MACQUARIE University

Game Semantics for Free Cartesian Closed Categories

Ruben Zilibowitz

INTRODUCTION

Typed functional programming has become an important part of modern day software development. Part of the reason for this is the mathematical nature of such languages. The origin of functional programming is the **lambda calculus** [5]. It is well know there is an isomorphism between **typed lambda calculus** and **free cartesian closed categories** (CCCs) [1]. **Game semantics** [2] provides us new tools for understanding properties of functional languages and proving correctness of programs, using ideas from game theory. We wish to focus on the game semantics of a free cartesian closed category as a definitive account of this has not yet appeared.

WHAT IS A GAME?

A game is played between two players, usually called **O** and **P** (Opponent and Player). A game is played on an **arena** which is an ordered labelled forest. Player O always starts first. The last player to make a valid move wins. Player O starts by choosing some root vertex in the forest. In a **linear game** [2] play proceeds by each player choosing a successor vertex to the last choice by the opposite player. In a **pointer game** [3] player P has additional moves available; at each move they can play a successor vertex to any previous O-move, not only the last one. Linear games can model linear types, whilst pointer games are needed to model more general purpose functional programming. There are ways of composing winning strategies, giving a categorical structure; the algorithms for doing this are called **interaction** [4].

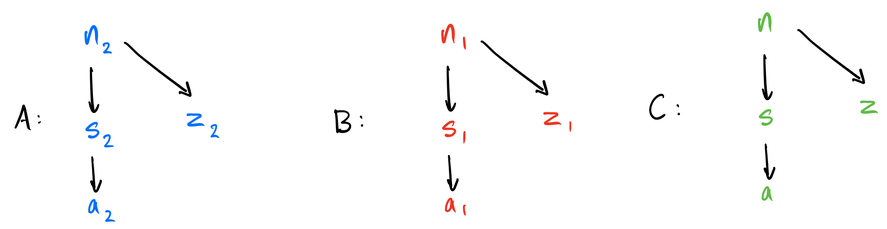
METHODS

Complement the theoretical approach by developing software for:
* Generating examples of games and strategies.
* Composition of winning strategies, aka “interaction".
* Lambda calculus computations such as beta reduction.
* Visualising games and strategies.

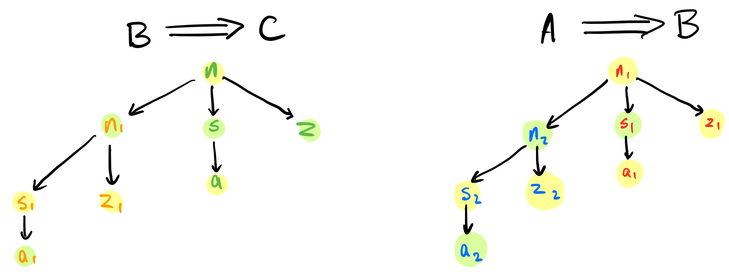
Compare results of equivalent algorithms such as interaction and beta reduction. Make conjectures, prove theorems, generalise.

EXAMPLE

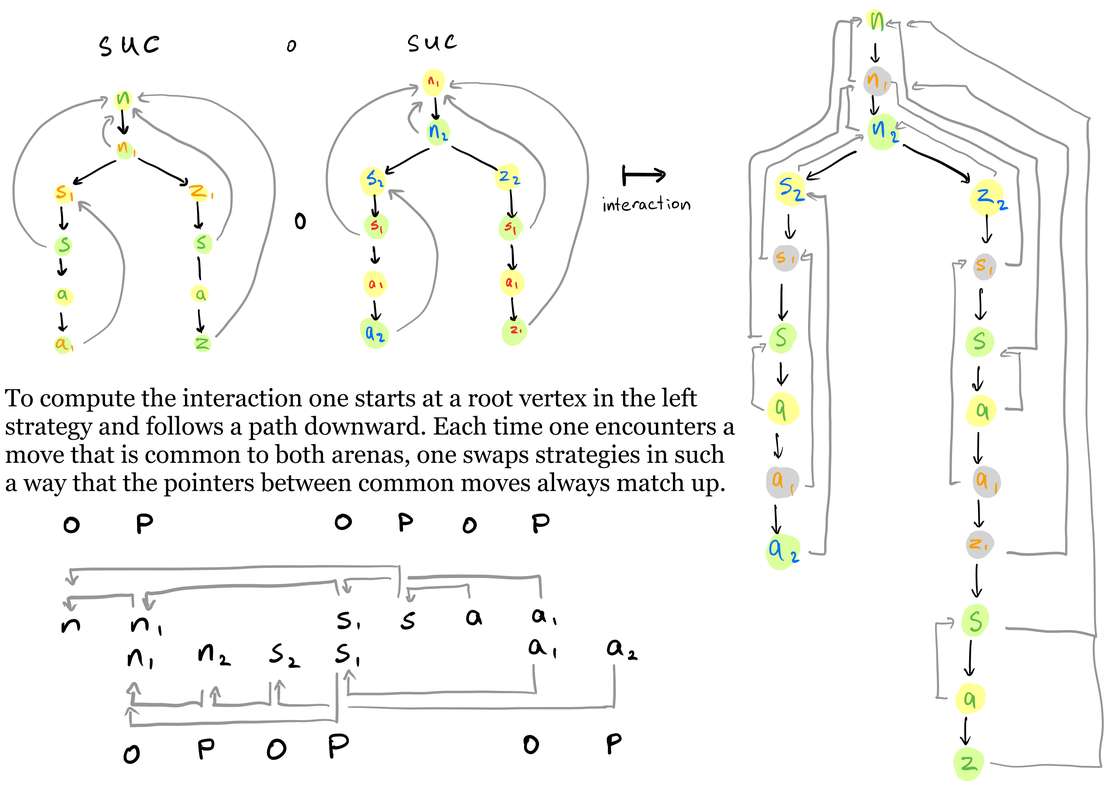
Starting with three arenas as follows:



We obtain arenas for the “internal-homs” as follows:



Winning P-strategies in pointer games are given by ordered labelled forests with **justification pointers** [4]. These are the grey arrows pointing to ancestor vertices in the below forests. Every P-move must point to some earlier O-move whose label is a parent in the arena. This example shows composition of two `suc` strategies, so named because they compute the successor on Church numerals.



To compute the interaction one starts at a root vertex in the left strategy and follows a path downward. Each time one encounters a move that is common to both arenas, one swaps strategies in such a way that the pointers between common moves always match up.

RESULTS

We start by considering a free CCC, F, over the group G with two elements. It is a theorem that we can obtain a natural isomorphism between identity functors $\phi : 1_F \to 1_F$ with the following property: for any lambda term f of type A we have $\phi_A \cdot f \mapsto_{\beta\star} \beta\eta(f)$, where $\beta\eta(f)$ denotes the short-beta-long-eta normal form [6] of f and $\mapsto_{\beta\star}$ denotes the system of reductions involving beta, projection, and cancelling constants.

If we instead consider starting with the group G being the free group on a countably infinite number of generators, we can extend our definition above so that $\phi_x = x$ where x is a labelled atomic type on the left and an invertible constant on the right. This leads to the following correspondences.

Game Semantics	Typed Lambda Calculus
Arena	Type with labelled atoms, aka “labelled type”
Arena for labelled type A	$\phi_A \circ \phi_A$
Winning P-strategy for arena A	Term of labelled type B, such that A is the tree of constants* in $\phi_B \circ \phi_B$
Winning P-strategy corresponding to term $f : A \to B$	$\phi_{A\to B} \cdot f \equiv_\beta \phi_B \circ f \circ \phi_A^{-1}$, for labelled types A and B
Interaction	Intermediate step in composition of terms
Interaction of winning P-strategies $g : B_2 \to C$ after $f : A \to B_1$	$(\phi_{B_2\to C} \cdot g) \circ (\phi_{A\to B_1} \cdot f)$, for labelled types A, B, C

In the right hand column in the above, we can normalise with $\mapsto_{\beta\star}$ and then compute a tree representation* of the lambda terms, closely related to Böhm trees [5], in order to visualise the data more clearly. These results provide us with some simple tools for performing important computations. There is the possibility of generalisations to finite coproducts among other things. The simplicity and generalisability make these ideas compelling.

FUTURE WORK

* Cartesian closed categories with finite coproducts.
* Parallelism techniques for interaction of winning strategies.
* Infinite games and recursive types.
* Cartesian closed categories with a (weak) natural numbers object

[1] Lambek, J., & Scott, P.J. (1986). Introduction to higher order categorical logic.
[2] Hyland, M. (1997). Game Semantics. In A. Pitts & P. Dybjer (Eds.), *Semantics and Logics of Computation*, pages. 131-184. Cambridge University Press.
[3] Harmer, R., Hyland, M., & Melliès, P. (2007). Categorical Combinatorics for Innocent Strategies. *22nd Annual IEEE Symposium on Logic in Computer Science (LICS 2007)*, 379-388.
[4] D.J.D.Hughes (2000). *Hypergame Semantics: Full Completeness for System F*. (Doctoral dissertation, Oxford University).
[5] Barendregt, H. (1985). The lambda calculus - its syntax and semantics. *Studies in logic and the foundations of mathematics*.
[6] Simpson, A.K. (1995). Categorical completeness results for the simply-typed lambda-calculus. *TLCA*.

ABN 90 952 801 237 | CRICOS Provider 00002J