Constructive Game Semantics for Instantaneous Reactions*
(Updated Version**)

Joaquín Aguado[1] and Michael Mendler[1]

Faculty of Information Systems and Applied Computer Sciences
The University of Bamberg, Germany
{joaquin.aguado, michael.mendler}@wiai.uni-bamberg.de

Abstract. This paper presents some results towards a game–theoretic account of the constructive semantics of step responses for synchronous languages, providing a coherent semantic framework encompassing both non-deterministic Statecharts (as per Pnueli & Shalev) and deterministic Esterel (Berry). In particular, it is shown that Esterel arises from a finiteness condition on strategies whereas Statecharts permits infinite games. Beyond giving a novel and unifying account of these concrete languages the paper sketches a general theory for obtaining different notions of constructive responses in terms of winning conditions for finite and infinite games and their characterisation as maximal post-fixed points of functions in directed complete semi-lattices of intensional truth–values. Proofs can be found in a technical report [6].

1 Introduction

The classical theory of games, originally developed in descriptive set theory and long used in economics and engineering control theory, has emerged as a surprisingly versatile mathematical tool also in Logic and Computer Science. The power of the game metaphor rests on its ability to handle combinatorially complex situations, specifically the alternation of quantifiers, in a natural and intuitive fashion [20, 30]. The intensionality of the game model opens up a promisingly wide playground for reconciling the algebraic and operational views in the semantics of proofs and programming languages [1]. The game–theoretic solution of the full–abstraction problem for the functional language PCF [3, 23], the game–theoretic analysis of proofs in multiplicative linear logic [2, 22, 8, 4, 16] are the most prominent cases in point. In this paper we would like to draw attention to an important aspect of games that deserves to be highlighted more explicitly than it is perhaps currently done. What could become the starting

* This work is supported by the EC within the TYPES network IST 510996.

** Slightly amended version of the paper *Constructive Semantics for Instantaneous Reactions* which appeared in *Games for Logic and Programming Languages* (GaLoP 2005), D. R. Ghica and G. McCusker (eds.), Edinburgh, 2 April 2005, pp. 16–31.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

point for many further applications is that games provide a powerful and intuitively rather appealing setting for studying non-monotonic problems with co- and contravariant logical dependencies. Such problems abound in Computer Science. Many of these arise from the need to handle open systems and maintain a compositional system-environment distinction.

Game theory handles cyclic systems of non-monotonic behaviours by capturing the system and environment dichotomy through the binary polarity of player and opponent, so that the swapping of roles gives constructive (intensional) meaning to negation. In this paper we demonstrate the versality of this idea for the semantics of step responses in synchronous programming. We describe the reaction of a composite system to stimuli from its environment as a game played by the individual sub-systems in which these negotiate between themselves the final outcome. This negotiation is governed by game rules determining particular notions of constructive response. We show how different forms of winning conditions can generate different response semantics with varying degrees of constructiveness. The games we are using are non-classical in the sense that they are not necessarily determined, i.e. there need not exist winning strategies for either player. This models the non-constructiveness of a system response. Our strategies, unlike those used in type theory, do not have computational meaning ("proofs", "program") themselves. They are extensional in that they generate constructive truth-values for the interpretation of signals. Our work differs from related work of de Alfaro and Henzinger [12, 13] where the game board (*interface automata*) represents explicit synchronisation dynamics. In our case of synchronous step responses we are interested only in the stationary behaviour. The execution sequences and interleavings on the game board (*mazes*) are abstracted away.

2 Synchronous Programming and Step Responses

Let us begin by discussing the central issues arising from the Synchrony Hypothesis. For a more general introduction the reader is referred to [18, 10].

In the synchronous model system execution is thought to be scheduled under the regime of an implicit global clock which marks off a succession of individual *reaction instants*. In every instant each component delivers a full response to the external stimuli imposed by the environment. According to the *Synchrony Hypothesis* all the internal signal exchanges needed to produce this reaction are abstracted from in the sense that both the input stimulus and the response are assumed to occur *"at the same instant."* This reflects the point of view of an external environment which is always significantly slower than the system it feeds. This compactification is the beauty of the synchronous model and makes up much of its algebraic appeal. It adopts the macro-step viewpoint of the environment specified in terms of plain propositional logic and truth values to record only the overall *presence* or *absence* of a signal at the respective instant.

To be definite let us fix a concrete system model. Let us assume that components communicate via *signals* $\mathsf{S} = \{a, b, c, \ldots\}$ each of which can be present or

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

absent for a given reactive instant. A component is built up from individual transitions that *emit* signals in reaction to certain *triggering* conditions becoming satisfied. A *transition* t of the form

$$c_1, c_2, \ldots, c_m, \neg b_1, \neg b_2, \ldots, \neg b_n / a_1, a_2, \ldots, a_k$$

is triggered by the presence of some signals $\mathsf{pos}(t) = \{c_1, \ldots, c_m\} \subseteq \mathsf{S}$ called *positive preconditions* of t and the absence of signals $neg(t) = \{b_1, b_2, \ldots, b_n\} \subseteq \mathsf{S}$ called *negative preconditions*. The result of its execution is that the *action* signals $act(t) = \{a_1, a_2, \ldots, a_k\} \subseteq \mathsf{S}$ are emitted, so that they can be picked up by other transitions to trigger further computations. A *reactive component* is a structure $C = (\mathsf{T}, act, trig)$ where T is the set of transitions and $trig : 2^{\mathsf{S}} \to 2^{\mathsf{T}}$ a *triggering function*. For each subset $E \subseteq \mathsf{S}$ the function *trig* picks out the set of transitions $trig(E) \subseteq \mathsf{T}$ that are triggered by E, assuming all $c \in E$ are present and all $b \in \mathsf{S} \setminus E$ are absent. If $t \in trig(E)$ we say t is *enabled* by E. Because of the negative trigger conditions the function *trig* is non-monotonic, in general. Increasing the set of signals $E_1 \subseteq E_2$ may both increase or decrease the number of transitions that are enabled.

We are interested in the overall response of a component C in reaction to an initial environment input E_0 and determined as the combined effort of all the transitions in C. This so-called *step response* is to be consistent with the abstract model of a transition as an implication

$$(c_1 \wedge c_2 \wedge \cdots \wedge c_m \wedge \neg b_1 \wedge \neg b_2 \wedge \cdots \wedge \neg b_n) \supset (a_1 \wedge a_2 \wedge \cdots \wedge a_k),$$

specifying the reaction as *"if all the c_i are present and all b_j absent, then all of the a_l are emitted and thus are present, too"*. Parallel composition of transitions would naturally be logical conjunction $t_1 \wedge t_2$. This propositional reading is appealing but what sort of logic do we get? The simple answer is: It depends. It depends on the properties of the intended operational behaviour that the synchronous abstraction is supposed to model, in particular on how precisely the transitions are scheduled and how the interaction between them is synchronised.

Let us see how one would determine the system response of C operationally for a single reactive instant under a given initial environment stimulus $E \subseteq \mathsf{S}$. All transitions in T are assumed to act concurrently with each other. The input E, thus, is sensed by all transitions simultaneously but only those in $trig(E)$ are enabled. Taking the role of a global scheduler we would now select some of these transitions, say $T_1 \subseteq trig(E)$, and execute them in parallel. How we determine T_1 will depend on the operational semantics we have in mind. The two extreme cases are executing only one transition at a time ($|T_1| = 1$) and executing all enabled transitions together in one go ($T_1 = trig(E)$). Firing T_1 emits the action signals $act(T_1) = \bigcup_{t \in T_1} act(t)$. These can now trigger further transitions, relative to the extended signal set $E_1 = E \cup act(T_1)$. Again we schedule a subset $T_2 \subseteq trig(E_1)$ of transitions enabled by E_1, and so on. In this way a chain reaction of transition firings $T_{i+1} \subseteq trig(E_i)$ and cumulated signal emissions $E_{i+1} = E_i \cup act(T_i)$ may ensue. We continue this process ensuring *maximal progress* for all system parts. When the activation sequence finally stabilises, i.e. $act(trig(E_n)) = E_n$, the

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

reactive instant is completed and the *step response* or *macro step* of C is the final accumulated signal set E_n. For contrast, the individual scheduling stages are sometimes called *micro steps*.

Within this generic model, which is typical for synchronous declarative languages many variations of scheduling strategies are possible. The crucial point here is how to deal with the potential inconsistencies introduced by the inhibitive effects of negative triggers. Consider the following situation: A transition $t_i = \neg b/a \in \mathit{trig}(E_i)$ is enabled by the absence of signal b from E_i, i.e., because of $b \notin E_i$. Transition t_i is fired and included into T_i. This produces action $a \in E_{i+1}$ which sets off further transitions in subsequent T_{i+1}, T_{i+2}, ... and eventually, because of that, signal b is produced, say $b \in \mathit{act}(t_n)$ where $t_n \in T_n$. Clearly, this is inconsistent with the firing of t_i in the first place which was done under the condition that b is absent. There is not a single canonical way to handle this. Likely, the full range of possibilities have not been explored yet, but already there is a profusion of different solutions adopted in the literature on synchronous languages [21, 31, 24, 28, 19, 33, 27, 10]. For more details, see [6].

Our claim is that the variety of semantics arising from the different options of handling negation can be described naturally and uniformly using the game-theoretic metaphor. Given the long-standing and sometimes heated debate about the "right" step semantics for synchronous programming it appears to be more than appropriate to search for a convincing unifying setting in which the different dialects can coexist, each having its own characteristic place and application. We present a natural hierarchy of three increasing levels of constructive strength in the interpretation of negation, covering both Pnueli & Shalev's version of Statecharts and Berry's Esterel. As the work is still tentative we do not claim to achieve more than outlining this programme here. The guiding idea is to try and characterise the different scheduling disciplines as instances of the ideal propositional view promised by the Synchrony Hypothesis according to which transitions are logical implications and parallel composition is logical conjunction. It is not difficult to see that this requires more than classical two-valued logic (*true* = presence, *false* = absence). Take the four transitions $t_1 := \neg b \supset a$, $t_2 := b \supset a$, $t_3 := \mathit{true} \supset a$, $t_4 := a \supset b$. In classical logic the conjunction $t_1 \wedge t_2$ is equivalent to t_3, so we would expect $t_1 \wedge t_2$ to be interchangeable with t_3. In particular $C_1 := t_1 \wedge t_2 \wedge t_4$ should be equivalent to $C_2 := t_3 \wedge t_4$ and thus represent the same operational behaviour. Now consider the operational semantics of [31] and run C_1 in the empty environment $E = \emptyset$: Signal b is absent in E, so transition t_1 fires. This sets off action a, triggering t_4, which produces signal b. This is inconsistent as b was assumed absent when t_1 was fired. Hence, following [31], we try to find another schedule that is safe. But there is none, so program C_1 fails, meaning it does not have a response in the empty environment. On the other hand, C_2 happily terminates producing response b in all circumstances. So, C_1 and C_2 are different, operationally.

It is not surprising that classical logic is not fine enough to model all the intensional aspects of scheduling under inhibiting as well as enabling effects. As seen above the single truth-value *false* cannot adequately model the meaning of

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

negation $\neg b$ when b is initially absent but occurring later. What is surprising is that for certain coherent scheduling regimes it is possible to maintain the abstract propositional viewpoint, i.e. avoid the complications of modelling scheduling sequences in detail, simply by choosing a constructive logic interpretation. For instance, in [26] it was shown that the simple twist of replacing the classical two-valued by an intuitionistic interpretation of signals (specifically, 3-valued Gödel logic) suffices to obtain a fully abstract and compositional model for the original macro-step semantics of Statecharts as given by Pnueli and Shalev [31]. Another example in this direction is [5] which explains the constructive semantics of Esterel naturally in terms of winning strategies in finite 2-player games. The present paper extends and systematises this work to show how non-classical truth values induced by logic games can be used to characterise different kinds of constructive single-step semantics for Statecharts-like languages.

3 Synchronous Reactions and Two–Player Symmetric Maze Games

How to Play. A reactive component C can be modelled as a *maze* M consisting of *rooms* and *directed corridors* between them. Every signal in S corresponds to a room in M and the corridors represent the *causal relations* between signals as given by the transitions. When one is in a room a and there is a corridor $a \longrightarrow b$, then it is possible to move into room b. In other words, the corridor $a \longrightarrow b$ denotes, according to C, that the status (present or absent) of a can be justified (computed) in terms of the status of b.

In the symmetric two-player game employed here, this maze is the board and the game figure is a token which is moved from room to room by the players taking turns according to certain *rules*. In general, the objective of the game is that of defending a set of rooms (region) according to a given *winning condition*. Thus, this game produces a pair of regions $P \subseteq \mathsf{S}$ and $O \subseteq \mathsf{S}$ one for each player. These two sets together constitute a possible response of C under a particular constructive semantics such that P and O will contain signals that are present and absent, respectively, when C is executed following the chosen operational model. The two players $\mathsf{P} = \{A, B\}$ are the *system* and the *environment* denoted respectively by A and B, where A plays for region P and B for region O. Hence, in all plays from P player A starts while in O his opponent B is the first to play. We will say A *defends* the *front-line* (P, O) if A has a winning strategy for all plays from P or O. The rules are given by two types of valid moves that we shall call *visible moves* and *secret moves*, respectively. In our games, we represent this statically in the maze by two types of corridors, *visible* and *secret*. Every time that the token is moved through a visible corridor the control is passed to the opponent and if the token is moved through a secret corridor the turn remains unchanged. In the same vein, it will be useful to distinguish between *visible rooms* and *secret rooms*. Visible rooms, like visible corridors, are atomic communication points between system and environment where information is exchanged, while secret rooms represent intermediate or auxiliary positions where no interaction

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

takes place. This models the difference between atomic signals a, b (visible rooms) and composite formulas such as $a \wedge \neg b$, $a \vee \neg b$ (secret rooms). This distinction will be relevant for certain cases of winning conditions.

Reactive Components as a Mazes. Since in our game P will correspond to the set of present signals and O to the set of absent signals the causality expressed in the transitions T of C can be represented in the maze M as follows.

For any $a \in act(t)$, transition $t \in T$ is expressing the fact that a is caused to be in P if for all $c \in pos(t)$, c is in P and for all $b \in neg(t)$, b is in O. This conjunction can be modelled in the maze by means of introducing an intermediate (secret) room, say y, and by adding a visible corridor between each $a \in act(t)$ and y, a visible corridor between y and each $c \in pos(t)$ and a secret corridor between y and each $b \in neg(t)$ as seen in Fig. 1, where visible rooms/corridors are marked ι and drawn with solid lines while secret rooms/corridors are drawn with dashed lines and marked τ. A transition like c/a with only one trigger and action can be coded without the intermediate room as a secret corridor from a to c and a transition $\neg b/a$ simply as a visible corridor from a to b. This short-hand will be sound for all semantics considered in this paper.

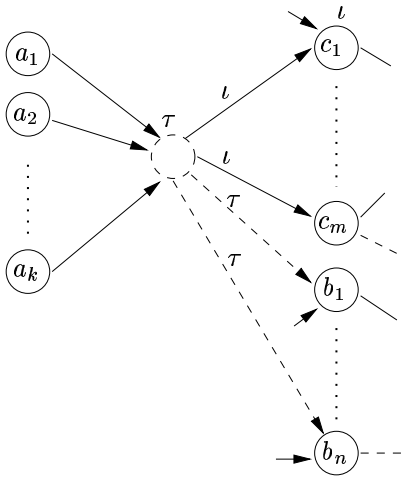


Fig. 1. Example transition.

Note that any environment stimulus can be accounted for as part of a reactive component. The situation where the environment provides x and z to C can be expressed as the parallel composition $C \mid (\cdot/x, z)$, where $\cdot/x, z$ is a transition with an empty precondition which produces signals x and z as required by the environment. A transition t with an empty precondition is reflected in the corresponding maze as a visible corridor from all $a \in act(t)$ to a *dungeon*, i.e., a room without exits.

Game Rules. We now informally present different alternative ways of playing the game. Let us use the maze of Figure 2 assuming that x is the only signal that can be used as stimulus by the environment. For the sake of convenience, in what follows whenever x is present we will avoid drawing a visible corridor to a dungeon from this room and instead simply assume that x is in P.

The synchronous reactive model requires constructive responses, meaning responses that are supported by some sort of justification which should come from the program. A natural way of obtaining various forms of constructive

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

arguments, would be to assume that at startup and in some pre-established order the players choose a finite amount (discrete quantity) of a particular resource that we may call *seeds*. During the game, and depending on the rules, a player has to pay with seeds for taking some action (*e.g.* visiting a room for the first time, using a corridor, taking his turn, etc.). A player wins if he can make his opponent get stuck, i.e., force him to play in a dungeon or make him run out of seeds. Note that this metaphor is not only useful for introducing constructive arguments, but also can reflect the underlying resources and features of the physical system (*e.g.* energy to maintain a signal, time, memory and so on) that the model abstracted from. The notion of seed is an intensional parameter that can be used to explain extensional but infinite winning conditions in terms of finite processes. In general, different intensional rules, involving seeds as suggested above, may give rise to the same extensional model.

In the *coherent* maze game, player B (the opponent) is the first to pick up his supply of seeds whereupon player A does the same with the advantage of knowing the resources of his opponent. In this game, *both* players are required to pay one (unit of) seed every time the token enters a visible room. Under these conditions A can win provided he can keep the play going for an arbitrary number of moves through visible rooms. One possible solution for Fig. 2 is given by $(P, O) = (\{y, r\}, \{z, s\})$. If the initial position of the token is r then evidently player A can use the secret corridor that forms a self-loop in this room repeatedly without ever getting stuck. If the initial position is y player A can move the token to room s and pass the turn to B. From this position, player B will either move to r and give A the chance to take over forever, or he can place the token back to y restarting the whole process. In any case, since the number of seeds on both sides is finite the play must eventually finish. Moreover, because A had chosen his number of seeds after B he will have more seeds available than B (assuming A is sensible enough), and so when this process ends it has to be because B runs out of seeds. Similarly, one argues that if B starts in any of the rooms $O = \{z, s\}$ player A can win by making B run out of seeds or get stuck.

In the *lazy* maze game, the rules are as before except now only the player who actually gets to occupy (receives the turn in) a visible room must pay. Hence, a coherent solution like the one before $(P, O) = (\{y, r\}, \{z, s\})$ is no longer acceptable, because although A can have as many seeds as he wants this number is finite. So if he decides to go in circles from room r to room r he will eventually get stuck there without B having lost a single seed. To win, player A must be able to hand over the turn to B arbitrarily often in a visible room. It is not hard to see that a solution for this game is given by $(P, O) = (\{y, s\}, \{x, r, z\})$. Starting in P player A can move to x or r

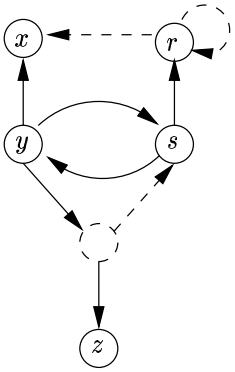


Fig. 2. Example maze.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

where opponent B will be stuck (in x) or cycle indefinitely (in r). Similarly, if B plays from O he will cycle or finish in a dungeon without having challenged A at all. Assuming that the presence of signal x is enforced by the environment we have in fact two solutions $(P,O) = (\{x,y,r\},\{z,s\})$ and also $(P,O) = (\{x,r,s\},\{z,y\})$, which illustrates the inherent non-determinism in this type of game.

In the *eager* maze game everything is as in the previous case but now the order in which the players choose their seeds is inverted, first A chooses and then B. Now loops like the one formed between y and s when $x,r \in P$ can no longer be supported by the fact that A has more seeds than B in such a way that depending on the starting position either $y \in P, s \in O$ or $s \in P, y \in O$ is defended by A. It is B who has the advantage of having more seeds, so basically A needs to solve the game by avoiding circular confrontations and instead try to push B into a dungeon as soon as possible. In our board the solution for this is given by $(P,O) = (\{y\},\{x,z\})$ which can easily be verified. This game type ensures determinism.

Charts and Mazes. Formally, a *maze* is a finite labelled transition system $M = (\mathsf{S}_\iota, \mathsf{S}_\tau, \xrightarrow{\iota}, \xrightarrow{\tau})$, consisting of finite and disjoint sets of *visible rooms* S_ι and *secret rooms* S_τ, together with transition relations $\xrightarrow{\gamma} \subseteq \mathsf{S} \times \mathsf{S}$, where $\gamma \in \{\iota,\tau\}$ is a secret or observable action and $\mathsf{S} = \mathsf{S}_\iota \cup \mathsf{S}_\tau$ the set of all rooms. The transitions represent valid moves (*corridors*) between rooms. A transition $s \xrightarrow{\iota} s'$ corresponds to a visible corridor connecting s with s' and $s \xrightarrow{\tau} s'$ to a secret corridor. For technical convenience we assume that rooms without exiting corridors (*dungeons*) are visible. In this way it is always observable when a player gets stuck. Moreover, we assume that no pair of secret rooms is connected by a secret corridor, which ensures that each move involves at least one visible interaction. We denote the opponent of player U by $\overline{U}$. More generally, for $\gamma \in \{\iota,\tau\}$ and $U \in \mathsf{P} = \{A,B\}$ we define $U^\gamma \in \mathsf{P}$ to be the unique U' such that $U = U'$ iff $\gamma = \tau$. We will assume throughout that M is a fixed, finitely branching, maze.

Configurations and Plays. A *(game) configuration* is a pair $c = (\mathsf{pos}(c), \mathsf{turn}(c)) \in \mathsf{S} \times \mathsf{P}$. The first part $\mathsf{pos}(c)$ is a position in M and $\mathsf{turn}(c)$ denotes the player who has the turn at this point. A *play* is a (possibly empty) finite or infinite sequence of *configurations*

$$\pi = (m_0,t_0)\cdot(m_1,t_1)\cdot(m_2,t_2)\cdots$$

consistent with the game rules, i.e., each step follows some corridor in M and the player's turn changes exactly if this corridor is visible. Formally, for all (m_i,t_i) that have a successor (m_{i+1},t_{i+1}) in π we must have $m_i \xrightarrow{\tau} m_{i+1}$ if $t_i = t_{i+1}$ or $m_i \xrightarrow{\iota} m_{i+1}$ if $t_i \neq t_{i+1}$. The *domain* $\mathrm{dom}(\pi) \in \omega + 1$ of a game path is the set of indices, i.e. $\mathrm{dom}(\pi) = \omega$ if the path is infinite and $\mathrm{dom}(\pi) \in \omega$ if it is finite. In handling such indices it is expedient to consider the domain as an

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

ordinal number. Specifically, $\omega = \{0, 1, 2, \ldots\}$ and each natural number $n \in \omega$ is identified with the set of its predecessors, i.e., $n = \{0, 1, 2, \ldots, n-1\}$. We can then present a path as a function $\pi : \mathrm{dom}(\pi) \to \mathsf{S} \times \mathsf{P}$. The empty play is denoted ϵ. From now on, let Π denote the set of plays in M. We write $\sqsubseteq$ for the *prefix* ordering on Π, i.e., $\pi_1 \sqsubseteq \pi_2$ if there exists a play σ such that $\pi_2 = \pi_1 \cdot \sigma$. A play π_1 is a *suffix* of another play π_2 if $\pi_2 = \sigma \cdot \pi_1$ for some finite play σ.

4 Strategies and Defensible Front-Lines

As usual a strategy for a player U is a subset of plays in which U's moves are uniquely determined at each stage of a play where he holds the turn, while keeping unconstrained the decisions of his opponent. A *strategy for* U, or U-*strategy*, is a non-empty $\sqsubseteq$-closed subset $\Sigma \subseteq \Pi$ of plays that is *continuous*, U-*deterministic* and $\overline{U}$-*closed*. Specifically, Σ is

- *continuous*, if $\pi \notin \Sigma$ implies there is a finite prefix $\pi' \sqsubseteq \pi$ with $\pi' \notin \Sigma$;
- U-*deterministic*, if $\pi \cdot (m, U) \cdot (m_i, U_i) \cdot \sigma_i \in \Sigma$ for $i = 1, 2$, then $m_1 = m_2$ and $U_1 = U_2$;
- $\overline{U}$-*closed*, if $\pi \cdot (m, \overline{U}) \cdot \sigma \in \Sigma$ then for all corridors $m \xrightarrow{\gamma} m'$ in the maze there is an extension $\pi \cdot (m, \overline{U}) \cdot (m', \overline{U}^{\gamma}) \cdot \sigma' \in \Sigma$.

Two features of U-strategies are noteworthy. Firstly, they are partial, i.e. player U is not forced to make a move. He may decide to stop the play for good, even if there is an outgoing corridor. Secondly, a strategy in general has many initial configurations generating several independent threads of plays from different initial positions with different starting players.

Throughout the paper we will fix $U = A$ and simply talk about a *strategy* when we mean A-strategy. Also, we will be interested only in *consistent* and *positional* strategies. A strategy is *consistent* if no position is occupied by more than one player, i.e., $\pi_i \cdot (m, U_i) \cdot \sigma_i \in \Sigma$ for $i = 1, 2$ and fixed $m \in \mathsf{S}$ implies $U_1 = U_2$. A strategy is *positional* if a player's every move only depends on the current position, not on the history of the play. Formally, $\pi \cdot (m, A) \cdot \sigma \in \Sigma$ iff $(m, A) \cdot \sigma \in \Sigma$. By default all strategies are assumed to be positional and consistent A-strategies.

Let us look at some consequences of consistency. For an A-strategy Σ let P_Σ and O_Σ be the sets of positions in Σ in which player A and B receive the turn, respectively. Formally, $P_\Sigma := \{m \mid \exists \pi, \sigma.\, \pi \cdot (m, A) \cdot \sigma \in \Sigma\}$ and dually $O_\Sigma := \{m \mid \exists \pi, \sigma.\, \pi \cdot (m, B) \cdot \sigma \in \Sigma\}$. Consistency of strategies, then, is equivalent to the condition $P_\Sigma \cap O_\Sigma = \emptyset$. We call pairs (P, O) of regions with the property that $P \cap O = \emptyset$ *front-lines* and (P_Σ, O_Σ) the front-line *defended* by strategy Σ. A front-line (P, O) where P is the complement of O is called *binary* or *two-valued*.

We will be interested in maximal front-lines that are defensible using particular types of strategies and characterise them in terms of post-fixed points. To understand the connection it is useful to view the maze M as a Kripke transition structure in labels $\gamma \in \{\iota, \tau\}$ in which regions may be specified using formulas of

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

propositional modal μ-calculus. We assume the reader is familiar with this language and its semantics (see e.g. [35, 17]). In this language we have the standard modalities $\langle\gamma\rangle$, $[\gamma]$ each of which is of type $2^{\mathsf{S}} \to 2^{\mathsf{S}}$ on sets of rooms, defined in the usual way:

$$\langle\gamma\rangle(R) := \{m \mid \exists m' \in R.\, m \xrightarrow{\gamma} m'\}$$
$$[\gamma](R) := \{m \mid \forall m' \in R.\, m \xrightarrow{\gamma} m'\},$$

for arbitrary $R \subseteq \mathsf{S}$. The logical connectives $\vee$, $\wedge$ correspond to union and intersection on 2^{S}, respectively, and $\neg$ is complementation, i.e., $R \vee S := R \cup S$, $R \wedge S := R \cap S$, $\neg R := 2^{\mathsf{S}} \setminus R$. Further, there are least fixed point $\mu X.\phi$ and greatest fixed point operators $\nu X.\phi$ with the usual interpretation assuming that the formula scheme ϕ is monotonic in the recursion variable X.

Let us observe that defending a front-line with an arbitrary strategy does not require much cleverness, since as long as from O there is no secret corridor into P and no visible corridor into O the trivial "empty" strategy will do, in which A does not make any move at all. The following Proposition 1 characterises defensible front-lines, without reference to strategies:

Proposition 1. *Let (P, O) be a front-line. Then (P, O) is defensible iff $O \subseteq [\tau]O \wedge [\iota]P$. Further, (P, O) is maximal defensible iff additionally $P = \neg O$.*

Using Proposition 1 it is possible to make a connection between maximal defensible front-lines and classical logic. Read each visible corridor $a \xrightarrow{\iota} b$ as the logical implication $\neg b \supset a$ and each secret corridor $a \xrightarrow{\tau} b$ as $b \supset a$. In this manner a finite maze M corresponds to a formula ϕ_M given by the conjunction of all its corridor implications. Taking the classical truth-value interpretation and identifying binary valuations $V : \mathsf{S} \to \mathbb{B}$ with subsets $V \subseteq \mathsf{S}$ we find that $(P, \neg P)$ is a maximal defensible front-line iff P as a binary truth valuation ($P(x) = 1$ iff $x \in P$) satisfies ϕ_M. In other words,

Proposition 2. *The maximal defensible front-lines (without further conditions on strategies) coincide with the classical binary truth-value models of ϕ_M.*

Thus, the notion of defensibility just coincides with classical logic, the weakest notion of synchronous response. In the next Sec. 5 we shall study specific *winning conditions* making the defence of front-lines constructive. Depending on which winning conditions we consider we will get different types of games on the same maze, implementing different degrees of causal justification.

Before we study winning conditions a couple of general observations are in order on the algebraic nature of our problem. Let FL_M be the set of front-lines of M, i.e., the set of $(P, O) \subseteq (2^{\mathsf{S}})^2$ such that $P \cap O = \emptyset$. Then, first observe that $(P, O) \in \mathsf{FL}_M$ iff (P, O) is a post-fixed point (pfp) of the "De-Morgan" function $\mathsf{dm} : (2^{\mathsf{S}})^2 \to (2^{\mathsf{S}})^2$ defined as $\mathsf{dm}(X, Y) := (\neg Y, \neg X)$, which interchanges the role of the players and complements their set of positions. From Prop. 1 we gather that defensible front-lines can be described as the pfps of $\mathsf{dfl}\text{-}M : \mathsf{FL}_M \to \mathsf{FL}_M$, where

$$\mathsf{dfl}\text{-}M(X, Y) := (X \vee \langle\tau\rangle X \vee \langle\iota\rangle Y, Y \wedge [\tau]Y \wedge [\iota]X),$$

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

taken as a function on FL_M. It is easy to verify that dfl-M indeed preserves front-lines, i.e. if $(P,O) \subseteq \mathsf{dm}(P,O)$ then dfl-$M(P,O) \subseteq \mathsf{dm}(\mathsf{dfl}$-$M(P,O))$. Having pfps the next question is what are the greatest fixed points? If f is a monotone function on a lattice then the greatest fixed point of f is the supremum of all pfps. However, for functions such as dm and dfl-M this does not work, in the former case since dm is not monotone and in the latter since FL_M is not a lattice. The sub-domain FL_M of front-lines is not completely without structure, though. It is easy to see that FL_M is a directed complete semi-lattice. By Zorn's Lemma such domains have maximal elements. So, we may not have unique *greatest* pfps but we can try and identify the *maximal* pfps instead.

5 Winning Conditions

A *winning condition*, for our purposes, is a property of (maximal) plays that is *time invariant*, i.e., invariant under shifting of the initial position. A rich source for winning conditions is provided by the different acceptance conditions for ω-regular languages over the alphabet of configurations. All usual acceptance criteria for infinite paths such as the Muller, Rabin, Streett, Parity, Büchi conditions (see [36, 17] for a comprehensive introduction) are time invariant, simply because they all are based on the set of configurations that occur infinitely often in a play. Obviously, this set does not change under shift of initial position. We shall only consider some special cases, here, related to the two well-known synchronous languages, Esterel and Statecharts. A more comprehensive study is left to future work.

Definition 1. *Let π be a play and U a player. We say that π is*

- U-live *if U always enables another visible move, i.e., for all $i \in dom(\pi)$ such that $\mathsf{turn}(\pi(i)) = U$ there exists $i < j < dom(\pi)$ such that $\mathsf{pos}(\pi(j)) \in S_\iota$.*
- U-reactive *if U always eventually hands over to $\overline{U}$ in a visible room. Formally, for all $i \in dom(\pi)$ if $\mathsf{turn}(\pi(i)) = U$ then there exists $i < j < dom(\pi)$ such that $\mathsf{pos}(\pi(j)) \in S_\iota$ and $\mathsf{turn}(\pi(j)) = \overline{U}$.*
- U-terminating *if all observable actions eventually stop and $\overline{U}$ is the last player, i.e., there exists $i < dom(\pi)$ such that $\mathsf{turn}(\pi(i)) = \overline{U}$ and for all $i < j < dom(\pi)$, we have $\mathsf{pos}(\pi(j)) \in S_\tau$ and $\mathsf{turn}(\pi(j)) = \overline{U}$.*

The conditions of Def. 1 reflect[1] the different intensional ways — discussed in Sec. 3 — of using seeds as additional (finite) resources. One can easily see that these conditions are general winning conditions according to the above definition, i.e., that they are time-invariant. On finite plays all three notions agree, i.e. if $\mathrm{dom}(\pi) < \omega$ then π is U-live iff it is U-reactive iff it is U-terminating. They simply state that player $\overline{U}$ is the last to hold the execution token and

[1] This intensional interpretion of the winning conditions depends on the restiction that dungeons are always visible and secret rooms cannot be connected by secret corridors, which may be dropped by adjusting the rules for paying seeds.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

thus responsible of "stopping" the computation. On infinite plays reactiveness is a proper strengthening of liveness, and termination a proper strengthening of reactiveness. A play may be U-live but not U-reactive, e.g., if player U keeps going forward indefinitely along secret corridors between visible rooms without ever again passing the turn to his opponent. This corresponds to "divergence" on the side of U. A play may be U-reactive but not U-terminating, e.g., if U infinitely passes the turn to his opponent (in visible rooms) who keeps challenging him over and over again without the play ever stopping. An U-terminating play must eventually leave the onus on the side of $\overline{U}$ forever.

As done in [7] we will discuss only a special case of the above winning conditions, namely when the maze does not have any secret rooms. So, unless specified otherwise, $\mathsf{S} = \mathsf{S}_\iota$. This will suffice to convey the basic ideas. The general case can be developed without difficulties by refinement from the simplified setting. To assume that all rooms are visible essentially amounts to the special case of synchronous programs in which all transitions have single triggers and actions, i.e. are of the form b/a or $\neg b/a$. As indicated before, these can be represented directly by secret or visible corridors, respectively, connecting the visible rooms a and b. As far as terminating and live winning strategies are concerned this is without loss of generality as all transitions can be broken down into these special cases: E.g., one can show that $c, \neg b/a, d$ has the same semantics as the set of transitions $\neg c/x$, b/x, $\neg x/a$, $\neg x/d$ together. This does not hold for reactiveness, though, as we will indicate later. Under the assumption $\mathsf{S} = \mathsf{S}_\iota$, or $\mathsf{S}_\tau = \emptyset$, the winning conditions specialise as follows ([7]): A play π is

- *U-live* if U always makes another move when he gets the turn, i.e., for all $i \in \mathrm{dom}(\pi)$ if $\mathsf{turn}(\pi(i)) = U$ then $i + 1 \in \mathrm{dom}(\pi)$
- *U-reactive* if U always eventually hands over to $\overline{U}$. Formally, for all $i < \mathrm{dom}(\pi)$ there exists $i \leq j < \mathrm{dom}(\pi)$ with $\mathsf{turn}(\pi(j)) = \overline{U}$
- *U-terminating* if it is finite and $\overline{U}$ is the last player, i.e., $\mathrm{dom}(\pi) < \omega$ and $\mathsf{turn}(\pi(\mathrm{dom}(\pi) - 1)) = \overline{U}$.

Given a winning condition Win we say that a strategy Σ is a *Win-strategy* if all prefix-maximal plays in Σ satisfy Win, and a front-line (P, O) is called *Win-defensible* if there exists a Win-strategy Σ that defends it, i.e. for which $(P, O) = (P_\Sigma, O_\Sigma)$.

5.1 Coherent Responses

The first constructive strengthening of strategies is A-liveness. In an A-live strategy the player must make sure he is never blocked, i.e. he always makes a move when he receives the turn. This means for every "present" signal in P the player must be able to offer a justifying transition. Of course, like for ordinary defence strategies, he must play consistently, i.e., always move from a ("safe") P-position in his territory while confining the opponent to region O and preventing him from conquering any position in P. Let us say a front-line (P, O) is *coherent* if it is defensible by an A-live strategy. In other words, a front-line is defended coherently

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

if player A is consistent and can avoid ever getting trapped in a dungeon. Thus, coherent front-lines admit infinite plays. Let $\mathsf{CFL}_M \subseteq 2^{\mathsf{S}} \times 2^{\mathsf{S}}$ be the collection of all coherent front-lines. Like for general defensibility a simple characterisation can be given that does not refer to strategies.

Proposition 3. *A front-line (P, O) is coherent iff both $O \subseteq [\tau]O \wedge [\iota]P$ and $P \subseteq \langle\tau\rangle P \vee \langle\iota\rangle O$.*

Using Prop. 3 one can show that for arbitrary P there exists O such that (P, O) is a coherent front-line iff $P \subseteq \mathsf{is\_present}(P)$ where $\mathsf{is\_present}(X) := \langle\tau\rangle X \vee \langle\iota\rangle(\nu Y.\,(\neg X \wedge [\tau]Y \wedge [\iota]X))$. Dually, a set O can be extended to a coherent front-line (P, O) iff $O \subseteq \mathsf{is\_absent}(O)$ for $\mathsf{is\_absent}(Y) := [\tau]Y \wedge [\iota](\nu X.(\neg Y \wedge (\langle\tau\rangle X \vee \langle\iota\rangle Y))$. These characterisations describe the verification procedures to check that particular signals are coherently present (absent) in the system response without specifying the full expected response up front. It is important to point out that the negations appearing in $\mathsf{is\_present}$ and $\mathsf{is\_absent}$ are crucial to make sure the front-lines extracted are consistent. As a consequence of this the arguments P and O, respectively, appear simultaneously co- and contravariantly in the right-hand sides of both inequations $P \subseteq \mathsf{is\_present}(P)$ and $O \subseteq \mathsf{is\_absent}(O)$. Therefore, we cannot hope to obtain unique maximal solutions.

Proposition 4. *(P, O) is a maximal coherent front-line iff it is a maximal fixed point of $\mathsf{cfl}\text{-}M : \mathsf{FL}_M \to \mathsf{FL}_M$, where $\mathsf{cfl}\text{-}M(X, Y) := (\,\langle\tau\rangle X \vee \langle\iota\rangle Y\,,\,[\tau]Y \wedge [\iota]X\,)$. Moreover, $(P, \neg P)$ is coherent iff $P = \langle\tau\rangle P \vee \langle\iota\rangle \neg P$.*

What do coherent front-lines mean for synchronous programming? Let us look at Esterel. Considering a corridor $x \xrightarrow{\tau} y$ as a statement $\mathsf{present}\, y\, \mathsf{then}\, x$ and $x \xrightarrow{\iota} y$ as $\mathsf{present}\, y\, \mathsf{else}\, x$ associates an Esterel program $\mathsf{esterel}(M)$ with our maze M. A front-line (P, O) of M then corresponds to a potential response of $\mathsf{esterel}(M)$ where all signals in P are considered present and all in O absent. Now, $(P, \neg P)$ is coherent if a signal is present $x \in P$ in the response iff there is a statement in $\mathsf{esterel}(M)$ that emits it, i.e. there is $\mathsf{present}\, y_1\, \mathsf{then}\, x$ and $y_1 \in P$ or $\mathsf{present}\, y_2\, \mathsf{else}\, x$ and $y_2 \in \neg P$; and a signal is absent $x \in \neg P$ exactly if all statements that can emit x are switched off. Two-valued coherent responses have been called *logically coherent* by Berry [9], hence our terminology. An Esterel program is *logically reactive* (*logically deterministic*) if it has at least (at most) one logically coherent and binary response.

What is constructive about coherence? Well, coherence is strongly related to the notion of *inertiality*. If a transition, say $c, \neg a/b$ represents a response function with inertial delay then it has the following extra property: If the input trigger $c \wedge \neg a$ becomes satisfied and then false again before output b could be produced, then the transition being "inertial" completely forgets its previous excitation. Another way to put this is: if $c \wedge \neg a$ holds during some non-empty interval $[s, t) \subset \mathbb{R}$ of time and the output b indeed reacts while it is on, say at $t_b \in [s, t)$, then b must remain (at least) present strictly beyond t, i.e. during some interval $[t_b, t + \epsilon)$, for $\epsilon > 0$. Inertiality is an important assumption in hardware design [11] and the key to implementing memory.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

It is not difficult to verify, then, that a component (represented in Fig. 3) such as $(c, \neg a/b) \mid (b, \neg a/c) \mid (\neg d, b, c/d)$ can exhibit a real-time waveform in which both b, c are constant true mutually supporting each other by inertiality when a is absent. This corresponds to the coherent front-line $(P_1, O_1) = (\{b, c\}, \{a, y, z\})$. Observe that (P_1, O_1) can not be extended to a two-valued solution. For even with inertial delays signal d cannot be guaranteed to stabilise when b, c are present. There is, however, a coherent and two-valued solution $(P_3, O_3) = (\{x, y, z\}, \{a, b, c, d\})$, which also means our example is non-deterministic. In contrast, the defensible front-line $(P_2, O_2) = (\{a, b, c, y, z\}, \emptyset)$ is not coherent since a, b, c would have to be maintained by the environment, which is not an autonomous and constructive response.

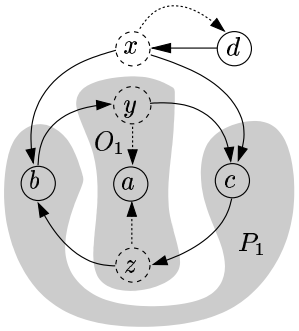


Fig. 3. Maze for component $(c, \neg a/b) \mid (b, \neg a/c) \mid (\neg d, b, c/d)$ with front-line (P_1, O_1).

5.2 Lazy Responses

Coherent strategies, though stronger than classical responses, still constitute a weak notion of winning. A player can defend his positions simply by avoiding ever to get stuck in a dungeon while maintaining consistency. This includes the possibility that A cycles along an infinite path of secret corridors where he keeps the turn forever. In this section we will strengthen the winning conditions so as to eliminate such behaviour. We still permit infinite plays but require the winning player to be reactive in the sense that he is never embarrassed about a move when challenged and *always generates a proper response* (hands over to opponent in a visible room) in finite time, though we do not insist he can stop the opponent from ever challenging him again. We shall call these defence strategies *lazy* and write LFL_M for the set of lazy defensible front-lines. For the example of Fig. 3 we find that the coherent valuation $(P_1, O_1) = (\{b, c\}, \{a, y, z\})$ is not lazy and thus ruled out. To defend room $b \in P_1$, say, player A would indefinitely send his opponent around the intermediate *secret* rooms y, z (corresponding to the composite trigger conjunctions $c \wedge \neg a$ and $b \wedge \neg a$) which violates reactiveness. Only $(P_3, O_3) = (\{x, y, z\}, \{a, b, c, d\})$ remains as a lazy front-line.

The term 'lazy' is inspired by the computational intuition that in the game the system player A produces a result to a request[2] from the environment player B. In a lazy computation it suffices that the system respond to every request

[2] As in the standard type-theoretic setting we imagine environment B making the first move with a choice of a secret corridor to any room in O or a visible corridor to any of the P positions. Logically this is the conjunction of all P with the negation of all O positions.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

from the environment in finite time, without necessarily being able to make the environment stop all further requests. Notice again, the central feature of these games is their use of infinite plays.

We will show how lazy front lines LFL_M are determined as the post-fixed points of a suitable monotone function $\mathsf{lfl}\text{-}M : \mathsf{FL}_M \to \mathsf{FL}_M$ such that all pfps of $\mathsf{lfl}\text{-}M$ are pfps of $\mathsf{cfl}\text{-}M$, which we abbreviate as $\mathsf{lfl}\text{-}M \leq \mathsf{cfl}\text{-}M$. This implies that $\mathsf{LFL}_M \subseteq \mathsf{CFL}_M$ as expected, i.e., all lazy front lines are necessarily coherent. Moreover, we show that lazy defence strategies are intimately related with Pnueli & Shalev's interpretation of Statecharts [31] in the sense that the maximal elements in LFL_M correspond to the step responses of M viewed as a Statechart. For the following we assume that all rooms are visible. For arbitrary mazes both Props. 5 and 6 need to be generalised slightly, by another least fixed point that implements finite skipping over secret rooms.

Proposition 5. *A front-line (P, O) is lazy iff $P \subseteq \mu X.(\langle\tau\rangle(P \wedge X) \vee \langle\iota\rangle O)$ and $O \subseteq [\tau]O \wedge [\iota]P$.*

Using the characterisation of Prop. 5 we can eliminate the negative part O from the definition of lazy front-lines as before: A set P can be lazily defended by the starting player, i.e. is part of a lazy front-line (P, O) for *some* O iff $P \subseteq \mu X.(\langle\tau\rangle(P \wedge X) \vee \langle\iota\rangle O^*)$ for the fixed ("loosest upper approximation") $O^* := \nu Y.\neg P \wedge [\tau]Y \wedge [\iota]P$. A similar statement can be made for the opponent part: A set O can be defended by the second player, i.e. is part of a lazy front-line (P, O) for *some* P iff $O \subseteq [\tau]O \wedge [\iota]P^*$ for the fixed set $P^* := \nu Y.(\neg O \wedge \mu X.(\langle\tau\rangle(Y \wedge X) \vee \langle\iota\rangle O))$. Note again how these "one-sided" formulations bring up the combined co- and contravariance inherent in our setting.

Prop. 5 states that (P, O) is a lazy front-line if it is a consistent pfp of the set function $\mathsf{lfl}\text{-}M : (2^{\mathsf{S}} \times 2^{\mathsf{S}}) \to (2^{\mathsf{S}} \times 2^{\mathsf{S}})$ defined as

$$\mathsf{lfl}\text{-}M(P, O) := (\,\mu X.(\langle\tau\rangle(P \wedge X) \vee \langle\iota\rangle O)\,,\, [\tau]O \wedge [\iota]P\,).$$

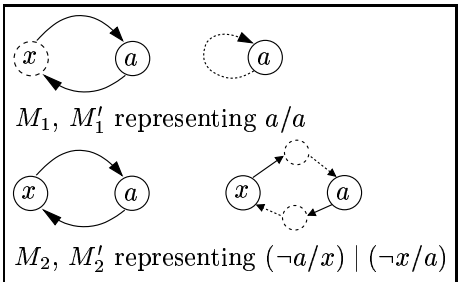
Clearly, $\mathsf{lfl}\text{-}M$ is monotone and $\mathsf{lfl}\text{-}M \leq \mathsf{cfl}\text{-}M$, i.e., all pfps of $\mathsf{lfl}\text{-}M$ are also pfps of $\mathsf{cfl}\text{-}M$. Also, $\mathsf{lfl}\text{-}M$ preserves front-lines, too, whence it is a monotone function on LFL_M. We may now rephrase Prop. 5, thus: A front-line is lazy iff it is a pfp of $\mathsf{lfl}\text{-}M : \mathsf{FL}_M \to \mathsf{FL}_M$. The fact that $\mathsf{lfl}\text{-}M \leq \mathsf{cfl}\text{-}M$ then explains why $\mathsf{LFL}_M \subseteq \mathsf{CFL}_M$. Again, from general results (see [6]) it follows that LFL_M is closed under directed union and preserved by the game function cfl.

Proposition 6. *Let (P, O) be a front-line. Then, (P, O) is maximal lazy if it is a maximal fixed point of $\mathsf{lfl}\text{-}M$. Specifically, $(P, \neg P)$ is lazy iff $P = \mu X.(\langle\tau\rangle(P \wedge X) \vee \langle\iota\rangle\neg P)$.*

As before with coherence, a maximal lazy front-line (P, O) need neither be uniquely defined nor two-valued. For the component $(\neg c/b) \mid (\neg b/c) \mid (c,\neg a,\neg b/a)$ we find that there are two lazy responses (= maximal lazy front-lines) $(P_1, O_1) = (\{b\}, \{a, c\})$ and $(P_2, O_2) = (\{c\}, \{b\})$. Of those only the former is two-valued. In the latter signal a remains in untenable no-man's-land. It cannot be defended consistently neither as part of P_2 nor of O_2.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

Note that in this example it does not matter if the intermediate room associated with the trigger conjunction $c \wedge \neg a \wedge \neg b$ of transition $c,\neg a,\neg b/a$ is visible or secret. We obtain the same lazy front-lines. This is not in general so. Consider the mazes M_1, M_1' and M_2, M_2' in Fig. 4 corresponding to components a/a and $(\neg a/x) \mid (\neg x/a)$, respectively, where M_1' is a short-hand for M_1 and M_2 a short-hand for M_2'. In M_1 the room x is secret which means that $(\{x\},\{a\})$ is the only non-trivial lazy front-line, while in M_2 both $(\{x\},\{a\})$ and $(\{a\},\{x\})$ are lazily defensible.



M_1, M_1' representing a/a

M_2, M_2' representing $(\neg a/x) \mid (\neg x/a)$

Fig. 4. The role of secret rooms.

We can now state the connection between lazy front-lines and Pnueli & Shalev Statecharts responses. We read each secret corridor $x \xrightarrow{\tau} y$ as a transition y/x and each visible corridor $x \xrightarrow{\iota} y$ as the transition $\overline{y}/x$ of Statecharts and maze M as the parallel composition $\mathsf{statecharts}(M)$ of all these transitions, which may thought of as the flat and normalised encoding of a complex hierarchical Statecharts automaton [25].

Theorem 1. *Let M be a finite maze and $\mathsf{statecharts}(M)$ the Statecharts program associated with M. Then, P is a Pnueli & Shalev response of $\mathsf{statecharts}(M)$ iff $(P,\neg P)$ is a lazy front-line in M.*

5.3 Eager Responses

Finally, we banish infinite plays and decree that player A must terminate his response in a finite number of steps by pushing the (control, execution) token back to the environment, which in turn must become satisfied and stop eventually. This is the winning condition of A-termination. We say a front-line (P,O) is *eager* if it is defensible by an A-terminating strategy. Let EFL_M be the set of eager front-lines. Eager defence is a finite process: A front-line (P,O) can only be defended if player A manages to drive the opponent into a dungeon (which by assumption is always visible). In this case all rooms in P are *winning* and all rooms in O are *losing positions*. A room that is not contained in any eagerly defensible front-line, and thus is neither a winning nor a losing position, is a *draw position*. In the previous example $(\neg c/b) \mid (\neg b/c) \mid (c,\neg a,\neg b/a)$ none of the two lazy responses $(P_1,O_1) = (\{b\},\{a,c\})$ and $(P_2,O_2) = (\{c\},\{b\})$ is eager as their defence involves infinite cycling between b and c. The only lazy front-line it has is $(\emptyset,\emptyset)$, i.e, all rooms are draw positions. Eager front-lines seem to be the strongest, most constructive notion of response so far considered in the literature on synchronous languages. As it turns out eager defence strategies correspond to the constructive semantics of Esterel [9]. An eagerly defensible (P,O) is a (partial) constructive Esterel response. Draw positions are (non-constructive) Esterel signals with undecided status.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

Theorem 2. *The greatest eager front-line* $(\mathsf{Win}, \mathsf{Lose})$ *of a finite maze* M *coincides with the constructive response of* $\mathsf{esterel}(M)$*, i.e., a signal* $a \in S$ *is present in* $\mathsf{esterel}(M)$ *iff* $a \in \mathsf{Win}$ *and* a *is absent in* $\mathsf{esterel}(M)$ *iff* $a \in \mathsf{Lose}$*. The program* $\mathsf{esterel}(M)$ *is constructive in the sense of Esterel iff* $(\mathsf{Win}, \mathsf{Lose})$ *is two-valued.*

There is also a fixed point characterisation of eager front-lines. It is known from the theory of Esterel (see [9, 5]) that $(\mathsf{Win}, \mathsf{Lose})$ can be obtained as the (2-dimensional) least fixed point of $\mathsf{cfl}\text{-}M : \mathsf{FL}_M \to \mathsf{FL}_M$, i.e., $(\mathsf{Win}, \mathsf{Lose}) = \mu(X, Y).\, \mathsf{cfl}\text{-}M(X, Y)$, where $\mathsf{cfl}\text{-}M(X, Y) = (\langle\tau\rangle X \vee \langle\iota\rangle Y\ ,\ [\tau]Y \wedge [\iota]X)$. Assuming $\mathsf{cfl}\text{-}M$ is continuous (always for finite or finitely branching mazes, in particular those generated from pure Esterel programs) this fixed point can be computed in the standard fashion by iteration, viz. as $\bigcup_{i<\omega} \mathsf{cfl}\text{-}M^i(\emptyset, \emptyset)$. The successive approximations $\mathsf{cfl}\text{-}M^i(\emptyset, \emptyset)$ not only accumulate the front-line $(\mathsf{Win}, \mathsf{Lose})$ but also construct an eager defence strategy for it, which essentially corresponds to the Must/Can analysis of Esterel [9].

The following Proposition 7 identifies a monotone function $\mathsf{efl}\text{-}M : \mathsf{FL}_M \to \mathsf{FL}_M$ satisfying $\mathsf{efl}\text{-}M \leq \mathsf{lfl}\text{-}M$ (i.e., a strengthening of the lazy semantics) such that eager front-lines coincide with the pfps of $\mathsf{efl}\text{-}M$.

Proposition 7. *A front-line* (P, O) *is eager iff it is a pfp of the function* $\mathsf{efl}\text{-}M$ *defined as* $\mathsf{efl}\text{-}M(P, O) := \mu(X, Y).\, \mathsf{cfl}\text{-}M(P \wedge X, O \wedge Y)$.

Thus, eager front-lines, too, are nothing but pfps of monotone functions on front-lines, this time of

$$\mathsf{efl}\text{-}M(P, O) := \mu(X, Y).\, \mathsf{cfl}\text{-}M(P \wedge X, O \wedge Y).$$

Again, the maximal pfps coincide with the maximal fixed-points, but now these are indeed the least fixed point of $\mathsf{cfl}\text{-}M$. Although eager responses may not be two-valued they are always deterministic.

6 Conclusions

In this paper we have identified four natural levels of semantics for synchronous (instantaneous) response in a game–theoretic setting as defensible front-lines according to increasing restrictions on winning conditions. The levels $\mathsf{DFL}_M \supset \mathsf{CFL}_M \supset \mathsf{LFL}_M \supset \mathsf{EFL}_M$ correspond to *classical*, *coherent*, *lazy*, and *eager* valuations, repectively. Each level is associated with a particular degree of computational constructiveness, DFL_M being the weakest and EFL_M the strongest, reflecting a characteristic operational interpretation of system execution. At DFL_M there is no constructiveness requirement, CFL_M is intimately linked with inertiality, LFL_M is Statecharts, and EFL_M corresponds to Esterel. The game theory gives a coherent interpretation for non-determinism and partiality of the non-classical semantics. We have shown that these semantics can be obtained algebraically as (maximal) post-fixed points of a decreasing sequence of monotone functions $\mathsf{dfl}\text{-}M > \mathsf{cfl}\text{-}M > \mathsf{lfl}\text{-}M > \mathsf{efl}\text{-}M$ on the directed complete partial ordering $\mathsf{FL}_M \subset (2^S)^2$ of front lines. In this way the game semantics turns algebraic

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

and induces suitable truth–values for presence and absence of signals at a given level. The explicit presentation of these truth-values, however, is only known for dfl-M, viz. classical Boolean logic, and for lfl-M where we get Gödel's 3-valued intuitionism. The truth-value interpretation of cfl-M and efl-M is left open, in particular it is not clear if these are finite-valued.

The levels of constructiveness characterised here using games and winning conditions also play an important role in Normal Logic Programming (NLP), which extends standard definite Horn clause programming by permitting negative literals in clause bodies and queries. Various types of models based on three- and many-valued interpretations have been developed in the literature for normal logic programs. We refer the reader to [34] for a survey of the classic results.

There are two important methodological differences between logic programming and synchronous languages: First, algebraic and logic models of NLP are judged according to their ability to reconstruct a fixed (standard) operational execution model, viz. *negation as finite failure* also known as *SLDNF resolution.* Where a many-valued semantics does not fit SLDNF completely, one seeks to identify restricted classes of programs for which they coincide. In synchronous programming, in contrast, one is not dealing with just a single operational model but with many of them, each accommodating different scheduling principles and implementation platforms. Second, since synchronous programs typically model embedded and reactive systems with some degree of (low-level) asynchrony it is essential that non-determinism and concurrency is represented adequately. NLP, on the other hand is based on a strong deterministic and sequential execution model, which even constrains the order in which clauses and literals are executed. In this sense the work presented here aims at a rather more general setting than what is considered in NLP. In another sense, though, our scope is more restricted here, viz., in considering only propositional programs. Current other work [29, 32], however, indicates that the game-theoretic point of view is useful also for First-order Logic Programming. We believe that generating constructive models of Horn clauses from different types of winning conditions may provide further insights into the relationship between operational and denotational semantics of Logic Programming. At the propositional level eager front-lines are related to the *3-valued models* of Fitting, lazy front-lines to the *stable models* and the *supported models* of NLP are the binary coherent front-lines (see [34] for definitions of these types of models and references).

To appreciate the versality of the games model suppose, e.g., we wanted to capture the standard sequential execution in literals a, b in a Prolog clause $c :- a, b$ such that the whole clause loops if a loops even if the second clause b has a finite failure. Under a symmetric interpretation of conjunction (or parallel models such as those considered in this paper) the clause would not hang up in executing a but fail instead on the grounds that a conjunction is false if any of the conjuncts is.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

We can model the asymmetric form of conjunction in terms of mazes as seen in Fig. 5, say under the eager[3] winning condition: Suppose that a is undefined, i.e. it cannot be won by the starting nor by the second player. Then, as we check easily, room c is undefined, too. The first player U cannot win: He can only go into the intermediate room x, from where $\overline{U}$ puts him into a which is undefined by assumption. Similarly, the first player will not lose: If $\overline{U}$ takes him to a he will not lose by assumption, if $\overline{U}$ takes him to b then he can avoid losing by going to intermediate room y. There $\overline{U}$ has two options, either to move down to b again, from where U can repeat, or to go up to a, either handing over the turn to U or not, depending on which of the two parallel corridors he choses. Yet, in either case U will not lose by assumption on a. Further, one can show that if a is decided, i.e. the first or the second player has a winning strategy from a, then no player will ever move from b into the intermediate room y since he would then give his opponent the option to move up into a under full control of who gets the turn in a. Since a is determined the opponent will win as first or second player, accordingly. But if room y is never used we may as well remove it together with all corridors connecting it with b and a. This yields the same maze as the one we get from the symmetric translation of $a, b/c$ (see Fig. 1). Thus, if a is determined our coding in Fig. 5 coincides with the standard symmetric conjunction. It would be interesting to explore this further and to undertake a systematic study of three-valued semantics of NLP in terms of games and winning conditions.

Regarding future work it should also be noted that we have not discussed the algorithmic computation of front lines. For classical semantics this is well-known and for Statecharts and Esterel these can be derived directly from the respective original work [31, 9]. The algorithmic construction of (binary) lazy front lines has been described in [31] in terms of a non-deterministic fixed point search with backtracking and in [14] deterministically using BDD-techniques. If the immediate benefits of the game-theoretic setting for improvements on the algorithmic side may be modest it can at least guide the search profitably as a new reference point. Also, not having considered any composition operations for mazes we must leave open questions of compositionality and full abstraction. This will be addressed in future work. From [26] it is known already that the game semantics from lfl-M yields a fully abstract model for Statecharts under parallel composition, a problem that had been open for a long time.

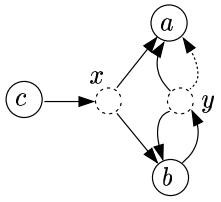


Fig. 5. The maze for the Prolog clause $c :- a, b$ where a is evaluated strictly before b.

[3] This winning condition corresponds directly to the standard least fixed-point interpretation employed in Prolog.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

The main contribution of this paper, so we believe, is to identify game theory as an expressive framework for studying seemingly disparate step semantics for synchronous languages (implementing the Synchrony Hypothesis) from a single vantage point. The field of Statecharts semantics (see e.g. [19]) in particular has been notoriously incoherent and controversial. We believe this is partly due to the lack of an adequate semantic framework to manage the subtleties of causal cycles. Game theory surely has a lot to offer here.

References

1. S. Abramsky. Games in the semantics of programming languages. In *11th Amsterdam Colloquium*, pages 1–5. Univ. of Amsterdam, 1997.
2. S. Abramsky and R. Jagadeesan. Games and full completeness for multiplicative linear logic. *Journal of Symbolic Logic*, 59(2):543–574, 1994.
3. S. Abramsky, R. Jagadeesan, and P. Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409–470, 2000.
4. S. Abramsky and P.-A. Melliés. Concurrent games and full completeness. In *Logic in Computer Science*, 1999.
5. J. Aguado, G. Lüttgen, and M. Mendler. A-maze-ing Esterel. In *Synchronous Languages, Applications, and Programming (SLAP'03)*, Porto, July 2003.
6. J. Aguado and M. Mendler. Constructive semantics for instantaneous reactions. Bamberger Beiträge zur Wirtschaftsinformatik und Angewandten Informatik 63, University of Bamberg, February 2005.
7. J. Aguado and M. Mendler. Constructive semantics for instantaneous reactions. In Ghica and McCusker [15], pages 16–31.
8. P. Baillot, V. Danos, T. Ehrhard, and L. Regnier. Believe it or not, AJM's games model is a model of classical linear logic. In *LICS '97*, pages 68–75, 1997.
9. G. Berry. The constructive semantics of pure Esterel. CMA, Ecole des Mines, INRIA, 1999. Draft version 3.0.
10. G. Berry. The foundations of Esterel. In *Essays in Honour of Robin Milner*. MIT Press, 2000.
11. Janusz A. Brzozowski and Carl-Johan H. Seger. *Asynchronous Circuits*. Springer-Verlag, New York, 1995.
12. L. de Alfaro and T. A. Henzinger. Interface automata. In *Joint 8th Europ. Softw. Eng. Conf. and 9th Int. Symp. Found. Softw. Eng. (ESEC/FSE 01)*, 2001.
13. L. de Alfaro, T. A. Henzinger, and M. Stoelinga. Timed interfaces. In *International Workshop on Embedded Software (EMSOFT 2002)*. Springer, 2002.
14. T. Friedrich. How to implement Statecharts intuitionistically. Darwin research project, Department of Computer Science, The University of Sheffield, 2002.
15. D. R. Ghica and G. McCusker, editors. *Games for Logic and Programming Languages (GaLoP 2005)*, Edinburgh, April 2005.
16. J.-Y. Girard. Locus solum: From the rules of logic to the logic of rules. *Mathematical Structures in Computer Science*, 11(3):301–506, June 2001.
17. E. Grädel, W. Thomas, and Th. Wilke, editors. *Automata Logics, and Infinite Games*. Springer, 2002.
18. N. Halbwachs. *Synchronous programming of reactive systems*. Kluwer, 1993.
19. D. Harel and A. Naamad. The STATEMATE semantics of Statecharts. *ACM Transactions on Software Engineering*, 5(4):293–333, October 1996.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.

20. J. Hintikka and G. Sandu. What is the logic of parallel processing? *International Journal of Foundations of Computer Science*, 6(1):27–49, 1995.
21. C. Huizing, R. Gerth, and W.-P. de Roever. Modeling Statecharts behavior in a fully abstract way. In M. Dauchet and M. Nivat, editors, *CAAP'88*, pages 271–294. Springer LNCS 299, 1988.
22. J. M. E. Hyland and C.-H. L. Ong. Fair games and full completeness for multiplicative linear logic without the mix-rule. Unpublished, 1992.
23. M. Hyland and L. Ong. On full abstraction for PCF: I, II and III. *Information and Computation*, 163(2):285–408, 2000.
24. N. Leveson, M. Heimdahl, H. Hildreth, and J. Reese. Requirements specification for process-control systems. *IEEE Trans. Softw. Eng.*, 20(9), 1994.
25. G. Lüttgen and M. Mendler. Statecharts: From visual syntax to model-theoretic semantics. In *Workshop on Integrating Diagrammatic and Formal Specification Techniques*, pages 615–621. Austrian Computer Society, 2001.
26. G. Lüttgen and M. Mendler. The intuitionism behind Statecharts steps. *ACM Transactions on Computational Logic*, 3(1):1–41, January 2002.
27. G. Lüttgen, M. von der Beeck, and R. Cleaveland. Statecharts via process algebra. In J.C.M. Baeten and S. Mauw, editors, *CONCUR'99*, pages 399–414. Springer LNCS 1664, 1999.
28. A. Maggiolo-Schettini, A. Peron, and S. Tini. Equivalences of Statecharts. In U. Montanari and V. Sassone, editors, *CONCUR'96*, pages 687–702. Springer LNCS 1119, 1996.
29. D. Miller and A. Saurin. A game semantics for proof search: Preliminary results. In Ghica and McCusker [15], pages 92–106.
30. A. Pietarinen and G. Sandu. Games in philosophical logic. *Nordic J. Philosophical Logic*, 4:143–173, 2000.
31. A. Pnueli and M. Shalev. What is in a step: On the semantics of Statecharts. In T. Ito and A.R. Meyer, editors, *TACS'91*, pages 244–264. Springer LNCS 526, 1991.
32. P. Rondogiannis and W. W. Wadge. An infinite-game semantics for negation in logic programming. In Ghica and McCusker [15].
33. V. A. Saraswat, R. Jagadeesan, and V. Gupta. Foundations of timed concurrent constraint programming. In *Logic in Computer Science*, 1994.
34. J. C. Shepherdson. Logics for negation as failure. In Y. N. Moschovakis, editor, *Logic from Computer Science*, pages 521–583. Springer, 1991.
35. C. Stirling. *Modal and Temporal Properties of Processes*. Springer, 2001.
36. W. Thomas. On the synthesis of strategies in infinite games. In *STACS'95*, pages 1–13. Springer LNCS 900, 1995.

J. Aguado, M. Mendler: Constructive Game Semantics for Instantaneous Reactions.
In: D. R. Ghica and G. McCusker (eds.) Games for Logics and Programming Languages (GaLoP 2005), Edinburgh, April 2005, pp. 16-31.