What is the real logic of games after all?

Giorgi Japaridze*

Abstract

Over the past several decades many versions of game semantics for logic were introduced, none of which, however, has been perceived as a refinement of or a reasonable alternative to classical semantics. Apparently there are three major reasons:

1) Those semantics are often not very natural and artificially adjusted to some non-semantical considerations, such as justifying certain existing axiomatic systems (intuitionism, linear logic).

2) Non-artificial game semantics either do not yield any new logic but classical logic, or have to totally depart from the original meaning of the language of logic as a formal counterpart of natural language that allows us to talk about the surrounding world.

3) The technical results (axiomatization, decidability/enumerability etc.) proven so far regarding the logics generated by relatively natural game semantics have not been sufficiently comprehensive and have been usually limited to only some rather restricted fragments of the language.

The present paper introduces and studies a game semantics which, as the author believes, does not suffer from the above shortcomings. The corresponding logic is fully axiomatized in a language that can be considered the union of the languages of classical and linear logics. This logic is a conservative extension of classical logic and of a certain version of linear logic, thus unifying these two logics (in a broad sense of the term "linear logic") within the framework of one general semantical approach.

Key words: game logics, game semantics, linear logic.

1 Introduction

The goal of the present paper is to introduce a mathematically elegant and intuitively convincing semantics for parallel game combinations and announce some main results regarding the corresponding logic, such as a full axiomatization of it at both the propositional and the predicate levels. These results are stated without proofs as their high degree of technical involvement, in conjunction with the space limitations, did not allow the author to include any

*Supported by Summer Research Grant from Villanova University

serious proofs here. A lengthy extended paper with detailed proofs for the main technical results claimed in this article is currently in the process of preparation.

Beginning from Lorenzen [8], a variety of game semantics have been studied by different authors. With some technical differences in game-playing rules, common to most of those semantics is the idea of considering sentences as games between two players — *proponent*, who is trying to "defend" the sentence, and *opponent*, who is "attacking" the sentence, where disjunction or existential quantifier signifies proponent's move/choice, and conjunction or universal quantifier is opponent's move/choice. The main motivation for Lorenzen was to find a semantical justification of Heyting's intuitionistic logic. After trial and error, the goal of obtaining completeness of that logic with respect to Lorenzen's semantics was achieved [5]. The value of this achievement, however, seems dubious, as it was obtained by artificially tuning the semantics to adjust it to the goal, at the expense of losing its naturality and modeling potential. After all, virtually every formal system can be "justified" with some sort of specially designed technical semantics, and the whole point is how natural and usable the semantics is with respect to which completeness is proven. Among the non-convincing features of Lorenzen's game semantics is treating atoms as "black boxes" ([2]) that cannot be opened by proponent, so that an atom can be defended only if opponent has previously claimed its truth, as well as a series of other supplementary rules governing the conditions under which statements can be attacked or re-attacked, yielding somewhat confusing asymmetries between the two players.

Among the most natural game semantics known so far is the one introduced by Blass [4]. The idea of understanding negation as role switch and disjunction, conjunction and quantifiers as choice operations, employed in [4], was not new and had been used, in an essentially the same way, by virtually all the authors who tackled game semantics. Blass, however, was apparently the first to add to considerations another, parallel way of combining games, thus opening the possibility of a game-semantical justification for the distinction between the two (additive and multiplicative) sorts of operators in substructural logics. Blass's semantics no longer treats atoms as "black boxes" and it enjoys perfect symmetry between the two players. As a result of this symmetry, however, the corresponding game logic collapses to classical logic as long as atoms are interpreted as elementary (no-move) games or even finite-depth games, — the natural requirement that, by the way, was present in Lorenzen's approach. To avoid trivialization of the logic of games, Blass insisted that atoms be interpreted as infinite-depth, indetermined (and thus highly non-constructive) games. This precluded his semantics from being considered a reasonable and appealing alternative to classical semantics (what infinite-depth game should be the interpretation of 0=0 after all?).

With atoms interpreted as infinite-depth games, Blass's semantics yields a substructural logic that is strictly between Affine logic (linear logic plus weakening) and classical logic. Unfortunately, however, this logic has not been axiomatized, and no other reasonable syntactic characterization (such as a decision algorithm) of it has been found, either. So far we only have two partial completeness results. In particular, it was shown in [4] that:

1. The multiplicative propositional fragment of Blass's logic (the propositional fragment where only multiplicative operators are allowed) is decidable, and it is a proper extension of the corresponding fragment of Affine logic.

2. The fragment whose formulas can be represented with purely additive (multiplicative-free) sequents is also decidable and it is exactly the corresponding fragment of Affine logic.

The question on decidability or recursive enumerability of the full (additive-multiplicative) logic, or even the full propositional fragment of it, however, remains open.

The fact that Blass's logic takes us far beyond Girard's original linear logic was perceived as its shortcoming by some authors who believed that linear logic must be the "right logic". Abramsky and Jagadeesan [1] revised Blass's semantics by considering a version of games that are always by opponent to start. This way, at the expense of losing symmetry between players and somewhat "denaturalizing" Blass's semantics, the goal of disvalidating the weakening rule has been achieved, even though the resulting logic still remains stronger than linear logic. In particular, in [1] it was shown that the multiplicative propositional fragment of the logic induced by this semantics is exactly the corresponding fragment of linear logic plus the mix rule (the latter being inadmissible in linear logic). Just as in [4], however, the question on decidability or enumerability of the full additive-multiplicative logic, or the propositional fragment of it, was not answered in [1].

Thus, neither linear logic, nor affine logic or even linear logic with mix can be considered the logic of games. What is then the "real logic of games"? The present paper contains an answer, with a full axiomatization of such a logic without any restrictions on its language. The language of the logic of games I consider here is a proper extension of the language of additive-multiplicative linear logic. There are two features that make the former stronger than the latter:

1) My language contains an additional sort of operators that allow us to capture games with imperfect information.

2) It also has two sorts of atoms: one for any games, and one for elementary (zero-depth) games. Elementary games are, in fact, nothing but predicates in the classical sense, which makes my semantics a generalization/refinement of classical semantics and makes my logic a conservative extension of classical logic, with classical operators understood as their multiplicative counterparts.

Besides considering a more expressive language of the logic of games, some of the features that distinguish my approach from that of Blass include:

1) I allow both of the players to have legal moves in the same position. This way we get a more general concept of games, as well as a more natural semantics for multiplicatives as parallel game combination operators. For example, my understanding of the multiplicative conjunction $A \wedge B$ is a game playing which

means playing the two games A and B in parallel, where both players are free to make a move in either component as long as that move is legal for that component. On the other hand, in Blass's games, only one of the players may have a legal move in any position. To comply to this requirement, Blass would not allow opponent to move in A as long as proponent has a legal move in B. This somewhat disqualifies plays over $A \wedge B$ as really parallel and independent plays of A and B: what can be done in one of the conjuncts depends on what is happening in the other conjunct.

2) I, as in [6], require that proponent's strategies be effective. This eliminates the necessity to appeal to infinite-depth, indetermined games in completeness proofs. This also yields a more interesting semantics as it makes games an adequate formalization of the intuitive notion of interactive computational problems, thus making the concept of winnability a generalization of computability/decidability. The requirement of effectiveness of proponent's strategies was absent in [4]. In fact, it would be redundant within the framework of Blass's approach as it would not affect the concept of validity in any way.

3) My games can contain variables. This makes games a generalization of classical predicates and enables us to define quantifiers as natural unary operations on games.

2 The propositional logic of racefree games

From now on I will refer to opponent and proponent with the more technical names P_0 and P_1, respectively.

Let us fix three infinite sets of expressions: *Mov* (*moves*), *Var* (*variables*) and *Dom* (*domain*). Without loss of generality we may assume that $Dom = \{0, 1, 2, \ldots\}$ and Mov is the set of all strings over the alphabet $\{0\text{-}9, a\text{-}z, .\}$. A move suffixed with "$P_0$" or "$P_1$" is called a *labeled move*, where the label P_i indicates which player has made the move. Sequences of labeled moves we call *runs*, and finite runs call *positions*.

The letters $\alpha, \beta, \ldots$ will be used as metavariables for moves, $\Gamma, \Delta, \ldots$ for runs, $x, y, \ldots$ for variables and $X, Y, \ldots$ for finite sets of variables. An *instance* of such an X is a function $f : X \rightarrow Dom$. If $X_1 \subseteq X_2$, f_1 is an instance of X_1 and f_2 is an instance of X_2 that agrees with f_1 on X_1, then f_1 is said to be the *restriction* of f_2 to X_1 and f_2 is an *extension* of f_1 to X_2.

Definition 2.1 Where n is a natural number, an n-ary *game* is a tuple

$$[M_0, M_1, X, F]$$

such that:

- M_0 and M_1 are functions that send positions to subsets of *Mov*. We call the elements of $M_i\langle\Gamma\rangle$ P_i's *legal moves* in position Γ. A *legal run* is a run where, for any initial segment $\langle\Gamma, \alpha P_i\rangle$, we have $\alpha \in M_i\langle\Gamma\rangle$. We assume that if Γ is an illegal run, then $M_0\langle\Gamma\rangle = M_0\langle\Gamma\rangle = \emptyset$.

- X is a set of n (distinct) variables.
- F is a function that returns one of the values P_0 or P_1 for every pair (Γ, f), where Γ is a legal run and f is an instance of X. F tells us who the winner is in a given run for a given instance of X. This function is extended to all runs by stipulating that an illegal run is lost by the player who has made the first illegal move.

A *strict game* is a game where, in every position, at most one of the players has legal moves. Games that only have a finite number of legal runs we call *finite*. A game with no infinitely long legal runs is said to be *finite-depth*. A finite-depth game is *bounded-depth* if there is an upper bound for the lengths of its legal runs; this bound is called the *depth* of the game. Games without any legal moves we call *elementary*. Every predicate $R(X)$ on Dom can be understood as the elementary game $(\emptyset, \emptyset, X, F)$ such that, for any instance f of X, $F(\langle\rangle, f) = P_1$ iff $R(f)$ is true (here $\emptyset$ stands for the function that returns $\emptyset$ for every position, and $\langle\rangle$ is the empty run). This makes predicates special cases of games, and classical logic a special case of the logic of games.

Here are definitions of some basic propositional operations on games:

Definition 2.2 (*Negation*)

$$\neg[M_0, M_1, X, F] = [M_0{}', M_1{}', X, F'],$$

defined as follows. Let $\overline{\Gamma}$ mean the result of interchanging the labels P_0 and P_1 in run Γ. Then, for every Γ, we have:

- $M_0'\langle\Gamma\rangle = M_1\langle\overline{\Gamma}\rangle$ and $M_1'\langle\Gamma\rangle = M_0\langle\overline{\Gamma}\rangle$.
- For every instance f of X, $F'(\Gamma, f) = P_1$ iff $F(\overline{\Gamma}, f) = P_0$.

Thus, $\neg G$ is nothing but G with the roles of the players switched.

Definition 2.3 (*Multiplicative conjunction*)

$$[M_0^l, M_1^l, X^l, F^l] \wedge [M_0^r, M_1^r, X^r, F^r] = [M_0, M_1, X^l \cup X^r, F],$$

defined as follows. For $d \in \{l, r\}$, let Γ^d stand for the result of removing from Γ all labeled moves except those that start with $d.$, and then deleting the prefix $d.$ in the remaining labeled moves. Then:

- For every position Γ, $i \in \{0, 1\}$ and $d \in \{l, r\}$, we have $M_i\langle\Gamma\rangle = \{d.\alpha \mid d \in \{l, r\},\ \alpha \in M_i^d\langle\Gamma^d\rangle\}$.
- For every instance f of $X^l \cup X^r$, whenever Γ is a legal run, we have $F(\langle\Gamma\rangle, f) = P_1$ iff $F^l(\langle\Gamma^l\rangle, f^l) = F^r(\langle\Gamma^r\rangle, f^r) = P_1$, where f^l and f^r are the restrictions of f to X^l and X^r, respectively.

Playing $G^l \wedge G^r$ means playing, in parallel, the two games G^l and G^r, where P_1 has to win in both components to be the winner. A move $l.\alpha$ (resp. $r.\alpha$) means that the move α was made in the left (resp. right) component. Obviously, when applied to elementary games, the operations $\neg$ and $\wedge$ again produce elementary games and, remembering that the latters are nothing but predicates, these two operators act exactly as the classical negation and conjunction. However, when not restricted to elementary games, their behavior rather resembles that of the corresponding *multiplicative* operators of linear logic.

Definition 2.4 (*Additive conjunction*)

$$[M_0^l, M_1^l, X^l, F^l] \sqcap [M_0^r, M_1^r, X^r, F^r] = [M_0, M_1, X^l \cup X^r, F],$$

where:

- $M_0\langle\rangle = \{l, r\}$, $M_1\langle\rangle = \emptyset$ and, for any run Γ, $i \in \{0,1\}$ and $d \in \{l, r\}$, we have $M_i\langle dP_0, \Gamma\rangle = M_i^d\langle\Gamma\rangle$.
- For every instance f of $X^l \cup X^r$, we have: $F(\langle\rangle, f) = P_1$ and, for any run Γ and $d \in \{l, r\}$, we have $F(\langle dP_0, \Gamma\rangle, f) = F^d(\langle\Gamma\rangle, f^d)$, where f^d is the restriction of f to X^d.

As its name indicates, $\sqcap$ corresponds to the additive conjunction of linear logic. In $G^l \sqcap G^r$ the opening move is by P_0, who has to choose between l (*left*) and r (*right*), after which the play continues according to the rules of the chosen subgame. If no initial choice was made, this game is considered won by P_1.

Definition 2.5 (*Trivially lost game*) $\bot$ is the elementary 0-ary game whose only possible run (empty run) is lost by P_1.

The remaining propositional game operations can be defined in terms of the above four operations as follows:

Definition 2.6

Additive disjunction	$A \sqcup B = \neg(\neg A \sqcap \neg B)$
Multiplicative disjunction	$A \vee B = \neg(\neg A \wedge \neg B)$
Multiplicative implication	$A \rightarrow B = \neg A \vee B$
Trivially won game	$\top = \neg\bot$

One can see that the only difference between $\sqcup$ and $\sqcap$ is that, in the former, it is P_1 rather than P_0 who makes the initial choice between *left* and *right*. $A \vee B$, just like $A \wedge B$, means a parallel combination of A and B, with the difference that, in order to win in $A \vee B$, for P_1 it is sufficient to win in one of the components. $\top$ is an elementary game automatically won by P_1. As for $\rightarrow$, it corresponds to the intuition of *reducing* game B to game A. To win this game, for P_1, means to win B as long as P_0 wins game A against P_1.

Let Γ and Δ be runs and $i \in \{0,1\}$. We say that Δ is a P_i-*delay* of Γ iff

- for each $j \in \{0,1\}$, the subsequence of P_j-labeled moves of Γ is the same as that of Δ, and
- for any $n, m \geq 1$, if the nth P_i-labeled move is made later than (is to the right of) the mth P_{1-i}-labeled move in Γ, then so is it in Δ;

in other words, in Δ each player has made the same sequence of moves as in Γ, only, in Δ, P_i might have been acting with some delay. Then we say that game $[M_0, M_1, X, F]$ is *racefree* if, for each $i \in \{0,1\}$, whenever f is an instance of X, Γ is a run with $F(\Gamma, f) = P_i$ and Δ is a P_i-delay of Γ, we have $F(\Delta, f) = P_i$.

Intuitively, a racefree game is a game where speed does not give any advantage to a player: if it has a "good" move in a given position, then that move will as well remain "good" after the other player has made several moves, so that no opportunities can be missed by not acting fast enough (see Example 2.8

below). Notice that all strict games are racefree. One can verify that $\neg$, $\wedge$ and $\sqcap$ preserve the racefree property of games. The closure of elementary games under these three operations forms a natural class of finite racefree games. And the closure of elementary games under $\neg$ and $\sqcap$ forms a class of finite strict games.

My philosophy is to allow P_0 follow any strategies while require that P_1 only follows effective strategies. A way to materialize this approach is to assume that P_1 is a (deterministic) Turing machine M with a special additional state called $MOVE$ and an additional read-only tape (along with the usual read/write work tape) called the *run tape*. The latter serves as a dynamic input for the machine, spelling, at any moment, the current position of the game, i.e. the sequence of moves made by the players so far. For simplicity we assume that M has no halt state, — the effect of halting, if we wish, can be achieved by infinitely looping without reading and writing.

A *configuration* is a tuple (S, W, w, R, r), where S is a state of M, W is contents of the work tape, w is a position of the read/write head on the work tape, R is contents of the run tape, and r is a position of the read head on the run tape. In the *initial configuration*, the run tape is blanc and the work tape contains an instance of the variables of the game played by the machine; in that configuration, as this can be understood, S is the start state of the machine and w, r are the leftmost positions on the corresponding tapes.

A configuration $C' = (S', W', w', R', r')$ that can legally follow the current configuration $C = (S, W, w, R, r)$ is said to be a *successor* of C. The conditions that S', W', w', r' should here satisfy must be clear: these four parameters are uniquely determined by the previous configuration and the transition function of the machine in the standard way. The only nondetermined and nonstandard parameter is R', which will be affected by the possible moves that P_0can make during the transition from C to C'. The condition that we require R' to satisfy is the following:

- If $S \neq MOVE$, then R' is the result of appending to R any (possibly empty) finite sequence of P_0-labeled moves.
- If $S = MOVE$, then R' is the result of appending to R any finite sequence of P_0-labeled moves followed by αP_1, where α is (the string spelled by) the initial segment of W up to the first blanc symbol.

Thus, M makes a move by entering the $MOVE$ state, in which case the contents of the work tape from its beginning up to the first blanc cell of the tape will be interpreted as the move it has made. While M is making this move (or just going to the next step without a move), any finite number of any moves by P_0 may be appended to the contents of the run tape, corresponding to the intuition that not only the strategy of P_0 can be arbitrary, but its speed as well, as long as P_0 can only make a finite number of moves per time unit. This sort of nondeterministic transition from R to R' creates different branches of computation that M may have to follow, each branch being a sequence of configurations where the first configuration is the initial configuration and every other configuration is a successor of the previous one. Each branch incrementally spells a (possibly infinite) run on the run tape. We call such a run a *run with* M. Then we say that

Definition 2.7 Machine M *wins* game $[M_0, M_1, X, F]$ iff for every instance f of X and every run Γ with M (with f on its work tape in the initial configuration) we have $F(\Gamma, f) = P_1$.

A game for which such a winning Turing machine M exists will be said to be *effectively winnable*.

The logic GM_{prop} that I am going to introduce has the following connectives: $\neg, \wedge, \vee, \rightarrow, \bot, \top, \sqcap, \sqcup$ and two sorts of infinitely many propositional atoms: *elementary atoms* and *general atoms*. An *interpretation* is a function * which assigns, to each elementary (resp. general) atom P, an elementary (resp. racefree) game P^*. This assignment naturally extends to all formulas by commuting with all operators: $\bot^* = \bot$; $(\neg Q)^* = \neg(Q^*)$; $(Q \wedge R)^* = Q^* \wedge R^*$; $(Q \sqcap R)^* = Q^* \sqcap R^*$; etc. A formula Q is said to be *valid* iff the game Q^* is effectively winnable for every interpretation *.

Example 2.8 The formula $A \vee \neg A$ is valid, whether A be an elementary atom, a general atom or a compound formula. Using the notation Γ^d from Definition 2.3, any legal run Γ of $A^* \vee \neg A^*$ will have two disjoint subruns: Γ^l and Γ^r that are legal runs of A^* and $\neg A^*$, respectively, with Γ won by P_1 if and only if either Γ^l or Γ^r is won. This game will be won by a machine that uses the following strategy: Ignore the input (an instance of the variables of the game); keep polling the run tape; every time you see that P_0 has made a move $l.\alpha$ (resp. $r.\alpha$), make the move $r.\alpha$ (resp. $l.\alpha$). In other words, keep mimicking, in A^*, P_0's moves in $\neg A^*$, and vice versa.

Of course, unless A^* is a strict game, the runs Γ^l and Γ^r may not be exactly symmetric. For example, if both of the players have a legal move in A^*, a fast P_0 can make moves in both A^* and $\neg A^*$ before the machine gets a chance to copy P_0's first move. Generally this could make winning the game impossible. But what saves the day is that the definition of interpretation requires A^* to be racefree. Even though the machine cannot ensure that $\Gamma^r = \overline{\Gamma^l}$ (see Definition 2.2 for notation $\overline{\Gamma}$), a little analysis of the situation can convince us that the above mimicking strategy, however, guarantees that Γ^r is a P_1-delay of $\overline{\Gamma^l}$. So that, if Γ^l is lost, which implies that $\overline{\Gamma^l}$ is won, then Γ^r, as a P_1-delay of $\overline{\Gamma^l}$, will be won, which makes the mimicking strategy a winning strategy for $A^* \vee \neg A^*$.

On the other hand, $A \sqcup \neg A$ is not valid, even when A is an elementary letter. To see this, it is sufficient to consider an interpretation that interprets A as an undecidable predicate.

By an *elementary formula* we mean a formula that does not contain general atoms, $\sqcap$ or $\sqcup$. A *surface occurrence* of a subformula is an occurrence that is not in the scope of $\sqcap$ or $\sqcup$. Understanding $A \rightarrow B$ as an abbreviation for $\neg A \vee B$, an occurrence is *negative*, if it is in the scope of an odd number of occurrences of $\neg$; otherwise it is *positive*. The *elementarization* of a formula Q is the result of replacing in Q all positive (resp. negative) surface occurrences of general atoms by $\bot$ (resp. $\top$), all (positive or negative) surface occurrences of the form $R_1 \sqcap R_2$ by $\top$ and all (positive or negative) surface occurrences of the form $R_1 \sqcup R_2$ by $\bot$.

Definition 2.9 The *axioms* of **logic** GM_{prop} are all the elementary formulas that are classical tautologies, and the *rules* of inference are:

1. $Q' \vdash Q$, where Q' has at most two occurrences of a certain elementary atom a, and Q is the result of replacing in Q' (each occurrence of) this atom by a general atom A.

2. $Q' \vdash Q$, where Q' is the result of replacing in Q a negative (resp. positive) surface occurrence of a subformula $R_1 \sqcap R_2$ (resp. $R_1 \sqcup R_2$) by R_1 or R_2.

3. $Q', \vec{S} \vdash Q$, where Q' is the elementarization of Q and $\vec{S}$ is a (the minimal) set of formulas satisfying the following condition: If Q has a positive (resp. negative) surface occurrence of a subformula $R_1 \sqcap R_2$ (resp. $R_1 \sqcup R_2$), then, for each $i \in \{1, 2\}$, $\vec{S}$ contains the result of replacing this occurrence in Q by R_i.

Theorem 2.10 $GM_{prop} \vdash Q$ iff Q is valid. Moreover:

a) There is an effective procedure that takes a GM_{prop}-proof of a formula Q and constructs a Turing machine that wins Q^* for every interpretation *.

b) If $GM_{prop} \not\vdash Q$, then Q^* is not effectively winnable for some interpretation * that interprets general atoms as finite strict games of depth ≤ 2.

One can show that GM_{prop} is decidable in polynomial space. How does GM_{prop} compare with other logics? Obviously its restriction to elementary formulas is classical propositional logic. And its *general-base fragment*, — the fragment where elementary atoms are forbidden, — is a proper extension of propositional Affine logic. Here are two examples of principles underivable in the latter but provable in GM_{prop}:

$$(A \wedge B) \vee (C \wedge D) \rightarrow (A \vee C) \wedge (B \vee D)$$

$$\big(A \wedge (C \sqcap D)\big) \sqcap \big(B \wedge (C \sqcap D)\big) \sqcap \big((A \sqcap B) \wedge C\big) \sqcap \big((A \sqcap B) \wedge D\big) \rightarrow (A \sqcap B) \wedge (C \sqcap D)$$

As these are apparently the shortest possible formulas that separate GM_{prop} from Affine logic, we can say that the latter is "almost complete" with respect to our semantics. As for the distance between the general-base fragment of GM_{prop} and classical logic, it is huge as GM_{prop} does not prove $A \rightarrow A \wedge A$.

3 The predicate logic of racefree games

Here come two quantifier-style operations on games.

Definition 3.1 (*Additive universal quantifier*)

$$\sqcap x[M_0, M_1, X, F] = [{M_0}', {M_1}', X\text{-}\{x\}, F'],$$

where:

- ${M_0}'\langle\rangle = Dom$, ${M_1}'\langle\rangle = \emptyset$ and, for any run Γ, $i \in \{0,1\}$ and $d \in Dom$, we have ${M_i}'\langle dP_0, \Gamma\rangle = M_i\langle\Gamma\rangle$.
- For every instance f of X-$\{x\}$, we have: $F'(\langle\rangle, f) = P_1$ and, for any run Γ and $d \in Dom$, we have $F'(\langle dP_0, \Gamma\rangle, f) = F(\langle\Gamma\rangle, f^d)$, where f^d is the extension of f to X such that $f^d(x) = d$.

The above operation is similar to $\sqcap$ and its meaning can be described by $\sqcap xG(x) = G(0)\sqcap G(1)\sqcap G(2)\sqcap \ldots$. At the beginning of the play over $\sqcap xG$, which can be written as $\sqcap xG(x)$, P_0 has to choose one particular element d of *Dom*. Then the play continues according to the rules of $G(d)$, where the latter is $G(x)$ with the possible instances of the variables restricted to those that return d for x. Note that effective winnability of a game G containing variables can be redefined in terms of effective winnability of 0-ary games by stipulating that G is effectively winnable iff its $\sqcap$-closure is so.

Definition 3.2 (*Blind universal quantifier*)

$$\forall x[M_0, M_1, X, F] = [M_0, M_1, X\text{-}\{x\}, F'],$$

where for every instance f of X-$\{x\}$ and any run Γ, we have $F'(\langle\Gamma\rangle, f) = P_1$ iff for every extension f' of f to X, $F(\langle\Gamma\rangle, f') = P_1$.

The operation $\forall x$, which obviously acts as the classical universal quantifier for elementary games, is not exactly similar to $\wedge$ when not restricted to elementary games, and has no counterpart in linear logic (see Section 4 for a definition of the "multiplicative universal quantifier"). This operation in fact produces what is called *games with imperfect information* ([2]). Intuitively, $\forall xG(x)$ is the same as $\sqcap xG(x)$, with the difference that, in $\forall xG(x)$, the particular value d that P_0 "selects" and for which the game $G(d)$ should be won, is invisible to P_1, so that it has to play blindly in a way that guarantees success no matter what this value d is.

For each of the above two sorts of universal quantifier we have a corresponding existential quantifier:

Definition 3.3 *Additive existential quantifier* $\sqcup xA(x) = \neg\sqcap x\neg A(x)$
Blind existential quantifier $\exists xA(x) = \neg\forall x\neg A(x)$

We define one more, rather technical, game operation:

Definition 3.4 (*Substitution of variables*) Suppose $G = [M_0, M_1, X, F]$ is a game, $\{y_1, \ldots, y_n\} = Y$ are (distinct) variables, and $z_1, \ldots, z_n$ is a sequence of n (not necessarily distinct) variables. We may assume that $X \cap Y = \{y_1, \ldots, y_k\}$ $(0 \leq k \leq n)$ and X-$Y = \{x_1, \ldots, x_m\}$ $(m \geq 0)$, so that $X = \{x_1, \ldots, x_m, y_1, \ldots, y_k\}$. Then

$$G(y_1/z_1, \ldots, y_n/z_n) = [M_0, M_1, X', F'],$$

where:

- $X' = \{x_1, \ldots, x_m, z_1, \ldots, z_k\}$.
- For any position Γ and instance f of X', $F'(\Gamma, f) = F(\Gamma, g)$, where g is the instance of X such that $g(x_i) = f(x_i)$ and $g(y_j) = f(z_j)$, for each i, j with $1 \leq i \leq m$, $1 \leq j \leq k$.

The language of logic GM_{pred} that I am going to introduce, in addition to the propositional connectives of GM_{prop}, has four quantifiers: $\forall, \exists, \bigsqcap, \bigsqcup$. The set *Var* that we fixed at the beginning of Section 2 serves as its set of individual variables. The language has two sorts of predicate letters: *elementary letters* and *general letters*. With each (elementary or general) letter P is associated an n-tuple of distinct variables, called its *canonical tuple*, where the number n is called the *arity* of P. We assume that for each n, there are infinitely many n-ary elementary and general letters. The expression $P(x_1, \ldots, x_n)$, where P is an n-ary elementary (resp. general) letter and $x_1, \ldots, x_n$ are (not necessarily distinct) variables, is called an elementary (resp. general) *atom*. If here $(x_1, \ldots, x_n)$ is the canonical tuple of P, then $P(x_1, \ldots, x_n)$ is an (elementary or general) *canonical atom*. *Formulas* are built in the standard way, only, for simplicity of analysis, we assume that no formula can contain both a free and a bound occurrence of the same variable.

An *interpretation* for a formula Q is a function * which assigns an elementary (resp. racefree) game $P^* = [M_0, M_1, X, F]$ to each elementary (resp. general) canonical atom $P(x_1, \ldots, x_n)$, such that, if a variable x is in X but not among $x_1, \ldots, x_n$, then x does not occur in Q. This assignment extends to all formulas as follows: For an atom $P(y_1, \ldots, y_n)$, $(P(y_1, \ldots, y_n))^* = P^*(x_1/y_1, \ldots, x_n/y_n)$, where $(x_1, \ldots, x_n)$ is the canonical tuple of P; $(Q \wedge R)^* = Q^* \wedge R^*$; $(\bigsqcap x Q)^* = \bigsqcap x(Q^*)$; etc.

A formula Q is said to be *valid* iff the game Q^* is effectively winnable for every interpretation * for Q. Redefining the terms "elementary formula", "surface occurrence" and "elementarization" for the predicate language, an *elementary formula* now is a formula which does not contain general letters, $\sqcap, \sqcup, \bigsqcap$ or $\bigsqcup$; a *surface occurrence* of a subformula is an occurrence that is not in the scope of one of these four additive operators; and the *elementarization* of a formula Q is the result of replacing in Q all positive (resp. negative) surface occurrences of general atoms by $\perp$ (resp. $\top$), all (positive or negative) surface occurrences of the form $R_1 \sqcap R_2$ or $\bigsqcap x R$ by $\top$, and all (positive or negative) surface occurrences of the form $R_1 \sqcup R_2$ or $\bigsqcup x R$ by $\perp$.

Definition 3.5 The *axioms* of **logic** GM_{pred} are all the elementary formulas provable in classical predicate calculus, and the *rules* of inference are:

1. $Q' \vdash Q$, where Q' has at most two occurrences of a certain elementary n-ary letter a, and Q is the result of replacing in Q' this letter by a general n-ary letter A.

2a. $Q' \vdash Q$, where Q' is the result of replacing in Q a negative (resp. positive) surface occurrence of a subformula $R_1 \sqcap R_2$ (resp. $R_1 \sqcup R_2$) by R_1 or R_2.

2b. $Q' \vdash Q$, where Q' is the result of replacing in Q a negative (resp. positive) surface occurrence of a subformula $\bigsqcap x R(x)$ (resp. $\bigsqcup x R(x)$) by $R(y)$ for some y (y not bound in Q').

3. $Q', \vec{S} \vdash Q$, where Q' is the elementarization of Q and $\vec{S}$ is a (minimal, finite) set of formulas satisfying the following conditions:

  a) If Q has a positive (resp. negative) surface occurrence of a subformula $R_1 \sqcap R_2$ (resp. $R_1 \sqcup R_2$), then, for each $i \in \{1,2\}$, $\vec{S}$ contains the result of replacing this occurrence in Q by R_i;

  b) If Q has a positive (resp. negative) surface occurrence of a subformula $\bigsqcap x R(x)$ (resp. $\bigsqcup x R(x)$), then $\vec{S}$ contains the result of replacing this occurrence in Q by $R(y)$ for some y not occurring in Q.

Theorem 3.6 $GM_{pred} \vdash Q$ iff Q is valid. Moreover:

a) There is an effective procedure that takes a GM_{pred}-proof of a formula Q and constructs a Turing machine that wins Q^* for every interpretation * for Q.

b) If $GM_{pred} \not\vdash Q$, then Q^* is not effectively winnable for some interpretation * for Q that interprets general atoms as finite strict games of depth ≤ 2.

Remember the definitions of the game properties "racefree", "bounded-depth", "finite-depth", "finite", "elementary, "strict" from Section 2. These definitions can be extended to game operations by stipulating that an operation is racefree (bounded-depth, etc.) if it preserves the racefree (bounded-depth, etc.) property of games. One can show that all of the game operations that we introduced so far are racefree, bounded-depth and finite-depth. So that, in view of the clause (b) of Theorem 3.6, GM_{pred} can be called not only the logic of racefree games, but also the logic of bounded-depth racefree games or the logic of finite-depth racefree games.

Because of space limitations, I did not have a chance to give a sufficiently detailed explanation of the (very natural) intuition behind the racefree property of games. To the reader who has difficulty visualizing this concept, I would suggest to understand "racefree games" as elements of the closure of elementary games under $\neg, \wedge, \vee, \rightarrow, \sqcap, \sqcup, \bigsqcap, \bigsqcup, \forall$ and $\exists$, which forms a natural subclass of racefree bounded-depth games, and for which GM_{pred} remains complete.

Obviously GM_{pred} is recursively enumerable. One can also show that the $\forall, \exists$-free fragment of GM_{pred} is decidable (in polynomial space), even though it contains the quantifier-type operators $\bigsqcap$ and $\bigsqcup$.

It is easy to see that the elementary fragment of GM_{pred} (the fragment where general letters and additives are forbidden) is exactly classical predicate logic, so that GM_{pred} is a conservative extension of classical logic. This is so because classical semantics is, in fact, nothing but the projection of our game semantics on the elementary fragment of the language. To appreciate the technical strength of Theorem 3.6, note that the language of GM_{pred} is by an order of magnitude more expressive than the classical language as it includes the language of additive-multiplicative linear logic as well.

The latter (the language of linear logic) is the fragment of the language of GM_{pred} where elementary letters and $\forall, \exists$ are forbidden. One can show that the corresponding fragment of GM_{pred} is a proper extension of predicate Affine logic; in particular, it is exactly logic ET that I introduced in [6]. This is a decidable logic that was defined nonaxiomatically, through a decision algorithm. The semantics with respect to which ET was shown to be sound and complete in [6], was similar to, but not the same as, the present game semantics. In particular, the games considered in [6], just as Blass's [4] games, were strict, and hence the multiplicative operations there had to be strict, while the multiplicative operations defined in the present paper are not strict.

Finally, the fragment of GM_{pred} where general letters are forbidden, — the *elementary-base fragment*, — is exactly logic L introduced in [7], where I axiomatically defined that logic with a (yet another) semantics in mind, with respect to which L was proven to be sound and complete. That semantics understood sentences as *tasks* that are to be carried out by an agent (machine, robot) working as a slave for its master (user, environment).

Thus, there are at least four technically quite different semantics that yield the same (relevant fragments of) logic GM_{pred}: our present game semantics, classical semantics and the two different semantics employed in [6] and [7]. Most likely Blass's game semantics can also be added to this list, — at least, the two syntactically defined fragments of the logic generated by that semantics (the propositional multiplicative fragment and the purely additive fragment) for which Blass had found completeness proofs, coincide with the corresponding fragments of GM_{pred}. This remarkably high degree of robustness of GM_{pred} with respect to technical variations of the notion of validity is a strong indication of its being natural, and a justification of my claim that GM_{pred} is "the real logic of games". After all, the Church-Turing thesis is based on a similar argument of robustness with respect to broad variations of how algorithms are formalized.

4 Beyond finite-depth operations

The class of natural racefree operations is not limited to those studied so far. Here are some more operations of this kind:

Definition 4.1 (*Multiplicative universal quantifier*)

$$\bigwedge x[M_0, M_1, X, F] = [{M_0}', {M_1}', X\text{-}\{x\}, F'],$$

defined as follows. For $d \in Dom$, let Γ^d mean the result of removing from Γ all labeled moves except those that start with $d.$, and then deleting the prefix $d.$ in the remaining labeled moves. Then:

- For every position Γ, and $i \in \{0,1\}$, we have $M_i{}'\langle\Gamma\rangle = \{d.\alpha \mid d \in Dom,\ \alpha \in M_i\langle\Gamma^d\rangle\}$.
- For every instance f of X-$\{x\}$, whenever Γ is a legal run, we have $F'(\langle\Gamma\rangle, f) = P_1$ iff $F(\langle\Gamma^d\rangle, f^d) = P_1$ for every d with $d \in Dom$, where f^d is the extension of f to X such that $f^d(x) = d$.

Thus, $\bigwedge x G(x)$ can be thought of as the infinite multiplicative conjunction $G(0)\wedge G(1)\wedge G(2)\wedge \ldots$.

Another natural racefree operation is $!$, a counterpart of what is called *storage* (*exponential*) in linear logic, for which I am only giving a semiformal definition:

Definition 4.2 (*Storage*) A legal run of $!G$ can be thought of as a tree whose arcs are labeled with labeled moves. Branches of this tree spell legal runs of G, and thus to each node of the tree corresponds a legal position of G spelled by the (sub)branch that ends in that node. In particular, the play starts with the root as the only node of the tree; at any time, P_1 can make a move α in (add an αP_1-arc to) any leaf, as long as α is a legal move for P_1 in the corresponding position of G; so can P_0, with the difference that the node where the move is made may as well be internal, in which case a new branch is created. Then, P_1 is considered the winner iff it is the winner, in the sense of G, in all branches of the tree.

Thus, in $!G$, P_0 is allowed to restart the game G an unlimited number of times without terminating the already-in-progress plays of G, creating, this way, more and more parallel plays of G with the possibility to try different strategies in each of them and become the winner as long as one of those strategies succeeds. Moreover, P_0 does not have to really restart G from the very beginning every time it "restarts" it, but rather it can select to continue it from any of the previous positions, thus depriving P_1 of the possibility to reconsider its previously made moves. Notice that $!$ is not a strict operation. A strict (and hence slightly less natural) version of $!$, called the *repetition operator*, was introduced by Blass in [3].

The operations $\bigwedge$ and $!$ have their duals:

Definition 4.3 *Multiplicative existential quantifier* $\bigvee x A(x) = \neg\bigwedge x \neg A(x)$
Costorage $?A = \neg !\neg A$

Unlike all the other operations studied in this paper, $\bigwedge x, \bigvee x, !$ and $?$ are no longer bounded-depth or finite-depth. This is my natural excuse for not including them in the present analysis of the logic of (bounded- or finite-depth) racefree games. Exploring the game logic whose language would include these four operators is an interesting and presumably challenging task for the future.

The following list summarizes the properties of our game operations.

- Racefree: All
- Bounded-depth, Finite-depth: All except $\bigwedge, \bigvee, !, ?$
- Elementary: All except $\sqcap, \sqcup, \bigsqcap, \bigsqcup$
- Finite: All except $\bigsqcap, \bigsqcup, \bigwedge, \bigvee, !, ?$
- Strict: All except $\wedge, \vee, \rightarrow, \bigwedge, \bigvee, !, ?$

Note that the obviously elementary operations $\bigwedge$ and $!$ (together with their duals $\bigvee$ and $?$) are hardly of any interest in the context of elementary games, for we have $\bigwedge xG = \forall xG$ and $!G = G$ as long as G is elementary.

Villanova University, Pennsylvania, USA

References

[1] *S.Abramsky and R.Jagadeesan, Games and full completeness for multiplicative linear logic.* Journal of Symbolic Logic 59 (1994), pp.543-574.

[2] *J. van Benthem, Logic in Games.* ILLC, University of Amsterdam, 2001.

[3] *A.Blass, Degrees of indeterminancy of games.* Fund.Math., **77** (1972), pp.151-166.

[4] *A.Blass, A game semantics for linear logic.* Annals of Pure and Applied Logic, **56** (1992), pp.183-220.

[5] *W.Felscher, Dialogues, strategies, and intuitionistic provability.* Annals of Pure and Applied Logic 28 (1985), no.3, pp.217-254.

[6] *G.Japaridze, A constructive game semantics for the language of linear logic.* Annals of Pure and Applied Logic, **85** (1997), pp.87-156.

[7] *G.Japaridze, The logic of tasks.* Annals of Pure and Applied Logic 117 (2002), pp.263-295.

[8] *P.Lorenzen, Ein dialogisches Konstruktivitskriterium.* Infinitistic Methods (PWN, Warsaw, 1959), pp.193-200.