Proceedings of the Fourteenth International Conference on Principles of Knowledge Representation and Reasoning

Ontology-Based Monitoring of Dynamic Systems*

Franz Baader
Theoretical Computer Science, TU Dresden
Nöthnitzer Str. 46, 01062 Dresden, Germany
baader@tcs.inf.tu-dresden.de

Abstract

Our understanding of the notion "dynamic system" is a rather broad one: such a system has states, which can change over time. Ontologies are used to describe the states of the system, possibly in an incomplete way. Monitoring is then concerned with deciding whether some run of the system or all of its runs satisfy a certain property, which can be expressed by a formula of an appropriate temporal logic. We consider different instances of this broad framework, which can roughly be classified into two cases. In one instance, the system is assumed to be a black box, whose inner working is not known, but whose states can be (partially) observed during a run of the system. In the second instance, one has (partial) knowledge about the inner working of the system, which provides information on which runs of the system are possible.

In this paper, we will review some of our recent work that can be seen as instances of this general framework of ontology-based monitoring of dynamic systems. We will also mention possible extensions towards probabilistic reasoning and the integration of mathematical modeling of dynamical systems.

Introduction

In this paper we use the term "dynamic system" to denote a system that shows dynamic behavior in that it changes its states over time. Here "system" is seen in a broad sense, encompassing both man-made systems and natural systems: it may be a computer system, air traffic observed by radar, a patient in an intensive care unit, or a biological cell. We make no general assumptions about the form of the system's states, except that the states can be described using an appropriate ontology language. These descriptions may be incomplete (partial) in the sense that they do not uniquely determine a single state, but only restrict the possible states to a subset of all states.

In the case of a *black box* system, we have no information on how the system works internally, i.e., we do not know which state is transformed into which other state. In this setting, we assume that the system is observed by certain "sensors" (e.g., heart-rate and blood pressure monitors for a patient), and the results of sensing are stored in a fact base expressed using the given ontology language. Based on the information stored in the fact base, the monitor is supposed to detect certain predefined situations (e.g., heart-rate very high and blood pressure low), which require a reaction (e.g., fetch a doctor or give medication). More precisely, sensor readings are available for different points in time, and thus we obtain a time-stamped sequence of fact bases, each describing (possibly in an incomplete way) the state of the system at the respective time point. The situations to be described may be concerned with not just one state of the system, but a sequence of states (e.g., blood pressure decreasing and heart-rate increasing for a certain time). To describe such situations, the ontology language needs to be combined with an appropriate temporal logic.

In the case of a *white box* system, we have some knowledge about the inner working of the system, i.e., we have a specification of how states of the system are transformed into each other. This description may, however, be nondeterministic in the sense that a given state may not have a uniquely determined successor state, but a set of possible successor states. Given a (possibly incomplete) description of an initial state, the specification then determines a set of possible runs of the system, and we may ask whether some or all of these runs satisfy a certain property, formulated in an appropriate temporal logic.

Of course, we may also have a *combination* of both settings, where a (partial) specification of the system is available, and in addition one can observe the system during one of its runs.

In the following, we will review some of our recent work that can be seen as instances of the general framework of ontology-based monitoring of dynamic systems outlined above. Because of space limitations, we cannot introduce the technical definitions and results in detail. For these and also for detailed descriptions of related work we refer the reader to the cited papers. The ontology languages used in these papers are based on description logics. Again, we do not introduce them in detail, but refer the reader to "The Description Logic Handbook" (Baader et al. 2003) for a comprehensive introduction. We will finish this short paper with mentioning possible extensions of the described approaches towards probabilistic reasoning and the integration of mathematical modeling of dynamical system.

*Partially supported by the Cluster of Excellence "Center for Advancing Electronics Dresden" and the Collaborative Research Center 912 "Highly Adaptive Energy-Efficient Computing."

Monitoring Black Box Systems

As mentioned in the introduction, an important ingredient of such a monitoring approach is an appropriate temporal logic, which enables the specification of situations whose definition depends on several of the observed states of the system.

The temporal logic $\mathcal{ALC}$-LTL

Motivated by a situation awareness application (Baader et al. 2009), in which the system was air and marine traffic observed by radar, we have introduced the temporalized description logic $\mathcal{ALC}$-LTL (Baader, Ghilardi, and Lutz 2008; 2012) as such a logic. It combines the ontology language $\mathcal{ALC}$ with the propositional linear temporal logic LTL. Since the temporal structure underlying LTL are the natural numbers, this means that we consider a discrete flow of time. In contrast to propositional LTL, the states are not propositional valuations, but $\mathcal{ALC}$ interpretations, i.e., relational structures with unary predicates (concepts) and binary predicates (roles).

An important design decision for $\mathcal{ALC}$-LTL was, on the one hand, to restrict the application of temporal operators to $\mathcal{ALC}$ axioms, i.e., terminological axioms of the form $C \sqsubseteq D$ and assertional axioms of the form $C(a)$ and $r(a, b)$, where C, D are concepts, r a role, and a, b individuals. From a syntactic point of view, $\mathcal{ALC}$-LTL is obtained from propositional LTL by allowing the use of $\mathcal{ALC}$ axioms in place of propositional variables. For example, the $\mathcal{ALC}$-LTL formula

$$\Diamond\Box(\mathsf{UScitizen} \sqsubseteq \exists \mathsf{insured\_by}.\mathsf{HealthInsurer})$$

says that there is a future time point from which on US citizens will always have health insurance, and the formula

$$\Diamond\big((\exists \mathsf{finding}.\mathsf{Concussion})(\mathsf{BOB}) \wedge \mathsf{Conscious}(\mathsf{BOB})\,\mathsf{U}\,(\exists \mathsf{procedure}.\mathsf{Examination})(\mathsf{BOB})\big)$$

says that, sometime in the future, Bob will have a concussion with no loss of consciousness between the concussion and the examination. However, what cannot be expressed in $\mathcal{ALC}$-LTL is the general concept of a concussion with no loss of consciousness since expressing this concept would require the application of temporal operators within the concept:

$$\exists \mathsf{finding}.\mathsf{Concussion} \sqcap \mathsf{Conscious}\,\mathsf{U}\,\exists \mathsf{procedure}.\mathsf{Examination}.$$

On the other hand, the second important design decision was to allow for rigid concepts and roles, i.e., concepts/roles whose interpretation does not vary over time. For example, the concept Human and the role has_father should probably be rigid since a human being will stay a human being and have the same father over his/her lifetime, whereas Conscious should be a flexible concept (i.e., not rigid) since someone that is conscious at the moment need not always by conscious. Similarly, insured_by should be modeled as a flexible role. Using a logic that cannot enforce rigidity of concepts/roles may result in unintended models, and thus prevent certain useful inferences to be drawn. For example, the concept description $\exists \mathsf{has\_father}.\mathsf{Human} \sqcap \Diamond\,(\forall \mathsf{has\_father}.\neg\mathsf{Human})$ is only unsatisfiable if both has_father and Human are rigid. It should be noted that rigid concepts and roles provide some information about the inner working of the system since they tell us that their interpretation cannot be changed by state transitions.

While rigid concepts and roles increase the expressive power of the formalism, they are problematic from a computational point of view. In fact, in a logic that allows the application of temporal operators within concepts, rigid roles can even cause undecidability (Gabbay et al. 2003). In $\mathcal{ALC}$-LTL, rigid concepts and roles are less harmful, but they still increase the complexity of reasoning: the satisfiability problem in $\mathcal{ALC}$-LTL is 2-ExpTime-complete if both rigid concepts and roles are allowed; NExpTime-complete if only rigid concepts, but no rigid roles are allowed; and ExpTime-complete if neither concepts nor roles may be rigid.

These results are relevant for the monitoring of dynamic systems since it is easy to see that the monitoring problem can indeed be expressed as a satisfiability problem in $\mathcal{ALC}$-LTL: assume that ψ is an $\mathcal{ALC}$-LTL formula expressing a critical situation (e.g., the formula talking about Bob from above); that ϕ_t is an $\mathcal{ALC}$-LTL formula describing the observations made until time point t (e.g., observations made at certain time points by nurses or heart-rate sensors, and compiled into an $\mathcal{ALC}$-LTL formula using the next-operator); and ϕ is a conjunction of terminological axioms expressing (atemporal) background information (e.g., a medical ontology). Then the critical situation can be detected by testing whether the $\mathcal{ALC}$-LTL formula $\Box\phi \wedge \phi_t \wedge \neg\psi$ is satisfiable.

Runtime verification

A potential problem with the monitoring approach sketched above is that the size of the formula ϕ_t increases with every additional time point at which an observation is made, and thus one needs to solve ever larger satisfiability problems. For propositional LTL, this problem has been addressed in the runtime verification community: for a given LTL formula ψ a monitor M_ψ (i.e., a deterministic finite automaton with state output) is constructed whose size depends only on the formula describing the critical situation, and not on the number of time points (Bauer, Leucker, and Schallhart 2011). At each time point, the observations lead to a transition of the monitor, and the output of the state reached through this transition says whether ψ has been violated or not.

Building on results from (Baader, Ghilardi, and Lutz 2008; 2012), we have shown in (Baader, Bauer, and Lippmann 2009) that this approach can be extended from propositional LTL to $\mathcal{ALC}$-LTL. The main advantage of this extension over the propositional case is, on the one hand, that one can employ ontologies defining the important concepts of the application domain (e.g., concepts defined in a medical ontology) to describe critical situations. On the other hand, in contrast to runtime verification for propositional LTL, where for every time point one assumes to have complete information about the values of the propositional variables, the approach developed in (Baader, Bauer, and Lippmann 2009) can also deal with incomplete information. Improving on the results in (Baader, Bauer, and Lippmann

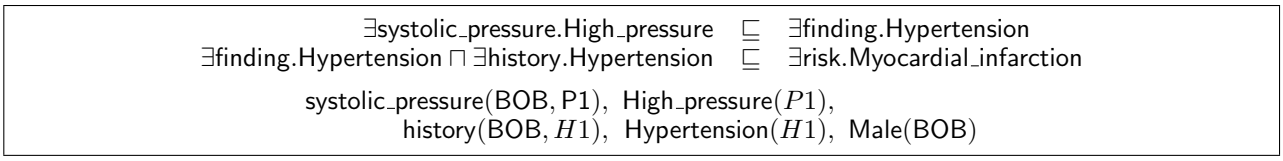

Figure 1: An ontology and a fact base

2009), where some of the monitor constructions were triple-exponential, we show in (Baader and Lippmann 2014) that, for a given $\mathcal{ALC}$-LTL formula ψ, a monitor M_ψ of double-exponential size can be constructed in double-exponential time. In general, this double-exponential blow-up cannot be avoided, even in the propositional case.

Temporalizing ontology-based data access

In (Baader, Borgwardt, and Lippmann 2013), we introduce an extension of $\mathcal{ALC}$-LTL where conjunctive queries using concepts and roles as predicates can be used in place of $\mathcal{ALC}$ axioms. For example, the conjunctive query

$$\exists y.\mathsf{risk}(x, y) \wedge \mathsf{Myocardial\_infarction}(y) \wedge \mathsf{Male}(x)$$

asks for male patients that are at risk to have a heart attack. The obvious difference to $\mathcal{ALC}$ axioms is the use of variables in conjunctive queries. On the one hand, there are the free variables (called answer variables): for these (x in the example) one wants to find named individuals such that replacing the variable by the individual name makes the query true. On the other hand, there are existentially quantified variables, which may refer to unnamed individuals. In our example, the myocardial infarction y need not have an explicit name in the fact base. Answering such queries w.r.t. an ontology is called *ontology-based data access*. Here one assume that one has a fact base, consisting of assertions, and an ontology, consisting of terminological axioms. As an example, assume that the fact base contains the assertions about the patient Bob shown in the lower part of Figure 1, which say that Bob has high blood pressure (obtained from sensor data), and is male and has a history of hypertension (obtained from the patient record). In addition, we have an ontology that says that patients with high blood pressure have hypertension and that patients that currently have hypertension and also have a history of hypertension are at risk for a heart attack, as shown in the upper part of Figure 1. The situation we want to recognize for a given patient x is whether this patient is a male person that is at risk for a heart attack, which is described by the conjunctive query from above. Given the information in the fact base and the axioms in the ontology, we can derive that Bob satisfies this query. Obviously, without the ontology this answer could not be derived.

The complexity of OBDA, i.e., the complexity of checking whether a given tuple of individuals is an answer of a conjunctive query in a fact base w.r.t. an ontology, has been investigated in detail for cases where the ontology is expressed in an appropriate description logic. One can either consider the *combined complexity*, which is measured in the size of the whole input (consisting of the query, the ontology, and the fact base), or the *data complexity*, which is measured in the size of the fact base only (i.e., the query and the ontology are assumed to be of constant size). The underlying assumption is that query and ontology are usually relatively small, whereas the size of the data may be huge. In the database setting (where there is no ontology and closed world assumption is used), answering conjunctive queries is NP-complete w.r.t. combined complexity and in AC^0 w.r.t. data complexity. For OBDA w.r.t. $\mathcal{ALC}$ ontologies, the complexity is considerably higher: ExpTime-complete w.r.t. combined complexity and coNP-complete w.r.t. data complexity (Calvanese, De Giacomo, and Lenzerini 1998; Lutz 2008; Calvanese et al. 2006).

In (Baader, Borgwardt, and Lippmann 2013), we have extended these results to temporal conjunctive queries, which are obtained from LTL by replacing propositional variables with conjunctive queries. The complexity again depends on whether rigid concepts/roles are allowed or not. For the combined complexity, the obtained complexity results are identical to the ones for $\mathcal{ALC}$-LTL, though the upper bounds are considerably harder to show. For data complexity, we obtain the same complexity as for atemporal OBDA (coNP-complete) if no rigid roles are allowed. With rigid roles, we have an ExpTime upper bound (in contrast to 2-ExpTime for combined complexity), but can only show a coNP lower bound.

Monitoring White Box Systems

In this setting, one assumes that one has some knowledge about the inner working of the system, i.e., a (possibly incomplete) specification of how states of the system are transformed into each other. In principle, such a specification could be provided by an appropriate temporal logic. However, there are some problems with this approach. First, such a specification may require a temporalized description logic in which temporal operators can be applied within concepts, which may cause computational problems. Second, in addition to saying what changes during a state transition, one also needs to specify what does not change. This so-called *frame problem* has been investigated in detail in the *reasoning about actions* community. In (Baader et al. 2005; Baader, Lippmann, and Liu 2010) we have introduced action formalisms based on description logics, which inherit Reiter's solution to the frame problem from situation calculus (Reiter 2001).

Based on an action theory defined in such a formalism, one can specify the inner working of a system (e.g., an autonomous robot) using the action programming language Golog (de Giacomo, Lespérance, and Levesque 2000). In this setting, rather than monitoring a single run of the system, we are interested in the *verification problem*: show that certain (temporal) properties are satisfied by any run of the

program. The first attempt to solve the verification problem for action theories based on description logics can be found in (Baader, Liu, and ul Mehdi 2010). However, instead of examining the actual execution sequences of a given Golog program, this approach considers infinite sequences of actions that are accepted by a given Büchi automaton $\mathcal{B}$. If this automaton over-approximates the program, i.e. all possible execution sequences of the program are accepted by $\mathcal{B}$, then any property that holds in all the sequences accepted by $\mathcal{B}$ is also a property that is satisfied by any execution of the program. As logic for specifying properties of infinite sequences of actions, the approach again uses $\mathcal{ALC}$-LTL. In (Baader and Zarrieß 2013), we improve on the results in (Baader, Liu, and ul Mehdi 2010) by directly considering all (finite and infinite) sequences of actions that are runs of a given Golog program.

Future Research

One important topic for future research are extensions towards probabilistic reasoning. Probabilities may come into our framework of ontology-based monitoring of dynamic systems for a variety of reasons: sensors used to populate the fact base may be erroneous with some probability; based on the observed run of the system, one may compute projections into the future, which are again equipped with a probability; the application of an action may have probabilistic outcomes (the action may, e.g., succeed only with some probability).

Another interesting topic is the integration of numerical sensor values. In the black box setting, these values can be represented using description logics with concrete domains (Lutz 2003), which may, however, cause computational problems. In the white box setting, one needs approaches for describing how the numerical values change of time. One possibility for expressing this is to use systems of differential equations, as done in mathematical modeling of dynamical systems (Scheinerman 2012).

References

Baader, F., and Lippmann, M. 2014. Runtime verification using a temporal description logic revisited. LTCS-Report 14-01, Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, Germany. See http://lat.inf.tu-dresden.de/research/reports.html.

Baader, F., and Zarrieß, B. 2013. Verification of Golog programs over description logic actions. In *Proc. FroCoS'13*, LNCS 8152, 181–196. Springer-Verlag.

Baader, F.; Bauer, A.; and Lippmann, M. 2009. Runtime verification using a temporal description logic. In *Proc. FroCoS'09*, LNCS 5749, 149–164. Springer-Verlag.

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P., eds. 2003. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge, UK: Cambridge University Press.

Baader, F.; Lutz, C.; Miličić, M.; Sattler, U.; and Wolter, F. 2005. Integrating description logics and action formalisms: First results. In *Proc. AAAI'05*, 572–577. AAAI Press.

Baader, F.; Bauer, A.; Baumgartner, P.; Cregan, A.; Gabaldon, A.; Ji, K.; Lee, K.; Rajaratnam, D.; and Schwitter, R. 2009. A novel architecture for situation awareness systems. In *Proc. Tableaux'09*, LNCS 5607, 77–92. Springer-Verlag.

Baader, F.; Borgwardt, S.; and Lippmann, M. 2013. Temporalizing ontology-based data access. In *Proc. CADE-24*, LNCS 7898, 330–344. Springer-Verlag.

Baader, F.; Ghilardi, S.; and Lutz, C. 2008. LTL over description logic axioms. In *Proc. KR'08*, 684–694. AAAI Press.

Baader, F.; Ghilardi, S.; and Lutz, C. 2012. LTL over description logic axioms. *ACM Trans. Comput. Log.* 13(3).

Baader, F.; Lippmann, M.; and Liu, H. 2010. Using causal relationships to deal with the ramification problem in action formalisms based on description logics. In *Proc. LPAR'10*, LNCS 6397, 82–96. Springer–Verlag.

Baader, F.; Liu, H.; and ul Mehdi, A. 2010. Verifying properties of infinite sequences of description logic actions. In *Proc. ECAI'10*, 53–58. IOS Press.

Bauer, A.; Leucker, M.; and Schallhart, C. 2011. Runtime verification for LTL and TLTL. *ACM Trans. Softw. Eng. Methodol.* 20(4).

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2006. Data complexity of query answering in description logics. In *Proc. KR'06*, 260–270. AAAI Press.

Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 1998. On the decidability of query containment under constraints. In *Proc. PODS'98*, 149–158. ACM Press.

de Giacomo, G.; Lespérance, Y.; and Levesque, H. J. 2000. Congolog, a concurrent programming language based on the situation calculus. *Artif. Intell.* 121(1-2):109–169.

Gabbay, D.; Kurusz, A.; Wolter, F.; and Zakharyaschev, M. 2003. *Many-dimensional Modal Logics: Theory and Applications*. Elsevier.

Lutz, C. 2003. Description logics with concrete domains—a survey. In *Advances in Modal Logics Volume 4*, 265–296. King's College Publications.

Lutz, C. 2008. The complexity of conjunctive query answering in expressive description logics. In *Proc. IJCAR'08*, LNCA 5195, 179–193. Springer-Verlag.

Reiter, R. 2001. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press.

Scheinerman, E. R. 2012. *Invitation to Dynamical Systems*. Dover Publications.