypermove μ consists in the single move $m_1 = (B \Rightarrow C, \phi, e_c)$, because e_c references an O-hole. We continue to proceed analogously with case (a)ii. Since $\dot{\beta}$ stores an arena E, $\dot{m}_1$ does not complete a hypermove. O is required to open a thread on E.
Since P played $m_1 = (B \Rightarrow C, \phi, e_c)$ subject to the copycat rule, the O-move $m_1^- = (F, \underline{*}, f)$ immediately preceding $m_1 = (B \Rightarrow C, \phi, e_c)$ (in other words, the last move of the previous O-hypermove) also references β_c, i.e., $\mathsf{ref}_{q'}(m_1^-) = \beta_c$:

$$\begin{array}{ccccccccc}
 & & \mathsf{O} & & \mathsf{O} & & & \mathsf{O} & \mathsf{P} \\
\boxed{q'} & \cdots & \beta_c & \cdots & (B \Rightarrow C, \underline{*}, d_c) & \cdots & \cdots & \vdots & (B \Rightarrow C, \phi, e_c) \\
 & & * & & & & & (F, \underline{*}, f) &
\end{array}$$

$$\begin{array}{cccccccc}
 & & \mathsf{P} & & \mathsf{P} & & & \mathsf{O} \\
\boxed{r'} & \cdots & \dot{\beta} & \cdots & (A \Rightarrow B, \psi, \dot{d}) & \cdots & \cdots & (A \Rightarrow B, \underline{*}, \dot{e}) \\
 & & E & & & & &
\end{array}$$

Let $\dot{f} = \mathsf{source}(f)$. Due to the typing of source, there are three cases to consider: A. $\dot{f} \in \mathsf{act}(r)$, B. $\dot{f} \in \overline{\mathsf{act}}(p)$, C. $\dot{f} \in \mathsf{act}(q)$. ($f = \bullet$ is impossible, since $f = \bullet$ only if $f \in \overline{\mathsf{act}}(p)$.)

A. *$\dot{f} \in \mathsf{act}(r)$. Thus $\mathsf{depth}(f) = 1$, so $F = B \Rightarrow C$ and since $\beta \in \mathsf{Hol}_B^{\downarrow}$, $f = h_c$ for some $h \in \mathsf{Act}_B$ (the only way taht f could reference β_c).*

$$\begin{array}{cccccccc} & & O & & O & & O & P \\ \boxed{q'} & \cdots & \beta_c & \cdots & (B\Rightarrow C,\underline{\ast},d_c) & \cdots & (B\Rightarrow C,\underline{\ast},h_c) & (B\Rightarrow C,\phi,e_c) \\ & & * & & & & & \end{array}$$

$$\begin{array}{cccccccc} & & P & & P & & & O \\ \boxed{r'} & \cdots & \dot\beta & \cdots & (A\Rightarrow B,\psi,\dot d) & \cdots & & (A\Rightarrow B,\underline{*},\dot e) \\ & & E & & & & & \end{array}$$

Since $\dot f \in \mathsf{act}(r')$, $\dot f = \dot h$, *another occurrence of* $h \in \mathsf{Act}_B$. *Let* $n_1 = (A \Rightarrow B, \zeta, \dot h)$ *be the move containing* $\dot h$, *which (by condition 4(a) of Definition 6.14 of well-formed interaction state) is in the same column as* $\dot m_1 = (B \Rightarrow C, \underline{\ast}, h_c)$:

$$\begin{array}{cccccccc} & & O & & O & & O & P \\ \boxed{q'} & \cdots & \beta_c & \cdots & (B\Rightarrow C,\underline{\ast},d_c) & \cdots & (B\Rightarrow C,\underline{\ast},h_c) & (B\Rightarrow C,\phi,e_c) \\ & & * & & & & & \end{array}$$

$$\begin{array}{cccccccc} & & P & & P & & P & O \\ \boxed{r''} & \cdots & \dot\beta & \cdots & (A\Rightarrow B,\psi,\dot d) & \cdots & (A\Rightarrow B,\zeta,\dot h) & (A\Rightarrow B,\underline{*},\dot e) \\ & & E & & & & & \end{array}$$

Since $\dot\beta$ *stores an arena* E, *immediately below* $n_1 = (A \Rightarrow B, \zeta, \dot h)$ *there must be a move* $n_2 = (E, \omega, g)$ *opening a thread on* E:

$$\begin{array}{cccccccccc}
 & & O & & O & & O & & P \\
\boxed{q'} & \cdots & \beta_c & \cdots & (B\Rightarrow C, \underline{*}, d_c) & \cdots & (B\Rightarrow C, \underline{*}, h_c) & & (B\Rightarrow C, \phi, e_c) \\
 & & * & & & & & &
\end{array}$$

$$\begin{array}{cccccccccc}
 & & P & & P & & P & & O \\
\boxed{r''} & \cdots & \dot\beta & \cdots & (A\Rightarrow B, \psi, \dot d) & \cdots & (A\Rightarrow B, \zeta, \dot h) & & (A\Rightarrow B, \underline{*}, \dot e) \\
 & & E & & & & (E, \omega, g) & &
\end{array}$$

Let $\dot n_2 = (E, \underline{*}, \dot g)$, a copy of $n_2 = (E, \omega, g)$ with stored arenas 'hidden'. Define r'' to be r' with $\dot n_2$ immediately below $\dot m_1 = (A\Rightarrow B, \underline{*}, \dot e)$, i.e.,

$$\begin{array}{cccccccccc}
 & & O & & O & & O & & P \\
\boxed{q'} & \cdots & \beta_c & \cdots & (B\Rightarrow C, \underline{*}, d_c) & \cdots & (B\Rightarrow C, \underline{*}, h_c) & & (B\Rightarrow C, \phi, e_c) \\
 & & * & & & & & &
\end{array}$$

$$\begin{array}{cccccccccc}
 & & P & & P & & P & & O \\
\boxed{r''} & \cdots & \dot\beta & \cdots & (A\Rightarrow B, \psi, \dot d) & \cdots & (A\Rightarrow B, \zeta, \dot h) & & (A\Rightarrow B, \underline{*}, \dot e) \\
 & & E & & & & (E, \omega, g) & & (E, \underline{*}, \dot g)
\end{array}$$

This completes a hypermove for O, since $\dot g$ cannot reference a P-hole (see Lemma 5.13, part 1). Put $(p, q', r'', 3, \mathsf{source}') \in \mathsf{Int}(A, B, C)$, where source' is the extension of source with $\mathsf{source}'(\dot e) = e_c$ and $\mathsf{source}'(\dot g) = g$.

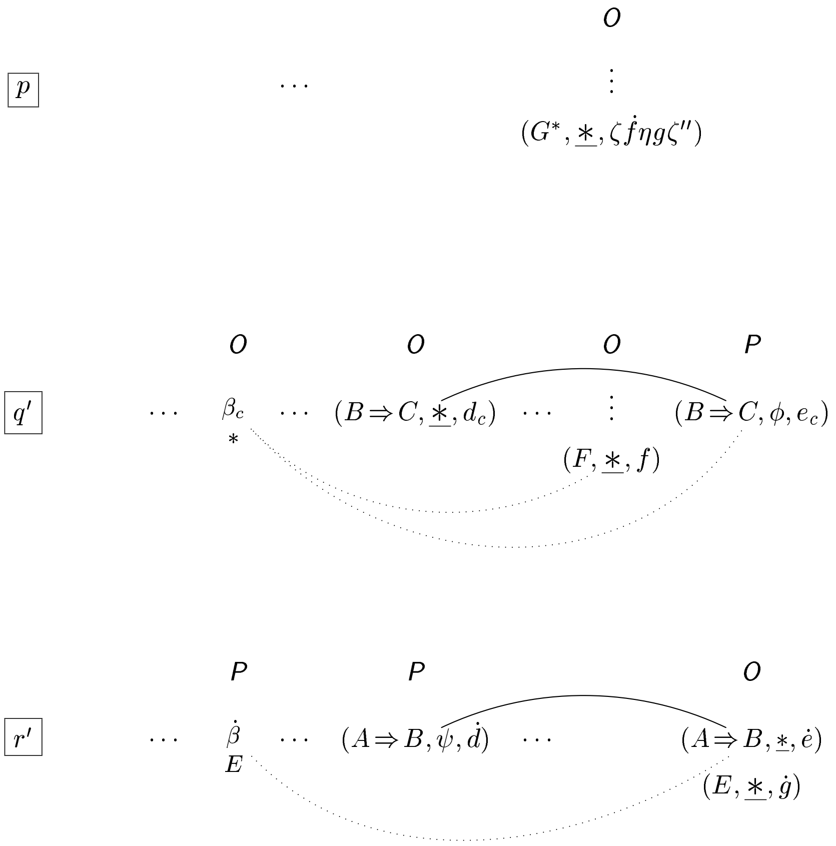
B. *$\dot c \in \overline{\mathsf{act}}(p)$. So $\mathsf{depth}(f) > 1$ (the only way an action with source in p could reference a hole β_c in a 'B component' of $B \Rightarrow C$), and $\dot f$ is part of a compound action $\zeta \dot f \zeta'$ located in a fully expanded arena G^*. (This case is nearly identical to (a)ii.B.)*

$$\begin{array}{llllll} & & & & \mathsf{O} & \\ \boxed{p} & & \cdots & & \vdots & \\ & & & & (G^*, \underline{*}, \zeta\dot{f}\eta g\zeta'') & \end{array}$$

$$\begin{array}{llllllll} & & \mathsf{O} & & \mathsf{O} & & \mathsf{O} & \mathsf{P} \\ \boxed{q'} & \cdots & \beta_c & \cdots & (B \Rightarrow C, \underline{*}, d_c) & \cdots & \vdots & (B \Rightarrow C, \phi, e_c) \\ & & * & & & & (F, \underline{*}, f) & \end{array}$$

$$\begin{array}{lllllll} & & \mathsf{P} & & \mathsf{P} & & \mathsf{O} \\ \boxed{r'} & \cdots & \dot{\beta} & \cdots & (A \Rightarrow B, \psi, \dot{d}) & \cdots & (A \Rightarrow B, \underline{*}, \dot{e}) \\ & & E & & & & \end{array}$$

Since $\mathsf{ref}_{q'}(f)$ is an O-hole of q', which was expanded to E in the full expansion G^*, $\dot{f}$ is not the last atomic action of $\zeta\dot{f}\zeta'$: the next live atomic action after $\dot{f}$ is an opening action of E, i.e. $\zeta' = \eta g\zeta''$ for η a trail above ζf and $g \in \mathsf{Act}_E^{op}$. O's action $\dot{e}$ references $\dot{\beta}$, storing E, so O is required to play an opening move on E. We obtain this opening move as $(E, \underline{*}, \dot{g})$, for $\dot{g}$ a fresh occurrence of g, i.e., $\dot{g} = g$ in Act_E.



This completes a hypermove for O*, since* $\dot{g}$ *cannot reference a* P*-hole (see Lemma 5.13, part 1).*

C. $\dot{f} \in \overline{\mathsf{act}}(q)$. *This case reduces either to case A or to case B above. Let* $n_1 = (F, \zeta, \dot{f})$ *be the move containing* $\dot{f}$*, and note that* $\mathsf{depth}(f) > 1$ *and* $\mathsf{depth}(\dot{f}) > 1$ *since* f *is a copycat action.*

$$\begin{array}{cccccccccccc}
 & & \mathsf{O} & & \mathsf{O} & & \mathsf{P} & & \mathsf{O} & & \mathsf{P} \\
\boxed{q'} & \cdots & \beta_c & \cdots & (B\Rightarrow C, \underline{*}, d_c) & \cdots & \vdots & \cdots & \vdots & & (B\Rightarrow C, \phi, e_c) \\
 & & * & & & & (F,\zeta,\dot f) & & (F,\underline{*},f) & &
\end{array}$$

$$\begin{array}{cccccccccccc}
 & & \mathsf{P} & & \mathsf{P} & & & & & & \mathsf{O} \\
\boxed{r'} & \cdots & \dot\beta & \cdots & (A\Rightarrow B, \psi, \dot d) & \cdots & & \cdots & & & (A\Rightarrow B, \underline{*}, \dot e) \\
 & & E & & & & & & & &
\end{array}$$

Since $\mathsf{depth}(\beta_c) = 1$ *and* $\mathsf{depth}(f) > 1$, *the only way* $\mathsf{ref}_{q'}(f) = \beta_c$ *is possible is with* $\beta_c \in \mathsf{Hol}^{\uparrow}_F$, *a global hole of* F *(as opposed to* $\beta_c \in \mathsf{Hol}^{\downarrow}_F$, *a local hole of* F*). Then because* $\dot f$ *is a copycat action,* $\mathsf{ref}_{q'}(\dot f) = \beta_c$ *also. We are in search of an opening move in* E, *to 'copy' as a move underneath* $\dot m_1 = (A\Rightarrow C, \underline{*}, \dot e)$. *We shall 'chain back' along moves*

$$\begin{array}{cccccccccccccccc}
l_u & \dot l_{u-1} & \cdots & l_{u-1} & \dot l_{u-2} & \cdots & l_{u-2} & \cdots & \cdots & \dot l_2 & \cdots & l_2 & \dot l_1 & \cdots & l_1 \\
\mathsf{O} & \mathsf{P} & & \mathsf{O} & \mathsf{P} & & \mathsf{O} & & & \mathsf{P} & & \mathsf{O} & \mathsf{P} & & \mathsf{O}
\end{array}$$

in q', *each of which references* β_c, *until we reach a suitable candidate* l_u *with* $\mathsf{source}(l_u) \in \mathsf{act}(r') + \overline{\mathsf{act}}(p)$, *and then appeal to case A or B according as* $\mathsf{source}(l_u) \in \mathsf{act}(r')$ *or* $\mathsf{source}(l_u) \in \overline{\mathsf{act}}(p)$.

The O*-moves* $l_i = (G_i, \underline{*}, g_i)$ *and* P*-moves* $\dot l_i = (G_i, \zeta_i, \dot g_i)$ *are determined as follows. Let* $l_1 = (G_1, \underline{*}, g_1)$ *be the* O*-move immediately preceding* $n_1 = (F, \zeta, \dot f)$, *i.e., the last move of the* O*-hypermove preceding the* P*-hypermove containing* n_1. *By the copycat rule,* $\mathsf{ref}_{q'}(l_1) = \mathsf{ref}_{q'}(n_1) = \beta_c$. *Having defined* $l_i = (G_i, \underline{*}, g)_i$, *if* $\mathsf{source}(l_i) \in \mathsf{act}(q')$, *define* $\dot l_i = \mathsf{source}(l_i) = (G_i, \zeta_i, \dot g_i)$ *and define* l_{i+1} *to be the* O*-move preceding* $\dot l_i$, *and repeat. (Thus* $\mathsf{ref}_{q'}(\dot l_i) = \beta_c$, *and by the copycat rule,* $\mathsf{ref}_{q'}(l_{i+1}) = \beta_c$ *also.) Otherwise (i.e., if* $\mathsf{source}(l_i) \in \overline{\mathsf{act}}(p) + \mathsf{act}(r')$*) terminate, setting* u *to be the current value of* i. *(Note that the algorithm must terminate, since* $l_{i+1} < l_i$.*)*

Now by construction either $\mathsf{source}(l_u) \in \mathsf{act}(r')$, *or* $\mathsf{source}(l_u) \in \overline{\mathsf{act}}(p)$. *In the former case, proceed as in case (b)ii.A. above, but with* l_u *in place of* $(F, *, f)$; *in the latter case, proceed as in case (b)ii.B. above, again with* l_u *in place of* $(F, *, f)$.

(c) $\dot b \in \overline{\mathsf{act}}(p)$, *so* m *was a transmission from* O *in* p. *Since* $\dot b = \mathsf{source}(b) \sim b$, *by definition of match (page 128) there are two subcases to consider: i.* $\mathsf{depth}(b) = \mathsf{depth}(\dot b) = 1$, $D = B \Rightarrow C$, $b = c \in \mathsf{Act}^{op}_C \hookrightarrow \mathsf{Act}_{\Rightarrow C}$ *and* $a = \langle\, c, e \,\rangle = e_c \in \mathsf{Act}^{op}_C \times \mathsf{Act}_B \hookrightarrow \mathsf{Act}_{B\Rightarrow C}$, *and ii. otherwise.*

i. $\mathsf{depth}(b) = \mathsf{depth}(\dot b) = 1$, $D = B \Rightarrow C$, $b = c \in \mathsf{Act}^{op}_C \hookrightarrow \mathsf{Act}_{B\Rightarrow C}$ *and* $a = \langle\, b, c \,\rangle = e_c \in \mathsf{Act}^{op}_C \times \mathsf{Act}_B \hookrightarrow \mathsf{Act}_{B\Rightarrow C}$. *So* $q' = q\mu$ *looks like this:*

$$
\begin{array}{cccc}
 & \mathsf{O} & & \mathsf{P} \\
\ldots & (B \Rightarrow C, \underline{*}, d_c) & \ldots & (B \Rightarrow C, \phi, e_c) \\
 & & & m_2 \\
 & & & \vdots \\
 & & & m_k
\end{array}
$$

We shall transmit P's move $m_1 = (B \Rightarrow C, \phi, e_c)$ accross to r as an opening hypermove of $A \Rightarrow B$. Let $\dot{m}_1 = (A \Rightarrow C, \underline{*}, \dot{e})$ be a 'copy' of $m_1 = (B \Rightarrow C, \phi, e_c)$ with stored arenas hidden, where $\dot{e}$ and e are occurrences of the same opening action of B (i.e., $\dot{e} = e$ in Act_B), treated as an opening action of $A \Rightarrow B$. Let $r' = r\dot{m}_1$, with $\dot{m}_1 = (A \Rightarrow C, \underline{*}, \dot{e})$ unjustified:

$$
\begin{array}{ccccc}
 & & \mathsf{O} & & \mathsf{P} \\
\boxed{q'} & \ldots & (B \Rightarrow C, \underline{*}, d_c) & \ldots & (B \Rightarrow C, \phi, e_c) \\
 & & & & m_2 \\
 & & & & \vdots \\
 & & & & m_k
\end{array}
$$

$$
\begin{array}{ccccc}
 & & & & \mathsf{O} \\
\boxed{r'} & & \ldots & \ldots & (A \Rightarrow B, \underline{*}, \dot{e})
\end{array}
$$

Define $\mathsf{T}(\dot{m}_1) = \mathsf{T}(m_1)$. Note that this completes an O-hypermove since $\dot{e}$, being an opening action, references either a global hole of H or one if its own local holes. Set

$$(p, q', r', 3, \mathsf{source}') \in \mathsf{Int}(A, B, C),$$

where source' extends source with $\mathsf{source}'(\dot{e}) = e_c$.

ii. *Otherwise. Thus $\mathsf{depth}(b) > 1$, $\mathsf{depth}(\dot{b}) > 1$, and $\dot{b}$ is part of a compound action $\zeta_1 \dot{b} \zeta_2$ located in a fully expanded arena E^*, where $\zeta_2 = \eta_1[b_1] \ldots \eta_w[b_w]$ and $\eta_i \in \mathsf{trails}(\zeta_1 \dot{b} \eta_1[b_1] \ldots \eta_{i-1}[b_{i-1}])$. Let $q' = q\mu$. Thus:*

$$\begin{array}{cccc}
 & & \mathsf{O} & \\
\boxed{p} & \ldots & (E^*, \underline{*}, \zeta_1 \dot{b} \zeta_2) & \ldots
\end{array}$$

$$\begin{array}{cccccc}
 & & \mathsf{O} & & \mathsf{P} \\
\boxed{q'} & \ldots & (D, \underline{*}, b) & \ldots & (D, \phi, a) \\
 & & & & m_2 \\
 & & & & \vdots \\
 & & & & m_k
\end{array}$$

Define actions a_{ij} and c_i, holes α_i, and $k_i \in \mathbb{N}$, by recursion on i as follows.

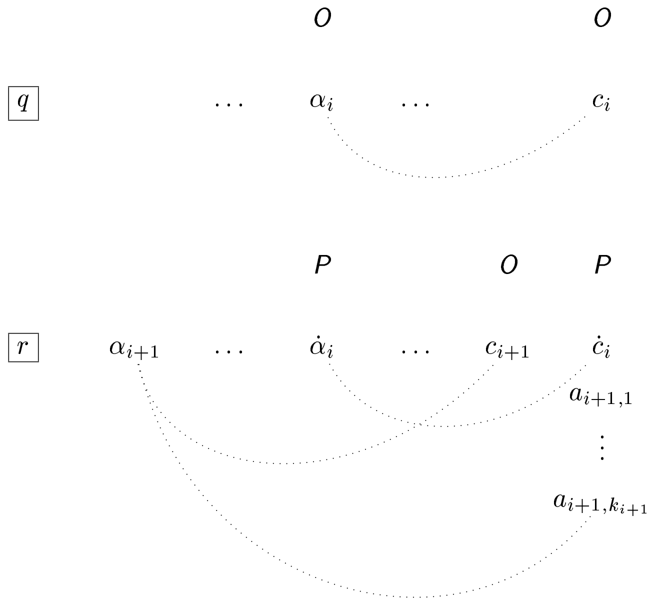
Base case. *Define $k_1 = k$. For $j = 1, \ldots, k$ define a_{1j} to be the action of m_j (so in particular, $a_{11} = a$). Define $\alpha_1 = \mathsf{ref}_{q'}(a_{1k})$, and define c_1 to be the last action of the O-hypermove preceding a_{1k}:*

$$\begin{array}{ccccc}
 & & & \mathsf{O} & \mathsf{P} \\
\boxed{q'} & \alpha_1 & \ldots & \vdots & a_{11} \\
 & & & c_1 & \vdots \\
 & & & & a_{1k_1}
\end{array}$$

(Note that $\alpha \in H$ is possible, though the diagram may suggest $\alpha \in \mathsf{Hol}(q')$.) By the copycat rule, $\mathsf{ref}_{q'}(c_1) = \mathsf{ref}_{q'}(a_{1k}) = \alpha_1$.

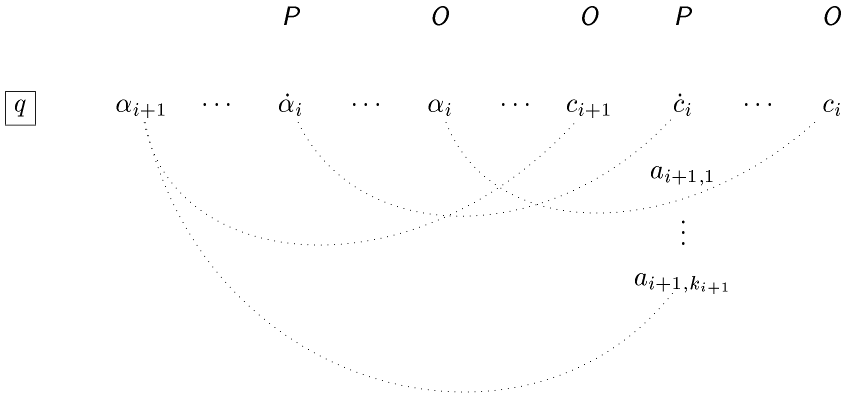
Recursion step. *Suppose a_{ij}, α_i, c_i and k_i have been defined for some $i \geq 1$. If $\alpha_i \in H$, $\mathsf{source}(c_i) \in \overline{\mathsf{act}}(p)$, or $\mathsf{source}(\alpha_i) \in \mathsf{Hol}(p)$, then stop. Otherwise $\alpha_i \in \mathsf{OHol}(q) + \mathsf{OHol}(r)$, $\mathsf{source}(\alpha_i) \in \mathsf{PHol}(q) + \mathsf{PHol}(r)$, and $\mathsf{source}(c_i) \in \mathsf{act}(q) + \mathsf{act}(r)$. Let $\dot{c}_i = \mathsf{source}(c_i)$ and $\dot{\alpha} = \mathsf{source}(\alpha_i)$. Without loss of generality, assume $\dot{c}_i \in \mathsf{act}(q)$. (For $\dot{c}_i \in \mathsf{act}(r)$ switch q and r in the passage below.) By inductive hypothesis, $c_i \in \mathsf{Oact}(q)$ is the last move of an O-hypermove, and $\alpha_i = \mathsf{ref}_q(c_i)$. There are three cases: (1) $\dot{c}_i \in \mathsf{act}(r)$; (2) $\dot{c}_i \in \mathsf{act}(q)$ and $\mathsf{ref}_q(\dot{c}_i) \in \mathsf{PHol}(q)$; (3) $\dot{c}_i \in \mathsf{act}(q)$ and $\mathsf{ref}_q(\dot{c}_i) \in \mathsf{OHol}(q)$.*

(1) $\dot{c}_i \in \mathsf{act}(r)$. Thus $\mathsf{depth}(c_i) = \mathsf{depth}(\dot{c}_i) = 1$, $\dot{\alpha}_i \in \mathsf{PHol}(r)$, and $\mathsf{ref}_r(\dot{c}_i) = \dot{\alpha}_i$.



Define k_{i+1} to be the number of actions below $\dot{c}_i$, and define $a_{i+1,j}$ to be the j^{th} such action ($k_{i+1} > 0$ since $\dot{c}_i$ references a P-hole.) Define c_{i+1} to be the last O-action of the O-hypermove preceding (the P-hypermove containing) $\dot{c}_i$. (In the diagram c_{i+1} need not be at the same depth as $\dot{c}_i$.) Define $\alpha_{i+1} = \mathsf{ref}_r(c_{i+1}) = \mathsf{ref}_r(a_{i+1,k_{i+1}})$ (the latter equality holding because of the copycat rule).

(2) $\dot{c}_i \in \mathsf{act}(q)$ and $\mathsf{ref}_q(\dot{c}_i) \in \mathsf{PHol}(q)$. Thus $\mathsf{depth}(c_i) = \mathsf{depth}(\dot{c}_i) > 1$, $\dot{\alpha}_i \in \mathsf{PHol}(q)$, and $\mathsf{ref}_q(\dot{c}_i) = \dot{\alpha}_i$.



($\dot{\alpha}_i$, α_i, $\dot{c}_i$ and c_i are in the same row, but α_{i+1} and c_{i+i} need not be in that row. In particular, $\alpha_{i+1} \in H$ is possible.) The definitions are exactly the same as in case (1). Define k_{i+1} to be the number of actions below $\dot{c}_i$, and define $a_{i+1,j}$ to be the j^{th} such action. (Note that $k_{i+1} > 0$ since $\dot{c}_i$ references a P-hole.) Define c_{i+1} to be the last O-action of the O-hypermove preceding (the P-hypermove containing) $\dot{c}_i$. Define $\alpha_{i+1} = \mathsf{ref}_r(c_{i+1}) = \mathsf{ref}_r(a_{i+1,k_{i+1}})$ (the latter equality holding because of the copycat rule).

(3) $\dot{c}_i \in \mathsf{act}(q)$ *and* $\mathsf{ref}_q(\dot{c}_i) \in \mathsf{OHol}(q)$. *Thus* $\mathsf{ref}_q(\dot{c}_i) = \mathsf{ref}_q(c_i) = \alpha_i$, *and there are no actions below* $\dot{c}_i$.

$$\begin{array}{ccccccccc} & & \mathsf{O} & & \mathsf{O} & \mathsf{P} & & \mathsf{O} \\ \boxed{q} & \ldots & \alpha_i & \ldots & c_{i+1} & \dot{c}_i & \ldots & c_i \end{array}$$

Define $k_{i+1} = 0$, $\alpha_{i+1} = \alpha_i$, *and define* c_{i+1} *to be the last* O*-action of the* O*-hypermove preceding (the* P*-hypermove containing)* $\dot{c}_i$. *(So* $\mathsf{ref}_q(c_{i+1}) = \alpha_i$, *by the copycat rule.)*

Let $l \geq 1$ *be the maximum* i *defined by the recursion above (which must terminate, because* $c_{i+1} \prec c_i$*). Define* s *to be the column obtained from the sequence*

$$a_{11} \cdots a_{1k_1} a_{21} \cdots a_{2k_2} \cdots a_{l1} \cdots a_{lk_l}$$

of all the a_{ij} *contructed in the recursion by inserting a 'linebreak' before every* a_{ij} *with* $j \geq 2$. *For example, given*

$$a_{11}a_{12}a_{13}a_{21}a_{22}a_{23}a_{31}a_{41}a_{42}a_{43}$$

s *is*

$$\begin{array}{c} a_{11} \\ a_{12} \\ a_{13}a_{21} \\ a_{22} \\ a_{23}a_{31}a_{41} \\ a_{42} \\ a_{43} \end{array}$$

Define $\tilde{a}_{ij}$ *to be the* j^{th} *action of the* i^{th} *row of* s, *let* y *be the number of rows of* s, *and let* x_i *be the number of elements of row* i. *For example, with* s *as above,* $\tilde{a}_{52} = a_{31}$, $y = 7$, *and* $x_5 = 3$. *Note that* $\tilde{a}_{11}$ *is an occurrence of* a, *the action of* P*'s move* (D, ϕ, a) *in* q'.

The column s *will constitute the live atomic actions of a* P*-hypermove in* p, *the 'team-response' of* P *in* q *and* P *in* r. *We break into two cases: A.* $\dot{c}_l \notin \overline{\mathsf{act}}(p)$, *and B.* $\dot{c}_l \in \overline{\mathsf{act}}(p)$.

A. $\dot{c}_l \notin \overline{\mathsf{act}}(p)$, *i.e.,* $\dot{c}_l \in \mathsf{act}(q) + \mathsf{act}(r)$. *Recall that* $\dot{b} = \mathsf{source}(b)$ *is a live atomic action of the* O*-move*

$$m = (E^*, \underline{*}, \zeta_1 \dot{b} \eta_1[b_1] \ldots \eta_w[b_w]).$$

Add the following hypermove ν *to* p, *with its first move justified by* m*:*

$$\begin{array}{c} (E^*,\, \theta_1,\, \zeta_1 \tilde{a}_{11}\, \eta_1[b_1]\tilde{a}_{12} \ldots \eta_{x_1-1}[b_{x_1-1}]\tilde{a}_{1x_1}\ \eta_{x_1}[b_{x_1}] \ldots \eta_w[b_w]) \\ (L_2,\, \theta_2,\, \tilde{a}_{21} \ldots \tilde{a}_{2x_2}) \\ (L_3,\, \theta_3,\, \tilde{a}_{31} \ldots \tilde{a}_{3x_3}) \\ \vdots \\ (L_y,\, \theta_y,\, \tilde{a}_{y1} \ldots \tilde{a}_{yx_y}) \end{array}$$

where the locations L_i and stores θ_i are is given as follows. Write $n_{ij} = (D_{ij}, \phi_{ij}, \tilde{a}_{ij})$ for the move containing the action $\tilde{a}_{ij}$. For $i = 2, \ldots, y$ define

$$L_i = D_{i1}^*,$$

the full expansion of the location of $\tilde{a}_{ij}$. Given an element e of a sequence t, recall that we define $t_{<e}$ and $t_{>e}$ by $t = t_{<e}\, e\, t_{>e}$. For $i = 1, \ldots, y$ let d^i be the action of the i^{th} move of ν, and for $j = 1, \ldots, x_i$ and $\beta \in \mathsf{Hol}_{D_{ij}}(\tilde{a}_{ij})$ define

$$\theta_i(d^i_{<\tilde{a}_{ij}}\ \beta\ d^i_{>\tilde{a}_{ij}}) = \phi_{ij}(\beta)^*,$$

the full expansion of the arena stored in β. (At this point it might be useful to look at section 6.1.4, about the holes of compound actions.)

Define p' to be the extension of p by ν as above, and put

$$(p', q', r, 1, \mathsf{source}') \in \mathsf{Int}(A, B, C),$$

where source' extends source with the addition of $\mathsf{source}'(\tilde{a}_{i,j}) = \tilde{a}_{i,j}$, and where the occurrence of $\tilde{a}_{ij}$ on the left is in ν, and the occurrence of $\tilde{a}_{ij}$ on the right is the original copy in the move n_{ij} of q or r. The hypersequences p' and q' now look like this:

$$\begin{array}{ccccc}
 & & \mathsf{O} & & \mathsf{P} \\
\boxed{p'} & \ldots & (E^*, \underline{*}, \zeta_1 \dot{b} \zeta_2) & \ldots & (E^*, \theta_1, \zeta_1 \dot{a} \hat{\zeta}_2) \\
 & & & & (L_2, \theta_2, s_2) \\
 & & & & \vdots \\
 & & & & (L_y, \theta_y, s_y)
\end{array}$$

$$\begin{array}{ccccc}
 & & \mathsf{O} & & \mathsf{P} \\
\boxed{q'} & \ldots & (D, \underline{*}, b) & \ldots & (D, \phi, a) \\
 & & & & \vdots \\
 & & & & m_k
\end{array}$$

where $\dot{a} = \tilde{a}_{11} = a_{11}$, a copy of a.

B. *$\dot{c}_l \in \overline{\mathsf{act}}(p)$. Proceed exactly as in the previous case, but then extend the last compound action s_y to form p'' from p' as follows:*

$$
\begin{array}{cccccccc}
 & & & \mathsf{O} & & \mathsf{O} & & \mathsf{P} \\
\boxed{p''} & & \cdots & (E^*, \underline{*}, \zeta_1 \dot{b} \zeta_2) & \cdots & (G^*, \underline{*}, \omega_1 \dot{c}_l \omega_2) & \cdots & (E^*, \theta_1, \zeta_1 \dot{a} \hat{\zeta}_2) \\
 & & & & & & & (L_1, \theta_2, s_2) \\
 & & & & & & & \vdots \\
 & & & & & & & (L_{h-1}, \theta_{h-1}, s_{h-1}) \\
 & & & & & & & (L'_y, \theta'_y, s_y \widetilde{\omega_2})
\end{array}
$$

$$
\begin{array}{cccccccccccc}
 & & \mathsf{O} & & & & \mathsf{O} & \mathsf{P} & & \mathsf{P} \\
\boxed{q'} & \cdots & (D, \underline{*}, b) & \cdots & \alpha_l & \cdots & \vdots & \vdots & \cdots & (D, \phi, a) \\
 & & & & & & c_l & a_{l,k_l} & & m_2 \\
 & & & & & & & & & \vdots \\
 & & & & & & & & & m_k
\end{array}
$$

The source $\dot{c}_l$ of c_l is part of a compound action $\omega_1 \dot{c}_l \omega_2$ in p. Here $\widetilde{\omega_2}$ is (a fresh copy of) the subsequence of ω_2 consisting in its live atomic actions. Given a live atomic action e of ω_2, write $\dot{e}$ for the corresponding occurrence of e in $\widetilde{\omega_2}$. Extend the store θ_y associated with s_y to form θ'_y as follows. Let $\omega = \omega_1 \dot{c}_l \omega_2$ and $\widetilde{\omega} = s_y \widetilde{\omega_2}$. For live $e \in \omega_2$ and $\beta \in \mathsf{Hol}(e)$ define

$$\theta'_y(\widetilde{\omega}_{<\dot{e}}\, \beta\, \widetilde{\omega}_{>\dot{e}}) \quad = \quad \langle \omega_{<e}\, \beta\, \omega_{>e}, \bullet \rangle,$$

the arena with the single action $\bullet$ referencing the global hole $\omega_{<e}\, \beta\, \omega_{>e}$. Finally, we have two cases to consider: (1) $\mathsf{ref}_p(\omega_1 \dot{c}_l \omega_2) = \widetilde{\omega}_{<\dot{e}}\, \beta\, \widetilde{\omega}_{>\dot{e}}$, or (2) otherwise. We consider (2) first.

(2) Thus $\mathsf{ref}_{p''}(s_y \widetilde{\omega_2})$ is not a P-hole, so ν completes a $(p'', q, r, 1, \mathsf{source}'')$ to $\mathsf{Int}(A, B, C)$, where source'' extends source' of case A with $\mathsf{source}''(\dot{e}) = e$ for each live atomic action $e \in \omega_2$ (equivalently, for each $\dot{e} \in \widetilde{\omega_2}$).

(1) Thus $\mathsf{ref}_{p''}(s_y \widetilde{\omega_2}) = \widetilde{\omega}_{<\dot{e}} \beta \widetilde{\omega}_{>\dot{e}}$, which (by construction) stores the singleton arena $\langle \omega_{<e} \beta\, \omega_{>e}, \bullet \rangle$, so we duly open a thread, forming p''' from p'':

$$\begin{array}{ccccccc}
 & & O & & O & & P \\
\boxed{p'''} & \cdots & (E^*, \underline{*}, \zeta_1 \dot{b} \zeta_2) & \cdots & (G^*, \underline{*}, \omega_1 \dot{c}_l \omega_2) & \cdots & (E^*, \theta_1, \zeta_1 \dot{a} \hat{\zeta}_2) \\
 & & & & & & (L_1, \theta_2, s_2) \\
 & & & & & & \vdots \\
 & & & & & & (L_{h-1}, \theta_{h-1}, s_{h-1}) \\
 & & & & & & (L'_y, \theta'_y, s_y \widetilde{\omega_2}) \\
 & & & & & & (\langle \omega_{<e}\, \beta\, \omega_{>e}, \bullet \rangle, \emptyset, \bullet)
\end{array}$$

$$\begin{array}{ccccccccccc}
 & & O & & & & O & P & & P \\
\boxed{q'} & \cdots & (D, \underline{*}, b) & \cdots & \alpha_l & \cdots & \vdots & \vdots & \cdots & (D, \phi, a) \\
 & & & & & & c_l & a_{l,k_l} & & m_2 \\
 & & & & & & & & & \vdots \\
 & & & & & & & & & m_k
\end{array}$$

Since $\bullet$ references $\omega_{<e}\, \beta\, \omega_{>e}$, an O-hole, this completes a hypermove. Put $(p''', q, r, 1, \mathsf{source}'') \in \mathsf{Int}(A, B, C)$, where source'' is as in case (2) above.

2. $\mathsf{next} = 3$. *This case is exactly as* $\mathsf{next} = 2$, *but with q and r exchanged, modulo the special case of an r-move in an A component of $A \Rightarrow B$ justified by an opening $A \Rightarrow B$ move. Then (similar to Hyland-Ong interaction) realign the justification pointer of the transmited move to point to the very first move of p.*

3. $\mathsf{next} = 1$. *Thus O is to move in p. Let $p' \in \mathsf{Pos}(A \Rightarrow C)$ be a position extending p by a hypermove μ. So p has the following shape:*

$$\begin{array}{ccc}
P & & O \\
\vdots & & \\
(E^*, \theta, \zeta b \zeta') & & (E^*, \underline{*}, \dot{\zeta} a \zeta'') \\
\vdots & & (F^*, \underline{*}, c_1 \ldots c_k)
\end{array}$$

where $\dot{\zeta}$ is an occurrence of ζ ‘dotted’ in order to distinguish it from the original, and the c_i are the atomic actions of the compound action $c_1 \ldots c_k$ constituting the second move of μ. (By Lemma 5.13 μ consists in at most two moves, and we code the case of μ a singleton with $k = 0$.) We have distinguished the live atomic actions b and a within the compound actions as those of the live atomic justification $b \curvearrowleft a$ behind the justification $\zeta b \zeta' \curvearrowleft \dot{\zeta} a \zeta''$.

Let $\dot{b} = \mathsf{source}(b)$. There are three cases: (a) $\dot{b} \in \mathsf{act}(q)$, (b) $\dot{b} \in \mathsf{act}(r)$, and (c) $\dot{b} \in \overline{\mathsf{act}}(p)$. (Note that $\dot{b} = \mathsf{dummy}$ (i.e., $b = \bullet$) is impossible, since $\bullet$, being the action of a singleton arena, could not justify another action.)

(a) $\dot{b} \in \mathsf{act}(q)$. *Let* $\dot{m} = (D, \phi, \dot{b})$ *be the move containing* $\dot{b}$. *We copy* a *as a move* $\dot{m}_1 = (D, \underline{*}, \dot{a})$ *justified by* $\dot{m}$, *forming* q'*:*

$$
\boxed{p'} \qquad
\begin{array}{cccc}
 & & \mathsf{P} & \mathsf{O} \\
\cdots & & \vdots & \\
 & & (E^*, \theta, \zeta b \zeta') & (E^*, \underline{*}, \dot{\zeta} a \zeta'') \\
 & & \vdots & (F^*, \underline{*}, c_1 \ldots c_k)
\end{array}
$$

$$
\boxed{q'} \qquad
\begin{array}{cccc}
 & \mathsf{P} & & \mathsf{O} \\
\cdots & \vdots & & \\
 & (D, \phi, \dot{b}) & \cdots & (D, \underline{*}, \dot{a}) \\
 & \vdots & &
\end{array}
$$

Set $\mathsf{T}(\dot{m}_1) = \mathsf{T}(m_1)$. *Let* $\alpha = \mathsf{ref}_q(\dot{a})$. *There are two cases: i.* $\alpha \in H + \mathsf{OHol}(q')$, *or ii.* $\alpha \in \mathsf{PHol}(q')$.

i. $\alpha \in H + \mathsf{OHol}(q')$. *Then* $\dot{m}_1 = (D, \underline{*}, \dot{a})$ *completes a hypermove for* O. *Put* $(p', q', r, 2, \mathsf{source}') \in \mathsf{Int}(A, B, C)$, *where* source' *extends* source *by setting* $\mathsf{source}(\dot{a}) = a$. *(Note incidentally that if* $\alpha \in H$ *then in fact* $\zeta'' = \zeta'$*.)*

ii. $\alpha \in \mathsf{PHol}(q')$. *Due to the conditions of being a well-formed interaction state (page 132),* $\mathsf{ref}_p(\dot{\zeta} a \zeta'')$ *is a* P*-hole* $\psi \dot{\alpha} \psi'$ *with* $\mathsf{source}(\psi \dot{\alpha} \psi') = \alpha$, *and* $\dot{\alpha}$ *a copy (occurrence) of* α. *In order for* $\dot{a}$ *to have referenced a* P*-hole,* a *must be the last live atomic action of* $\dot{\zeta} a \zeta''$ *(by construction of expansion). Let* $G = \mathsf{store}(\alpha)$. *Then by well-formedness of the interaction state* $\mathsf{store}(\psi \dot{\alpha} \psi') = G^*$, *so in fact* $G = F$*:*

$$
\boxed{p'} \qquad
\begin{array}{ccccc}
 & & & \mathsf{P} & \mathsf{O} \\
\cdots & \vdots & \cdots & \vdots & \\
 & \psi \dot{\alpha} \psi' & & (E^*, \theta, \zeta b \zeta') & (E^*, \underline{*}, \dot{\zeta} a \zeta'') \\
 & F^* & & \vdots & (F^*, \underline{*}, c_1 \ldots c_k)
\end{array}
$$

$$
\boxed{q'} \qquad
\begin{array}{cccccc}
 & & & \mathsf{P} & & \mathsf{O} \\
\cdots & \vdots & \cdots & \vdots & & \\
 & \alpha & & (D, \phi, \dot{b}) & \cdots & (D, \underline{*}, \dot{a}) \\
 & F & & \vdots & &
\end{array}
$$

Since $\mathit{ref}_{q'}(\dot{a}) = \alpha$, *O is obliged to open a thread on* F. *By definition of expansion, since* $c_1 \ldots c_k$ *is an opening (compound) action of* F^*, *we play the opening move* $(F, \underline{*}, \dot{c}_1)$ *for* $\dot{c}_1$ *a fresh occurrence of* c_1, *forming* q''*:*

$$
\begin{array}{llllll}
 & & & & P & O \\
\boxed{p'} & \cdots & \vdots & \cdots & \vdots & \\
 & & \psi\dot{\alpha}\psi' & & (E^*, \theta, \zeta b \zeta') & (E^*, \underline{*}, \dot{\zeta} a \zeta'') \\
 & & F^* & & \vdots & (F^*, \underline{*}, c_1 \ldots c_k)
\end{array}
$$

$$
\begin{array}{lllllll}
 & & & & P & & O \\
\boxed{q'} & \cdots & \vdots & \cdots & \vdots & & \\
 & & \alpha & & (D, \phi, \dot{b}) & \cdots & (D, \underline{*}, \dot{a}) \\
 & & F & & \vdots & & (F, \underline{*}, \dot{c}_1)
\end{array}
$$

This completes a hypermove for O. Put $(p', q'', r, 2, \mathit{source}') \in \mathit{Int}(A, B, C)$, *where* source' *extends* source *by setting* $\mathit{source}(\dot{a}) = a$ *and* $\mathit{source}(\dot{c}_1) = c_1$. *Set* $T(F, \underline{*}, \dot{c}_1) = T(D, \underline{*}, \dot{a})$.

(b) $\dot{b} \in \mathit{act}(r)$. *Exactly as the previous case, but with* r *in place of* q, *hence adding* $(p', q, r'', 3, \mathit{source}') \in \mathit{Int}(A, B, C)$.

(c) $\dot{b} \in \overline{\mathit{act}}(p)$. *So* $\dot{b}$ *is part of a compound action* $\xi\dot{b}\dot{\zeta}' \in \mathit{act}(p)$. *We 'copycat' O, defining* p' *as:*

$$
\begin{array}{lllll}
 & O & P & O & P \\
\boxed{p'} & (G^*, \underline{*}, \xi\dot{b}\dot{\zeta}') \cdots & (E^*, \theta, \zeta b \zeta') & (E^*, \underline{*}, \dot{\zeta} a \zeta'') & (G^*, \phi, \dot{\xi}\dot{a}\dot{\zeta}'') \\
 & & & (F^*, \underline{*}, c_1 \ldots c_k) &
\end{array}
$$

Just as ζ *and* $\dot{\zeta}$ *are copies, so* ξ *and* $\dot{\xi}$ *are copies, and* $\dot{a}$ *is a copy (fresh occurrence) of* a. *Define the store* ϕ *as follows. Since we have chosen the live atomic justification* $\dot{b}\curvearrowleft\dot{a}$ *to 'copycat' the live atomic justification* $b\curvearrowleft a$, *we have* $\zeta\dot{b}\zeta' \sim \dot{\xi}\dot{a}\dot{\zeta}''$. *Thus, by Lemma 6.9, there is an isomorphism of holes* $f : \mathit{Hol}(\dot{\xi}\dot{a}\dot{\zeta}'') \cong \mathit{Hol}(\zeta\dot{b}\zeta')$. *Define* $\phi(\beta) = \langle f(\beta), \bullet \rangle$, *the singleton arena with action* $\bullet$ *referencing the O-hole* $f(\beta)$ *of the O-move* $(E^*, \underline{*}, \dot{\zeta} a \zeta'')$. *Due to the typing of* $\mathit{ref}_{p'}$, *there are three subcases: i.* $\mathit{ref}_{p'}(\dot{\xi}\dot{a}\dot{\zeta}'') \in \mathit{PHol}(p)$, *ii.* $\mathit{ref}_{p'}(\dot{\xi}\dot{a}\dot{\zeta}'') \in \mathit{OHol}(p)$, *iii.* $\mathit{ref}_{p'}(\dot{\xi}\dot{a}\dot{\zeta}'') \in H$.

i. $\mathit{ref}_{p'}(\dot{\xi}\dot{a}\dot{\zeta}'')$ *is a P-hole, say* $\varphi_1\alpha\varphi$ *(see below). Then since we are in copycat threads,* α *has a source* $\dot{\alpha}$ *in an earlier O-hole* $\varphi_2\dot{\alpha}\dot{\varphi}$. *In particular* $k = 0$, *since* $\dot{\zeta} a \zeta''$ *references an O-hole, in other words O's hypermove consists of a single compound action* $\dot{\zeta} a \zeta''$. *Since* $\dot{\xi}\dot{a}\dot{\zeta}''$ *references a P-hole, P is required to play an opening move on the polymorphic arena inside the hole. By well-formedness of the interaction state, the stored arena is the singleton arena*

$\langle \varphi_2\dot{\alpha}\dot{\varphi}, \bullet\rangle$. So we complete a hypermove for P by playing $\bullet$, the only possible action, to form p''.

$$
\begin{array}{cccccccccccc}
 & O & & P & & O & & P & & O & & P \\
\cdots & \begin{matrix}\varphi_2\dot{\alpha}\dot{\varphi} \\ * \end{matrix} & \cdots & \begin{matrix}\varphi_1\alpha\varphi \\ \langle \bullet, \varphi_2\dot{\alpha}\dot{\varphi}\rangle \end{matrix} & \cdots & (G^*, \underline{*}, \xi\dot{b}\dot{\zeta}') & \cdots & (E^*, \theta, \dot{\zeta}b\zeta') & & (E^*, \underline{*}, \dot{\zeta}a\zeta'') & & (G^*, \phi, \dot{\xi}\dot{a}\dot{\zeta}'') \\
 & & & & & & & & & & & (\langle \bullet, \varphi_2\dot{\alpha}\dot{\varphi}\rangle, \emptyset, \bullet)
\end{array}
$$

Add $(p'', q, r, 1, \textsf{source}')$ to $\textsf{Int}(A, B, C)$, where $\textsf{source}'$ extends $\textsf{source}$ with $\textsf{source}'(\dot{a}) = a$, and $\textsf{source}'(\dot{d}) = d$ for the corresponding copies of live actions $\dot{d} \in \dot{\zeta}''$ and $d \in \zeta''$. Set $\textsf{T}(\langle \bullet, \varphi_2\dot{\alpha}\dot{\varphi}\rangle, \emptyset, \bullet) = \textsf{T}(G^*, \phi, \dot{\xi}\dot{a}\dot{\zeta}'')$.

ii. *$\textsf{ref}_{p'}(\dot{\xi}\dot{a}\dot{\zeta}'')$ is an O-hole. The situation is similar to the previous case i, but this time it is O who is obliged to play $\bullet$, rather than P. P's hypermove consists of the single action $\dot{\xi}\dot{a}\dot{\zeta}''$.*

$$
\begin{array}{cccccccccccc}
 & O & & P & & O & & P & & O & & P \\
\cdots & \begin{matrix}\varphi_2\dot{\alpha}\dot{\varphi} \\ * \end{matrix} & \cdots & \begin{matrix}\varphi_1\alpha\varphi \\ \langle \bullet, \varphi_2\dot{\alpha}\dot{\varphi}\rangle \end{matrix} & \cdots & (G^*, \underline{*}, \xi\dot{b}\dot{\zeta}') & \cdots & (E^*, \theta, \dot{\zeta}b\zeta') & & (E^*, \underline{*}, \dot{\zeta}a\zeta'') & & (G^*, \phi, \dot{\xi}\dot{a}\dot{\zeta}'') \\
 & & & & & & & & & (\langle \bullet, \varphi_2\dot{\alpha}\dot{\varphi}\rangle, \emptyset, \bullet) & &
\end{array}
$$

Let p'' be this extension of p'. Add $(p'', q, r, 1, \textsf{source}')$ to $\textsf{Int}(A, B, C)$, where $\textsf{source}'$ is as in the previous case.

iii. *$\textsf{ref}_{p'}(\dot{\xi}\dot{a}\dot{\zeta}'')$ is a global hole $\alpha \in H$:*

$$
\begin{array}{cccccccccc}
\alpha & & & O & & P & & O & & P \\
 & & \cdots & (G^*, \underline{*}, \xi\dot{b}\dot{\zeta}') & \cdots & (E^*, \theta, \dot{\zeta}b\zeta') & & (E^*, \underline{*}, \dot{\zeta}a\zeta'') & & (G^*, \phi, \dot{\xi}\dot{a}\dot{\zeta}'')
\end{array}
$$

In this case both O and P's last move complete hypermoves. Let p'' be this extension of p'. Add $(p'', q, r, 1, \textsf{source}')$ to $\textsf{Int}(A, B, C)$, where $\textsf{source}'$ is as in case i.

6.2 Interaction is well-defined

In this section we prove that the interaction algorithm is well-defined.

Technically, the following theorem is the cornerstone of the thesis, and is the culmination of all the auxiliary concepts in earlier chapters and sections, such as expansion, live atomic enabling, match, source, and so on. Due to the level of detail in the definition of the interaction algorithm, a lot of the work has already been done. The key case below is 1.(c).ii.A, the 'team-response' in p of P in q and P in r.

THEOREM 6.16 *Interaction is well-defined. In other words, every sourced interaction state $\rho = (p, q, r, \mathsf{next}, \mathsf{source})$ added to the set $\mathsf{Int}(A, B, C)$ of valid interaction states in Definition 6.15 is well-formed.*

Proof By induction on length $|\rho|$, and by case analysis on the algorithm. The case enumerations below correspond to the cases in the algorithm. The base cases $|\rho| = 0$ and $|\rho| = 1$ are clearly well-formed. Througout the sequel, "condition i" of an unqualified definition means "condition i of the definition of well-formed interaction state". (The definition of well-formed interaction state is on page 132.)

1. (a) Because $m \curvearrowleft m_1$, we have $b \vdash_D a$, so $(D, \psi, \dot{b}) \curvearrowleft m_1^*$ respects enabling and location. Since m_1^* is an O-move, it stores no arenas, so the scope condition of being a hypersequence is satisfied.

   i. Condition 1 of well-formed interaction state is trivial. Condition 2 is verified since $b \overline{\curvearrowleft} a$ and $\dot{b} \overline{\curvearrowleft} \dot{a}$, with $\dot{b} = \mathsf{source}(b)$ and $a = \mathsf{source}(\dot{a})$. Condition 3 is inherited from ρ, since p is left unchanged. Condition 4 does not apply to a, the m_i, or b, since each are of depth strictly greater than 1.

   ii. We have to check conditions for the second O-move $(E, \underline{*}, d)$.

      A. All events take place in q'' alone, still at depth greater than 1, so all that remains is to verify condition 2. Condition 2 holds since $\mathsf{source}(\dot{e}) = e$ and both e and $\dot{e}$ are unjustified.

      B. Just as for A., the only condition of concern is condition 2. The interaction state now involves a move of p, since this time $\dot{e} \in \overline{\mathsf{act}}(p)$. Since $\dot{e}$ is unjustified, we must verify that the live atomic action e of $d = \zeta\dot{c}\eta\zeta''$ is not live atomic justified. Because $\dot{c}$ is a live atomic action, e is not the first live atomic action of d. By definition of the live atomic actions of compound actions (section 4.4.3), only the first live atomic action of a sequence can be live atomic justified. Hence e is not live atomic justified.

   (b) Since $\mathsf{depth}(d_c) = \mathsf{depth}(e_c) = \mathsf{depth}(\dot{d}) = \mathsf{depth}(\dot{e}) = 1$, condition 1 is satisfied. Condition 2 does not apply to the new moves of q' and r' since $k = 0$.

   i. The addition of the copied O-move $(E, \underline{*}, \dot{g})$ is analogous to case (a)ii.A.

   ii. The addition of the copied O-move $(E, \underline{*}, \dot{g})$ is analogous to case (a)ii.B.

   iii. By construction, this case of the algorithm reduces to case A or case B (by 'backchaining' along the l_i and $\dot{l}_i$). Thus well-formedness is inherited from A and B. (Note how crucial the copycat condition is in this step of the algorithm, which ensures that every action of the 'chain' reference β_c.)

   (c) i. This case is familiar from first-order (λ-calculus) interaction. The only addition to ρ is the unjustified opening move $(A \Rightarrow B, \underline{*}, \dot{e})$ of $A \Rightarrow B$. Since $\mathsf{depth}(\dot{e}) = \mathsf{depth}(e) = 1$, conditions 1 and 2 are satisfied. Condition 3 does not apply, since no holes have been added to $\mathsf{PHol}(p)$. Condition 4(a) holds since we defined $\mathsf{T}(\dot{m}_1) = \mathsf{T}(m_1)$. Condition 4(b) holds since $\dot{m}_1$ is unjustified. Conditions 4(c) and 4(d) do not apply.

ii. A. First we must show that the hypermove ν is well-defined, i.e., that $p' = p\nu \in \mathsf{Pos}(A \Rightarrow C)$. We start by showing that

$$\zeta_1 \dot{b} \zeta_2 \quad \vdash_{E^*} \quad \zeta_1 \dot{a} \hat{\zeta}_2, \tag{6.1}$$

i.e., that the first action of ν is justified by $\zeta_1 \dot{b} \zeta_2$ in the location E^*. In particular, $\zeta_1 \dot{a} \hat{\zeta}_2$ is indeed an action of E^*, which is not obvious. By inductive hypothesis, since p is a well-formed interaction state, E^* is the location of the justifier $\zeta_1 \dot{b} \zeta_2$ of $\zeta_1 \dot{a} \hat{\zeta}_2$. By definition of full expansion (page 131),

$$E^* = \underline{\mathsf{rename}}((\mathsf{lookup}^*)^{w'}(E)),$$

with rename in the definition determined by source on $\mathsf{OHol}(q) + \mathsf{OHol}(r)$, and $w' \geq w$, where w is the index appearing in

$$\zeta_1 \dot{b} \zeta_2 = \zeta_1 \dot{b} \eta_1[b_1] \ldots \eta_w[b_w].$$

Recall the definition of enabling for compound actions (page 59):

$$\begin{array}{lllcl}
f\eta & \vdash_{\phi^*(K)} & f'\eta' & \Leftrightarrow & f \vdash_K f' \text{ and } \eta = \eta' \\
f\eta & \vdash_{\phi^*(K)} & f'\eta' g' & \Leftrightarrow & f \vdash_K f' \text{ and } \eta = \eta' \text{ and } \vdash g' \\
f\eta g & \vdash_{\phi^*(K)} & f'\eta' & \Leftrightarrow & f \vdash_K f' \text{ and } \eta g = \eta' \text{ and } \vdash g \\
f\eta g & \vdash_{\phi^*(K)} & f'\eta' g' & \Leftrightarrow & \begin{cases} f \vdash_K f',\ \eta g = \eta',\ \vdash g \text{ and } \vdash g', & \text{or} \\ g \vdash g' \text{ and } f\eta = f'\eta' & \end{cases}
\end{array}$$

In order to show

$$\zeta_1 \dot{b} \zeta_2 \quad \vdash_{E^*} \quad \zeta_1 \dot{a} \hat{\zeta}_2,$$

we start by showing

$$\zeta_1 \dot{b} \quad \vdash_{\mathsf{lookup}^*(E_0)} \quad \zeta_1 \dot{a},$$

where $E_0 = (\mathsf{lookup}^*)^{w''}(E)$ and $w'' = w' - w - 1$. (If $\zeta_1 = \epsilon$ then trivially $\zeta_1 \dot{b} \vdash \zeta_1 \dot{a}$ since $b \vdash a$, and our arguments later still apply.)

First we must show that $\zeta_1 \dot{a}$ is indeed an action of $\mathsf{lookup}^*(E_0)$. Recall the definition of the actions of an expanded arena (page 59):

$$\begin{array}{rcl}
\mathsf{Act}_{\phi^*(K)} & = & \{\ f\eta\ :\ \eta \in \mathsf{trails}_\phi(f) \text{ and } \mathsf{ref}_K(f) \notin \mathsf{dom}(\phi)\ \} \quad \cup \\
& & \{f\eta g\ :\ \eta \in \mathsf{trails}_\phi(f),\ \mathsf{ref}_K(f) \in \mathsf{dom}(\phi), \\
& & \qquad\qquad \text{and } g \in \mathsf{Act}_{\phi(\mathsf{ref}_K(f))}\}
\end{array}$$

Since $(E^*, \underline{*}, \zeta_1 \dot{b} \zeta_2)$ is in p, by inductive hypothesis $\zeta_1 \dot{b} \zeta_2$ is an action of E^*, and so $\zeta_1 \dot{b}$ is an action of $\mathsf{lookup}^*(E_0)$. Since $\dot{b} = \mathsf{source}(b)$, by the typing of source the atomic action $\dot{b}$ is a live atomic action. Hence $\zeta_1 \dot{b}$ arises as an action of $\mathsf{lookup}^*(E_0)$ via the second set in the union, i.e., with $K = E_0$, $\phi = \mathsf{lookup}$, $\zeta_1 = f\eta$, and $g = \dot{b}$. Thus by taking $g = \dot{a}$ in the same expression, we obtain $\zeta_1 \dot{a}$ as an action of $\mathsf{lookup}^*(E_0)$, as required.

Since $(D, \underline{*}, b) \curvearrowleft (D, \phi, a)$ in q' (and q' is by induction hypothesis a play), we have $b \vdash_D a$. Thus by the fourth case in the definition of enabling quoted above, with $f\eta = f'\eta' = \zeta_1$, $g = \dot{b}$, and $g' = \dot{a}$ (and hence with the second clause of the "or" on the right hand side),

$$\zeta_1 \dot{b} \quad \vdash_{\mathsf{lookup}^*(E_0)} \quad \zeta_1 \dot{a}.$$

Recall that

$$\begin{aligned} \zeta_1 \dot{b} \zeta_2 &= \zeta_1 \dot{b} \eta_1 [b_1] \ldots \eta_w [b_w] \\ \zeta_1 \dot{a} \hat{\zeta}_2 &= \zeta_1 \tilde{a}_{11}\, \eta_1 [b_1] \tilde{a}_{12} \ldots \eta_{x_1 - 1} [b_{x_1 - 1}] \tilde{a}_{1x_1}\ \eta_{x_1} [b_{x_1}] \ldots \eta_w [b_w] \end{aligned}$$

For $j = 0, \ldots, w$ define σ_j and $\widetilde{\sigma}_j$ by

$$\begin{aligned} \zeta_1 \dot{b} \zeta_2 &= \sigma_j \eta_{j+1} \ldots [b_w] \\ \zeta_1 \dot{a} \widetilde{\zeta}_2 &= \widetilde{\sigma}_j\, \eta_{j+1} \ldots [b_w] \end{aligned}$$

(So $\sigma_0 = \zeta_1 \dot{b}$ and $\widetilde{\sigma}_0 = \zeta_1 \dot{a}$.) Define $E_j = (\mathsf{lookup}^*)^j (E_0)$. Having just shown that $\sigma_0 \vdash_{\mathsf{lookup}^*(E_0)} \widetilde{\sigma}_0$, we now show that

$$\sigma_{j-1} \vdash_{\mathsf{lookup}^*(E_{j-1})} \widetilde{\sigma}_{j-1} \quad \Rightarrow \quad \sigma_j \vdash_{\mathsf{lookup}^*(E_j)} \widetilde{\sigma}_j, \tag{6.2}$$

hence by induction

$$\sigma_w \vdash_{\mathsf{lookup}^*(E_w)} \widetilde{\sigma}_w$$

i.e,

$$\zeta_1 \dot{b} \zeta_2 \quad \vdash_{E^*} \quad \zeta_1 \dot{a} \widetilde{\zeta}_2,$$

our objective 6.1.
Suppose

$$\sigma_{j-1} \vdash_{\mathsf{lookup}^*(E_{j-1})} \widetilde{\sigma}_{j-1}.$$

Our goal is to deduce

$$\sigma_j \quad \vdash_{\mathsf{lookup}^*(E_j)} \quad \widetilde{\sigma}_j. \tag{6.3}$$

We start by showing that

$$\sigma_j, \widetilde{\sigma}_j \quad \in \quad \mathsf{lookup}^*(E_j) \tag{6.4}$$

Since $\zeta_1 \dot{b} \zeta_2$ is in p, $\sigma_j \in \mathsf{lookup}^*(E_j)$. By inductive hypothesis $\widetilde{\sigma}_{j-1} \in E_j = \mathsf{lookup}^*(E_{j-1})$. We show that

$$\widetilde{\sigma}_j \quad \in \quad \mathsf{lookup}^*(E_j)$$

by appealing once more to the definition of the actions of an expanded arena:

$$\begin{aligned} \mathsf{Act}_{\mathsf{lookup}^*(E_j)} \quad = \quad & \{\, f\eta \ :\ \eta \in \mathsf{trails}_{\mathsf{lookup}}(f) \text{ and } \mathsf{ref}_{E_j}(f) \notin \mathsf{dom}(\mathsf{lookup}) \,\} \\ \cup \quad & \{ f\eta g \ :\ \eta \in \mathsf{trails}_{\mathsf{lookup}}(f),\ \mathsf{ref}_{E_j}(f) \in \mathsf{dom}(\mathsf{lookup}), \\ & \qquad \text{and } g \in \mathsf{Act}_{\mathsf{lookup}(\mathsf{ref}_{E_j}(f))} \} \end{aligned}$$

Define $\widetilde{\alpha}_{ij} = \mathit{ref}(\widetilde{a}_{ij})$. We shall require the following facts:

$$\begin{aligned}
\widetilde{\alpha}_{1j} &\in \mathit{OHol}(q) + \mathit{PHol}(r) && (6.5)\\
\mathit{ref}(\widetilde{a}_{1j}) &\text{ is } \text{global} && (6.6)\\
\mathit{source}(\widetilde{\alpha}_{1j}) &\in \mathit{PHol}(q) + \mathit{PHol}(r) && (6.7)\\
\widetilde{\alpha}_{1x_1} &\in \mathit{PHol}(\widetilde{a}_{1x_1}) && (6.8)
\end{aligned}$$

where $1 \leq j < x_1$. (We relegate the proofs to the end of this section.)

We consider two cases: (1) $j \leq x_1$, and (2) $j > x_1$.

(1) $j \leq x_1$. So by definition of $\widetilde{\sigma}_j$,

$$\widetilde{\sigma}_j \quad = \quad \widetilde{\sigma}_{j-1}\eta_j[b_j]\widetilde{a}_{1j}.$$

By definition of references in an expansion (page 60),

$$\mathit{ref}_{E_j}(d\eta) \quad = \quad \begin{cases} \mathit{ref}(d)\,\eta' & \text{if } \mathit{ref}(d) \text{ is local} \\ \mathit{ref}(d) & \text{if } \mathit{ref}(d) \text{ is global} \end{cases}$$

$$\mathit{ref}_{E_j}(d\eta e) \quad = \quad \begin{cases} d\,\eta\,\mathit{ref}(e) & \text{if } \mathit{ref}(e) \text{ is local} \\ \mathit{ref}(e) & \text{if } \mathit{ref}(e) \text{ is global} \end{cases}$$

where $\eta' = [\eta, \mathit{act}_{E_{j-1}}(\mathit{ref}(d)), d]$ (defined on page 60). Since

$$\widetilde{\sigma}_{j-1} \quad = \quad \widetilde{\sigma}_{j-2}\eta_{j-1}[b_{j-1}]\widetilde{a}_{1(j-1)},$$

taking $d = \widetilde{\sigma}_{j-2}$, $\eta = \eta_{j-1}[b_{j-1}]$ and $e = \widetilde{a}_{1(j-1)}$ in the definition of reference, we obtain

$$\mathit{ref}_{E_j}(\widetilde{\sigma}_{j-1}) \quad = \quad \mathit{ref}(\widetilde{a}_{1(j-1)})$$

In the conditional of the definition of reference, the bottom case ($\mathit{ref}(\widetilde{a}_{1(j-1)})$ is global) applies, since earlier we defined

$$\widetilde{\alpha}_{1(j-1)} = \mathit{ref}(\widetilde{a}_{1(j-1)}),$$

and $\widetilde{a}_{1(j-1)}$ is global by fact 6.6.

Recall that

$$\begin{aligned}
\mathit{Act}_{\mathit{lookup}^*(E_j)} \quad = \quad & \{\, f\eta \;:\; \eta \in \mathit{trails}_{\mathit{lookup}}(f) \text{ and } \mathit{ref}_{E_j}(f) \notin \mathit{dom}(\mathit{lookup}) \,\} \\
\cup \quad & \{ f\eta g \;:\; \eta \in \mathit{trails}_{\mathit{lookup}}(f),\ \mathit{ref}_{E_j}(f) \in \mathit{dom}(\mathit{lookup}), \\
& \qquad \text{and } g \in \mathit{Act}_{\mathit{lookup}(\mathit{ref}_{E_j}(f))}\}
\end{aligned}$$

By taking $f = \widetilde{\sigma}_{j-1}$, $\eta = \eta_j[b_i]$, and $g = \widetilde{a}_{1j}$, since

$$\widetilde{\sigma}_j \quad = \quad \widetilde{\sigma}_{j-1}\eta_j[b_j]\widetilde{a}_{1j},$$

we obtain $\widetilde{\sigma}_j \in \mathit{Act}_{\mathit{lookup}^*(E_j)}$ as required. Note that

$$\mathit{ref}_{E_j}(f) \in \mathit{dom}(\mathit{lookup})$$

holds because

$$\mathit{OHol}(q) + \mathit{PHol}(r) \subseteq \mathit{dom}(\mathit{lookup})$$

and (as we showed just above)

$$\mathit{ref}_{E_j}(f) = \widetilde{\alpha}_{1(j-i)}$$

and by fact 6.5,

$$\widetilde{\alpha}_{1(j-i)} \in \mathit{OHol}(q) + \mathit{PHol}(r).$$

The condition

$$g = \widetilde{a}_{1j} \in \mathit{Act}_{\mathit{lookup}(\mathit{ref}_{E_j}(f))}$$

holds because of the following argument. Since

$$\mathit{ref}_{E_j}(f) = \widetilde{\alpha}_{1(j-i)}$$

we have

$$\mathit{lookup}(\mathit{ref}_{E_j}(f)) = \mathit{lookup}(\widetilde{\alpha}_{1(j-1)}).$$

By definition of *lookup* (page 130), since $\widetilde{\alpha}_{1(j-1)} \in \mathit{OHol}(q)$,

$$\mathit{lookup}(\widetilde{\alpha}_{1(j-1)}) \quad = \quad \mathit{store}(\mathit{source}(\widetilde{\alpha}_{1(j-1)})).$$

Now because of the 'linebreak' construction, $\widetilde{a}_{1(j-1)} = a_{i_j 1}$ for some $i_j \geq 1$, and (refer to the two diagrams following "**Recursion step**" of page 146) by definition $a_{i_j 1}$ is immediately below $\dot{c}_{i_j}$. Since $\mathit{ref}(\dot{c}_{i_j}) = \dot{\alpha}_{i_j}$, and by definition $\dot{\alpha}_{i_j} = \mathit{source}(\alpha_{i_j})$ and $\alpha_{i_j} = \widetilde{\alpha}_{1(j-1)}$, we have

$$\mathit{ref}(\dot{c}) = \mathit{source}(\widetilde{\alpha}_{1(j-1)}).$$

Thus $\widetilde{a}_{1(j-1)}$ is an opening action of $\mathit{lookup}(\mathit{ref}_{E_j}(f))$, so in particular

$$\widetilde{a}_{1j} \in \mathit{Act}_{\mathit{lookup}(\mathit{ref}_{E_j}(f))}$$

as required. Thus we have completed case (1) $j \leq x_1$ of subgoal 6.4.

(2) $j > x_1$. This is analogous to case (1). Instead of using the $f\eta g$ component of $\mathit{Act}_{\mathit{lookup}^*(E_j)}$, we use the $f\eta$ component. Fact 6.8 plays the same role in the verification that $f\eta \in \mathit{Act}_{\mathit{lookup}^*(E_j)}$ as fact 6.5 played in (1) in verifying $f\eta g \in \mathit{Act}_{\mathit{lookup}^*(E_j)}$.

So we have completed subgoal 6.4. To complete goal 6.1, all that remains is to verify 6.2:

$$\sigma_{j-1} \vdash_{\mathit{lookup}^*(E_{j-1})} \widetilde{\sigma}_{j-1} \quad \Rightarrow \quad \sigma_j \vdash_{\mathit{lookup}^*(E_j)} \widetilde{\sigma}_j.$$

Recall once again the definition of enabling of compound actions:

$$\begin{array}{llcl}
f\eta & \vdash_{\phi^*(K)} \; f'\eta' & \Leftrightarrow & f \vdash_K f' \text{ and } \eta = \eta' \\
f\eta & \vdash_{\phi^*(K)} \; f'\eta' g' & \Leftrightarrow & f \vdash_K f' \text{ and } \eta = \eta' \text{ and } \vdash g' \\
f\eta g & \vdash_{\phi^*(K)} \; f'\eta' & \Leftrightarrow & f \vdash_K f' \text{ and } \eta g = \eta' \text{ and } \vdash g \\
f\eta g & \vdash_{\phi^*(K)} \; f'\eta' g' & \Leftrightarrow & \begin{cases} f \vdash_K f',\ \eta g = \eta',\ \vdash g \text{ and } \vdash g', & \text{or} \\ g \vdash g' \text{ and } f\eta = f'\eta' \end{cases}
\end{array}$$

We break into the same two conditions, (1) $j \leq x_1$ and (2) $j > x_1$.

(1) $j \leq x_1$. Thus

$$\sigma_{j-1} \vdash_{\mathit{lookup}^*(E_{j-1})} \widetilde{\sigma}_{j-1} \quad \Rightarrow \quad \sigma_j \vdash_{\mathit{lookup}^*(E_j)} \widetilde{\sigma}_j.$$

is

$$\begin{array}{ccc} \sigma_{j-1} & \vdash_{E_j} & \widetilde{\sigma}_{j-1} \\ & \Rightarrow & \\ \sigma_{j-1}\eta_j[b_j] & \vdash_{\mathit{lookup}^*(E_j)} & \widetilde{\sigma}_{j-1}\eta_j[b_j]\widetilde{a}_{1j} \end{array}$$

and so we appeal to line 2 of the definition of enabling of compound actions, with $f = \sigma_{j-1}$, $K = E_j$, $f' = \widetilde{\sigma}_{j-1}$, $\eta = \eta_j[b_j]$ and $\eta' = \eta_j[b_j]\widetilde{a}_{1j}$, and $\phi = \mathit{lookup}$.

(2) $j > x_1$. Thus

$$\sigma_{j-1} \vdash_{\mathit{lookup}^*(E_{j-1})} \widetilde{\sigma}_{j-1} \quad \Rightarrow \quad \sigma_j \vdash_{\mathit{lookup}^*(E_j)} \widetilde{\sigma}_j.$$

is

$$\begin{array}{ccc} \sigma_{j-1} & \vdash_{E_j} & \widetilde{\sigma}_{j-1} \\ & \Rightarrow & \\ \sigma_{j-1}\eta_j[b_j] & \vdash_{\mathit{lookup}^*(E_j)} & \widetilde{\sigma}_{j-1}\eta_j[b_j] \end{array}$$

and so we appeal to line 1 of the definition of enabling of compound actions, with $f = \sigma_{j-1}$, $K = E_j$, $f' = \widetilde{\sigma}_{j-1}$, $\eta = \eta_j[b_j]$ and $\eta' = \eta_j[b_j]$, and $\phi = \mathit{lookup}$.

Thus we have proved our goal

$$\zeta_1 \dot{b} \zeta_2 \quad \vdash_{E^*} \quad \zeta_1 \dot{a} \hat{\zeta}_2,$$

The verification of the first move of ν was the tough case. For $i = 2, \ldots y$, to verify that $s_i \in \mathit{Act}^{op}_{L_i}$ is essentially the same, though simplified by the fact that each of the η_j are empty. The location $L_i = D^*_{il}$ are well-defined as $\mathit{store}(\mathit{ref}(s_i))$, since for any compound action d, $\mathit{ref}(d)$ is (by definition) determined by the last atomic live action of d. The copycat rule (required for p' to be a play) is satisfied because the copycat rule is satisfied at every step of "**Recursion step'**" (from page 146): in cases (1) and (2) $\mathit{ref}(a_{i+1,k_{i+1}}) = \mathit{ref} c_{i+1} = \alpha_{i+1}$, and in (3) $\mathit{ref}(\dot{c}_{i+1}) = \mathit{ref} c_{i+1} = \alpha_i = \alpha_{i+1}$.

Finally, we must return to prove facts 6.5–6.8. (Refer to the diagrams in cases (1), (2), and (3) of "**Recursion step**", page 146.)

(6.5) $\widetilde{\alpha}_{1j} \in \mathit{OHol}(q) + \mathit{PHol}(r)$. Since $j < x_1$, the recursion has not terminated. Thus in the paragraph "**Recursion step**" (page 146), $\alpha_i \in \mathit{OHol}(q) + \mathit{OHol}(r)$ holds.

(6.6) $\mathit{ref}(\widetilde{a}_{1j})$ is global. By definition of 'linebreak' in construction of s (and, in particular, s_1), we have $\widetilde{a}_{1j} = a_{i_j 1}$ for some $i_j \geq 1$, so $\mathit{ref}(\widetilde{a}_{1j}) = \alpha_{i_j+1}$. (Refer to the diagrams of cases (1) and (2) of the recursion.) By the previous fact (if α_{i_j+1} is not already global, because of the termination condition $\alpha_{i_j+1} \in H$, for i_j+1 maximal) α_{i_j+1} is an O-hole. Now the only way an opening P-move (viz. a_{i_j}) can reference an O-hole (viz. α_{i_j+1}) is if that O-hole is global, since there are no O-holes in the thread of an opening action.

(6.7) $\mathit{source}(\widetilde{\alpha}_{1j}) \in \mathit{PHol}(q)+\mathit{PHol}(r)$. Because $j < x_1$, the recursion is not yet terminated, so the condition $\mathit{source}(\alpha_i) \in \mathit{PHol}(q)+\mathit{PHol}(r)$ of the paragraph "**Recursion step**" (page 146) holds.

(6.8) $\widetilde{\alpha}_{1x_1} \in \mathit{PHol}(\widetilde{a}_{1x_1})$. Since $\widetilde{a}_{1x_1}$ is the last action $a_{i_{x_1}1}$ of s_1, by definition of the 'linebreak' condition, the next action of s must be $a_{i_{x_1}2}$, and hence $k_{i_{x_1}} > 1$. Thus (refer to the diagrams of cases (1) and (2) of the recursion) there are at least two actions beneath $\dot{c}_{i_{x_1}-1}$, and in particular $a_{i_{x_1}1}$ must reference a P-hole.

B. By construction, this case is merely the extension of case A with an additional •-move. Since this is a •-move, conditions 1,2, and 4 are immediate. Condition 3(a) does not apply, and condition 3(b) holds by construction of the stored singleton arenas.

2. This case was by construction case 1 with q and r swapped, so the same correctness argument carries through (with q and r swapped).

3. (a) i. If $\mathit{depth}(\zeta b \zeta') = 1$, then $E^* = A \Rightarrow B$ and $D = B \Rightarrow C$. So since $\dot{b} \curvearrowleft \dot{a}$, $\mathit{depth}(\dot{a}) = 1$. Condition 2 holds, because $\dot{b} \curvearrowleft \dot{a}$ and $b \overline{\curvearrowleft} a$. Condition 3 is vacuous, since we are not adding any new P-moves to ρ. Condition 4 holds trivially.

   ii. With the addition of the move $(F, \underline{*}, \dot{c}_1)$, since $\mathit{depth}(\dot{c}_1) > 1$, the only case of concern is condition 2. This holds since both $c_1 \dots c_k$ and $\dot{c}$ are unjustified.

   (b) By definition, this case parallels case (a).

   (c) For each of i., ii., the reasoning is the same as case 1.(c).ii.B. Case iii. is immediate, since no additional •-move has been played.

□

6.3 Composition

Having defined the interaction algorithm in the previous section, we are ready to derive from it the composition of strategies.

Definition 6.17 *Given polymorphic arenas A, B, C over a common set of holes H and strategies $\sigma : A \to B$ and $\tau : B \to C$, their composite $\sigma;\tau : A \to C$ is defined as the set of positions*

$$\{\, p \;:\; (p, q, r, \mathit{next}, \mathit{source}) \in \mathit{Int}(A,B,C),\ \ulcorner q \urcorner \in \tau \text{ and } \ulcorner r \urcorner \in \sigma \}$$

in $A \to C$.

We must verify that $\sigma;\tau$ is a well-defined strategy.

Proposition 6.18 *The composite $\sigma;\tau \subseteq \mathit{Pos}(A \Rightarrow C)$ as defined above is a strategy for $A \Rightarrow C$.*

Proof We must verify that $\sigma;\tau$ is non-empty, prefix-closed, P-deterministic and O-contingent complete. It is non-empty because $(\epsilon, \epsilon, \epsilon, 1, \epsilon) \in \mathit{Int}(A, B, C)$. It is prefix-closed because the set of valid interaction states was built up inductively on length. P-determinism is inherited from the P-determinism of σ and τ, because every move played by P in p was the aggregate of actions each with source a P-move in q or r, together with copy-cat actions, which are deterministic. O-contingent completeness comes from the fact that every time $\mathit{next} = 1$ in the interaction algorithm, we allowed every possible O-move extending p. □

PROPOSITION 6.19 *Composition of strategies is associative.*

Proof Because parallel composition of processes is associative. Given strategies $\sigma : A \to B$, $\tau : B \to C$ and $\nu : C \to D$, define valid interaction states $\rho = (p, q, r, s, \mathit{next}, \mathit{source}) \in \mathit{Int}(A, B, C, D)$ analogously to $\mathit{Int}(A, B, C)$, with $p \in \mathit{Pos}(A \Rightarrow D)$, p a play on $(C \Rightarrow D)$, q a play on $(B \Rightarrow C)$, and r a play on $(A \Rightarrow B)$. Define $\sigma;\tau;\nu$ as the set of positions

$$\{\ p\ :\ (p, q, r, \mathit{next}, \mathit{source}) \in \mathit{Int}(A, B, C),\ \ \ulcorner q \urcorner \in \nu, \ulcorner r \urcorner \in \tau \text{ and } \ulcorner s \urcorner \in \sigma\ \}$$

Since the interaction algorithm merely copies moves around, both $(\sigma;\tau);\nu$ and $\sigma;(\tau;\nu)$ are equal to $\sigma;\tau;\nu$. □

6.3.1 Identities

Given a polymorphic arena $A \in \mathbb{PA}_H$, the identity strategy id_A on $A \Rightarrow A$ is a copycat strategy. Distinguish the input and output components of $A \Rightarrow A$ by writing subscripts: $(A \Rightarrow A) = (A_1 \Rightarrow A_2)$. Given an action or hole θ in A_1, write $\overline{\theta}$ for the corresponding action or hole in A_2, and vice versa.

We define $\mathit{id}_A \subseteq \mathit{Pos}(A_1 \Rightarrow A_2)$ in stages. On the depth 1 actions, things are exactly as in the Hyland/Ong definition of an identity strategy:

$$\begin{array}{cccccccc} O & P & O & P & O & P & O & P \\ \\ a_1 & \overline{a}_1 & \overline{a}_2 & a_2 & a_3 & \overline{a}_3 & \overline{a}_4 & a_4 \end{array}$$

First, define $S \subseteq (\mathit{Act}_{A_1} + \mathit{Act}_{A_2})^*$ to consist of sequences s of actions from A_1 and A_2 such that

$$s = s'ab \ \Rightarrow \begin{cases} b = \overline{a} & \text{if } |s| \text{ is even} \\ a \vdash b & \text{if } |s| \text{ is odd} \end{cases}$$

where $a \vdash b$ stands for $a \vdash_{A_1} b$ or $a \vdash_{A_2} b$ depending on whether a is in A_1 or A_2.

For each $s \in S$ add justification pointers $\curvearrowleft : s \rightharpoonup s$ by defining $\curvearrowleft(a)$ to be the predecessor of a if a is in position $2, 3, 5, 7, 9, \ldots$, or the action three before a if a occurs in position $4, 6, 8, \ldots$.

Now turn each action a of s into a move $(A_1 \Rightarrow A_2\phi, a)$ as follows. If a is in odd position, then define ϕ to be constantly $*$. If a is in even position, so that the preceding move in s is a copy $\overline{a}$ of a in the opposite copy of A, for each $\alpha \in \mathit{Hol}_A(a)$ define $\phi : \mathit{Hol}_A(a) \to \mathbb{PA}$ by $\phi(\alpha) = \langle \overline{\alpha}, \bullet \rangle$, the singleton polymorphic arena referencing $\overline{\alpha}$. In other words, P 'copies' the hidden contents of O-holes by storing singleton polymorphic arenas referencing the corresponding holes.

The final step is to turn the justified sequence of moves into a play of $A_1 \Rightarrow A_2$. To do so, we shall add a move $\bullet$ underneath some of the P-moves (where $\bullet$ is the action of all the stored singleton arenas), in order to form two-move P-hypermoves. Specifically: whenever the reference $\mathit{ref}_s(a)$ of an action is a P-hole of s, add the move $\bullet$ underneath a. Note that the copycat condition is satisfied because of the $\bullet$ moves, and the identity is clearly a winning strategy.

Given a strategy $\sigma : A \Rightarrow B$, the composite $\mathit{id}_A;\sigma$ is σ, as is $\sigma;\mathit{id}_B$. At the level of actions, this is for the usual reason in game semantics, that 'copying does nothing'. Likewise at the level of stored arenas: P copies the $*$ contents of O holes as his own stored arenas.

6.3.2 Winning strategies compose

PROPOSITION 6.20 *The composition of winning strategies is winning.*

Proof Suppose $\sigma : A \to B$ and $\tau : B \to C$ are winning strategies, in other words, both σ and τ are total and finite. We show a contradiction when (i) $\sigma;\tau : A \to C$ is infinite, and (ii) $\sigma;\tau : A \to C$ is not total. To make the arguments easier to follow, we first consider (1) the case when A, B, and C have no local holes, and then (2) the general case.

(1i) Suppose that A, B and C have no local holes, and that $\sigma;\tau : A \to C$ is infinite. Every strategy is finitely branching (by Lemma 5.9, and the fact that it is finitely branching at P-hypermoves because of P-determinism), so by König's lemma there must be an infinite sequence of positions $p_0 \sqsubset p_1 \sqsubset p_2 \sqsubset \dots$. Correspondingly, by definition of composition, there must be an infinite sequence of valid interaction states $\rho_0 \sqsubset \rho_1 \sqsubset \rho_2 \sqsubset \dots$ in $\mathit{Int}(A,B,C)$. Let $\rho_i = (p_i, q_i, r_i, \mathit{next}_i, \mathit{source}_i)$, and define $\rho = (p, q, r, \mathit{next}, \mathit{source})$ to be the infinite interaction state that is the limit of the ρ_i.

Since A, B and C have no local holes, every hypermove of p, q and r consists in a single move, which in turn consists in an action with no storage of arenas. Hence the source of $\mathit{source}(a)$ of every P-action a of the infinite position p is either in q or r. So since every P-action is the direct result of a P-action in q or r, (at least) one of the plays q or r must be infinite. Without loss of generality, suppose q is infinite.

Since q is infinite and τ is finite, some view $v \in \tau$ must recur an infinite number of times. In other words, there exists an infinite increasing sequence of indices $i_1 < i_2 < i_3 < \dots$ such that $\ulcorner q_{i_j} \urcorner = v$. Write a_i for the last action of q_i. Since the last action of a play is by definition in the view, $a_{i_j} = a$ for a the last action[3] of the postition v. Note that a_{i_k} is the only of the a_{i_j} that appears in the view[4] $\ulcorner q_{i_k} \urcorner$. For suppose a_{i_l} is in $\ulcorner q_{i_k} \urcorner$ for $l < l$. Then $\ulcorner q_{i_k} \urcorner = \ulcorner q_{i_l} \urcorner \cdot s \cdot a_{i_k}$ for some subsequence s of the actions between a_{i_l} and a_{i_k}, hence $v = v \cdot s \cdot a_{i_k}$, a contradiction.

For a_{i_j} not to appear in $\ulcorner q_{i_{j+1}} \urcorner$, it must be deleted in the process of taking the view. By definition of the view algorithm, there must be an O-pointer 'skipping' over a_{i_j}. More precisely, there exists an O-action b_j and a P-action b'_j such that $b'_j \curvearrowleft b_j$ and $b'_j < a_{i_j} < b_j < a_{i_{j+1}}$.

The only way for an O-action b of the play q_i (hence of q) to have a non-trivial justification pointer, i.e., to be justified not by the preceding P-move, but an earlier one, is if b is an

[3]Note that this equality $a_{i_j} = a$, a_{i_j} is an action-*occurrence* in q and a is merely an action of $B \Rightarrow C$.

[4]Similar to the action-occurrence/action distinction, we consider $\ulcorner q_{i_k} \urcorner$ as a subsequence of q_{i_k} (and hence of q), rather than 'forgetting where its actions came from' and regarding it merely as an element of $\mathit{Pos}(B \Rightarrow C)$. So it makes sense to ask whether some action of q_{i_k} (and hence of q) appears in $\ulcorner q_{i_k} \urcorner$ or not.

action of[5] B. This is because of the following reasoning. Certainly b is[6] either in B or C. If it is in C, then $\mathsf{source}(b)$ was in p. O's pointers are by definition trivial in p (because p is a position, rather than a play), and by the definition of the interaction algorithm on O-moves in p, when b is passed accross to q, it is provided with the corresponding trivial pointer there. Hence b must have been in B.

Thus we have found an infinite subsequence $b_0 < b_1 < b_2 < \dots$ of actions in q all of which are[7] in B. Since B is finite, one of the actions b of B must occur infinitely often. In other words, there exist $j_0 < j_1 < j_2 < \dots$ such that[8] $b_{j_i} = b$ for $i \geq 0$.

Without loss of generality, the enabling action b' of b in $B \Rightarrow C$ occurs only finitely often. Otherwise take $b = b'$, and repeat. (This terminates because there are finitely many actions b'' in B such that $b'' \vdash_B b$. Also, note in passing that $b' \in \mathsf{Act}_C \hookrightarrow \mathsf{Act}_C^{\mathrm{op}} \times \mathsf{Act}_B + \mathsf{Act}_C$ if and only if b is an opening action of B.) Since there are an infinite number of occurrences of b in q and only a finite number of occurrences of b', one of the occurrences of b' must justify an infinite number of occurrences of b. Now by Martin Hyland's result on timewasting for dialogue games and innocent strategies (quoted in [NO]), necessarily there are an infinite number of distinct views in q. (The timewasting result applies because, in the interaction algorithm every move by σ and τ was based on the view in $A \Rightarrow B$ or $B \Rightarrow C$ alone, so each of σ and τ are playing innocently in B.) The infinite number of distinct views in q contradicts the finiteness of σ.

(1ii) Suppose that A, B, and C have no local holes, and that $\sigma;\tau : A \to C$ is not total. Then p is finite, and is terminated by an O action a, with timestamp i. So after timestamp i there are no further moves by σ in A or by τ in C. Thus σ and τ engage in 'infinite chit-chat' between p and q.

(2i) Suppose that $\sigma;\tau : A \to C$ is infinite. Recall that the depth of a hypermove $m_1 \dots m_k$ in a hypersequence h is the depth of the row in which the first move m_1 appears when we display h (the formal definition was Definition 5.3.2, page 69). Then there are two cases: (a) there are an infinite number of moves in p of depth 1, or (b) after a final depth 1 move m, every move of p is of depth 2 or more.

In case (a), by Lemma 5.10 every move at depth 1 is located in $A \Rightarrow C$, so there are an infinite number of moves in $A \Rightarrow C$. So then we apply (1i) to reach a contradiction.

Suppose case (b) applies. In the interaction algorithm, P-moves m in p of depth 2 or more arise in one of two ways: (A) by copycat (when $\mathsf{source}(m)$ is in p), or (B) by σ or τ (when $\mathsf{source}(m)$ is in q or r).

Claim. An infinite number of P-moves of p are of type (B).

Proof of claim. Suppose only finitely many moves of p arise from σ or τ. Then after some move n, every P-move is a copycat move. The only polymorphic arenas stored by P in such copycat moves are singletons, so the size of the playing area remains constant after n. So since P is playing copycat to O, who always justifies by the immediately preceding move, O will eventually reach the leaf of an arena. When P copies this leaf action in his next move, O will have been run out of moves. So p is prefix-maximal, contradicting finiteness. (*QED Claim.*)

Since there are an infinite number of moves of type (B), i.e., transmissions from σ or τ, one of q or r is infinite. Without loss of generality, suppose q is infinite.

---

[5] When we say an action b of $B \Rightarrow C$ "is an action of B" we mean morally that b is in the input component corresponding to "$B \Rightarrow$". Technically speaking, since $\mathsf{Act}_{B \Rightarrow C}$ is $\mathsf{Act}_C^{\mathrm{op}} \times \mathsf{Act}_B + \mathsf{Act}_C$, by "is an action of B" we mean $b \in \mathsf{Act}_C^{\mathrm{op}} \times \mathsf{Act}_B$. So in thinking of $b \in \mathsf{Act}_B$, b implicitly carries a 'tag' indictating which opening action of C the copy of B containing b is grafted on to in the function space construction.

[6] Modulo the previous footnote.

[7] Modulo the last but one footnote.

[8] See footnote 3.

In particular, there are an infinite number of moves at depth 1 in q. For if not, there is some move n' in q after which every move is of depth 2 or more. So by definition of the interaction algorithm, every O-move after n' is a copycat move, and τ is being played against itself. So since τ is finite, q is finite, which is a contradiction.

Now because q has an infinite number of moves at depth 1, by Lemma 5.10 these are all located in $B \Rightarrow C$. So there are an infinite number of moves in $B \Rightarrow C$, and the reasoning of (1i) applies to q.

(2ii) Suppose $\sigma;\tau$ is not total. Then p is finite, terminated by an O-move. Since σ and τ are total, at least one of q and r are infinite. Without loss of generality, assume q is infinite. There are an infinite number of moves at depth 1 in q. Otherwise after some move m every move is at depth 2 or more, so every O-move after m is a copycat move (τ is being played 'against himself'), hence q is finite by the finiteness of τ. Once again, by Lemma 5.10, every move of q at depth 1 is in $B \Rightarrow C$, so there are an inifinite number of moves in q in $B \Rightarrow C$, and we can apply the reasoning of (1i) once again, in order to reach a contradiction.

$\square$

Note that it was finiteness that was contradicted when we assumed that $\sigma : \tau : A \to C$ was not total.

Chapter 7

The categorical model

We take the work of the previous chapters and construct a $2\lambda\times$-hyperdoctrine $\mathbb{H}$, and hence obtain a model of system F. The first step is to define an auxiliary $2\lambda\times$-hyperdoctrine $\mathbb{G}$, which contains 'partial' as well as winning strategies. Then we take $\mathbb{H}$ to be the subcategory of $\mathbb{G}$ consisting of the winning strategies.

7.1 The auxiliary hyperdoctrine $\mathbb{G}$

We define a hyperdoctrine $\mathbb{G}$, i.e., a $\mathbb{B}$-indexed cartesian closed category $\mathbb{G} : \mathbb{B}^{op} \to \mathsf{CCCat}$. For $n \geq 0$ let $[n]$ denote the set $\{1, \ldots, n\}$ of positive natural numbers up to and including n, and write $\mathbb{PA}_n$ as shorthand for $\mathbb{PA}_{[n]}$, the set of polymorphic arenas over $[n]$, i.e. with n global holes $\{1, \ldots, n\}$. Given $1 \leq m \leq n$ write $\bullet^m_n$ for the single-action polymorphic arena of $\mathbb{PA}_n$ in which the action references the global hole $m \in [n]$.

7.1.1 The base category $\mathbb{B}$

The base category $\mathbb{B}$ of $\mathbb{G}$ is defined as follows. It is very similar in spirit to the base category of the term model of system F.

Objects The set $\mathbb{N} = \{0, 1, 2, \ldots\}$ of natural numbers.

Morphisms A morphism in $\mathbb{B}(m, n)$ consists of an n-tuple of polymorphic arenas in $\mathbb{PA}_m$. Thus a morphism $\alpha : m \to n$ in $\mathbb{B}$ is a function $\alpha : [n] \to \mathbb{PA}_m$.

Composition Given $\alpha : k \to m$ and $\beta : m \to n$, the composite of $\alpha;\beta : k \to n$ is given by expansion (analogous to substitution in the base category of the term model). Specifically, as a function $[n] \to \mathbb{PA}_k$, the composite $\alpha;\beta$ is given by $(\alpha;\beta)(i) = \alpha^*(\beta(i))$, the expansion of the polymorphic arena $\beta(i) \in \mathbb{PA}_m$ by the assignment $\alpha : [m] \to \mathbb{PA}_k$ of polymorphic arenas to the global holes of $\beta(i)$, as given in Definition 4.2, page 59.

Identities The identity $n \to n$, as a function $[n] \to \mathbb{PA}_n$, takes i to $\bullet^i_n$, i.e. the singleton polymorphic arena whose action references $i \in [n]$.

Products The product of m and n is their numeric sum $n+m$. The projections $\pi_m : m+n \to m$ and $\pi_n : m+n \to n$, as functions $\pi_m : [m] \to \mathbb{PA}_{m+n}$ and $\pi_n : [n] \to \mathbb{PA}_{m+n}$, are given by $\pi_m(i) = \bullet^i_{m+n}$ and $\pi_n(i) = \bullet^{m+i}_{m+n}$. Again, just think of the term model.

Distinguished object The distinguished object from which all other objects are generated by finite products is 1.

7.1.2 The fibre $\mathbb{G}_n$

The cartesian closed category $\mathbb{G}_n$ is defined as follows.

Objects The set $\mathbb{PA}_n$ of polymorphic arenas over $[n]$.

Morphisms A morphism $\sigma : A \to B$ is a strategy on the polymorphic arena $A \Rightarrow B$.

Composition Given strategies $\sigma : A \to B$ and $\tau : B \to C$, composition is as given in Definition 6.17 at the end of the previous chapter.

Identities Copycat strategies, as given in Definition 6.3.1 at the end of the previous chapter.

Products and Exponentials Product $A \times B$ and exponential $A \Rightarrow B$ of arenas was defined in Chapter 4.

Terminal object The terminal object is the empty polymorphic arena.

PROPOSITION 7.1 *$\mathbb{G}_n$ is cartesian closed.*

Proof The natural isomorphism $\mathbb{G}_n(A, B \times C) \cong \mathbb{G}_n(A, B) \times \mathbb{G}_n(A, C)$ is given as follows. Suppose σ is a strategy on $A \Rightarrow B \times C$. By construction of $\Rightarrow$ and $\times$, $A \Rightarrow B \times C$ is $(A \Rightarrow B) \times (A \Rightarrow C)$. Define $\overline{\sigma} \in \mathbb{G}_n(A, B) \times \mathbb{G}_n(A, C)$ as (σ_1, σ_2), where

$$\sigma_1 = \{\, p \in \sigma \ : \ p \in \mathit{Pos}(A \Rightarrow B) \subseteq \mathit{Pos}((A \Rightarrow B) \times (A \Rightarrow C)) \,\}$$

and

$$\sigma_2 = \{\, p \in \sigma \ : \ p \in \mathit{Pos}(A \Rightarrow C) \subseteq \mathit{Pos}((A \Rightarrow B) \times (A \Rightarrow C)) \,\}$$

This is clearly a natural isomorphism, with inverse $(\sigma_1, \sigma_2) \mapsto \sigma_1 \cup \sigma_2$.

By construction of $\Rightarrow$ and $\times$, $A \Rightarrow (B \Rightarrow C)$ is exactly $A \times B \Rightarrow C$. So $\mathit{Pos}(A, B \Rightarrow C) = \mathit{Pos}(A \times B, C)$, and $\mathbb{G}_n(A, B \Rightarrow C) \cong \mathbb{G}_n(A \times B, C)$ directly.

The empty arena E is terminal because $A \Rightarrow E = E$ and there is a unique strategy on E, namely $\{\epsilon\}$. $\square$

Note that, as required in the definition of $2\lambda\times$-hyperdoctrine, the set of objects of $\mathbb{G}_n = \mathbb{PA}_n$ is exactly the homset $(n, 1)$, because this homset is by definition the set of '1-tuples' of polymorphic arenas over $[n]$.

7.1.3 Reindexing functors

Given a morphism $\alpha : n \to m$ in the base category, we define a strict cartesian closed reindexing functor $\alpha^* : \mathbb{G}_m \to \mathbb{G}_n$.

On objects Given $A \in \mathbb{G}_m$, i.e., a polymorphic arena with m holes, define $\alpha^*(A)$ to be the expansion $\alpha^*(A)$ of A along $\alpha : [m] \to \mathbb{PA}_n$, an assignment of polymorphic arenas to the global holes of A.

On morphisms Given a strategy σ on $A = B \Rightarrow C$, the strategy $\alpha^*(\sigma)$ on $\alpha^*(A)$ is defined as follows. By definition of a strategy, σ is a set of positions on A. We use σ as the 'backbone' of a set of positions on $\alpha^*(A)$, and then use copycat for the 'ribs' consisting of moves that go outside this core set.

Recall from page 60 of the section on expansion that a path down the forest of $\alpha^*(A)$ corresponds to a path (the 'backbone') in A ending at an action a referencing a global hole $i \in [m]$, followed by a path (the 'rib') in $\alpha(i)$, together with an accumulation of post-fix tags on the actions of the path in A. Let $\widetilde{\mathit{Plays}}(\alpha^*(A))$ be the subset of $\mathit{Pos}(\alpha^*(A))$ consisting of positions such that all threads on $\alpha^*(A)$ (as opposed to threads on imported arenas) are paths for which the 'rib' is at most one move.

Let $\pi : \widetilde{\mathit{Plays}}(\alpha^*(A)) \to \mathit{Pos}(A)$ be the projection (i.e. surjection) defined by deleting the post-fix tags from threads in $(\alpha^*(A))$, including the tag consisting of the first 'rib'-move when it arises. (At this point it is probably useful to look at the column on page 60 characterising a path in an expanded arena.) Then by live atomic enabling (section 4.4.2) π is a 'local homeomorphism' in the following sense: given any $p \in \widetilde{\mathit{Plays}}(\alpha^*(A))$ with $\pi(p)\mu \in \mathit{Pos}(A)$ for some hypermove μ, there is a unique hypermove $\tilde{\mu}$ such that $p\tilde{\mu} \in \widetilde{\mathit{Plays}}(\alpha^*(A))$ and $\pi(p\tilde{\mu}) = \pi(p)\mu$ (trivially so, infact, if the first move of μ is not located in A).

Define the 'backbone' strategy $\sigma_1 \subseteq \widetilde{\mathit{Plays}}(\alpha^*(A)) \subseteq \mathit{Pos}(A)$ by iterating the 'local homeomorphism', starting at the empty hypersequence. Thus each position of σ_1 is a position of σ with some additional post-fix tags on the moves of threads in $\alpha^*(A)$.

We extend σ_1 with 'copycat ribs' in order to define $\alpha^*(\sigma)$. Suppose $p\mu \in \mathit{Pos}(A)$, with $p \in \sigma_1$, and with the last move of $\pi(p)$ referencing a global hole i. Thus the last action of p is of the form $a\nu b$ for $a \in A$, ν a sequence of post-fix tags, and b an opening action of $\alpha(i)$. Suppose further that μ was a P-hypermove. Then by the copycat rule, $\pi(p\mu)$ also references i, so $p\tilde{\mu}$ is also a compound move ending in b. Now to form $\alpha^*(\sigma)$, for each such p and μ add all extensions of $p\mu$ obtained by playing the copycat strategy against O if the live atomic enabler of his next action is b. In other words, P plays copycat on the copy of the 'rib' arena $\alpha(i)$ that got pasted onto a in the expansion of the 'backbone' A.

The functor α^* is trivially strict cartesian closed because of the simplicity of the definitions of product and exponential for polymorphic arenas as a graphical operations.

7.1.4 Indexed products

Let $\pi_n : n+1 \to n$ in $\mathbb{B}$ be projection onto n, i.e. $\pi_n : [n] \to \mathbb{PA}_{n+1}$ maps i to $\bullet^i_{n+1}$. Then the reindexing functor $\pi_n^* : \mathbb{G}_n \to \mathbb{G}_{n+1}$ simply adds an extra global hole to a polymorphic arena A, and acts as the identity on morphisms[1]. Define a right adjoint $\Pi_n : \mathbb{G}_{n+1} \to \mathbb{G}_n$ to π_n^* on objects by $\Pi_n(A) = \forall(n+1)(A)$, the quantification of $n+1$ in A, as defined on page 52. On morphisms, $\Pi_n(\sigma)$ is essentially the identity: for each position $p \in \sigma$ simply add an extra O-import $*$ to the first move of p (corresponding to the global hole $n+1$, which has now become local).

Because of the syntactic flavour of polymorphic arenas, the fact that the two functors are adjoint is just as trivial as in the term model. We need to verify that for all arenas with global holes $[n]$, $\mathbb{G}_n(B, \forall(n+1)(A))$ is naturally isomorphic to $\mathbb{G}_{n+1}(\pi_n^*(B), A)$. In other words, strategies on $B \Rightarrow \forall(n+1)(A)$ correspond naturally to strategies on $\pi_n^*(B) \Rightarrow A$. This

[1] Modulo some additional (but redundant) tags that get put on to the actions of A.

is immediate, since the two arenas are the same apart from the fact that the hole $n+1$ is global in the former and (copies of it are) local in the latter.

We must verify the Beck-Chevalley condition, that 'quantification commutes with expansion'. Formally, we require that the diagram on page 20 commutes, i.e., for all $\alpha : m \to n$ in the base $\mathbb{B}$, the functors $\mathbb{G}_{n+1} \xrightarrow{\Pi_n} \mathbb{G}_n \xrightarrow{\alpha^*} \mathbb{G}_m$ and $\mathbb{G}_{n+1} \xrightarrow{(\alpha \times id_1)^*} \mathbb{G}_{m+1} \xrightarrow{\Pi_m} \mathbb{G}_m$ are the same. On an object A, the first functor moves (copies of) the global hole $n+1$ down into the roots of A, then α^* expands actions referencing $1, \ldots, n$ into polymorphic arenas $\alpha(1), \ldots, \ldots \alpha(n)$. The second functor expands the actions $1, \ldots, n$, and essentially sets actions a referencing $n+1$ to reference $m+1$, because id_1 expands a with $\bullet^{m+1}_{m+1}$; then it moves $m+1$ down into the roots. By similar considerations, the two functors are also the same on morphisms, and the canonical natural transformation is an identity.

We have shown:

PROPOSITION 7.2 $\mathbb{G}$ *is a* $2\lambda\times$*-hyperdoctrine.*

Hence, as detailed in [Cro94], $\mathbb{G}$ is a model of system F.

7.2 The hypergame model $\mathbb{H}$

Define $\mathbb{H}$ as the subcategory of $\mathbb{G}$ obtained by reducing the homsets in the fibres to the winning strategies. We need to check that $\mathbb{H}$ is a well-defined $2\lambda\times$-hyperdoctrine.

The fibre $\mathbb{H}_n$ is closed under composition, since winning strategies compose (Proposition 6.20). Identities are winning strategies, as is the unique strategy on the empty arena. The product adjunction puts winning strategies in bijection with pairs of winning strategies, and the exponential adjunction, being trivial, puts winning strategies in bijection with winning strategies.

Given a morphism $\alpha : n \to m$ in the base category $\mathbb{B}$, and a winning strategy σ on A, $\alpha^*(\sigma)$ is winning. The underlying 'backbone' of $\alpha^*(\sigma)$ is σ itself, which is total, and as soon as P elects to go into a 'rib' $\alpha(i)$, $\alpha^*(\sigma)$ becomes the copycat strategy between instances of the arena $\alpha(i)$, and copycat strategies are winning. Finally, there are only a finite number of copycat strategies added to σ in the construction of $\alpha^*(\sigma)$, because, since arenas are finite, only a finite number of holes of A were expanded.

Since the right adjoint $\Pi_n : \mathbb{G}_{n+1} \to \mathbb{G}_n$ to $\pi_n^* : \mathbb{G}_n \to \mathbb{G}_{n+1}$ acts essentially trivially on morphisms, it maps winning strategies to winning strategies. The natural isomorphism $\mathbb{G}_n(B, \forall(n+1)(A)) \cong \mathbb{G}_{n+1}(\pi_n^*(B), A)$ is by definition essentially trivial, and so puts winning strategies in bijection with one another.

Thus we have shown:

PROPOSITION 7.3 $\mathbb{H}$ *is a* $2\lambda\times$*-hyperdoctrine.*

Hence $\mathbb{H}$ is a model of system F, as detailed in [Cro94].

Chapter 8

Full completeness

In this chapter we prove our main result.

THEOREM 8.1 (FULL COMPLETENESS) *Every morphism (i.e. winning strategy) σ of $\mathbb{H}$ defines an η-long, β-normal term $\widehat{\sigma}$, whose interpretation is σ.*

Before proving the theorem, we require the analogous result at the level of types:

PROPOSITION 8.2 *Every polymorphic arena $A \in \mathbb{H}_n$ defines a prenex normal type $\widehat{A}$ of system F with free type variables amongst $X_1, \ldots, X_n$, and whose interpretation is A.*

Proof We define $\widehat{A}$ by recursion on the depth of the forest of A. If A is the empty polymorphic arena, define $\widehat{A} = \mathit{Unit}$. If A is of depth $d \geq 1$, first consider the case that A is a tree. Let L be the set of local holes attached to the root a, and without loss of generality assume $L = [k]$ for some $k \geq 0$. Let $a_1, \ldots, a_m$ be the children of a. Define A_i to be the polymorphic arena over $[n+k]$ with actions $\{ \, b \in A \; : \; a_i \vdash_A^* b \, \}$ and structure inherited from A, but with references to $j \in L$ now taken to be global references to $n+j$. Being of strictly smaller depth, the A_i define types $\widehat{A}_i$ with free variables amongst $X_1, \ldots, X_{n+k}$. Now define

$$\widehat{A} \;=\; \forall X_{n+1} \ldots \forall X_{n+k}.(\widehat{A}_1 \to \ldots \to \widehat{A}_m \to X_l),$$

where $l = j$ if $\mathit{ref}_A(a) \in [n]$, and $l = n + j$ if $\mathit{ref}_A(a) = j \in L$.

For the more general case when A is a forest consisting of tree polymorphic arenas $B_1, \ldots, B_k$ defining types $\widehat{B}_1, \ldots, \widehat{B}_k$ by the above method, define

$$\widehat{A} \;=\; \widehat{B}_1 \times (\ldots \times (\widehat{B}_{k-2} \times (\widehat{B}_{k-1} \times \widehat{B}_k)) \ldots).$$

The fact that the interpretation of $\widehat{A}$ is A is a simple structural induction on the height of the type. □

Proof (of Theorem 8.1) Let σ be a winning strategy for $A \in \mathbb{H}_n$. We define the normal form t_σ by recursion on the size of σ.

The construction of $\widehat{A}$ from A in the previous proposition induces a linear ordering on the opening actions of A, the set of children of every action of A, and the set of local holes of every action of A. Thus, with $l = |\mathit{Hol}_A(a)|$, the set of local holes $\mathit{Hol}_A(a)$ is canonically isomorphic to $[l] = \{1, 2, \ldots, l\}$, and a store $\phi : \mathit{Hol}_A(a) :\rightarrow \mathbb{PA}$ corresponds to a sequence of l arenas.

If $\sigma = \{\epsilon\}$, then necessarily $A = \mathit{Unit}$, and we define $t_\sigma = \langle\ \rangle$. Otherwise, consider the case when A is a tree. Let a be the opening action of A, and let $b_{11}, \ldots, b_{1u_1}$ be the children of a. (Note that $u_1 = 0$ is impossible, for then A is an arena with the single action a, P has no response to an opening hypermove in A, and there are no winning strategies for A.)

Let $[l]$ be the set of local holes $\mathit{Hol}_A(a)$ of a. O's (unique) opening hypermove on A consists in the single move $(A, \underline{*}, a)$, where the store $\underline{*} : [l] \to \mathbb{PA}_n + \{*\}$ is constantly $*$. After P's response hypermove determined by σ, we reach the following position (with the justification pointer omitted):

$$
\begin{array}{ll}
(A, \underline{*}, a) & (A, B_{11}B_{12}\ldots B_{1z_1}, b_{1e_1}) \\
& (B_{1f_2}, B_{21}B_{22}\ldots B_{2z_2}, b_{2e_2}) \\
& \vdots \\
& (B_{(i-1)f_i}, B_{i1}B_{i2}\ldots B_{iz_i}, b_{ie_i}) \\
& \vdots \\
& (B_{(v-1)f_v}, B_{v1}B_{v2}\ldots B_{vz_v}, b_{ve_v})
\end{array}
$$

where $1 \le f_i \le z_{i-1}$, $1 \le e_i \le u_i$, and for $i = 2, \ldots, l$, $b_{i1}, b_{i2}, \ldots, b_{iu_i}$ are the opening actions of $B_{(i-1)f_i}$. (Following the convention set out above, we identify the store of a move with a finite sequence of arenas.)

We construct an arena $A' \in \mathbb{H}_{n+l}$ such that there is a bijection between positions of A that begin with the two hypermoves above, and positions of A'.

Let $a_{i1}, \ldots, a_{iw_i}$ be the children of b_{ie_i}. Define

$$
\begin{aligned}
A' &= \prod_{\substack{1 \le i \le v \\ 1 \le j \le w_i}} (C \Rightarrow D_{ij}) \\
C &= \prod_{1 \le k \le u_1} C_i
\end{aligned}
$$

where $(C \Rightarrow D_{ij})$ represents the positions of A that result from O playing the child a_{ij} of b_{ie_i} as his next action, and C_k represents P's option to play the child b_{1k} justified by the action a of O's first move.

The arena C_k is the subtree of A below b_{1k}. Formally, $C_k \in \mathbb{H}_{n+l}$ has the set of actions $\{\, c \in \mathit{Act}_A \ :\ b_{1k} \vdash_A^* c \,\}$ and structure inherited from A, but with references to local holes $j \in \mathit{Hol}_A(a)$ of the root a of A redirected to reference the global hole $n + j$.

The arena D_{ij} is the expansion of the subtree of A below a_{ij} by the arenas stored in P's first hypermove. Formally, $D_{ij} \in \mathbb{H}_{n+l}$ is $\phi_i^*(\tilde{D}_{ij})$, where the assignment $\phi_i : \mathit{Hol}_A(b_{ie_i}) \to \mathbb{PA}_{n+l}$ is the store $B_{i1}B_{i2}\ldots B_{iz_i}$ of P's i^{th} move, and $\tilde{D}_{ij} \in \mathbb{H}_{n+l} + \mathit{Hol}_A(b_{ie_i})$ is obtained as follows. The set of actions of $\tilde{D}_{ij}$ is $\{\, d \in \mathit{Act}_A \ :\ a_{ij} \vdash_A^* d \,\}$, and structure is inherited from A, apart from the redirection of references to local holes $j \in \mathit{Hol}_A(a)$ of a to reference the global hole $n + j$, and the set of (previously local) holes $\mathit{Hol}_A(b_{ie_i})$ now considered to be global.

Given $p \in \mathit{Pos}(A)$ extending the pair of hypermoves above, define $\overline{p} \in \mathit{Pos}(A')$ as follows. First, form $\tilde{p}$ from p by deleting the first two hypermoves, and changing the location of all depth 1 moves from A to A'. Reset dangling P justification pointers that previously targetted $(A, \underline{*}, a)$ to target the first O-action of $\tilde{p}$.

Now we must deal with opening moves located in the arenas stored by P in the second hypermove of p. Although these arenas have been deleted the beginning of the position, their ghosts are present in A' because of the expansions forming the D_{ij}. Suppose $\mu = m_1 m_2 \dots m_d$ is a hypermove in which m_1 references the hole of

$$\hat{m} = (B_{(r-1)f_r}, B_{r1}B_{r2}\dots B_{rz_r}, b_{re_r})$$

containing B_{rs}, and formally set $f_1 = 0$ and $B_{00} = A$ in order to account for the case $r = 1$. Thus m_2 is an opening move (B_{rs}, ϕ, b) on B_{rs}. Now m_1, in order to reference a hole of $\hat{m}$, must be hereditarily justified by $\hat{m}$, hence is in the same location as $\hat{m}$, so $m_1 = (B_{(r-1)f_r}, \psi, c)$. Reset $\mu = m_1 m_2 \dots m_d$ to be $\overline{\mu} = \overline{m} m_3 \dots m_d$, where $\overline{m} = (A', [\phi, \psi], bc)$, $[\phi, \psi]$ is the source tupling of ϕ and ψ (since expansion takes disjoint unions of holes), and bc is the compound action formed from b and c.

The retargetted justification pointers are well-defined, because any such justified P-action is b_{ik}, an opening move of C, which is in negative position in the function space component $C \Rightarrow D_{ij}$ of A', and hence (by definition of the enabling relation of $\Rightarrow$) is enabled by the opening action a_{ij} of D_{ij}. The construction $p \mapsto \overline{p}$ is clearly a (prefix-preserving) bijection.

Via the bijection of positions just defined, σ determines a winning strategy σ' for A', which by product factorisation of A is equivalent to a winning strategy σ'_{ij} for each $C \Rightarrow D_{ij}$. Since σ' is strictly smaller than σ, by the induction hypothesis we have normal forms $\widehat{\sigma}_{ij}$ defined by the σ'_{ij}.

We define the normal form $\widehat{\sigma}$ as follows. For $2 \le i \le v$ define the context ("e_ith projection")

$$\pi_i[-] = \begin{cases} \mathit{snd}\dots\mathit{snd}\,\mathit{fst}[-], & \text{if } 1 \le e_i \le u_i, \\ \mathit{snd}\dots\mathit{snd}\,\mathit{snd}[-], & \text{if } e_i = u_i, \end{cases}$$

where the number of consecutive occurrences of *snd* in the first case is $e_i - 1$, and in the second case is e_i. Then for $2 \le i \le v$ define the contexts

$$\begin{aligned} E_1[-] &= [-]\widehat{B}_{11}\widehat{B}_{12}\dots\widehat{B}_{1z_1}\widehat{\sigma}_{11}\dots\widehat{\sigma}_{1w_1} \\ E_i[-] &= (\pi_i[-])\widehat{B}_{i1}\widehat{B}_{i2}\dots\widehat{B}_{iz_i}\widehat{\sigma}_{i1}\dots\widehat{\sigma}_{iw_i} \end{aligned}$$

Define

$$t_\sigma = \Lambda X_1 \dots \Lambda X_k . \lambda x_1^{\widehat{C}_1} \dots \lambda x_{u_1}^{\widehat{C}_{u_1}} . E_v E_{v-1} \dots E_2 E_1 [x_{e_1}]$$

The validation that the interpretation of $\widehat{\sigma}$ is indeed σ is a simple structural induction.

Finally, in the case that A has more than one component, we take the terms generated by σ restricted to each of the components, and then form the appropriate $\langle\ ,\ \rangle$-pairings of these terms that correspond to the product type $\widehat{A}$ as defined in the proposition above. □

An immediate corollary of the construction of the term t_σ in the proof of the theorem is:

COROLLARY 8.3 *If T is a type with no negative quantifiers then every winning strategy σ for the polymorphic arena interpreting T defines a distinct η-long β-normal form t_σ whose interpretation is σ. So in particular, $\beta\eta$-normal forms of type T are in bijection with winning strategies for the polymorphic arena interpreting T.*

This corollary was applied in section 5.7, where we obtained results about $\beta\eta$-normal inhabitants of the encodings of inductive datatypes in system F. Note that in the main theorem one does not have uniqueness of the η-long, β-normal form, since type arguments are identified up to prenex equivalence. One has uniqueness in the corollary because when there are no negative quantifiers, terms do not contain type arguments.

One way of thinking about full completeness is that the model is isomorphic to a quotient of the syntax. The quotient induced by $\mathbb{H}$ on the product- and unit-free fragment system F includes $\beta\eta$ together with the isomorphism induced on terms (because terms contain types) by identifying every type with its prenex normal form. So the equational theory of the model is very close to initial.

Chapter 9

Conclusions

We began in Chapter 1 by introducing a methodology for extracting strategies from normal forms of the simply typed lambda calculus as 'top-down terms'. By applying this methodology to normal forms of system F, we obtained the key idea for a fully complete game semantics: *types as second-order moves.*

Hyland/Ong and Nickau, in their work on first-order games, interpreted a type as an arena or 'board' on which to play a game. In order to turn the types as second-order moves idea into a working model, we required the interpretation of polymorphic types as some suitably generalised notion of 'board'. Our solution in Chapter 4 was simple: decorate Hyland/Nickau/Ong arenas with 'holes' corresponding to quantified type variables. Polymorphic arenas turned out to be in bijection with prenex types, and can be presented as a kind of Böhm tree.

The technically most demanding step in the construction of the hypergame model was to define a composition of strategies. Our detailed analysis of first-order interaction in Chapter 3 uncovered a neglected difference between the approaches of Hyland/Ong and Nickau. For technical reasons, outlined at the end of Chapter 3, we chose Nickau interaction as the first-order basis of interaction in the hypergame model.

The interaction algorithm was presented in Chapter 6. The main idea that was not present in first-order interaction is that moves are not only transmitted between the three agents of an interaction, but can be 'reflected back' aginst the same agent, by a form of copycat. This copycat component of interaction corresponds to the uniformity of polymorphic terms of system F. P cannot see polymorphic arenas stored in holes by O, so the only way to proceed on such arenas is by playing copycat.

By full completeness, the model is parametric, in the informal Strachey sense that every function acts 'uniformly'. The model is not Reynolds relationally parametric, using a result of [PA93]: $\forall X . X \to X$ is not terminal. Note that the term model of system F is not relationally parametric for the same reason, so this is not a symptom of non-uniformity.

On the product- and unit-free fragment system F, the equational theory of the model includes $\beta\eta$, together with the equivalence induced on terms by identifying every type with its prenex normal form (because terms contain types). So the equational theory of the model is very close to initial. A conjecture, and a topic of future research, is that this is exactly the equational theory of the model.

A long-term objective of the line of work in this thesis is to take the 'top-down term' methodology and apply it to other calculi, including other vertices of the lambda cube [Bar92], such as F_ω and dependent types.

Appendix A

Counting inhabitants of types

We consider an example of the copycat rule at work, supplementing the text of section 1.1.1. This is the syntactic argument corresponding to the proof of proposition 5.19.

Given types U and V, the system F encoding of the product of U and V is

$$U \tilde{\times} V \quad = \quad \forall Y.\, (U \to V \to Y) \to Y,$$

for some Y not occurring free in U or V. Take

$$\begin{array}{rcccl} U & = & \texttt{Bool} & = & \forall X.\, X \to X \to X \\ V & = & \texttt{Nat} & = & \forall X.\, (X \to X) \to X \to X \end{array}$$

Here is a possible discussion of a term of type $\texttt{Bool}\tilde{\times}\texttt{Nat}$, following our usual protocol:

$$\begin{array}{rrclrcl} \text{I choose } t & t & = & \Lambda Y.\, \lambda p^{\texttt{Bool}\to\texttt{Nat}\to Y}.? & t & : & \forall Y.\, (\texttt{Bool} \to \texttt{Nat} \to Y) \to Y \\ \text{You choose } p & t & = & \Lambda Y.\, \lambda p^{\texttt{Bool}\to\texttt{Nat}\to Y}.pbn & p & : & \texttt{Bool} \to \texttt{Nat} \to Y \\ \text{I choose } b & b & = & \Lambda X.\, \lambda x^X.\lambda y^X.? & b & : & \forall X.\, X \to X \to X \\ \text{You choose } x & b & = & \Lambda X.\, \lambda x^X.\lambda y^X.x & x & : & X \end{array}$$

Your first move was forced, as t has only one abstracted term variable. The head-variable p required two arguments in order for the body to be of ground type (and hence for t to be η-long): b of type `Bool`, and n of type `Nat`. I chose to inspect b; you followed up with the strategy for `true`.

I could have asked to inspect n instead:

$$\begin{array}{rrclrcl} \text{I choose } t & t & = & \Lambda Y.\, \lambda p^{\texttt{Bool}\to\texttt{Nat}\to Y}.? & t & : & \forall Y.\, (\texttt{Bool} \to \texttt{Nat} \to Y) \to Y \\ \text{You choose } p & t & = & \Lambda Y.\, \lambda p^{\texttt{Bool}\to\texttt{Nat}\to Y}.pbn & p & : & \texttt{Bool} \to \texttt{Nat} \to Y \\ \text{I choose } n & n & = & \Lambda X.\, \lambda f^{X\to X}.\lambda x^X.? & n & : & \forall X.\, (X \to X) \to X \to X \\ \text{You choose } f & n & = & \Lambda X.\, \lambda f^{X\to X}.\lambda x^X.fa_1 & f & : & X \to X \\ \text{I choose } a_1 & a_1 & = & ? & a_1 & : & X \\ \text{You choose } f & a_1 & = & fa_2 & f & : & X \to X \\ \text{I choose } a_2 & a_2 & = & ? & a_2 & : & X \\ \text{You choose } x & a_2 & = & x & x & : & X \end{array}$$

This time you followed up with the strategy for the Church numeral

$$\underline{2} = \Lambda X.\, \lambda f^{X\to X}.\lambda x^X.f(fx)$$

Taking the two together, these discussions define the encoded pair

$$\langle \texttt{true}, \underline{2} \rangle = \Lambda Y.\, \lambda p^{\texttt{Bool} \to \texttt{Nat} \to Y}.p\,\texttt{true}\,\underline{2} \quad : \quad \texttt{Bool} \mathbin{\tilde{\times}} \texttt{Nat}$$

So every pair of strategies, one on `Bool` and one on `Nat`, gives rise to a strategy on $\texttt{Bool} \mathbin{\tilde{\times}} \texttt{Nat}$, injectively. But is pairing surjective? In other words, do all inhabitants of $\texttt{Bool} \mathbin{\tilde{\times}} \texttt{Nat}$ arise by pairing? Because of the copycat rule, we shall see that the answer is yes.

Recall the first seven moves of the previous dialogue, in which I was inspecting the second component of the pair:

I choose t	t	$=$	$\Lambda Y.\, \lambda p^{\texttt{Bool} \to \texttt{Nat} \to Y}.?$	$t \,:\, \forall Y.\, (\texttt{Bool} \to \texttt{Nat} \to Y) \to Y$
You choose p	t	$=$	$\Lambda Y.\, \lambda p^{\texttt{Bool} \to \texttt{Nat} \to Y}.pbn$	$p \,:\, \texttt{Bool} \to \texttt{Nat} \to Y$
I choose n	n	$=$	$\Lambda X.\, \lambda f^{X \to X}.\lambda x^X.?$	$n \,:\, \forall X.\, (X \to X) \to X \to X$
You choose f	n	$=$	$\Lambda X.\, \lambda f^{X \to X}.\lambda x^X.fa_1$	$f \,:\, X \to X$
I choose a_1	a_1	$=$	$?$	$a_1 \,:\, X$
You choose f	a_1	$=$	fa_2	$f \,:\, X \to X$
I choose a_2	a_2	$=$	$?$	$a_2 \,:\, X$

At this point, you have three abstracted term variables to choose from: p, f, and x. In the last discussion you played x, yielding the Church numeral $\underline{2}$; if you play f you'll be on the road to defining some higher numeral. What happens if you play p?

I choose t	t	$=$	$\Lambda Y.\, \lambda p^{\texttt{Bool} \to \texttt{Nat} \to Y}.?$	$t \,:\, \forall Y.\, (\texttt{Bool} \to \texttt{Nat} \to Y) \to Y$
You choose p	t	$=$	$\Lambda Y.\, \lambda p^{\texttt{Bool} \to \texttt{Nat} \to Y}.pbn$	$p \,:\, \texttt{Bool} \to \texttt{Nat} \to Y$
I choose n	n	$=$	$\Lambda X.\, \lambda f^{X \to X}.\lambda x^X.?$	$n \,:\, \forall X.\, (X \to X) \to X \to X$
You choose f	n	$=$	$\Lambda X.\, \lambda f^{X \to X}.\lambda x^X.fa_1$	$f \,:\, X \to X$
I choose a_1	a_1	$=$	$?$	$a_1 \,:\, X$
You choose f	a_1	$=$	fa_2	$f \,:\, X \to X$
I choose a_2	a_2	$=$	$?$	$a_2 \,:\, X$
You choose p	a_2	$=$	$pb'n'$	$p \,:\, \texttt{Bool} \to \texttt{Nat} \to Y$

This suggests some kind of attempt on your part to recursively store a pair within the component of a pair—and certainly, should you succeed, the answer to our surjectivity question would be no. But we have a violation of the copycat rule: the type of a_2, being X, is of the form "$\cdots X$", but the body $pb'n'$ is of type Y.

So the copycat rule means that after my first move, either b or n, determining in which component (`Bool` or `Nat`) to play, you are 'stuck' playing in that component: you can never 'get back out' by playing p. Thus winning strategies on $\texttt{Bool} \mathbin{\tilde{\times}} \texttt{Nat}$ are in bijection with pairs of winning strategies, one on `Bool` and one on `Nat`. So (informally), we have deduced that in (product- and unit-free) system F, closed, η-long, β-normal terms of type $\texttt{Bool} \mathbin{\tilde{\times}} \texttt{Nat}$ are in bijection with pairs of closed, η-long, β-normal terms, one of type `Bool` and one of type `Nat`.

Bibliography

[Abr97] Samson Abramsky. Semantics of interaction. In A. M. Pitts and P. Dybjer, editors, *Semantics and Logics of Computation*, Publications of the Newton Institute. Cambridge University Press, 1997.

[AHM98] Samson Abramsky, Kohei Honda, and Guy McCusker. A fully abstract game semantics for general references. In *Proceedings, Thirteenth Annual IEEE Symposium on Logic in Computer Science*, pages 334–344. IEEE Computer Society Press, 1998.

[AJ94] Samson Abramsky and Radha Jagadeesan. Games and full completeness for multiplicative linear logic. *Journal of Symbolic Logic*, 59(2):543 – 574, June 1994. Also appeared as Technical Report 92/24 of the Department of Computing, Imperial College of Science, Technology and Medicine.

[AJM94] Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. Full abstraction for PCF (extended abstract). In Masami Hagiya and John C. Mitchell, editors, *Theoretical Aspects of Computer Software. International Symposium TACS'94*, number 789 in Lecture Notes in Computer Science, pages 1–15, Sendai, Japan, April 1994. Springer-Verlag.

[AM95] Samson Abramsky and Guy McCusker. Games and full abstraction for the lazy λ-calculus. In *Proceedings, Tenth Annual IEEE Symposium on Logic in Computer Science*, pages 234–243. IEEE Computer Society Press, 1995.

[AM96] Samson Abramsky and Guy McCusker. Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions (extended abstract). In *Proceedings of 1996 Workshop on Linear Logic*, volume 3 of *Electronic notes in Theoretical Computer Science*. Elsevier, 1996.

[AM98] Samson Abramsky and Guy McCusker. Full abstraction for Idealized Algol with passive expressions. To appear in Theoretical Computer Science, 1998.

[AM99a] S. Abramsky and G. McCusker. Game semantics. In U. Berger and H. Schwichtenberg, editors, *Computational Logic: Proceedings of the 1997 Marktoberdorf International Summer School*, NATO ASI Series F: Computer and Systems Sciences. Springer-Verlag, 1999.

[AM99b] Samson Abramsky and Paul-André Melliès. Concurrent games and full completenes. In *Proceedings, Fourteenth Annual IEEE Symposium on Logic in Computer Science*, pages 431–442. IEEE Computer Society Press, 1999.

[Bar84] Henk P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North-Holland, revised edition, 1984.

[Bar92] Henk P. Barendregt. Lambda calculi with types. In S. Abramsky, D. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science*. Oxford University Press, 1992.

[BC82] Gérard Berry and P.-L. Curien. Sequential algorithms on concrete data structures. *Theoretical Computer Science*, 20:265–321, 1982.

[Bla72] A. Blass. Degrees of indeterminacy of games. *Fundamenta Mathematica*, 77, 1972.

[Bla92] Andreas Blass. A game semantics for linear logic. *Annals of Pure and Applied Logic*, 56, 1992.

[CGW89] T. Coquand, C. Gunter, and G. Winskel. Domain theoretic models of polymorphism. *Information and Computation*, 81:123–167, 1989.

[Con76] J. H. Conway. *On Numbers and Games*. Academic Press, 1976.

[Cro94] Roy Crole. *Categories for Types*. Cambridge University Press, 1994.

[DHR96] Vincent Danos, Hugo Herbelin, and Laurent Regnier. Game semantics and abstract machines. In *Proceedings, Eleventh Annual IEEE Symposium on Logic in Computer Science* [IEE96], pages 394–405.

[Fel86] Walter Felscher. Dialogues as a foundation for intuitionistic logic. In D. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic, Volume III*, pages 341–372. D. Reidel Publishing Company, 1986.

[Gan93] Robin Gandy. Dialogues, blass games and sequentiality for objects of finite type. Unpublished manuscript, 1993.

[Gir71] J.Y. Girard. Une extension de l'interprétation de gödel à l'analyse, et son application à l'élimination des coupures dans l'analyse et la théorie des types. In J.E. Fenstad, editor, *Proceedings of the Scandinavian Logic Symposium*. North-Holland, 1971.

[Gir86] Jean-Yves Girard. The system F of variable types, fifteen years later. *Theoretical Computer Science*, 45:159–192, 1986.

[Gir87] Jean-Yves Girard. Linear Logic. *Theoretical Computer Science*, 50(1):1–102, 1987.

[Gir89] Jean-Yves Girard. Towards a geometry of interaction. In J. W. Gray and Andre Scedrov, editors, *Categories in Computer Science and Logic*, volume 92 of *Contemporary Mathematics*, pages 69–108. American Mathematical Society, 1989.

[GLT89a] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Cambridge University Press, 1989.

[GLT89b] Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989.

[Gun92] Carl A. Gunter. *Semantics of Programming Languages: Structures and Techniques.* Foundations of Computing. The MIT Press, 1992.

[Her] Hugo Herbelin. Using types to control reduction in system f. Working draft.

[HM99] R. S. Harmer and G. A. McCusker. A fully abstract game semantics for finite nondeterminism. In *Proceedings, Fourteenth Annual IEEE Symposium on Logic in Computer Science.* IEEE Computer Society Press, 1999.

[HO93] J. M. E. Hyland and C.-H. L. Ong. Fair games and full completeness for Multiplicative Linear Logic without the mix-rule. Unpublished manuscript, 1993.

[HO94] J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF: I, II and III. Accepted for publication in Information and Computation in 1997, 1994.

[Hoa85] Charles Anthony Richard Hoare. *Communicating Sequential Processes.* Prentice-Hall, 1985.

[How] Howard. The formulae-as-types notion of construction. In *[?].*

[HS80] J.R. Hindley and J.P. Seldin, editors. *To H.B. Curry: Essays on combinatory logic, Lambda Calculus and Formulism.* Academic Press, 1980.

[HS99] J. M. E. Hyland and A. Schalk. Astract games for linear logic. In *Proceedings, Category Theory in Computer Science*, Electronic Notes in Computer Science, 1999. to appear.

[Hug97] Dominic J. D. Hughes. Games and definability for System F. In *Proceedings, Twelfth Annual IEEE Symposium on Logic in Computer Science* [IEE97], pages 76–86.

[HY97] Kohei Honda and Nobuko Yoshida. Game theoretic analysis of call-by-value computation. In P. Degano, R. Gorrieri, and A. Marchietti-Spaccamela, editors, *Proceedings, 25th International Colloquium on Automata, Languages and Programming: ICALP '97*, volume 1256 of *Lecture Notes in Computer Science*, pages 225–236. Springer-Verlag, 1997.

[Hyl88] J. Martin E. Hyland. A small complete category. *J. Pure Appl. Logic*, 40:135–165, 1988.

[Hyl97] J. M. E. Hyland. Game semantics. In A. M. Pitts and P. Dybjer, editors, *Semantics and Logics of Computation*, Publications of the Newton Institute. Cambridge University Press, 1997.

[IEE96] IEEE Computer Society Press. *Proceedings, Eleventh Annual IEEE Symposium on Logic in Computer Science*, 1996.

[IEE97] IEEE Computer Society Press. *Proceedings, Twelfth Annual IEEE Symposium on Logic in Computer Science*, 1997.

[KP93] G. Kahn and G. D. Plotkin. Concrete domains. In *Böhm Festschrift Special Issue*, Theoretical Computer Science. 1993. First appeared (in French) as technical report 338 of INRIA-LABORIA, 1978.

[Lai97] Jim Laird. Full abstraction for functional languages with control. In *Proceedings, Twelfth Annual IEEE Symposium on Logic in Computer Science* [IEE97], pages 58–67.

[Lam68] Lambek. Deductive systems and categories, i. *Math. Systems Theory*, 2:278–318, 1968.

[Law69] Lawvere. Adjointness in foundations. *Dialectica*, 23:281–269, 1969.

[LL78] P. Lorenzen and K. Lorenz. *Dialogische Logik.* Wissenschaft Buchgeselleschaft, 1978.

[LM84] G. Longo and E. Moggi. Cartesian closed categories of enumerations for effective type-structures. In G. Kahn, D. MacQueen, and G.D. Plotkin, editors, *Symposium on Semantics of Data Types*, volume 173 of *LNCS*. Springer Verlag, 1984.

[McC79] N. McCracken. *An investigation of a programming language with a polymorphic type structure.* PhD thesis, Syracuse University, 1979.

[McC96a] Guy McCusker. *Games and Full Abstraction for a Functional Metalanguage with Recursive Types.* PhD thesis, Department of Computing, Imperial College, University of London, 1996.

[McC96b] Guy McCusker. Games and full abstraction for FPC. In *Proceedings, Eleventh Annual IEEE Symposium on Logic in Computer Science* [IEE96], pages 174–183.

[McC98] G. McCusker. *Games for recursive types.* BCS Distinguished Dissertation. Cambridge University Press, 1998.

[Mil80] Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science.* Springer-Verlag, Berlin, 1980.

[MO99] A. S. Murawski and C.-H. L. Ong. Exhausting Strategies, Joker Games and Full Completeness for IMLL with Unit. To appear in the ENTCS proceedings of *Category Theory and Computer Science 1999*, 34 pp., ftp-able from Ong's homepage, 1999.

[Nic94] H. Nickau. Hereditarily sequential functionals. In *Proceedings of the Symposium on Logical Foundations of Computer Science: Logic at St. Petersburg*, Lecture notes in Computer Science. Springer, 1994.

[Nic96] H. Nickau. *Hereditarily Sequential Functionals: A Game-Theoretic Approach to Sequentiality.* PhD thesis, Universität-Gesamthochschule-Siegen, 1996.

[NO] H. Nickau and C.-H L. Ong. Games for first order and second order intuitionistic propositional proofs. Working draft.

[O'H96] P O'Hearn. Parametric polymorphism. In *Domains and Denotational Semantics: History, Accomplishments and Open Problems*, volume 59, pages 227–256. 1996.

[Ong96] C. H. L. Ong. A semantic view of classical proofs: type-theoretic, categorical and denotational characterizations. In *Proceedings, Eleventh Annual IEEE Symposium on Logic in Computer Science* [IEE96], pages 230–241.

[PA93] G. Plotkin and M. Abadi. A logic for parametric polymorphism. *Lecture Notes in Computer Science*, 664, 1993.

[Pit87] A. M. Pitts. Polymorphism is set theoretic, constructively. In D.H. Pitt, A. Poignè, and D. E. Rydchard, editors, *Category Theory and Computer Science, Proc. Edinburgh 1987*, Lecture Notes in Computer Science, pages 12–39. Springer-Verlag, 1987.

[Plo80] Gordon Plotkin. Lambda-definability in the full type hierarchy. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 363–373. Academic Press, 1980.

[Rey74] John C. Reynolds. Towards a theory of type structure. In *Paris colloquium on programming*, volume 19 of *LNCS*, 1974.

[Rey84] John C. Reynolds. Polymorphism is not set-theoretic. In Gilles Kahn, David B. MacQueen, and Gordon D. Plotkin, editors, *Semantics of Data Types*, volume 173 of *Lecture Notes in Computer Science*, pages 145–156, Berlin, June 1984. Springer-Verlag.

[Sco72] D. S. Scott. Continuous lattices. In E. Lawvere, editor, *Toposes, Algebraic Geometry and Logic*, pages 97–136. Springer-Verlag, Berlin, 1972. Lecture Note in Mathematics 274.

[Sto79] J. E. Stoy. *Denotational Semantics: The Scott-Strachey Approach to Programming Language Theory.* The MIT Press, 1979. The MIT Press Series in Computer Science.

[Str67] C. Strachey. Fundamental concepts in programming languages. Lecture notes for the International Summer School in Computer Programming, Copenhagen, 1967.

[TS96] A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory.* Cambridge University Press, 1996. Cambridge Tracts in Computer Science, 43.