Axiomatizing an Algebra of Step Reactions for Synchronous Languages

Gerald Lüttgen[1] and Michael Mendler[2]

[1] Department of Computer Science, Sheffield University, 211 Portobello Street, Sheffield S1 4DP, U.K., g.luettgen@dcs.shef.ac.uk
[2] Fakultät für Wirtschaftsinformatik und Angewandte Informatik, Universität Bamberg, D–96045 Bamberg, michael.mendler@wiai.uni-bamberg.de

Abstract. This paper introduces a novel algebra for reasoning about step reactions in synchronous languages, such as macro steps in Harel, Pnueli and Shalev's Statecharts and instantaneous reactions in Berry's Esterel. The algebra describes step reactions in terms of configurations which can both be read in a standard operational as well as in a model–theoretic fashion. The latter arises by viewing configurations as propositional formulas, interpreted intuitionistically over finite linear Kripke structures. Previous work by the authors showed the adequacy of this approach by establishing compositionality and full–abstraction results for Statecharts and Esterel. The present paper generalizes this work in an algebraic setting and, as its main result, provides a sound and complete equational axiomatization of step reactions. This yields, for the first time in the literature, a complete axiomatization of Statecharts macro steps, which can also be applied, modulo encoding, to Esterel reactions.

1 Introduction

Synchronous languages provide a popular framework for designing and programming *event–based reactive systems*. Prominent examples of such languages include Harel's *Statecharts* [2], which is a graphical language that extends finite–state machines by concepts of state hierarchy, concurrency, and event priority, and Berry's *Esterel* [1], which is a textual language having similar features to Statecharts. Today, both languages are supported by commercial tools that mainly focus on generating running code. The development of semantic–based verification tools, however, is still in its infancy, which is partly due to the lack of sufficiently simple compositional semantics.

The semantics of Statecharts, as conceived by Pnueli and Shalev [15], and of Esterel are based on the idea of *cycle–based reaction*, where first the input events, as defined by a system's environment, are sampled at the beginning of each cycle, then the system's reaction in form of the emission of further events is determined, and finally the generated events are output to the environment. Statecharts and Esterel differ in the details of what exactly constitutes a cycle, which is also called a *macro step* in Statecharts and an *instantaneous reaction* in Esterel. Moreover, Esterel refers to events as signals. Both languages have in common that they obey the semantic principles of *synchrony* and *causality*. The

synchrony requirement reflects the mechanism behind cycle–based reaction and is mathematically modeled via the *synchrony hypothesis*. This hypothesis ensures that reactions and propagations of events are instantaneous, which models an idealized system behavior and is practically justified by the observation that reactive systems usually perform much faster than their environments. Causality refers to the requirement that the reason for an event to be generated in a system reaction can be traced back to the input events provided by the environment. Esterel differs from Statecharts in that it further adopts the principles of *reactivity* and *determinism*. Reactivity implies that in each cycle, a system response, in the form of generated events, can be constructed for any input an environment may provide. Determinism requires for this response to be unique.

This brief discussion highlights the variety of possible choices when defining a semantics for step reactions, with different choices implying subtly different semantics. Recent research by the authors, aiming at a unifying semantic framework for synchronous languages, has concentrated on employing ideas from *intuitionistic logic* for describing step reactions [8–10]. Intuitionistic logic, in contrast to classical logic, is constructive and thus truly reflects the operational character of step reactions in the light of causality: it rejects the principle of the excluded middle, viz., that events are either always present or always absent throughout a reaction. This axiom cannot be maintained for a compositional semantics that allows the system environment to inject events *during* a step reaction. Indeed, our intuitionistic setting led to compositional and fully–abstract characterizations of Statecharts macro steps and Esterel reactions [9, 10].

This paper introduces a simple yet expressive algebra for describing and reasoning about step reactions in terms of so–called *configurations* and presents an equational axiomatization for it. In particular, this gives for the first time in the literature a sound and complete axiomatization for Statecharts macro steps, which can also be applied, modulo encoding, to Esterel reactions. The step algebra's semantics is inspired by the authors' previous work and reads configurations as propositional formulas, interpreted intuitionistically over finite linear Kripke structures, to which we refer as *sequence structures* (cf. Sec. 2). Our axiomatization is then built on top of this algebra (cf. Sec. 3), and its proof of completeness combines techniques used in process algebras and logics (cf. Sec. 4). At its heart, the proof employs a process–algebraic notion of *normal form* that in turn is defined by model–theoretic means. Our axioms not only shed light on the semantics of step reactions, but also provide groundwork for an exact axiomatic comparison of popular synchronous languages.

2 Step Algebra

This section introduces our step algebra for reasoning about those step reactions that may be specified within *event–based synchronous language*. Usually, synchronous languages, such as *Statecharts* [2, 15] or *Esterel* [1] (with its graphical front–end *SyncCharts*), enrich the notation of finite state machines by mechanisms for expressing hierarchy, concurrency, and priority. This makes it possible

to refine a single state by a state machine, to run several state machines in parallel which may coordinate each other via broadcasting events, and to give some transitions precedence over others, respectively. The underlying semantics is based on a nested two–level execution model. The top–level synchronous computation cycle consists of a sequence of *macro steps*, each of which represents a single reaction of the system to some environment stimulus. If the notion of a reaction is understood, this synchronous level can be handled by conventional automata models and poses no semantic challenge. Hence, the interesting question is what exactly constitutes a reaction. Although the environment treats reactions as atomic, each individual reaction is indeed a computation in its own right, a sequence of *micro steps* in which concurrent components of the system under consideration are engaged. This introduces subtle issues regarding compositionality and full-abstraction, which are well documented in the literature (cf. Sec. 5). Moreover, many synchronous languages —and variants of Statecharts in particular— are distinguished by how exactly they define valid sets of concurrent transitions that can fire together in a single reaction. The work presented here features an algebra of individual reactions that arose from the systematic study of these problems and variations in the literature. We start off with an informal description of step reactions.

In event–based synchronous languages, each transition t is labeled by two sets of events, which are referred to as *trigger* and *action*, respectively. The trigger is a set of positive events P and negative events $\overline{N}$, taken from a countable universe of events Ev and their negated counterparts in $\overline{Ev} =_{\mathrm{df}} \{\overline{e} : e \in Ev\}$, respectively. For convenience, we define $\overline{\overline{e}} =_{\mathrm{df}} e$. Intuitively, t is enabled and forced to fire if the transition's environment signals all events in P but none in N. The effect of firing t is the generation of all events in the transition's action $A \subseteq Ev$. These events might in turn trigger transitions in different parallel components, thus enabling a *causal* chain reaction whose length is bounded by the number of parallel components within the program under consideration. A *step reaction* is then the set of all events that are already valid at the beginning of the step or generated during the step. When constructing steps in the suggested operational manner, it is possible to experience inconsistencies, namely when a firing transitions generates some event e, whose absence, i.e., its negation $\overline{e}$, was assumed when firing a previous transition in the step. Since an event cannot be both present and absent within the same step reaction, due to the principle of *global consistency*, the choice sequence leading to the inconsistency is rejected, and a different sequence must then be chosen. If no consistent sequence is possible, the step construction fails altogether. Alternatively, one could also say that the step construction remains unfinished; it waits for the environment to provide additional events to disable the transition(s) that produced the inconsistency.

The semantic subtlety of step reactions arises precisely from the capability of defining transitions whose enabledness disables other transitions, as well as from the interpretation of negated trigger events. In the light of this discussion, the key operators for combining transitions in synchronous languages are *parallel composition* and *event negation*. State hierarchy is merely a notational convenience

rather than a semantically relevant operator. Observe that parallel composition and event negation also allow one to express *nondeterministic choice* [8]. For example, a choice between two transitions $P_1, \overline{N_1}/A_1$ and $P_2, \overline{N_2}/A_2$ might be written as the parallel composition $P_1, \overline{N_1}, \overline{e_2}/A_1, e_1 \parallel P_2, \overline{N_2}, \overline{e_1}/A_2, e_2$, where e_1, e_2 are distinguished events not occurring in the triggers or actions of the two original transitions and where the comma notation stands for union, i.e., $X, Y =_{\mathrm{df}} X \cup Y$ and $X, x =_{\mathrm{df}} X \cup \{x\}$. Finally, we often write x for the singleton set $\{x\}$.

Syntax. For the purposes of this paper, it is convenient to work with a quite general syntax for reactions, which allows us to encode several dialects of synchronous languages, including Statecharts and Esterel. In fact, it may be seen as the smallest common superset of the kernel of both languages. The terms describing step reactions, to which we also refer as *configurations*, are defined inductively as follows:

$$C \quad ::= \quad 0 \mid A \mid I/C \mid C \| C$$

where $A \subseteq Ev$ and $I \subseteq Ev \cup \overline{Ev}$. Intuitively, 0 stands for the configuration with the empty behavior, $A \subseteq Ev$ denotes the signaling of all events in A, configuration I/C encodes that configuration C is triggered by the presence of the positive events in I and the absence of the negative events in I, and $C_1 \| C_2$ describes the parallel composition of configurations C_1 and C_2. Observe that the semantics of configuration 0 coincides with the semantics of $A = \emptyset$; nevertheless, it seems natural to include 0. For notational convenience, we let the transition slash / to have lower binding power than parallel composition $\|$ and interpret a nesting of transition slashes in a right–associative manner.

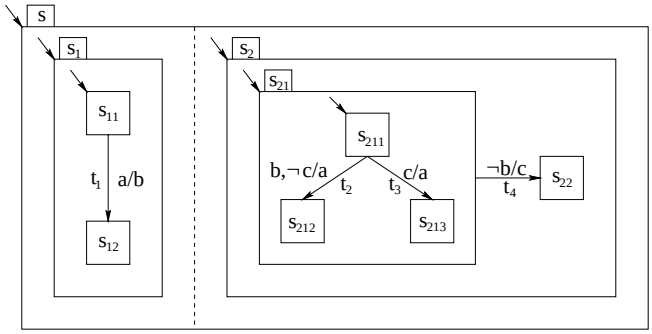

Fig. 1. Example Statechart

We illustrate our syntax by means of an example. Consider the Statechart depicted in Fig. 1 and assume that all components are in their initial states marked by small unlabeled arrows. Then the first Statechart step determining the initial Statechart reaction, may be encoded in our syntax as the configuration

$$C_{ex} \quad =_{\mathrm{df}} \quad a/b \parallel b, \overline{c}, \overline{e_3}, \overline{e_4}/a, e_2 \parallel c, \overline{e_2}, \overline{e_4}/a, e_3 \parallel \overline{b}, \overline{e_2}, \overline{e_3}/c, e_4 \ .$$

Although the main body of this paper focuses on Statecharts, it is worth noting again that reactions of Esterel programs can be encoded in our syntax as well. In

particular, our syntax admits nested triggers, such as in $I/(J/C)$, and parallel composition in a transition's action, such as in $I/(C\|D)$. While these features are only present in Esterel but not in Statecharts, their inclusion will prove notationally convenient, but not essential (cf. Sec. 5).

Semantics. In order for a semantics on configurations to be useful for the purposes of this paper, it must meet several requirements. First, it must be *compositional* to be axiomatizable, i.e., it must give rise to a semantic equivalence on configurations that is a congruence. Second, it should be compatible with the existing semantics of the synchronous languages of interest, in particular with Statecharts' and Esterel's semantics. Unfortunately, many semantics for synchronous languages, including the one of Statecharts as originally conceived by Harel, Pnueli and Shalev [2, 15], are not compositional. Recent research by the authors has revealed an appealing model–theoretic framework for studying Statecharts' and Esterel's semantics, which is based on reading configurations as simple propositional formulas that are *intuitionistically* interpreted over finite linear Kripke structures [8–10]. This model–theoretic approach allows not only for a compositional semantics but also for establishing full–abstraction results. The present paper generalizes this work in an algebraic setting.

The key idea is to consider a single step reaction as a *stabilization* process, where the synchronous environment is only concerned with the final response, while the system under consideration takes care of the actual sequence of events that leads to the stationary state. The main feature that distinguishes a stabilization process from an arbitrary computation is that it is a *monotonically increasing approximation* of the final step reaction. This means that once an event from the final step reaction has become present or asserted by the firing of a transition generating it, the event will remain present until the stationary state is reached. Formally, we interpret configurations over finite, nonempty, strictly increasing sequences $M = (M_1, M_2, \ldots, M_n)$, where $n \in \mathbb{N}$ and $M_i \subseteq Ev$, to which we refer as *sequence structures*. 'Strictly increasing' means that $M_i \subsetneq M_{i+1}$, for all $1 \leq i < n$. We say that M satisfies C, in signs $M \models C$, if the following holds for all $1 \leq i \leq n$:

$M_i \models 0$	*always*
$M_i \models A$	*if* $A \subseteq M_i$
$M_i \models I/C$	*if* $(I \cap Ev \subseteq M_i$ *and* $\overline{(I \cap \overline{Ev})} \cap M_n = \emptyset)$ *implies* $M_i \models C$
$M_i \models C_1 \| C_2$	*if* $M_i \models C_1$ and $M_i \models C_2$

This definition is a shaved version of the standard semantics obtained when reading a configuration as a formula in propositional intuitionistic logic [16], i.e., when taking events to be atomic propositions and replacing $\overline{e}$ by the negation $\neg e$, concatenation of events in sets and '$\|$' by conjunction '$\wedge$', and the transition slash '/' by implication '$\supset$'. An empty trigger and the configuration 0 are identified with *true*. Then we have $M \models C$ if and only if C is valid in the intuitionistic Kripke structure M. Note that, for sequence structures $M = (M_1)$ of length one, the notions of sequence model and classical model coincide; hence, intuitionistic logic is a refinement of classical logic, and we simply write M_1 for (M_1). The

utility of intuitionistic logic comes into play when ensuring global consistency within reaction sequences. This is because intuitionistic logic interprets negation globally for a sequence M, and not locally for single states M_i.

Our semantics suggests the following equivalence on configurations. Configurations C_1, C_2 are *step congruent*, in signs $C_1 \simeq C_2$, if $M \models C_1 \;\Leftrightarrow\; M \models C_2$ holds for all sequence structures M.

Proposition 1 (Congruence). *The equivalence $\simeq$ is indeed a congruence, i.e., $C_1 \simeq C_2$ implies $C_1 \| D \simeq C_2 \| D$ and $I/C_1 \simeq I/C_2$, for all configurations C_1, C_2, D and for all triggers $I \subseteq Ev \cup \overline{Ev}$.*

It was proved in [9] that step congruence $\simeq$ is *compositional* and *fully–abstract* with respect to Statecharts macro–step semantics, according to the operational semantics of Pnueli and Shalev [15]. More precisely, for any two Statecharts configurations C, D, we have $C \simeq D$ if and only if $C \| P$ and $D \| P$ have exactly the same step responses for all parallel environments P. The remainder of this paper presents an axiomatic characterization of our step congruence.

3 Axiomatization

Our axioms system is displayed in Table 1, where $A, B, N, P \subseteq Ev$, $I, I_1, I_2 \in Ev \cup \overline{Ev}$, and $e \in Ev$, and where C, C_1, C_2, C_3 are configurations. We write $\vdash C_1 = C_2$ to state that $C_1 = C_2$ can be derived from the axioms via standard equational reasoning. Axioms (A1)–(A6) and (B1)–(B4) are fairly natural and do not need much operational justification. When taking them together, it is easy to see that every configuration is equivalent to a flat parallel composition of transitions, without nested triggers and where ordering and duplication is immaterial. Note that Axioms (B3) and (B4) can actually be deduced from Axioms (A1)–(A6), (B1), and (B2) by induction on the structure of C. Because of their fundamental nature, however, we have included them as first–class axioms.

Table 1. Axioms for the step congruence

(A1)	$\emptyset = 0$	(A2)	$A \,\|\, B = A \cup B$
(A3)	$\emptyset/C = C$	(A4)	$I/0 = 0$
(A5)	$I_1, I_2/C = I_1/(I_2/C)$	(A6)	$I/(C_1 \,\|\, C_2) = I/C_1 \,\|\, I/C_2$
(B1)	$C_1 \,\|\, C_2 = C_2 \,\|\, C_1$	(B2)	$(C_1 \,\|\, C_2) \,\|\, C_3 = C_1 \,\|\, (C_2 \,\|\, C_3)$
(B3)	$C \,\|\, C = C$	(B4)	$C \,\|\, 0 = C$

(C1)	$P, I/P = 0$	
(C2)	$C = C \,\|\, I/C$	
(C3)	$A \,\|\, A, I/C = A \,\|\, I/C$	
(C4)	$P, \overline{N}/C = 0$	*if* $P \cap N \neq \emptyset$
(D1)	$P, \overline{N}/A = P, \overline{N}/A, B$	*if* $N \cap A \neq \emptyset$
(D2)	$P, \overline{N}/A = P, e, \overline{N}/A \,\|\, P, \overline{N}, \overline{e}/A$	*if* $N \cap A \neq \emptyset$
(D3)	$\overline{N}/C \,\|\, P, \overline{N}/A = \|\{\overline{N}, \overline{e}/C : e \in P\} \,\|\, P, \overline{N}/A$	*if* $N \cap A \neq \emptyset$ *and* $P \neq \emptyset$

We concentrate on explaining the remaining, more interesting axioms. Axiom (C1) describes that, if the firing of a transition merely reproduces in its action some of the events required by its trigger, then we might just as well not fire the transition at all. Hence, it is equivalent to configuration 0. Axiom (C2) states that by adding in parallel to a configuration C a guarded version I/C of it, the behavior remains unchanged. This is intuitively clear since the extra I/C component, when and if it fires at all, only produces the behavior encoded by C, which is already present anyway. Logically speaking, guarding is a *weakening* operation. Axiom (C3) is perhaps the most important equation, as it emulates the firing of transitions. The left–hand side $A \parallel A, I/C$ represents a situation in which some events A have become present while at the same time there is a pending transition $A, I/C$ that is waiting, among other preconditions I, for the events in A. Hence, it is safe to cancel out all dependencies of C on A and to replace $A, I/C$ by I/C. Hence, Axiom (C3) is nothing but a version of the *cut* rule known from logic. Axiom (C4) deals with inconsistencies in triggers. If a configuration C is guarded by a trigger $P, \overline{N}$ in which some event is required to be both present and absent, i.e., $P \cap \overline{N} \neq \emptyset$, then this guarded configuration will never become active. In this case, $P, \overline{N}/C$ is obviously equivalent to 0.

The remaining Axioms (D1)–(D3) are concerned with conflicts between the trigger and action of a transition. They axiomatize the effect of transitions that produce a failure under certain trigger conditions. More precisely, these axioms involve a transition $P, \overline{N}/A$ with $N \cap A \neq \emptyset$, whose firing leads to a global inconsistency. Such a transition rejects the completion of all macro steps in which its trigger $P, \overline{N}$ is true. Thus, since $P, \overline{N}/A$ can never fire in a consistent way, the step construction cannot terminate in a situation in which trigger $P, \overline{N}$ holds true. In other words, whenever all P have become present the step construction must continue until at least one event in N is present, in order to inactivate the transition. If this does not happen the step construction fails. Axioms (D1)–(D3) formalize three different consequences of this. Axiom (D1) reflects the fact that, since $P, \overline{N}/A$ can never contribute to a completed step if $N \cap A \neq \emptyset$, we may add arbitrary other events B to its action, without changing its behavior. Logically, this axiom corresponds to the laws $e \wedge \neg e \equiv \mathit{false}$ and $\mathit{false} \supset B$, for any B. Axiom (D2) offers a second way of reading the inconsistency between triggers and actions. Since at completion time any event e is either present or absent, the same rejection that $P, \overline{N}/A$ produces can be achieved by $P, \overline{N}, e/A \parallel P, \overline{N}, \overline{e}/A$. This is because if e is present at completion time, then $P, \overline{N}, e/A$ raises the failure; if e is absent, then $P, \overline{N}, \overline{e}/A$ does the job. This corresponds to the law $\neg(\neg e \wedge \neg\neg e)$ in intuitionistic logic. To see this, simply instantiate P as *true* and A and N both as *false*. Then, Axiom (D2) turns into the equivalence $(\mathit{true} \wedge \neg \mathit{false}) \supset \mathit{false} \equiv ((\mathit{true} \wedge \neg \mathit{false} \wedge e) \supset \mathit{false}) \wedge (\mathit{true} \wedge \neg \mathit{false} \wedge \neg e) \supset \mathit{false}$, which simplifies to $\neg(\neg e \wedge \neg\neg e)$. Finally, consider Axiom (D3) that encodes the following intuition. Instead of saying that $P, \overline{N}/A$ generates a failure, if all events in P are present and all events in N are absent, we might say that, if all events in N are absent, then at least one of the events in P must be absent, provided the step under consideration is to be completed without failure. But then any parallel

component of the form $\overline{N}/C$, which becomes active on the absence of all events in N, can be replaced by the parallel composition $\|\{\overline{N}, \overline{e}/C \ : \ e \in P\}$. The reason is that, if $\overline{N}/C$ fires at all in the presence of transition $P, \overline{N}/A$, then at least one of the weaker transitions $\overline{N}, \overline{e}/C$ will be able to fire at some point, depending on which of the events in P it is that will be absent to avoid failure. Again there is a logic equivalent for this, namely the law $\neg(p_1 \wedge p_2) \supset \neg p_1 \vee \neg p_2$ that holds for *linear* Kripke models. This can be seen by instantiating N and A as *false*, P as $p_1 \wedge p_2$, and C as $\neg p_1 \vee \neg p_2$. It is not difficult to show that then Axiom (D3) becomes equivalent to $((\neg p_1 \vee \neg p_2) \wedge \neg(p_1 \wedge p_2)) \equiv \neg(p_1 \wedge p_2)$, which in turn is logically equivalent to $\neg(p_1 \wedge p_2) \supset \neg p_1 \vee \neg p_2$. Last, but not least, it is important to note that configuration $P, \overline{N}/A$, for $N \cap A \neq \emptyset$, is not the same as configuration 0, since the former inevitably produces a failure if its trigger is true, while 0 does not respond at all, not even by failure.

Theorem 1 (Correctness). *Let* $\vdash C_1 = C_2$. *Then,* $C_1 \simeq C_2$.

The correctness of each of our axioms can be established directly along our notion of sequence models. However, since this is exactly the standard interpretation of propositional intuitionistic formulas over finite linear Kripke structures, one may simply employ the wealth of knowledge on intuitionistic logic for the proof [16].

4 Completeness

The proof of completeness of our step–congruence axiomatization employs a notion of *normal form*. As usual, the idea is to first show that every configuration can be rewritten into one in normal form using our axioms and then establish the desired completeness result for configurations in normal form. The purpose of the normal form is to lay out explicitly, in a canonical syntactic form, the behavior offered by a configuration relative to a fixed and finite set of relevant events. Typically, these are all the events that occur in the configurations we wish to normalize. For simplicity, let us take Ev to be this finite set; the complement A^c of any set $A \subseteq Ev$ is then also finite. A normal form relative to Ev is a parallel composition of simple transitions

$$(\|_{i \in I} P_i, \overline{N_i}/A_i) \ \| \ (\|_{j \in J} E_j, \overline{E_j^c}/Ev) \ .$$

The transitions are grouped into two categories, indexed by I and J, respectively. The former category encodes individual, partial or complete, step reactions, whereas the latter category records the conditions under which the step construction fails (to complete). A transition $P_i, \overline{N_i}/A_i$ of the first kind specifies that, if the events in P_i are known to be present and those in N_i are absent, then A_i is one possible reaction of the configuration to P_i. A transition $E_j, \overline{E_j^c}/Ev$ of the second kind enforces that the step construction cannot consistently complete with just the events in E_j present. In order to complete the reaction at least one event outside E_j must become available as well. In the light of this discussion, a normal form may be seen as a "response table", where, given a set of environment events, one may look up the associated partial or complete step reaction

or learn about immanent failure. This response–table interpretation is reflected in a number of structural properties on normal–form configurations.

Definition 1. *A configuration C is in* normal form*, if it has the shape*

$$(\|_{i\in I} P_i, \overline{N_i}/A_i) \parallel (\|_{j\in J} E_j, \overline{E_j^c}/Ev)\,,$$

where I, J are disjoint finite index sets, $E_j \subsetneq Ev$, for all $j \in J$; and if it satisfies:

1. *$P_i \subseteq A_i$ and $P_i \cap N_i = \emptyset$, for all $i \in I$;*
2. *$B \models C$ iff $\forall j \in J.\, B \neq E_j$;*
3. *$B \models C$ iff $\exists i \in I.\, B = N_i^c$;*
4. *$B \models C$ and $P \subseteq B$ implies $\exists i \in I.\, P_i = P$ and $B = N_i^c$;*
5. *$(P_i, N_i^c)^* = A_i$, for all $i \in I$ with $N_i \cap A_i = \emptyset$; and*
6. *$N_i \cap A_i = \emptyset$, for all $i \in I$.*

where $(P,N)^* =_{df} \bigcap\{E : (E,N) \models C,\, P \subseteq E \subseteq N\}$ and $B \subseteq Ev$ arbitrary.

Conds. (2)–(5) encode structural properties that refer to our model–theoretic semantics. It is through these that the normal form obtains its essential semantic relevance. The other conditions, Conds. (1) and (6), are simple local consistency requirements. Note that the side condition $N_i \cap A_i = \emptyset$ of Cond. (5) is redundant due to Cond. (6); however, its presence will simplify matters later. It seems important to stress that Conds. (2)–(5) could of course be rewritten in purely syntactic terms, simply by expanding the definition of $\models$ in the respective cases. This, however, would only obscure the purpose of these conditions, which is to permit an easy correspondence between syntax and semantics in the completeness proof. Note that there may be other notions of normal form; we do not claim that Def. 1 is necessarily the best choice for performing equivalence proofs.

Proposition 2. *For any configuration C there exists a configuration C' in normal form such that $\vdash C = C'$.*

Proof. Let C be an arbitrary configuration. Because of Axioms (A3)–(A6) we may assume without loss of generality that C is given as a flat parallel composition of simple transitions $P, \overline{N^c}/A$. We will rewrite C using our axioms in six steps, obtaining configurations C_i, for $1 \leq i \leq 6$, such that C_i satisfies normal–form Conds. (1) through (i). We say that C_i is in i–normal form, or i–nf for short. At each stage we define the *J–part* of C_i to be the collection of all transitions of the form $B, \overline{B^c}/Ev$, where $B \subsetneq Ev$. All other transitions make up the *I–part*. In this way each C_i naturally splits into the form $(\|_{i\in I} P_i, \overline{N_i}/A_i) \parallel (\|_{j\in J} E_j, \overline{E_j^c}/Ev)$ such that $E_j \subsetneq Ev$, for all $j \in J$.

We will employ the associativity and commutativity Axioms (B1) and (B2) for parallel composition without explicit mentioning, wherever convenient. Furthermore, observe that since all C_i have the same semantics, it does not matter whether we read validity $\models$ in Conds. (2)–(5) relative to C or C_i.

1. Assume Cond. (1) is violated by a transition $P, \overline{N}/A$ in C, i.e., $P \not\subseteq A$ or $P \cap N \neq \emptyset$. In the second case we can simply drop the transition because of Axioms (C4) and (B4). In the former case, we can transform $P, \overline{N}/A$ so that Cond. (1) is satisfied:

$$\begin{aligned} \vdash P, \overline{N}/A &= P, \overline{N}/A \,\|\, 0 && \text{(B4)} \\ &= P, \overline{N}/A \,\|\, P, \overline{N}/P && \text{(C1)} \\ &= P, \overline{N}/(A \| P) && \text{(A6)} \\ &= P, \overline{N}/A, P && \text{(A2)} \end{aligned}$$

   Making these first adjustments yields C_1, with $\vdash C = C_1$, where C_1 is in 1–nf. All successive transformations to C_1 either introduce new transitions that satisfy Cond. (1) or, if not, we can repeat this step to clean out or transform the transitions such that Cond. (1) does hold.
2. Next we consider Cond. (2), starting off with direction ($\Longrightarrow$). Let $B \models C_1$, i.e., B is a classical model, and further $B = E_j$ for some $j \in J$. Then $B \subsetneq Ev$ and also $Ev \subseteq B$, since $B \models B, \overline{B^c}/Ev$. This is an obvious contradiction, whence this direction of Cond. (2) is automatically fulfilled in C_1.
   For the other direction ($\Longleftarrow$) we show that, for any $B \not\models C_1$, the equivalence $\vdash C_1 = C_1 \,\|\, B, \overline{B^c}/Ev$ is derivable. If we apply this for every such B we get our desired 2–nf C_2, subsuming the new transitions $B, \overline{B^c}/Ev$ in the J–part of the 2–nf. Note that always $Ev \models C$, which means $B \subsetneq Ev$ in such a case. The transformation $\vdash C_1 = C_1 \,\|\, B, \overline{B^c}/Ev$ is obtained in the following fashion. Since by assumption $B \not\models C_1$, there must be some transition $P, \overline{N}/A$ in C_1 such that $B \not\models P, \overline{N}/A$. Hence, $\vdash C_1 = C_1' \,\|\, P, \overline{N}/A$, where C_1' is C_1 without the transition $P, \overline{N}/A$. Now observe that $B \not\models P, \overline{N}/A$ implies $P \subseteq B$ and $N \cap B = \emptyset$, but $A \not\subseteq B$. We then reason as follows, abbreviating $P, \overline{N}/A \,\|\, B, \overline{B^c}/B$ by D.

$$\begin{aligned} &\vdash P, \overline{N}/A \\ &= P, \overline{N}/A \,\|\, B, \overline{B^c}/B && \text{(B4,C1)} \\ &= D \,\|\, B, P, \overline{B^c}, \overline{N}/A \,\|\, B, P, \overline{B^c}, \overline{N}/B && \text{(C2, A5, twice)} \\ &= D \,\|\, B, \overline{B^c}/A \,\|\, B, \overline{B^c}/B && (P \subseteq B,\ N \cap B = \emptyset, \text{ i.e., } N \subseteq B^c) \\ &= D \,\|\, B, \overline{B^c}/(A \| B) && \text{(A6)} \\ &= D \,\|\, B, \overline{B^c}/A, B && \text{(A2)} \\ &= D \,\|\, B, \overline{B^c}/Ev && (\text{D1},\ A \not\subseteq B, \text{ i.e., } A \cap B^c \neq \emptyset) \\ &= P, \overline{N}/A \,\|\, B, \overline{B^c}/B \,\|\, B, \overline{B^c}/Ev \\ &= P, \overline{N}/A \,\|\, B, \overline{B^c}/Ev && \text{(A6, A2)} \end{aligned}$$

   This shows $\vdash C_1 = C_1' \,\|\, P, \overline{N}/A = C_1' \,\|\, P, \overline{N}/A \,\|\, B, \overline{B^c}/Ev = C_1 \,\|\, B, \overline{B^c}/Ev$.
3. The direction ($\Longrightarrow$) of Cond. (3) can be trivially satisfied by inserting parallel transitions $B, \overline{B^c}/B$ for those $B \subsetneq Ev$ that satisfy $B \models C_2$, via Axioms (B4) and (C1). This preserves Conds. (1) and (2). Note that we accommodate $B, \overline{B^c}/B$ in the I–part.
   Suppose direction ($\Longleftarrow$) is violated by $B \not\models C_2$, for which there exists a transition $P_i, \overline{N_i}/A_i$ with $B = N_i^c$ in the I–part of C_2. We must have $N_i =$

$B^c \neq \emptyset$, for otherwise $B = N_i^c = Ev$, contradicting $B \not\models C_2$. By Cond. (2), there exists a transition $B, \overline{B^c}/Ev$ in the J–part of C_2. Hence,

$$\vdash C_2 = C_2' \,\|\, P_i, \overline{B^c}/A_i \,\|\, B, \overline{B^c}/Ev\,.$$

We distinguish several cases. If $A_i = \emptyset$, then $P_i, \overline{B^c}/A_i$ is the same as $P_i, \overline{B^c}/\emptyset$ which can be eliminated from C_2 right away by Axioms (A1), (A4), and (B4). If $P_i \cap \overline{B^c} \neq \emptyset$, we can drop $P_i, \overline{B^c}/A_i$ by way of Axioms (B4) and (C4). Hence, assume that $A_i \neq \emptyset$ and $P_i \cap B^c = \emptyset$. Now, if $B = \emptyset$, then $B^c = Ev$ and $P_i = \emptyset$. Hence, we use Axioms (A2) and (A6) to derive $\vdash\ P_i, \overline{B^c}/A_i \,\|\, B, \overline{B^c}/Ev = \overline{Ev}/A_i \,\|\, \overline{Ev}/Ev = \overline{Ev}/Ev = B, \overline{B^c}/Ev$, which gets rid of the culprit $P_i, \overline{B^c}/A_i$. It remains to tackle the situation in which $B \neq \emptyset$. But then, since also $B^c \neq \emptyset$, we can use the equational rewriting

$$\begin{array}{lll} \vdash\ P_i, \overline{B^c}/A_i \,\|\, B, \overline{B^c}/Ev & & \\ = \overline{B^c}/(P_i/A_i) \,\|\, B, \overline{B^c}/Ev & & \text{(A5)} \\ = \,\|\, \{\overline{B^c}, \overline{e}/(P_i/A_i) \,:\, e \in B\} \,\|\, B, \overline{B^c}/Ev & & \text{(D3)} \\ = \,\|\, \{P_i, \overline{B^c}, \overline{e}/A_i \,:\, e \in B\} \,\|\, B, \overline{B^c}/Ev & & \text{(A5)} \end{array}$$

to replace in C_2, effectively, the offending $P_i, \overline{B^c}/A_i$ by the parallel composition of transitions $P_i, \overline{B^c}, \overline{e}/A_i$, for $e \in B$, each of which has a negative trigger strictly larger than the one in $P_i, \overline{B^c}/A_i$ we started off with.

By iterating these transformations over all B's and i's such that $B \not\models C_2$ and $B = N_i^c$, Cond. (3) ($\Longleftarrow$) can be achieved. The normalization must terminate since the sets B to consider become smaller and smaller in the process. Note that the resulting configuration C_3 also satisfies Conds. (1) and (2), whence it is a 3–nf.

4. Cond. (4) may be achieved by inserting into C_3 the transitions $P, \overline{B^c}/P$, for all P, B such that $P \subseteq B \models C_3$. The insertions may be done via Axioms (B4) and (C1). Note that the resulting configuration C_4 still satisfies Conds. (1)–(3), since $P \subseteq B$ is equivalent to $P \cap B^c = \emptyset$, whence it is a 4–nf.
5. Consider an arbitrary transition $P_i, \overline{N_i}/A_i$, satisfying $N_i \cap A_i = \emptyset$, in the I–part of C_4. We will show how to enforce Cond. (5) for this transition. Under the assumptions, we know $P_i \subseteq A_i \subseteq N_i^c$ and $N_i^c \models C_4$ by Conds. (1) and (3). In order to show $(P_i, N_i^c)^* = A_i$, it is sufficient to establish the following two properties:

   (a) $(A_i, N_i^c) \models C_4$; and

   (b) $(X, N_i^c) \models C_4$ and $P_i \subseteq X \subseteq N_i^c$ implies $A_i \subseteq X$, for any $X \subseteq Ev$.

   Assume that Property (5a) is not yet satisfied, i.e., $(A_i, N_i^c) \not\models C_4$. Then, there must be a transition $P, \overline{N}/A$ in C_4 such that $(A_i, N_i^c) \not\models P, \overline{N}/A$. This transition could be of the form $P_k, \overline{N_k}/A_k$, for some $k \in I$, or of the form $E_j, \overline{E_j^c}/Ev$, for some $j \in J$.

   Because of $N_i^c \models C_4$, we have $N_i^c \models P, \overline{N}/A$. But then, $(A_i, N_i^c) \not\models P, \overline{N}/A$ means that $(A_i, N_i^c) \models P, \overline{N}$ and $(A_i, N_i^c) \not\models A$. Hence, in particular, $A_i \supseteq P$, $A_i \not\supseteq A$, and $N_i^c \cap N = \emptyset$, i.e., $N \subseteq N_i$.

We now show that $P_i, \overline{N_i}/A_i \,\|\, P, \overline{N}/A = P_i, \overline{N_i}/A_i, A \,\|\, P, \overline{N}/A$ by the following calculations, where $N_1 =_{\mathrm{df}} N_i \setminus N$ and $A_1 =_{\mathrm{df}} A_i \setminus P$:

$$
\begin{array}{ll}
\vdash P_i, \overline{N_i}/A_i \,\|\, P, \overline{N}/A & \\
= P_i, \overline{N_1}, \overline{N}/A_1, P \,\|\, P, \overline{N}/A & \\
= P_i, \overline{N_1}, \overline{N}/A_1 \,\|\, P_i, \overline{N_1}, \overline{N}/P \,\|\, P, \overline{N}/A & (A2, A6) \\
= P_i, \overline{N_1}, \overline{N}/A_1 \,\|\, \overline{N}/(P_i, \overline{N_1}/P \,\|\, P/A) & (A5, A6) \\
= P_i, \overline{N_1}, \overline{N}/A_1 \,\|\, \overline{N}/(P_i, \overline{N_1}/P \,\|\, P_i, \overline{N_1}/(P/A) \,\|\, P/A) & (C2) \\
= P_i, \overline{N_1}, \overline{N}/A_1 \,\|\, \overline{N}/(P_i, \overline{N_1}/(P \,\|\, P/A) \,\|\, P/A) & (A6) \\
= P_i, \overline{N_1}, \overline{N}/A_1 \,\|\, \overline{N}/(P_i, \overline{N_1}/(P \,\|\, A) \,\|\, P/A) & (C3) \\
= P_i, \overline{N_1}, \overline{N}/A_1, P, A \,\|\, P, \overline{N}/A & (A2, A5, A6) \\
= P_i, \overline{N_i}/A_i, A \,\|\, P, \overline{N}/A &
\end{array}
$$

This allows us to replace transition $P_i, \overline{N_i}/A_i$, for which $(A_i, N_i^c) \not\models C_4$ by the transition $P_i, \overline{N_i}/A_i, A$. If now $N_i \cap (A_i \cup A) = \emptyset$, i.e., $A_i \cup A \subseteq N_i^c$, then we find $(A_i \cup A, N_i^c) \models P, \overline{N}/A$ and $A_i \cup A \supsetneq A_i$ since $A_i \not\supseteq A$. Thus, using this technique, one can saturate the A_i until, for all transitions $P, \overline{N}/A$, there exists no $i \in I$ such that $N_i \cap A_i = \emptyset$ and $(A_i, N_i^c) \not\models P, \overline{N}/A$. This will ensure Property (5a), for all $i \in I$.

It remains to establish Property (5b). Let $(X, N_i^c) \models C_4$ for some $X \subseteq Ev$ such that $P_i \subseteq X \subseteq N_i^c$. Hence, $(X, N_i^c) \models P_i, \overline{N_i}/A_i$. Since $(X, N_i^c) \models P_i, \overline{N_i}$, we consequently know that $(X, N_i^c) \models A_i$, i.e., $A_i \subseteq X$ as desired. Let C_5 denote the 5–nf configuration resulting from this normalization step.

6. Let us assume that some transition $P, \overline{N}/A$ in C_5 violates Cond. (6). Then, using Axiom (D1) we rewrite $\vdash P, \overline{N}/A = P, \overline{N}/Ev$ first, and then by repeated applications of Axiom (D2) we obtain $\vdash P, \overline{N}/Ev = \|\{E, \overline{E^c}/Ev : P \subseteq E \subseteq N^c\}$. In this way, the offending original transition $P, \overline{N}/A$ in C_5 can be eliminated completely in terms of transitions indexed by J. This establishes Cond. (6) and does not destroy any of the conditions previously established. The result is a 6–nf C_6 with $\vdash C = C_6$.

Configuration C_6 is now the desired normal form of C. □

Theorem 2 (Completeness). *Let $C_1 \simeq C_2$. Then, $\vdash C_1 = C_2$.*

Proof. Let, w.l.o.g., C_1, C_2 be in normal form such that $C_1 \simeq C_2$, i.e., $M \models C_1$ iff $M \models C_2$. Due to symmetry and Axioms (B1)–(B4), it suffices to show that every parallel component, i.e., transition, of C_1 also occurs in C_2.

Consider a transition of the form $P_i, \overline{N_i}/A_i$ occurring in C_1. Since C_1 is in normal form, we know by Cond. (3) that $N_i^c \models C_1$. Hence, by premise $C_1 \simeq C_2$, we have $N_i^c \models C_2$. We may now apply Cond. (4), since $P_i \subseteq N_i^c$ by Cond. (1), to obtain some $i' \in I$ such that $P_{i'}, \overline{N_{i'}}/A_{i'}$ is a transition in C_2 with $N_{i'} = N_i$ and $P_{i'} = P_i$. By Cond. (5), $A_{i'} = (P_{i'}, N_{i'}^c)^* = (P_i, N_i^c)^* = A_i$. Note that the definitions of $(P_{i'}, N_{i'}^c)^*$ in C_2 and $(P_i, N_i^c)^*$ in C_1 coincide, because of $C_1 \simeq C_2$.

Consider a transition of the form $E_j, \overline{E_j^c}/Ev$ in C_1. Since C_1 is in normal form, we know by Cond. (2) that $E_j \not\models C_1$. Hence, $E_j \not\models C_2$ by the premise $C_1 \simeq C_2$. Further, by Cond. (2) applied to normal form C_2, we conclude the existence of some $j' \in J$ such that $E_{j'}, E_{j'}^c/Ev$ is a transition in C_2 with $E_{j'} = E_j$. □

5 Discussion and Related Work

There exists a wealth of related work on the semantics of synchronous languages, especially Statecharts. Our paper focused on the most popular original semantics of Harel's Statecharts, as defined by Pnueli and Shalev in their seminal paper [15]. Since this semantics combines the synchrony hypothesis and the causality principle, it cannot be compositional if step reactions are modeled by input–output–functions over event sets, according to a result by Huizing and Gerth [5]. Within the traditional style of labeled transition systems, researchers have then concentrated on providing compositionality for Pnueli and Shalev's semantics either by taking transition labels to be partial orders encoding causality [7, 12] or by explicitly including micro–step transitions [11]. Our step algebra is related to the former kind of semantics, where causality is encoded via intuitionistically interpreted sequence structures. However, in contrast to the other mentioned work, our logical approach lends itself to establishing full–abstraction results [9] and the equational axiomatization of Statecharts presented here.

A different approach to axiomatizing Statecharts was suggested by de Roever et al. for an early and lateron rejected Statecharts semantics that does not obey global consistency [3]. In their setting, it is admissible for a firing transition to generate an event, whose absence was assumed earlier in the construction of the macro step under consideration. This leads to a very different semantics than the one of Pnueli and Shalev [15], for which Huizing, Gerth, and de Roever gave a denotational account in [6]. This denotational semantics provided the groundwork for an axiomatization by Hooman, Ramesh, and de Roever [4]. However, in contrast to our work that *equationally* axiomatized the step congruence underlying Pnueli and Shalev's semantics, Hooman et al. supplied a Hoare–style axiomatization for both liveness and safety properties of Statecharts, which was proved to be sound and complete with respect to the denotational semantics of Huizing et al. [6]. A similar approach was taken by Levi regarding a process–algebraic variant of Pnueli and Shalev's Statecharts and a real–time temporal logic [7]. It should be noted that the settings of de Rover et al. and of Levi deal with sequences of macro steps and not just single macro steps, as our step algebra does. However, extending the step algebra and its axiomatization to sequences of macro steps should not be difficult. In such a more general development the configuration algebra introduced here would play the role of a synchronization algebra [17], around which a macro–step process language would be built.

The results of this paper are not restricted to Statecharts but can also be applied to other languages, in particular to Berry's Esterel [1]. The authors have shown in [10], using the same model–theoretic framework of intuitionistic sequence structures as for Statecharts, how the instantaneous core of Esterel can be faithfully and compositionally encoded in terms of propositional formulas. This is done in such a way that the operational execution of the encoding produces the same responses as the execution of the original program under the semantics of Esterel [1]. It is not difficult to see that the propositional formulas corresponding to Esterel configurations build a subclass in our step algebra, when taking $Ev =_{\mathrm{df}} \{s{=}1, s{=}0 \ : \ s \textit{ is a signal}\}$, where

s=1 stands for signal s is present 'high' and s=0 for s is present 'low'. This subclass, however, requires the full syntax of our step algebra, which allows for nested transition triggers. For example, the instantaneous Esterel program `present` a `then present` b `else emit` c `end end` would be translated into the configuration a=1/(b=0/c=1); see [10] for details. Because of the existence of this encoding of Esterel reactions into our step algebra, which preserves Esterel's semantics, the axiomatization presented here can directly be used to reason about Esterel reactions. For the sake of completeness, it needs to be mentioned that some initial work on axiomatizing Esterel has been carried out within an encoding of Esterel programs in a variant of the duration calculus [14]. However, this work aims at an axiomatic semantics for Esterel rather than an equational axiomatization of the underlying step congruence.

The step algebra presented in this paper focused on the most essential operators in synchronous languages. In the future we would like to enrich our algebra and its axiomatization to accommodate an operator for *event scoping*, or signal hiding, which is used in Esterel [1] and Argos [13]. Moreover, instead of encoding the external–choice operator +, as found in the hierarchy operator of Statecharts, via parallel composition and negated events, it is possible to include + as a primitive operator in our step algebra. To do so, one only needs to add a silent, non–synchronizing event in the action of every transition; see [9].

Our syntax is a common superset of the kernel languages of Statecharts and Esterel. Specifically, the applications of Axioms (A5), (A6), and (C2) introduce nested triggers and parallel compositions in the action part of a transition, which do not exist in standard Statecharts configurations. However, it can be shown that any axiomatic proof can be performed properly within the Statecharts fragment, when using the following axioms instead of their original counterparts:

(A6') $I/A \parallel I/B = I/(A \cup B)$

(C2') $I/A = I/A \parallel I, J/A$

(C3') $I/A \parallel A, J/B = I/A \parallel A, J/B \parallel I, J/B$

(D3') $I, \overline{N}/B \parallel P, \overline{N}/A = \{I, \overline{N}, \overline{e}/B \ : \ e \in P\} \parallel P, \overline{N}/A$, if $N \cap A \neq \emptyset$, $P \neq \emptyset$

6 Conclusions and Future Work

This paper presented a uniform algebra, to which we referred as step algebra, for reasoning about step reactions in synchronous languages, such as those originating from Statecharts and Esterel. The algebra covers single reactions, and as such constitutes a first important step towards an axiomatization of Statecharts and related languages. Its semantics was inspired by previous work of the authors, which adapted ideas from intuitionistic logics for defining a compositional semantics for step reactions. Our main result is a sound and complete axiomatization of the resulting step congruence in our step algebra, whose completeness proof mixes techniques from process algebra and logic. This yields, for the first time in the literature, a complete axiomatization of Statecharts macro steps, in the sense of Pnueli and Shalev. Modulo a simple syntactic translation, this axiomatization can be adapted to instantaneous reactions in Esterel as well. We

believe that our approach provides important groundwork for comparing popular synchronous languages by means of axioms, an approach that already proved successful in process algebra, and also for developing suitable compositional verification methods.

Regarding future work, we plan to integrate other operators employed in synchronous languages into our step algebra, in particular an operator for event scoping. Additionally, our algebra should be extended to step sequences, by adding prefixing operators and recursion.

Acknowledgments. We would like to thank the anonymous referees for their valuable comments and suggestions. The second author was supported by EPSRC grant GR/M99637 and the EC Types Working Group IST–EU–29001.

References

1. G. Berry. The constructive semantics of pure Esterel, 1999. Draft Version 3. Available at http://www-sop.inria.fr/meije/Personnel/Gerard.Berry.html.
2. D. Harel. Statecharts: A visual formalism for complex systems. *SCP*, 8:231–274, 1987.
3. D. Harel, A. Pnueli, J. Pruzan-Schmidt, and R. Sherman. On the formal semantics of Statecharts. In *LICS '87*, pp. 54–64. IEEE Computer Society Press, 1987.
4. J.J.M. Hooman, S. Ramesh, and W.-P. de Roever. A compositional axiomatization of Statecharts. *Theoretical Computer Science*, 101:289–335, 1992.
5. C. Huizing. *Semantics of Reactive Systems: Comparison and Full Abstraction.* PhD thesis, Eindhoven Univ. of Technology, 1991.
6. C. Huizing, R. Gerth, and W.-P. de Roever. Modeling Statecharts behavior in a fully abstract way. In *CAAP '88*, vol. 299 of *LNCS*, pp. 271–294, 1988.
7. F. Levi. *Verification of Temporal and Real-Time Properties of Statecharts.* PhD thesis, Univ. of Pisa-Genova-Udine, 1997.
8. G. Lüttgen and M. Mendler. Statecharts: From visual syntax to model-theoretic semantics. In *Integrating Diagrammatic and Formal Specification Techniques*, pp. 615–621. Austrian Computer Society, 2001.
9. G. Lüttgen and M. Mendler. The intuitionism behind Statecharts steps. *ACM Trans. on Computational Logic*, 3(1):1–41, 2002.
10. G. Lüttgen and M. Mendler. Towards a model-theory for Esterel. In *Synchronous Languages, Applications, and Programming*, vol. 65:5. ENTCS, 2002. To appear.
11. G. Lüttgen, M. von der Beeck, and R. Cleaveland. Statecharts via process algebra. In *CONCUR '99*, vol. 1664 of *LNCS*, pp. 399–414, 1999.
12. A. Maggiolo-Schettini, A. Peron, and S. Tini. Equivalences of Statecharts. In *CONCUR '96*, vol. 1119 of *LNCS*, pp. 687–702, 1996.
13. F. Maraninchi. Operational and compositional semantics of synchronous automaton compositions. In *CONCUR '92*, vol. 630 of *LNCS*, pp. 550–564, 1992.
14. P.K. Pandya, Y.S. Ramakrishna, and R.K. Shyamasundar. A compositional semantics of Esterel in Duration Calculus. In *AMAST '95*, vol. 936 of *LNCS*, 1995.
15. A. Pnueli and M. Shalev. What is in a step: On the semantics of Statecharts. In *TACS '91*, vol. 526 of *LNCS*, pp. 244–264, 1991.
16. D. van Dalen. Intuitionistic logic. In *Handbook of Philosophical Logic*, vol. III, chap. 4, pp. 225–339. Reidel, 1986.
17. G. Winskel. A compositional proof system on a category of labelled transition systems. *Inform. and Comp.*, 87(1/2):2–57, 1990.