1 Abramsky-Jagadeesan games: a review

1.1 Games and strategies

We use the definition of games found $\otimes$ in [3], which is essentially that of [2] and [1].

Definition 1.1. A *game* A is a tuple

$$(M_A, \lambda_A, b_A, P_A)$$

where

- M_A is a set of moves.
- $\lambda_A \colon M_A \to \{O, P\}$ is a function designating each move as a P-move or an O-move.
- $b_A \in \{O, P\}$ specifies a starting player.
- $P_A \subset M_A^*$ is a prefix-closed set of legal plays in the game such that P-moves and O-moves alternate.

Notation 1.2. If $b_A = P$, we say that A is a *positive game*, and if $b_A = O$, we say that A is a *negative game*.

Given $n \in \mathbb{N}$, write $P_A(n)$ for the set of plays in P_A of length n.

We shall define a function $\zeta_A \colon P_A \to \{O, P\}$ by setting $\zeta_A(s) = O$ if s ends in an O-move and $\zeta_A(s) = P$ if s ends in a P-move. ζ_A then tells us which player played last, or who is responsible for the current state of the game. If $\zeta_A(s) = O$, we say that s is an *O-play*; if $\zeta_A(s) = P$, we say that s is a *P-play*.

We write ϵ for the empty sequence in M_A^*.

We now give a definition of a (player) strategy for a game. A strategy can be thought of as a partial function from O-plays to P-moves, telling the player which move to play in a particular position. We follow [3] and others in presenting a strategy as the game-tree of the positions that can be obtained by following the strategy; our presentation differs slightly from that in [3] in that our strategies contain all O-plays and P-plays that arise, rather than the P-plays on their own. The two definitions are clearly equivalent, but ours will generalize more readily later on.

Definition 1.3. Let A be a game. A *strategy* for A is a prefix-closed subset $\sigma \subset P_A$ such that:

- σ contains at most one P-reply to any O-play. I.e., if $s \in \sigma$ is an O-play and a, b are P-moves such that $sa, sb \in \sigma$, then $a = b$.
- σ contains all O-replies to P-plays. I.e., if $s \in \sigma$ is a P-play and a is an O-move such that $sa \in P_A$, then $sa \in \sigma$.
- If $\sigma = \emptyset$ then $b_A = P$ and if $\epsilon \in \sigma$ then $b_A = O$.

We say the strategy σ is *total* if it is non-empty and contains P-replies to all its O-plays; i.e., if $s \in \sigma$ is an O-play then there is some (necessarily unique) P-move a such that $sa \in \sigma$.

1.2 Connectives on games

1.2.1 Negation

If A is any game, we write ${}^{\perp}A$ for the game obtained by switching the roles of O and P. In particular, if P is a positive game then ${}^{\perp}P$ will be a negative game, and if N is a negative game then ${}^{\perp}N$ will be a positive game.

Officially:

$$ {}^{\perp}A = (M_A, \neg \circ \lambda_A, \neg b_A, P_A) $$

where $\neg\colon \{O, P\} \to \{O, P\}$ switches O and P.

1.2.2 Tensor and implies

Our main recipe for combining two games into one will be playing games in parallel. This means that we set up copies of each game side by side and a starting player, and on each go the player whose turn it is may choose to play in either game, so long as their move makes a valid play in that game.

Let us see what this means in practice:

- If M, N are negative games, then the negative game obtained by playing M, N in parallel is called the *tensor product* $M \otimes N$. Observe that player P is forced to play in whichever game player O has just played

in (since the only moves available in the other game are O-moves), but that player O may play in either game when it is his turn.

- There is a dual to the tensor product. If P, Q are positive games, then the positive game obtained by playing P, Q in parallel is called the *par* $P \parr Q$. In this game, it is player P who may play in either of the two games when it is her turn, while player O is restricted to playing in whichever game player P has just played in.

- Now suppose that P is a positive game and that N is a negative game. In order to play these two games in parallel, we must be careful to choose a starting player. Let us suppose for now that O starts. Then O must make the first move in the game N. Thereafter, it is P who may choose which game to make her next move in, while O must play in whichever game P has just played in.

  The case we are most interested in is when the game P is the negation ${}^{\perp}M$ of some negative game M. Then the negative game obtained by playing ${}^{\perp}M$ and N in parallel is called the *implication* of N from M, and is written $M \multimap N$.

Exercise 1. *Suppose M, N are negative games and suppose that player P has strategies for the games M and $M \multimap N$. Show how she may combine these strategies to obtain a strategy for the game N.*

We shall now give a formal definition of playing two games in parallel.

Notation 1.4. Let X, Y be sets. If $s \in (X \sqcup Y)^*$ is any sequence made up of terms from both X and Y then we write $s|_X$ for the subsequence of s made up of all those terms that came from X and $s|_Y$ for the subsequence of s made up of all those terms that came from Y.

Now suppose $S \subset X^*$ is a set of sequences of elements of X and that $T \subset Y^*$ is a set of sequences of elements of Y. We write $S \| T$ for the subset of $(X \sqcup Y)^*$ made up of those alternating sequences s such that $s|_X \in S$ and $s|_Y \in T$.

Definition 1.5. Let A, B be games, and let $b \in \{O, P\}$ be a choice of starting player.

The *game with starting player b obtained by playing A,B in parallel* is given by

$$A \|_b B = (M_A \sqcup M_B, \lambda_A \sqcup \lambda_B, b, P_A \| P_B)$$

If M, N are negative games, we write $M \otimes N$ for $M \|_O N$.

If P, Q are positive games, we write $P \parr Q$ for $P \|_P Q$.

If M, N are negative games, we write $M \multimap N$ for $N \|_O {}^{\perp}M$.

1.2.3 Sequoid

If A, B are games, then the we can weaken the game $A \|_b B$ by requiring that the starting player b play the first move in game A. In this case, we must have $b = b_A$, or the resulting game has no legal plays at all. For this reason, we may suppress mention of the starting player b in our notation.

Definition 1.6. We use the notation from [3]. If X, Y are sets and $S \subset X^*, T \subset Y^*$ then we write $S \|_L T \subset S \| T$ for the set of all alternating sequences $s \in S \| T$ that are either empty or have their first term in S. Then we define

$$A \between B = (M_A \sqcup M_B, \lambda_A \sqcup \lambda_B, b_A, P_A \|_L P_B)$$

The starting player b_B for the game B is then the player who has the ability to switch games, while the other player must play in whichever game their opponent has just played in. If $b_B = O$ then we write $A \oslash B$ for $A \between B$ and if $b_B = P$ then we write $A \lhd B$ for $A \between B$.

In particular, if M, N are negative games, then $M \oslash N$ is the weakening of the tensor $M \otimes N$ in which player O is forced to make his first move in the game M.

1.3 Categorical Semantics

We build a category $\mathcal{G}$ whose objects are negative games. If M, N are negative games, then the morphisms $M \to N$ will be strategies for $M \multimap N$.

1.3.1 Identity morphisms

Let N be a negative game. Then $N \multimap N$ is given by the games N and ${}^{\perp}N$ played in parallel, with player P switching games. The identity morphism $N \to N$ is the *copycat strategy*:

- O makes a move in N.

- P plays that move in ${}^{\perp}N$.
- O replies to the move in ${}^{\perp}N$.
- P plays O's reply in N.
- ... and so on.

This strategy is total, since P always has a reply to O's move (play that move in the other game).

1.4 Composition of morphisms

Suppose L, M, N are negative games and we have strategies σ for $L \multimap M$ and τ for $M \multimap N$. We want to be able to compose these strategies to give us a strategy $\tau \circ \sigma$ for $L \multimap N$.

Let us put ourselves in the position of player P. We need to know which move she makes in response to each opponent move. Player P first of all sets up the games $L \multimap M$ and $M \multimap N$ side by side. The rough idea is that she can use the strategy σ on the left hand side and the strategy τ on the right hand side, while playing the copies of M and ${}^{\perp}M$ against each other in order to obtain plays in $L \multimap N$. Let us look at this a bit more closely.

We are playing in the game $L \multimap N$.

- O must make his first move in the game N.
- P copies this move as the first move in her copy of $M \multimap N$.
- P now uses the strategy τ for $M \multimap N$ to reply to O's move, within the game $M \multimap N$. If her reply is in the game N, then she copies her reply over to $L \multimap N$, and play continues.
- Otherwise, her reply is in the game ${}^{\perp}M$. In that case, she copies that move over as the first move in her copy of $L \multimap M$.
- She then uses her strategy σ to reply to that move in $L \multimap M$. If her reply is in the game ${}^{\perp}L$, then she makes that play in the game $L \multimap N$, and play continues with the next opponent move.
- Otherwise, her reply is in the game M. She copies that move over to $M \multimap N$ and replies to it using τ.

- She continues in this way until she makes a play either in ${}^{\perp}L$ or in N. The then waits for O's response, and replies to that in the same way.

There is a chance that P ends up switching back and forth between the copies of M and ${}^{\perp}M$ forever, and never makes her reply to O, even if the strategies σ and τ were total. For this reason, the composition of total strategies need not be total, and so we must deal with general strategies in this games model. We shall shortly consider ways to get around this.

Definition 1.7. Let L, M, N be games and let $\sigma\colon L \to M, \tau\colon M \to N$ be strategies. Define

$$\tau \| \sigma = \{s \in (M_L \sqcup M_M \sqcup M_N)^* \colon s|_{L,M} \in \sigma, s|_{M,N} \in \tau\}$$

$\tau\|\sigma$ is the set of all plays that occur when P plays in $L \multimap M$ according to σ and in $M \multimap N$ according to τ, identifying the moves in M in the two games. We can recover the composed strategy $\tau \circ \sigma$ for $L \multimap N$ by forgetting the moves in M:

$$\tau \circ \sigma = \{s|_{L,N} \colon s \in \tau \| \sigma\}$$

If $K \xrightarrow{\rho} L \xrightarrow{\sigma} M \xrightarrow{\tau} N$ are morphisms, then we have

$$\begin{aligned} \tau \circ (\sigma \circ \rho) &= \{s|_{K,N} \colon s \in (K \sqcup L \sqcup M \sqcup N)^*, s|_{K,L} \in \rho,\ s|_{L,M} \in \sigma,\ s|_{M,N} \in \tau\} \\ &= (\tau \circ \sigma) \circ \rho \end{aligned}$$

so this composition is associative, and it is easy to check that the copycat strategy is the identity.

1.5 Sequoidal category structure

We can endow $\mathcal{G}$ with a richer categorical structure, using the tensor and sequoid connectives. In investigating this, we arrive at the definition of a *sequoidal category*, introduced in [4].

1.5.1 Symmetric monoidal closed category structure

The operation $\otimes$ makes $\mathcal{G}$ into a symmetric monoidal closed category:

First, we need to show that $\otimes\colon \mathcal{G} \times \mathcal{G} \to \mathcal{G}$ is a functor. This comes down to finding, for each pair $K \xrightarrow{\sigma} L, M \xrightarrow{\tau} N$ of strategies, a strategy

$$K \otimes M \xrightarrow{\sigma \otimes \tau} L \otimes N$$

The strategy $\sigma \otimes \tau$ is the obvious one: reply to opponent moves in M or N according to τ and reply to opponent moves in K or L according to σ. It is easy to see that this is indeed a functor.

The tensor unit is the empty game $I = \{\emptyset, \emptyset, O, \{\epsilon\}\}$ with no moves. The unitors and associators are given by copycat strategies in the natural way.

This monoidal category is then closed with respect to $\multimap$: we have a natural bijection

$$\mathcal{G}(L \otimes M, N) \cong \mathcal{G}(L, M \multimap N)$$

1.5.2 Strict morphisms

We might try and do something similar with the sequoid. However, in this case we do not get a functor $\oslash\colon \mathcal{G} \times \mathcal{G} \to \mathcal{G}$. Indeed, suppose $K \xrightarrow{\sigma} L, M \xrightarrow{\tau} N$ are strategies as before. We can try and combine them as before to get a strategy for

$$(K \oslash M) \multimap (L \oslash N)$$

where P replies to O-moves in M or N using τ and replies to O-moves in K or L using σ. But consider this sequence of events:

- O makes his first move in L, as required by the definitions of $\oslash$ and $\multimap$.
- The strategy τ tells P to reply to O in the game L, rather than in K.
- After P has played in L, O switches games and plays a move in N.
- Now the strategy σ tells P to reply to this move in M.

Unfortunately for P, she cannot make a move in M, since no move has been made in K yet.

We can fix this problem by requiring that the morphism τ be *strict*. Given games K, L, a *strict morphism* is a strategy for $K \multimap L$ such that player P's first move is in the game K. In our case above, if τ is a strict strategy then P will make her first move in K and there will be no subsequent problems. The copycat morphism is certainly strict, as is the composition of two strict morphisms, so we get a full-on-objects subcategory $\mathcal{G}_s$ of $\mathcal{G}$, where the objects are the negative games and the morphisms are the strict morphisms. The sequoid then gives us a functor

$$\oslash\colon \mathcal{G}_s \times \mathcal{G} \to \mathcal{G}_s$$

In the world of programming languages, strict morphisms correspond to functions that always evaluate their argument. Suppose we define a game $\mathbf{N}$ by

$$\mathbf{N} = (\mathbb{N} \sqcup \{q\}, (n \mapsto P,\ q \mapsto O), O, \{\epsilon, q\} \cup \{qn \colon n \in \mathbb{N}\})$$

then a strategy for this game corresponds to a program that returns a natural number. A strict morphism $\mathbf{N} \to \mathbf{N}$, then, corresponds to a program that takes in one natural number and returns another one:

- O plays the move q in the game $\mathbf{N}$ on the right.
- P plays the move q in the game ${}^{\perp}\mathbf{N}$ on the left.
- O responds to P by playing a natural number m in ${}^{\perp}\mathbf{N}$.
- P then plays a natural number n in $\mathbf{N}$.

Since the number n is allowed to depend on m, the strict morphisms $\mathbf{N} \to \mathbf{N}$ correspond to functions $\mathbb{N} \to \mathbb{N}$.

There are non-strict strategies for $\mathbf{N} \multimap \mathbf{N}$, where P responds immediately to O by playing a natural number in $\mathbf{N}$ on the right, ending the game immediately. These correspond to non-strict functions, which always return the same value without evaluating their argument.

Remark 1.8. In the programming language point of view, there is a difference between a function that returns a constant value without evaluating the argument and a function that evaluates the argument but nevertheless returns the same value every time. In the latter case, the program will fail to terminate if the program for evaluating the argument does not terminate, whereas the non-strict version may terminate anyway, since the non-terminating bit of code is never run.

1.5.3 Sequoidal categories

The definition of a sequoidal category is a generalization of the category of games we are constructing.

The functor $\oslash \colon \mathcal{G}_s \times \mathcal{G} \to \mathcal{G}$ actually gives us a monoidal-category action of $\mathcal{G}$ on $\mathcal{G}_s$. That is, the functor $\_ \oslash (M \otimes N)$ is (naturally isomorphic to) the composition of the functors $\_ \oslash M$ and $\_ \oslash N$. In other words, for each L, M, N we have isomorphisms

$$L \oslash (M \otimes N) \cong (L \oslash M) \oslash N$$

satisfying certain coherence conditions. Once again, the isomorphism is the obvious copycat one - in both games, O must start in the game L but thereafter may switch to any game.

References

[1] Samson Abramsky and Radha Jagadeesan. Games and full completeness for multiplicative linear logic. *CoRR*, abs/1311.6057, 2013.

[2] Andreas Blass. A game semantics for linear logic. *Annals of Pure and Applied Logic*, 56(1–3):183 – 220, 1992.

[3] Martin Churchill, Jim Laird, and Guy McCusker. Imperative programs as proofs via game semantics. *CoRR*, abs/1307.2004, 2013.

[4] J. Laird. A categorical semantics of higher-order store. In *Proceedings of CTCS '02*, number 69 in ENTCS. Elsevier, 2002.