

Conciseness Considerations on Logics of Action¹

COSTAS KOUTRAS,² ATHENA GAGA,³ and PAVLOS PEPPAS⁴

ABSTRACT

In recent years a number of formal results have been established on the *correctness* of some well-known logics of actions. Correctness however is only one side of the coin; for a theory of action to qualify as a solution to the frame problem, the *conciseness* of its representations is also relevant. In a previous paper, [15], we developed a general framework within which logics of action can be assessed both in terms of correctness as well as conciseness. Herein we apply the results obtained in [15] to evaluate the logic of action proposed by Kartha and Lifschitz in [7].

Keywords: Frame Problem, Reasoning about Action, Action Languages, Knowledge Representation.

1 INTRODUCTION

The central problem in *Reasoning about Action*, and indeed one of the most significant problems in the area of *Knowledge Representation*, is what is known as the *frame problem* [13]. Loosely speaking, the frame problem is the problem of reasoning *effectively* about the behavior of a dynamic system (more precisely, the problem of *designing a formal logic* capable of carrying out such reasoning). As first observed by McCarthy and Hayes in [13], if one uses first-order logic to reason about a dynamic system, one needs to state explicitly, not only what changes during the occurrence of an action (i.e. statements such as “shooting the gun will kill Fred”), but also *everything that remains the same* (i.e. statements of the form “shooting the gun will not change the color of the gun”, “shooting the gun will not turn on the light”, “shooting the gun will not cause an earthquake”, etc.). Statements of the later kind are called *frame axioms*. Clearly, writing down all the frame axioms for domains of a reasonable size and complexity⁵ is an intractable task, and certainly a cumbersome approach to reason about such domains.

Over the last thirty years, much of the research in Reasoning about Action has been devoted to the design of more “effective” logics, capable of reasoning about dynamic domains without relying on frame axioms. Indeed, a number of such logics of action have been proposed (see [17] for a detailed account on the work in

¹ Published in the *Journal of Intelligent Systems*, 13(1): 71-94, Freund and Pettman, 2004. A preliminary version of this paper appears at the Proceedings of the 6th *Hellenic European Conference on Computer Mathematics and its Applications*, Athens, 2003, under the title “A Formal Conciseness Assessment of AR_0 ”.

² Dept of Computer Science and Technology, University of Peloponnese, Tripolis, Greece, ckoutras@cc.ece.ntua.gr

³ 28th of October 7, Agia Paraskeui, Athens 15231, Greece, athegg@yahoo.co.uk

⁴ Dept of Business Administration, University of Patras, Patras 265 00, Greece, ppeppas@otenet.gr

⁵ The frame axioms become much more elaborate in domains where actions have *side effects*.

this area). However, for many of these logics, the burden of the frame axioms was lifted at the expense of the *correctness* of the produced inferences.

This phenomenon was particularly evident in the '80s: in more than one occasion, a logic of action was proposed as a solution to the frame problem, only to be quickly rendered inadequate a few months later by a counterexample for which the logic produced the wrong inferences [18], [4], [1], [6], etc.

By the early '90s it was evident that the correctness of a proposed logic of action ought to be taken more seriously; in particular, it was acknowledged by many researchers [10], [7], [14], [16], that a logic of action should be *formally* assessed in terms of correctness before any claims to the frame problem could be made. To this end, a number of formal frameworks were introduced, the most elaborate of which was the one developed by Sandewall [16].

While formally assessing correctness is clearly a step in the right direction, all of the above frameworks neglected a second and equally important aspect of a logic of action; namely, the *conciseness* of its representations. Recall that the reason for moving away from first-order logic in the first place,⁶ was the large size of its representations due to the frame axioms. Hence evaluating a logic of action in terms of conciseness, is as important as evaluating it in terms of correctness. Moreover, a simple criterion of conciseness such as “no frame axioms” would not work for domains of some complexity – for such domains at least *some* background information about the “physics” of the domain needs to be stated explicitly [5], [11], [18], no matter how effective the logic of action might be (at see [15] for details).

In a previous paper, [15], we developed a rather general framework that provides the means for *formally* assessing a logic of action both in terms of correctness and conciseness. This, to our knowledge, was one of the first such proposals.⁷ Our main aim in this paper is to use the formal tools introduced in [15], to assess the performance of a well-known logic of action, proposed by Kartha and Lifschitz in [7]. The central result of the paper is a theorem showing that the logic of Kartha and Lifschitz is both correct and *sufficiently concise* with respect to a certain class of domains. To our knowledge, this theorem is the first of its kind; that is, the first result that *formally* evaluates the performance of a logic of action in terms of conciseness. In that sense, we consider the contribution of this article to be twofold: Firstly, it proves the appropriateness of a particular logic of action in solving the frame problem (for a given class of domains). Secondly, and perhaps more importantly, it demonstrates how such results can be obtained with the aid of the formal tools introduced in [15].

A few remarks are due regarding this second point. The framework introduced in [15] was developed in rather abstract terms. While this makes the framework very general, it also means that considerable effort needs to be invested in “re-packaging” a specific logic of action in order to fit the constructs and terminology of the framework. The exercise we undertake herein – that is, the re-packaging of Kartha and Lifschitz logic followed by an assessment of its conciseness using the results in [15] – is therefore quite instructive in demonstrating a methodology for obtaining conciseness results. Moreover, it illustrates the practical value of the framework introduced in [15].

The rest of the paper is structured as follows. In the following section we briefly review the basic concepts on conciseness introduced in [15]. In section 3 we re-package the theory of action presented in [7] to fit into the

⁶ More precisely, from *situation calculus* [13].

⁷ It should be noted that others have also worked on similar notions like *compactness* and *space efficiency* – see for example [2], [3], [8] – but in most cases this work relates to broader issues in Knowledge Representation and is not specifically tailored to the frame problem.

framework of [15], and we identify the class of domains against which the theory will be evaluated. Section 4 contains the main conciseness results of the paper. Finally in section 5 we make some concluding remarks.

2 A FORMAL ACCOUNT OF CONCISENESS

There are two methodological doctrines on whose basis the framework in [15] was developed. The first, mainly due to Sandewall [16], is the idea of assessing a theory of action against a *predefined class of domains*, rather than trying to prove *universal* claims about the theory. Sandewall calls these predefined classes of domains *ontological families*. Examples of ontological families are, the class of domains where all actions are *deterministic*, the class of domains for which there are no dependencies between fluents (and therefore actions have no implicit ramifications), etc.

The second central characteristic of the framework in [15], is that conciseness is defined in *comparative* rather than absolute terms. In other words, given two theories of action R and R' , our framework provides means for determining whether R is (in a certain sense) “more concise” than R' , without however referring explicitly to some “degree of conciseness” for neither R nor R' .⁸

Based on the above two methodological doctrines, formal definitions were developed in [15] for the concepts of a *dynamic domain*, a *scenario*, an *ontological family*, and a *reasoning system* (corresponding to a theory of action). With the aid of these definitions, the notions of *correctness* and *conciseness* of a reasoning system relative to an ontological family, were formally defined. We next review these basic definitions and results.

2.1 Dynamic Domains and Scenarios

One of the cornerstones of our framework is the concept of a *history*. Intuitively, for a given dynamic domain, a history represents a possible development of the domain in time. For example, in a setting where time is assumed to be discrete, one can think of a history as an alternating sequence of states and actions. Aiming for generality, the framework in [15], makes no assumptions about the internal structure of histories; instead it treats histories as primitive objects, based on which the rest of the formal constructs are built. The set of all histories is denoted by H and is assumed to be given *a priori*.

A *dynamic domain* Δ (or simply *domain*) is defined as a set of histories:

Definition 1: A dynamic domain Δ is a set of histories; i.e. $\Delta \subseteq H$.

The intended interpretation of Δ is that it contains all possible developments of the domain it represents. We note that, while every domain is defined as a set of histories, an arbitrary set of histories may not necessarily be a domain. The set of all domains is denoted by U and is given *a priori*.

Next, *ontological families* and *scenarios* are defined as follows:

Definition 2: An ontological family F is a set of domains; i.e. $F \subseteq U$.

⁸ Notice that both in [15] and in this article, we often abuse terminology and talk about the conciseness of a theory of action, instead of the conciseness of its *representations*.

Definition 3: A *scenario* Σ is a set of histories; i.e. $\Sigma \subseteq H$.

We note that when it comes to scenarios, sets of histories have a different meaning than what they have for domains. In particular they represent *partial* information about a history of the domain at focus (in the same way that a set of Kripke's possible worlds represent propositions). According to this reading, as far as the scenario $\Sigma = \{h_1, h_2, \dots, h_n\}$ is concerned, any of the histories $h_1, h_2, \dots, h_n$ is equally likely to be the *actual* history.

Similarly to domains, not every set of histories is necessarily a scenario. The set of all scenarios is denoted by G . There are only two assumptions we make about G . Firstly, we assume that G is large enough to distinguish between any two different domains. More precisely, we assume that, for any two different domains $\Delta, \Delta' \in U$, there is a scenario $\Sigma \in G$, such that $\Delta \cap \Sigma \neq \Delta' \cap \Sigma$. The second assumption about G is that the intersection of a scenario with a domain, is always a scenario. In other words, for any scenario $\Sigma \in G$ and any domain $\Delta \in U$, $\Sigma \cap \Delta \in G$.

Given a scenario Σ related to a domain Δ , the *completion* of Σ in Δ is a new scenario Σ' defined as the set $\Sigma' = \Sigma \cap \Delta$. The scenario Σ' represents the *conclusions* (together with the initial premises) that can be drawn from Σ given our background knowledge about Δ .⁹ In particular, Σ' is generated from Σ by removing all histories that are not possible in Δ ; notice that these are the only histories that can be *safely* dismissed.

2.2 Reasoning Systems

The framework in [15] also introduces the notion of a *reasoning system* that is meant to be a general model for a theory of action. Formally, a reasoning system R is defined as follows:

Definition 4: A *reasoning system* R is a tuple $R = \langle D, S, \eta, \theta, C \rangle$ where,

- D is a nonempty set, the elements of which we shall call domain representations.
- S is a nonempty set, the elements of which we shall call scenario representations.
- η is a subset of $U \times D$, called the domain encoding scheme; we assume that for all $\Delta \in U$ there is at least one $d \in D$ (called an encoding of Δ) such that $\langle \Delta, d \rangle \in \eta$.
- θ is a subset of $G \times S$, called the scenario encoding scheme; we assume that for all $\Sigma \in G$ there is at least one $s \in S$ (called an encoding of Σ) such that $\langle \Sigma, s \rangle \in \theta$.
- C is a function from $D \times S$ to S that we shall call the scenario closure operator.

We shall briefly explain the intended meaning of the above definition by examining the way it applies to situation calculus [13]. A domain representation conveys *background information* about the dynamic domain at focus, and in situation calculus takes the form of a set of sentences like $(holds(GunLoaded, s) \rightarrow \neg holds(AliveFred, Result(shoot, s)))$ or $(holds(GunLoaded, s) \rightarrow holds(GunLoaded, Result(wait, s)))$. A scenario representation on the other hand describes observations about a particular history of the domain; a typical scenario representation in situation calculus consists of sentences like $holds(AliveFred, S_0)$ or

⁹ This definition of Σ' is in line with our Kripke-like interpretation of sets of histories; under this reading, drawing conclusions from a scenario Σ is equivalent to eliminating histories from Σ .

$\neg \text{holds}(\text{GunLoaded}, S_0)$.¹⁰ Given a scenario and domain representation, situation calculus can draw further conclusions using first-order logic as its *inference engine*. The inference engine is one of the central components of a theory of action, and in the above definition it is modeled by the scenario closure operator C . Finally, the domain and scenario encoding schemes, associate domains and scenarios with their corresponding representations; it is essentially through these encoding schemes that the representations in D and S acquire their intended meaning. A detailed discussion on the motivation behind the above definition can be found in [15].

Notice that the above definition makes no assumptions about the *internal structure* of a domain or scenario representation. This makes reasoning systems a rather general model of a theory of action. On the other hand, a number of restrictions, presented below, are placed on the constituents of a reasoning system in order for them to convey the intended meaning. First however, some additional notation.

For a scenario representation $s \in S$, by $[s]_\theta$ we shall denote the set of all scenarios encoded by s ; i.e. $[s]_\theta = \{\Sigma \in G : \langle \Sigma, s \rangle \in \theta\}$. Similarly, for a domain representation $d \in D$, by $[d]_\eta$ we shall denote the set of all domains encoded by d ; i.e. $[d]_\eta = \{\Delta \in U : \langle \Delta, d \rangle \in \eta\}$.

The first requirement, expressed by condition (R1) below, essentially says that the language of a reasoning system is expressive enough to distinguish between two different scenarios:

(R1) For all $s \in S$, $[s]_\theta$ is a singleton.

The remaining two requirements on a reasoning system, are both related to the *correctness* of the domain encoding scheme. Once again some additional terminology needs to be introduced before these requirements can be expressed formally.

Let Δ be a domain in U , and d, d' two domain representation in D . We shall say that d and d' are *equivalent* iff for all scenario representations $s, s' \in S$, if $[s]_\theta = [s']_\theta$ then $[C(d, s)]_\theta = [C(d', s')]_\theta$. We shall say that the domain Δ and the domain representation d are *in agreement* iff for all $s \in S$, $\Delta \cap [s]_\theta = [C(d, s)]_\theta$. The conditions (R2) and (R3) below ensure that domains are encoded correctly:¹¹

(R2) For all $d, d' \in D$, if d and d' are equivalent, then $[d]_\eta = [d']_\eta$.

(R3) For all $\Delta \in U$, and all $d \in D$, if $\Delta \in [d]_\eta$ and Δ is not in agreement with d , then Δ is not in agreement with any element of D .

Having built formal models for theories of action and ontological families, we can now define formally the notions of correctness and conciseness:

¹⁰ Although in situation calculus both scenario and domain representations consist of sentences of the same formal language, there is a fundamental difference between *observations* about the value of fluents at particular situations (scenario representations), and *background information* about the “physics” of a domain (domain representations). See [7] and [15] for more details.

¹¹ We note that conditions (R2) and (R3) are expressed slightly differently in this paper compared to their original formulation in [15]. This change does not effect the essence of the conditions, and it was made in order to increase readability in view of the forthcoming discussions (especially in relation to the proofs that follow).

Definition 5: A reasoning system $R = \langle D, S, \eta, \theta, C \rangle$ is correct with respect to a dynamic domain Δ iff for all encodings d of Δ , i.e. $\Delta \in [d]_\eta$, Δ and d are in agreement. R is correct with respect to an ontological family F iff R is correct with respect to every domain in F .

Turning to conciseness next, recall that it is defined in relative terms. Moreover, since the framework in [15] was developed with no reference to the internal structure of domain representation, comparative conciseness was defined not in terms of some syntactic criterion, but by means of the set of dynamic domains that each domain representation encodes (refer to [15] for a detailed discussion on the motivation behind this approach).

More precisely, for a domain representation d , consider the set $[d]_\eta$ of all domains encoded by d . One way to read $[d]_\eta$ is the following: given the information represented by d , any of the domains in $[d]_\eta$ is equally likely to be the one that d refers to. Hence the smaller $[d]_\eta$ is, the more information d conveys.

Consider now two domain representations d and d' in two reasoning systems $R = \langle D, S, \eta, \theta, C \rangle$ and $R' = \langle D', S', \eta', \theta', C' \rangle$ respectively. It is safe to say that d' is at least as *informative* as d , iff $[d']_{\eta'} \subseteq [d]_\eta$. Associating the information value of a domain representation with its degree of conciseness, we say that d is at least as concise as d' iff $[d']_{\eta'} \subseteq [d]_\eta$. Finally we say that the reasoning system R *at least as concise* as the reasoning system R' , denoted $R \blacktriangleright R'$, iff for every domain representation $d \in D$, the *corresponding* domain representation $d' \in D'$, is not more concise than d .

In order however for the above definition of comparative conciseness to acquire a precise meaning, one needs to define formally the notion of *correspondence* between domain representations in different reasoning systems. This is the purpose of the next set of definitions.

Let $R = \langle D, S, \eta, \theta, C \rangle$ and $R' = \langle D', S', \eta', \theta', C' \rangle$ be two reasoning systems. Extending our previous definition of *equivalence* between domains to apply across reasoning systems, we shall say that $d \in D$ and $d' \in D'$ are equivalent iff for any two scenario representations $s \in S$, and $s' \in S'$, if $[s]_\theta = [s']_{\theta'}$, then $[C(d, s)]_\theta = [C'(d', s')]_{\theta'}$. A *faithful* mapping between D and D' is defined as a function f from D to D' , such that for all $d \in D$, if d is not equivalent with $f(d)$, then d is not equivalent with any element of D' . The function f is meant to map a domain representation d in D , to its counterpart $f(d)$ in D' .

Finally, we say that an encoding scheme η'' is *compatible* with the reasoning system R iff replacing η with η'' in R does not violate (R2) and (R3).

Drawing together the above ideas and definitions, we formally define comparative conciseness as follow [15]:

Definition 6: For any two reasoning systems $R = \langle D, S, \eta, \theta, C \rangle$ and $R' = \langle D', S', \eta', \theta', C' \rangle$, we shall say that R is at least as concise as R' , denoted $R \blacktriangleright R'$, iff there exist an encoding scheme η'' compatible with R' , and a faithful mapping $f: D \rightarrow D'$ such that for all $d \in D$, $[f(d)]_{\eta''} \subseteq [d]_\eta$.

The last definition of this section concerns the *parsimony* of a reasoning system R with respect to an ontological family F .

Definition 7: A reasoning system R is *parsimonious* with respect to an ontological family F iff it satisfies the following two conditions:

(P1) R is correct with respect to F .

(P2) For all reasoning systems R' , if R' is correct with respect to F then $R \blacktriangleright R'$.

Essentially, R is parsimonious with respect to F , iff it is the most concise out of all reasoning systems that are correct with respect to F . Clearly, if R is parsimonious with respect to F , then it is safe to say that R (apart from correct) is *sufficiently concise* to qualify as a solution to the frame problem for F . Therefore parsimony is a *formal* criterion provided by the framework in [15], against which logics of action can be assessed.

3 REPACKAGING AR_0

As already mentioned, the framework introduced in [15] and presented in the previous section, was developed in rather general terms. While this generality secures a wide range of applicability for the framework's assessment methods and results, on the other hand it means that some effort is required in order to apply these methods to a particular logic of action; more specifically, the various aspects of the logic need to be recast in a way that fits the definitions of the framework. This exercise is performed in the present section for the logic of action developed in [7]. In doing so, we will be able not only to assess this logic in terms of conciseness (see section 4), but also to illustrate a methodology for obtaining conciseness results with the aid of the framework in [15].

The contribution that Kartha and Lifschitz made in [7] is twofold. Firstly, they introduced a high level language, called AR_0 , for describing dynamic domains, along with semantics that essentially assign sets of histories to these domain descriptions. Let us denote by F_A the ontological family consisting of all sets of histories corresponding to AR_0 domain descriptions. The second major contribution in [7], is the development of a logic of action, which we shall denote L_A , that is shown to be *correct* with respect to F_A . Our aim in this section is to define F_A and L_A in terms of the constructs and terminology in [15]. In particular, after reviewing the syntax of AR_0 (section 3.1), we shall repackage its semantics in terms of histories and give a formal definition of the ontological family F_A (section 3.2). Then we turn to L_A . In section 3.3 we convert the logic presented in [7] into a reasoning system. This will prepare the ground for section 4 where the parsimony of L_A is proved with respect to F_A , using the tools in [15].

3.1 The Syntax of AR_0

Let $V = \{v_1, v_2, \dots, v_n\}$ be a nonempty finite set of symbols called *fluents*, and $E = \{e_1, e_2, \dots, e_m\}$, a nonempty finite set of symbols called *actions*. A certain subset Π of V is chosen and is identified as the *frame*; the elements of Π are called *frame fluents*. There are four types of *propositions* in AR_0 , called *value propositions*, *effect propositions*, *release propositions*, and *constraints*, and which respectively are expressions of the form shown below:

- (S1) **C after A** (*value proposition*).
- (S2) **A causes C if P** (*effect proposition*).
- (S3) **A releases F if P** (*release proposition*).
- (S4) **always C** (*constraint*)

In the above expressions, C and P are propositional combinations of fluents, A is an action in E , and $\mathbf{A}$ a string of actions.

The intuitive meaning of (S1) is that C is true after the sequence of actions A is performed at the initial state, the meaning of (S2) is that C is true whenever the action A is applied to a state that satisfies P , the meaning of (S3) is that whenever the action A is applied to a state that satisfies P , then the fluent F is exempt from complying with the law of inertia; in other words, the truth value of F after A is not determined by the value of F at the original state.¹² Finally, the intuitive meaning of (S4) is that C should be true at all states.

Given two value propositions $p_1 = (C_1 \text{ after } A_1)$, and $p_2 = (C_2 \text{ after } A_2)$, we shall say that p_1 and p_2 are *compatible* iff A_1 is a substring of A_2 or vice versa. A *scenario description* Y is defined as a nonempty finite set of pairwise compatible value propositions.

A *domain description* Q on the other hand is defined as a finite set of effect propositions, release propositions, and constraints (expressions of the form (S2) - (S4)).

We note that in [7], a domain description also includes value propositions. However, since value propositions essentially correspond to observations, we have separated them from the rest in order to comply with the methodology of our framework. This splitting of propositions in two sets, is part of the "repackaging" of AR_0 mentioned above, and does not affect the essence of the language.

A more important divergence from AR_0 is our requirement for compatibility between value propositions. Kartha and Lifschitz in [7], allow a scenario description to refer to *more than one* possible developments (or courses of action) of a dynamic domain. This option is not present under our compatibility requirement: all value propositions within a scenario description have to refer to the *same* history. The main reason for adopting this restriction herein is because it simplifies considerably the notation in the forthcoming discussion, thus allowing us to demonstrate more effectively the ideas and results of the paper – generality is sacrificed for the sake of clarity. It should be noted however that, with the appropriate adjustments, the compatibility requirement can be lifted without affecting the conciseness results of section 4.

3.2 Semantics of AR_0 (Repackaged)

Formulating semantics for AR_0 essentially amounts to defining for each domain description, the set of histories that correspond to it (i.e., all possible developments in time of the domain at hand). In [7], this set of histories was defined in terms of partial functions from strings of actions to states. Herein we shall recast this definition (without however changing the essence of the semantics) in a way that fits the terminology of the framework in [15].

We start with the notion of a *state* defined as a subset of V . Essentially a state w assigns truth values to fluents via the convention that fluents in w take the truth value "true" while all the rest take the value "false". For an arbitrary propositional combination of fluents C , the truth value of C at w is defined in the obvious way. We shall write $w \models C$ to denote the fact that C is true at w . We shall denote the set of all states by W .

A history in this context is essentially a sequence of alternating states and actions. For technical reasons, we shall assume the existence of a "dummy" action e^* (not already existing in E). Consider now an infinite sequence $h = \langle w_1, a_1 \rangle; \langle w_2, a_2 \rangle; \dots; \langle w_j, a_j \rangle; \dots$ of ordered pairs $\langle w_j, a_j \rangle$, where $w_i \in W$ is a state and a_j is either an action in E or the "dummy" action e^* . For any number $j \geq 1$, let us denote by $h_w(j)$ the state of the j -th pair in the sequence h , and by $h_a(j)$ the action of the same pair. We shall say that h is a *history* iff the following two conditions are satisfied:

¹² Release propositions were introduced in AR_0 to deal with non-determinism. See [7] for more details.

(H1) There exists a $k \geq 1$ such that $h_e(k) = e^*$.

(H2) For all $i \geq 1$, if $h_e(i) = e^*$ then for all $j \geq i$, $h_e(j) = e^*$, and $h_w(j) = h_w(i)$.

Essentially, these two conditions ensure that all histories are such that from some point onwards there is no activity – i.e. no change of state and no non-dummy action.

Let us denote by H_A the set of all such histories. We define the set of all domains U_A as the powerset of H_A ; in symbols $U_A = 2^{H_A}$.

Consider now a domain description Q . Kartha and Lifschitz [7] define two functions related to Q , which are denoted by $\text{Res}_0(A, w)$ and $\text{Res}(A, w)$ respectively. Both these functions map a state and an action to a set of states. In particular, for an action $A \in E$ and a state w , $\text{Res}_0(A, w)$ is defined as the set of all states w' such that for each effect proposition (A **causes** C if P) in Q , $w' \vdash C$ whenever $w \vdash P$. Based on $\text{Res}_0(A, w)$ the function $\text{Res}(A, w)$ is defined as the subset of $\text{Res}_0(A, w)$ containing the states that differ *minimally* from w .

More precisely, let $\text{Diff}(w, w')$ denote the symmetric difference of two states w and w' ; i.e., $\text{Diff}(w, w') = (w - w') \cup (w' - w)$. Moreover, for an action A and a state w let $I(A, w)$ be the set of frame fluents F such that for every release proposition (A **releases** F if P) in Q , it is not the case that $w \vdash P$. Intuitively, $I(A, w)$ contains all the fluents that are subjected to inertia during the application of A at w . The result function Res , mapping an action A and a state w to the set of states $\text{Res}(A, w)$, is defined as follows: $w' \in \text{Res}(A, w)$ iff $w' \in \text{Res}_0(A, w)$ and there is no w'' in $\text{Res}_0(A, w)$ such that $\text{Diff}(w, w'') \cap I(A, w) \subset \text{Diff}(w, w') \cap I(A, w)$.

For the special case of the dummy action e^* , we define $\text{Res}_0(e^*, w) = \text{Res}(e^*, w) = \{w\}$.

A state $w \in W$ is said to be *valid* iff it satisfies all constraint propositions in Q ; i.e., for each constraint (**always** C) in Q , $w \vdash C$. We shall say that a history $h \in H$ is *consistent* with Q , iff the following conditions are satisfied:

(M1) For all $j \geq 1$, $h_w(j)$ is valid.

(M2) For all $j \geq 1$, $h_w(j+1) \in \text{Res}(h_e(j), h_w(j))$.

We shall denote the set of all histories that are consistent with a domain description Q by $M(Q)$. We shall call such a set of histories an *AR₀ domain*, and we define F_A to be the ontological family consisting of all such domains; in symbols, $F_A = \{M(Q) : Q \text{ is a domain description in } AR_0\}$.

Next we turn to scenario descriptions and their corresponding histories. For a sequence of actions A , let $l(A)$ denote the length of the sequence, and let $A(k)$, for $1 \leq k \leq l(A)$, denote the k -th element of the sequence. We shall say that a history $h \in H_A$ is *consistent* with the scenario description Y iff the following conditions are satisfied:

(SC1) For every value proposition (C **after** A) in Y , and for all $1 \leq k \leq l(A)$, $h_e(k) = A(k)$.

(SC2) For every value proposition (C **after** A) in Y , $h_w(l(A)+1) \vdash C$.

(SC3) If $h_e(k) \neq e^*$ then for some (C **after** A) in Y , $k \leq l(A)$.

Condition (SC1) says that the actions that occur in the history h are precisely those mentioned in the scenario description Y , condition (SC2) says that the propositions that hold at various time points in h agree with what is stated in Y , and finally (SC3) says that there is no activity in h after the last action mentioned in Y .¹³

We shall denote by $M(Y)$ the set of all histories that are consistent with a scenario description Y . We shall call such a set of histories an AR_0 scenario, and we define G_A to be the set of all such scenarios; i.e. $G_A = \{M(Y) : Y \text{ is a scenario description in } AR_0\}$.

Recall from section 2, that there are two conditions that G_A ought to satisfy; firstly, that it can differentiate between any two domains, and secondly, that the intersection of a scenario and a domain is always a scenario. The following results show that both these conditions are met:

Lemma 1: *Let Δ, Δ' be any two domains in U_A . If $\Delta \neq \Delta'$ then there is a scenario $\Sigma \in G_A$ such that $\Delta \cap \Sigma \neq \Delta' \cap \Sigma$.*

Proof. Let $\Delta, \Delta' \in U_A$ be such that $\Delta \neq \Delta'$. Without loss of generality, we can then assume that there is a history $h \in H_A$ such that $h \in \Delta$ and $h \notin \Delta'$. From (H1) and (H2) we derive that there is a time-point $k \geq 1$ after which no activity exists in h . Let C_i be the conjunction of all literals¹⁴ that hold at $h_w(i)$ for $1 \leq i \leq k$. Moreover, let A_i be the sequence of actions that have occurred in h up to the time-point i ; i.e. $A_i = h_e(1); h_e(2); \dots h_e(i-1)$ (clearly, A_1 is the empty string, denoted by $\emptyset$). Finally, for $1 \leq i \leq k$, let p_i be the value proposition $p_i = (C_i \text{ after } A_i)$, and let Y be the scenario description consisting of all these propositions; i.e. $Y = \{p_i : 1 \leq i \leq k\}$. It is not hard to verify that h belongs to the scenario $M(Y)$. Consequently, $\Delta \cap M(Y) \neq \Delta' \cap M(Y)$ as desired. $\square$

Lemma 2: *For any domain $\Delta \in U_A$ and scenario $\Sigma \in G_A$, $\Delta \cap \Sigma \in G_A$.*

Proof. Let Δ be an arbitrary domain in U_A , and Σ any scenario in G_A . If $\Delta \cap \Sigma = \emptyset$, then clearly $\Delta \cap \Sigma$ is a member of G_A . Assume therefore that $\Delta \cap \Sigma \neq \emptyset$, and let Y be a (nonempty) AR_0 scenario description such that $\Sigma = M(Y)$. Since Y contains only finitely many elements, there is a value proposition $(C_i \text{ after } A_i)$ in Y such that the length of A_i is greater or equal to the length of any other action string that appears in Y . Let us denote by k the length of A_i ; i.e. $k = l(A_i)$. Then, by (SC3), for any history h in Σ , there is no activity after time-point k ; in symbols, for all $j > k$, $h_e(j) = e^*$ and $h_w(j) = h_w(k+1)$. This again entails that there are only *finitely many* histories in Σ , since any two histories $h, h' \in \Sigma$ can only differ up to time-point $k+1$. Consequently, there are only finitely many histories in $\Delta \cap \Sigma$. We can then enumerate $\Delta \cap \Sigma$ and write it as the set of histories $\Delta \cap \Sigma = \{h^1, \dots, h^j\}$. Let r_j^i be the conjunction of all literals that hold in $h_w^i(j)$, and let C_j be the sentence $C_j = r_j^1 \vee r_j^2 \vee \dots \vee r_j^j$. Moreover, let A_j be the action string consisting of the $j-1$ first actions of the history h^1 ; i.e. $A_j = h_e^1(1); h_e^1(2); \dots; h_e^1(j-1)$; for $j=1$ we set A_j to the empty string. Notice that, because of (SC1), (SC3), all histories in $\Delta \cap \Sigma$ agree on the actions that have occurred (i.e. $h_e^x(j) = h_e^y(j)$, for all $j \geq 1$, and $1 \leq x, y \leq j$), and therefore the action string A_j appears in all histories in $\Delta \cap \Sigma$. Finally, let p_j be the value proposition $p_j = (C_j \text{ after } A_j)$, and let Y be the scenario description $Y = \{p_j : 1 \leq j \leq k+1\}$. It is not hard to verify that $M(Y) = \Delta \cap \Sigma$, and therefore $\Delta \cap \Sigma \in G_A$ as desired. $\square$

¹³ Condition (SC3) is not required by Kartha and Lifschitz in [7]. Once again however, it is adopted for the sake of compliance with the requirements in [15] (in particular, it is necessary for proving Lemma 2 below). Nevertheless, (SC3) does not change the essence of Kartha and Lifschitz's approach.

¹⁴ A *literal* is either a fluent or the negation of a fluent.

3.3 The Logic L_A (Repackaged)

The language AR_0 and its semantics were developed primarily with the aim of *specifying* domains of a certain type.¹⁵ Reasoning about such domains however is a different story; a logic of action is required whose inferences are correct with respect to the semantics of the previous section.

Such a logic of action was indeed introduced in [7]. We shall call this logic L_A . The logic L_A is based on *nested circumscription* [9], and was developed specifically to deal with AR_0 domains. Our aim in this section is to re-construct L_A in terms of the concepts and terminology of the framework in [15].

The language of L_A is a multi-sorted first order language with the following sorts: a sort for fluents with variables $f, f_1, f_2, \dots$, a sort for actions with variables $a, a_1, a_2, \dots$, and a sort for situations, with variables $s, s_1, s_2, \dots$. In addition, the language of L_A has predicate variables $\lambda, \lambda_1, \lambda_2, \dots$, that range over properties of fluents.

In terms of constants, the language of L_A contains the fluents in V , the actions in E , and the situation constants S_0 and $\perp$. Following the standard situation calculus notation, L_A also contains the function constant *Result* which maps an action and a situation to a situation, and the predicate *Holds* whose two arguments are a fluent and a situation. Finally, L_A contains the predicate constant *FrameFluent* whose single argument is a fluent.

The well formed formulas (*wff*) in L_A are defined in the usual way. To increase readability, we shall adopt the notational conventions in [7]. In particular, for a situation s and a propositional combination C of fluent variables or fluent constants, by $T(C, s)$ we denote the wff resulting from C when replacing each fluent F (variable or constant) with the term $\text{Holds}(F, s)$. For a sequence of actions $A = a_1; a_2; \dots; a_k$, by $[A]$ we denote the term $\text{Result}(a_k, \text{Result}(a_{k-1}, \dots, \text{Result}(a_1, S_0), \dots))$.

Among the *wff* of L_A we identify the formulas listed below that play a special role in this logic and which are essentially the counterparts in L_A , of value, effect, release propositions, and constraints:

$$(F1) \quad [A] \neq \perp \wedge T(C, [A]).$$

$$(F2) \quad \forall s. (T(P, s) \wedge \text{Result}(A, s) \neq \perp \rightarrow T(C, \text{Result}(A, s))).$$

$$(F3) \quad \forall s. (T(P, s) \rightarrow \text{Ab}(F, A, s)).$$

$$(F4) \quad \forall s. T(C, s)$$

In the above formulas, A is a sequence of action constants, P and C are propositional combinations of fluents, A is an action constant, and F is a fluent constant.

A *domain representation* d in L_A is a finite set of sentences of the form (F2), (F3), and (F4). We shall denote the set of all such domain representations by D_A .

A *scenario representation* s in L_A is a nonempty finite set of sentences of the form (F1), such that for some sentence $([A] \neq \perp \wedge T(C, A))$ in s , the action string A subsumes all other action strings that appear in s (in other

¹⁵ Namely, domains that comply with the commonsense law of *inertia* while at the same time permitting *non-determinism* and actions with *indirect effects* - refer to [7] for more details.

words, all other action strings in s are substrings of A). We shall call A the *chronicle* of s and denote it by $\text{Ch}(s)$. We shall denote the set of all scenario representations by S_A .

There is clearly a straightforward mapping β from domain representations to domain descriptions. In particular, consider an arbitrary domain representation $d \in D$ and let d_{F2} , d_{F3} , and d_{F4} , be the subsets of d containing the formulas of the form (F2), (F3), and (F4) respectively. For each formula $E = \forall s. (T(P,s) \wedge \text{Result}(A,s) \neq \perp \rightarrow T(C, \text{Result}(A,s)))$ in d_{F2} , we define $\beta(E)$ to be the effect proposition $\beta(E) = (A \text{ causes } C \text{ if } P)$. For each formula $R = \forall s. (T(P,s) \rightarrow \text{Ab}(F,A,s))$ in d_{F3} , we define $\beta(R)$ to be the release proposition $\beta(R) = (A \text{ releases } F \text{ if } P)$. Finally, for each formula $Q = \forall s. T(C,s)$ in d_{F4} , we define $\beta(Q)$ to be the constraint, $\beta(Q) = (\text{always } C)$. Based on the above we define the domain description $\beta(d)$ corresponding to d to be the set of propositions $\beta(d) = \{ \beta(E): E \in d_{F2} \} \cup \{ \beta(R): R \in d_{F3} \} \cup \{ \beta(Q): Q \in d_{F4} \}$.

Similarly, there is an obvious mapping γ from scenario representations to scenario descriptions. In particular, let s be an arbitrary scenario representation, and let $V = ([A] \neq \perp \wedge T(C, [A]))$ be a formula in s . We define $\gamma(V)$ to be the value proposition, $\gamma(V) = (C \text{ after } A)$. The scenario description $\gamma(s)$ corresponding to s is then defined as $\gamma(s) = \{ \gamma(V): V \in s \}$.

With the aid of β and γ , we can now define the domain encoding scheme η_A and the scenario encodings scheme θ_A for L_A . In particular, for a domain $\Delta \in U_A$ and a domain representation $d \in D_A$, $\Delta \in [d]_{\eta_A}$ iff either $\Delta = M(\beta(d))$, or $\Delta \notin F_A$ and $M(\beta(d)) = \emptyset$. Similarly, for a scenario $\Sigma \in G_A$ and a scenario representation $s \in S_A$, $\Sigma \in [s]_{\theta_A}$ iff $\Sigma = M(\gamma(s))$. It is not hard to verify that every domain $\Delta \in U_A$ has an encoding $d \in D_A$ under η_A (i.e. $\Delta \in [d]_{\eta_A}$), and every scenario $\Sigma \in G_A$ has an encoding $s \in S_A$ under θ_A (i.e. $\Sigma \in [s]_{\theta_A}$).

There is a final component that needs to be defined in order for L_A to be fully converted into a reasoning system as described in the previous section: the reasoning engine.

As already mentioned, the language of L_A is essentially that of a multi-sorted first-order logic. The inferences however are drawn based on *nested circumscription*. More precisely, consider the following groups of formulas:

- (G1) $v_i \neq v_j$ for all $v_i, v_j \in V$ such that $i \neq j$.
- (G2) $e_i \neq e_j$ for all $e_i, e_j \in E$ such that $i \neq j$.
- (G3) $S_0 \neq \perp$.
- (G4) $\forall f. (f = v_1 \vee f = v_2, \dots \vee f = v_n)$.
- (G5) $\forall f. (\text{FrameFluent}(f) \leftrightarrow \bigvee_{v_i \in \Pi} f = v_i)$.
- (G6) $\forall a. (a = e_1 \vee a = e_2, \dots \vee a = e_m)$.
- (G7) $\exists s_1, \dots, s_k. (\bigwedge_{1 \leq i \leq j \leq k} s_i \neq s_j)$, where $k = 2^n$.
- (G8) $\forall s, f, a. (\text{FrameFluent}(f) \wedge \neg \text{Ab}(f, a, s) \rightarrow \text{Holds}(f, \text{Result}(a, s)) \leftrightarrow \text{Holds}(f, s))$.
- (G9) $\forall s, f, a. (\neg \text{Ab}(\lambda) \rightarrow \exists s. (s \neq \perp \wedge \forall f. (\text{Holds}(f, s) \leftrightarrow \lambda(f))))$
- (G10) $\forall a, s. (\text{Result}(a, s) = \perp \rightarrow \text{Ab}(a, s))$.

$$(G11) \quad \forall a. (\text{Result}(a, \perp) = \perp.$$

The groups (G1) - (G3) are *unique name* axioms, groups (G4) - (G6) are *domain closure* axioms, group (G7) states that the universe of situations is sufficiently large, (G8) essentially expresses the *commonsense law of inertia*, (G9) is the *existence of situations* axiom, (G10) says that unless an action a is abnormal, it is applicable at a state s , and (G11) states that the application of any action a at the state $\perp$ results again in the state $\perp$.

Consider next a domain representation d , and let d_{F2} , d_{F3} and d_{F4} be the sentences in d of type (F2), (F3), and (F4) respectively (consequently, $d = d_{F2} \cup d_{F3} \cup d_{F4}$). Let $\text{NC}(d)$ be the theory resulting from the following nested circumscription operator:

$$\{ \text{Result: (G8), } d_{F3}, \{ \text{Result: (G10), (G11), } d_{F2}, \{ \text{Holds: (G9), } d_{F4} \} \} \}$$

We are now one step away from defining the scenario closure operator C_A for L_A . In particular, let d and s be a domain and scenario representation respectively. We define $K(d, s)$ to be the theory:

$$K(d, s) = \text{Cn}(\text{NC}(d) \cup s \cup \{ (G1), \dots, (G7) \})$$

Next consider a substring A of the chronicle $\text{Ch}(s)$ of s . By $[[A, s, d]]$ we denote the set of states, $[[A, s, d]] = \{w \in W : w \vdash \varphi \text{ for all } \varphi \text{ such that } K(d, s) \vdash T(\varphi, [A])\}$. In other words, $[[A, s, d]]$ contains all states that are consistent with the sentences which, according to $K(d, s)$, hold after the action string A .

We define the closure of s under d , denoted $C_A(d, s)$ to be the following scenario representation:

$$C_A(d, s) = \{ ([A] \neq \perp) \wedge T(C, A) : A \text{ is a substring of } \text{Ch}(s), \text{ and } C = \bigvee_{w \in [[A, s, d]]} (\wedge w) \}$$

In the above definition $(\wedge w)$ denotes the conjunction of all literals that hold at a state w . Therefore $\bigvee_{w \in [[A, s, d]]} (\wedge w)$ is a sentence in *disjunctive normal form* essentially representing all that is known to be true – according to the theory $K(d, s)$ – after the action string A is executed.

We finally assemble the various components described above into a reasoning system R_A , which we define to be the tuple $R_A = \langle D_A, S_A, \eta_A, \theta_A, C_A \rangle$.

4 RESULTS ON CONCISENESS

Having transformed the logic L_A introduced in [7], into a reasoning system R_A , it is now possible to use the formal tools of the framework in [15] to make assessments about the *conciseness* of L_A .

First however we need to verify that R_A does indeed satisfy the requirements for a reasoning system as described in section 2.2.

Lemma 3: *The tuple $R_A = \langle D_A, S_A, \eta_A, \theta_A, C_A \rangle$ is a reasoning system; i.e. it satisfies the conditions (R1) - (R3).*

Proof. Condition (R1), follows trivially from the definition of θ_A . For (R2), let $d, d' \in D$ be any two equivalent domain representations. According to the definition of η_A , in order to show that $[d]_{\eta_A} = [d']_{\eta_A}$, it suffices to show that $M(\beta(d)) = M(\beta(d'))$. Assume on the contrary that $M(\beta(d)) \neq M(\beta(d'))$. Without loss of generality we can then assume that for some history $h \in H_A$, $h \in M(\beta(d))$ and $h \notin M(\beta(d'))$. Let $k \geq 1$ be the smallest number such that $h_e(k) = e^*$ (by (H1) such a k indeed exists). Let C_i be the conjunction of literals that hold at the state $h_w(i)$ and let $\mathbf{A}_i$ be the sequence of actions that have occurred at h up to time-point i ; i.e., $\mathbf{A}_i = h_e(1); h_e(2); \dots; h_e(i-1)$ (we take $\mathbf{A}_1$ to be the empty string). Define s to be the scenario representation, $s = \{ ([\mathbf{A}_i] \neq \perp) \wedge T(C_i, \mathbf{A}_i) : 1 \leq i \leq k \}$. It is not hard to verify that $h \in [s]_{\theta_A}$. Then, from the definition of C_A it follows that $h \in [C_A(d, s)]_{\theta_A}$ and $h \notin [C_A(d', s)]_{\theta_A}$. This however contradicts the equivalence of d and d' .

Finally for (R3), let $\Delta \in U_A$ be a domain and $d \in D_A$ be a domain representation such that $\Delta \in [d]_{\eta_A}$ and Δ is not in agreement with d . From the definition of η this entails that $\Delta \notin F_A$, which again entails that Δ is not in agreement with any domain representation in D_A as desired. $\square$

Having established that R_A is indeed a reasoning system, we next turn to its properties in relation to the ontological family F_A . It has already been shown in [7] that R_A is *correct* with respect to F_A . In this paper we prove that R_A is also *parsimonious* with respect to F_A .

Recall from section 2 that a reasoning system is parsimonious with respect to an ontological family if it is the *most concise* reasoning system among all those that are correct with respect to the particular ontological family. Clearly, having to compare R_A with all other reasoning systems that happen to be correct with respect to F_A , is practically impossible. Fortunately though there is a much better way to determine the parsimony of a reasoning system. In particular, let us call a reasoning system R *maximally correct* with respect to an ontological family F iff R is correct with respect to F and moreover, there is no proper superset of F with respect to which R is correct. The following result was proved in [15]:

Theorem 1: *A reasoning system R is parsimonious with respect to an ontological family F iff R is maximally correct with respect to F .*

Based on the above characterization of parsimony, we can now prove the central result of this article:

Theorem 2: *The reasoning system R_A is parsimonious with respect to the ontological family F_A .*

Proof. The correctness of R_A with respect to F_A has already been established in [7]. Consequently, given Theorem 1, all that is left to show is that there is no proper superset of F_A with respect to which R_A is correct. Assume on the contrary that such a superset, call it $F^\#$, exists, and let Δ be a domain in $F^\# - F_A$. Let d be an encoding of Δ in R_A , i.e. $\Delta \in [d]_{\eta_A}$. Let Q be the AR_0 domain description $Q = \beta(d)$, and let $\Delta' \in F_A$ be the domain $\Delta' = M(Q)$. If $M(\beta(d)) \neq \emptyset$, then from the definition of η_A it follows that $[d]_{\eta_A} = \{\Delta'\}$, which again entails that $\Delta' = \Delta$, contradicting our assumption that $\Delta \notin F_A$. Assume therefore that $M(\beta(d)) = \emptyset$. Then it is not hard to verify that for all $s \in S_A$, $[C_A(d, s)]_{\theta_A} = \emptyset$. Given that d is assumed to be correct with respect to Δ , this again entails that $\Delta = \emptyset$, which once again contradicts our assumption that $\Delta \notin F_A$. Consequently, R_A is maximally correct with respect to F_A , and therefore, in view of Theorem 1, R_A is parsimonious wrt F_A as desired. $\square$

5 CONCLUSION

In a previous article [15] we developed a general formal framework for assessing theories of action in terms of *both* correctness and conciseness. Herein we have applied the main ideas and results of that framework to

evaluate the logic of action proposed by Kartha and Lifschitz in [7]. In one of the main theorems of this paper we have shown that Kartha and Lifschitz's logic is not only correct with respect to AR_0 domains, but it also *sufficiently concise* (alias, *parsimonious*) to qualify as a solution to the frame problem (for AR_0).

We note that in assessing Kartha and Lifschitz's theory of action, most of the effort went into repackaging the theory to fit the models of the framework in [15] (sections 3.1, 3.2, 3.3); once the repackaging was completed, the tools of the framework (in particular, Theorem 1), reduced considerably the complexity of the assessment. This is no accident. While the framework in [15] offers some powerful assessment tools, it is on the other hand developed at a high level of abstraction that makes its usage non-trivial. In that sense, the re-packaging of Kartha and Lifschitz's theory of action undertaken herein, is in itself an important contribution of this article, since it illustrates a methodology for obtaining conciseness results with the aid of the (admittedly abstract) tools in [15].

As already mentioned, Theorem 2 is, to our knowledge, the first *formal* assessment of a theory of action in terms of conciseness. Much more of course is left to be done. A first obvious direction for future work is to consider ways of simplifying the framework in [15] (perhaps at the expense of generality) in order to make it more accessible. A second very interesting avenue for future research is to bring *computational complexity* in the picture; that is, to consider the relationship between the conciseness of a theory of action, and the computational complexity of its reasoning engine. Preliminary studies indicate that the two are closely related.

6 ACKNOWLEDGMENTS

We are grateful to Norman Foo, John Slaney, Sylvie Thiebaux, and the anonymous referees for many useful comments on this article. The third author is also grateful to the School of Computer Science and Engineering at the University of New South Wales for funding his visits to Sydney and making possible the above mentioned interactions.

REFERENCES

- [1] A. Baker "A simple solution to the Yale shooting problem", *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pp 11-20, 1989.
- [2] M. Cadoli, "Space Efficiency of Propositional Knowledge Representation Formalism", *Journal of Artificial Intelligence Research*, vol 13, pp 1-31, 2000.
- [3] R. Greiner and C. Elkan, "Measuring and Improving the Effectiveness of Representations", *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1991.
- [4] S. Hanks and D. McDermott, "Default reasoning, non-monotonic logics, and the frame problem", *Proceedings of the National (U.S.A.) Conference on Artificial Intelligence*, pp. 328-333, 1986.
- [5] A. Hass, "The case for domain-specific frame axioms", *Proceedings of the 1987 Workshop on the Frame Problem*, 1987.

- [6] G. N. Kartha, "Two counterexamples related to Baker's approach to the frame problem", *Artificial Intelligence*, pp. 379-391, vol.69, 1994.
- [7] G. N. Kartha and V. Lifschitz, "Actions with indirect effects (preliminary report)", *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 341-350, Morgan Kaufmann, 1994.
- [8] P. Liberatore, "Compilability and Compact Representations of Revision of Horn Knowledge Bases", *ACM Transactions on Computational Logic*, vol. 1(1), pp 131-161, 2000.
- [9] V. Lifschitz, "Nested abnormality theories", *Manuscript*, 1994.
- [10] F. Lin and Y. Shoham, "Provably correct theories of action (preliminary report)", *Proceedings of the National (U.S.A.) Conference on Artificial Intelligence*, pp. 349-354, 1991.
- [11] N. McCain and H. Turner, "A causal theory of ramifications and qualifications", *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pp. 1978-1984, Morgan Kaufmann, 1995.
- [12] J. McCarthy, "Applications of circumscription to formalizing common sense knowledge", *Artificial Intelligence*, vol.28:89-116, 1986.
- [13] J. McCarthy and P. Hayes, "Some philosophical problems from the standpoint of Artificial Intelligence", *Machine Intelligence*, vol.4:463-502, 1969.
- [14] P. Peppas and W. Wobcke, "On the use of epistemic entrenchment in reasoning about action", *Proceedings of the Tenth European Conference on Artificial Intelligence*, pages 403-407, John Wiley and Sons, 1992
- [15] P. Peppas, C. Koutras and M.-A. Williams, "Prolegomena to Concise Theories of Action", *Studia Logica*, Kluwer Academic Publishers, 2001.
- [16] E. Sandewall, *Features and Fluents*, Oxford University Press, 1994.
- [17] M. Shanahan, *Solving the Frame Problem*, MIT Press, 1997.
- [18] M. Winslett, "The Possible Models Approach", *Proceedings of the National (U.S.A.) Conference on Artificial Intelligence (AAAI'88)*, 1988.