An embedding of Concrete Data Structures into Dialogue Games

Clément Jacq and Paul-André Melliès

Laboratoire IRIF, Université Paris Diderot

Abstract

Dialogue games were recently introduced by Mellies [2014-2016] as an attempt to unify the two traditional (but still disconnected) frameworks for game semantics:

- concrete data structures and sequential algorithms introduced by Berry and Curien in the early 1980s
- arena games introduced by Hyland, Nickau and Ong in the mid 1990s.

The key idea underlying the notion of dialogue game is that every move m of an arena game should be decomposed as a pair $m = (alpha, val)$ consisting of a cell α and of a value val.
This decomposition enables one to interpret every move m of an arena game as the action for Player or for Opponent of filling a given cell with a given value. An important novelty of dialogue games with respect to concrete data structures is that a cell or a value may be either Opponent or Player. The only constraint is that a value should always fill a cell of the same polarity, and that it should always justify a cell of the opposite polarity.

Once understood this decomposition of an arena move as a pair consisting of a cell and of a value, the connection between dialogue games and arena games becomes essentially immediate. Despite the fact that they share the same philosophy of cells and values, the connection between dialogue games and concrete data structures is more subtle.

In this talk, we will explain how to connect the sequential algorithm model designed by Berry and Curien in the early 1980s with this recent notion of dialogue games. To that purpose, we start from the nice graph game model of intuitionistic linear logic designed by Hyland and Schalk in the early 2000s.
We exhibit a number of interesting and sometimes unexpected structures of this model by embedding it into a specific intuitionistic hierarchy of dialogue game. This embedding enables us to recover the graphical notations of positions in dialogue games. Take for instance the game $A = (Bool \otimes Bool) \multimap Bool$. The translation as a graph:

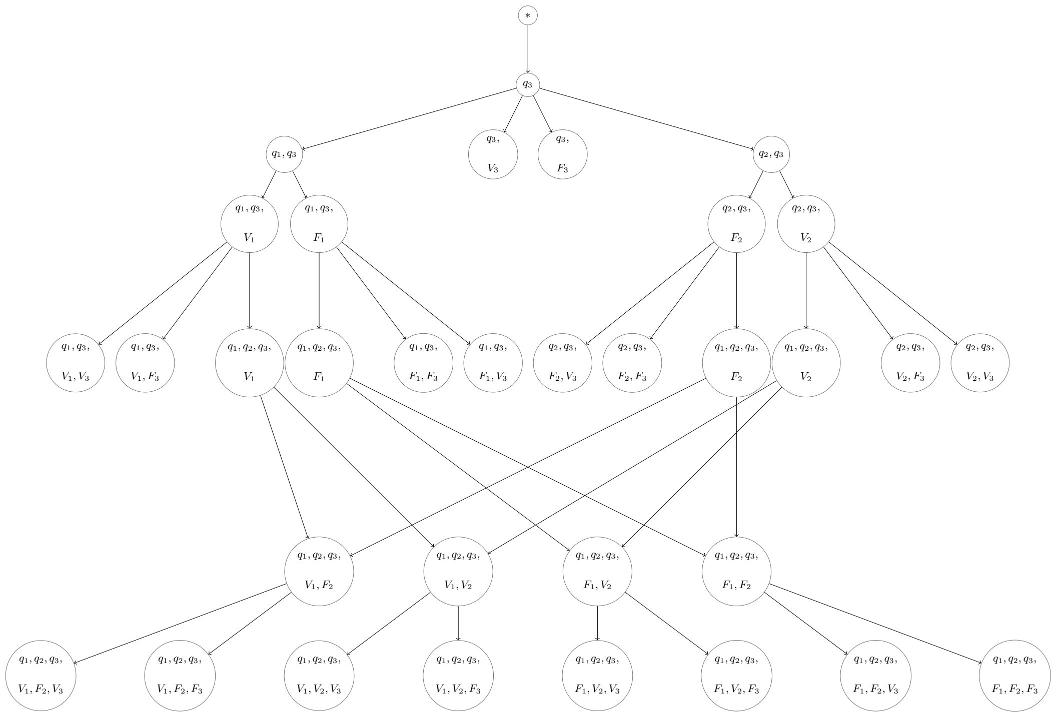

Fig. 1: The graph game for the type $(Bool \otimes Bool) \multimap Bool$

By embedding it in the dialogue game $B = \neg((\neg\neg(1+1)) \otimes (\neg\neg(1+1)) \otimes \neg(1+1))$ every position x of the graph game A may be depicted as follows:

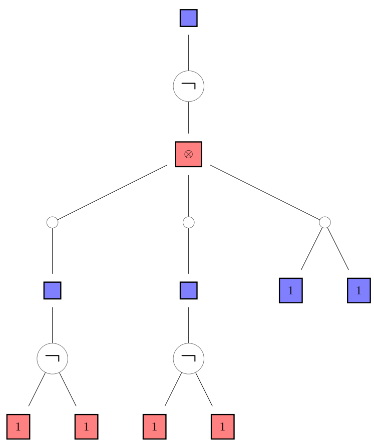

Fig. 2: The dialogue game for the type $(Bool \otimes Bool) \multimap Bool$