Applying Knowledge Representation and Reasoning to (Simple) Goal Models

Alexander Borgida*, Jennifer Horkoff†, John Mylopoulos†
*Dept of Computer Science, Rutgers University, USA
†DISI, University of Trento, Italy

***Abstract*—We consider simple i^*-style goal models with influence (contribution) links and AND/OR refinement (decomposition), and formalize them by translation into three standard logics that are actively studied in AI Knowledge Representation and Reasoning (KR&R): propositional logic, FOL and description logics (the first formalization is well known). In each case, this provides a semantics for the notation, on which we can base the definition of forward ("what if?") and backward ("how is this achievable?") reasoning, of interest to requirements engineers. We consider the manner in which AI KR&R research provides off-the-shelf algorithms that can be used to solve these tasks. We compare the representations by reporting known worst-case complexity results for the reasoning, as well as other criteria such as size/understandability of axiomatization, and ease of extension of modeling language.**

I. Introduction

Requirements Engineering research often introduces new language/notations. Whether used to communicate ideas, to check for consistency, or to solve more general problems, such languages need to be given a semantics. The purpose of this paper is to illustrate the use of standard AI KR&R formalisms for this purpose, and to argue for the benefits of doing so.

Specifically, we take a subset, i^*-CORE, of the i^* notations/languages, choose a seminal existing semantics for it [8], and express this in three logics: propositional logic, FOL, and description logics, (the non-propositional logic formalizations are somewhat novel). We then consider the problems of "propagating satisfaction forward" and "backward" in such models – typical tasks that requirements engineers perform with them. For each of the above formalizations of i^*-CORE, we then cast about in the KR&R literature for analyses of worst-case complexity concerning reasoning and off-the-shelf tools, showing that there is a wealth of resources available[1].

We also consider the perspicuity of the formalizations, and the ease with which some changes/additions to i^*-CORE can be specified. We summarize some of the advantages/disadvantages of the approaches at the end.

Our comparison can help guide future work in selecting an appropriate formal representation of goal models, and more generally RE languages.

II. Background: The Goal Model

Goal models have been widely studied in RE as a means to capture and reason over stakeholder goals and high-level requirements. Although several types of goal models exist, in this work we focus on i^*-style goal models, as their use of "softgoals" – goals without clear-cut criteria for satisfaction, and contribution relationships provide both powerful expression in terms of requirements tradeoffs and challenges in terms of formalization and reasoning. (See [19] for the foundational description, and [1] for a list of i^*-related publications.) We introduce a simple variant of an i^*-style goal modeling notation/language here, which we call i^*-CORE, and illustrate it using a small model for a hotel information system derived from an example in [11] (see Figure 1).

A (graphical) i^*-CORE model starts from nodes corresponding to goals, e.g., *Maximize profit*, *Make use of IT*. Note that in our simple i^*-CORE model, we use only the goal concept, without explicitly identifying softgoals. Goals can be decomposed/refined using the familiar AND/OR relations, e.g., the hotel *Makes use of IT* (*g1*) by *Developing a hotel website* (*g2*) AND *Acquiring a booking system* (*g3*) (written in text as $\{g2, g3\} \xmapsto{AND} g1$); the booking system can be developed *in-house* (*t1*) OR can be *rented* (*t2*), written as $\{t1, t2\} \xmapsto{OR} g3$. We will call leaf-level goals without further refinement *tasks*.

Inspired by work on non-functional requirements, a goal can influence (called contribution in i^*) another positively or negatively. For example, *Renting the booking system (t2)* will have a partial positive effect on *Maximize profit (g4)*, written as $t2 \xmapsto{+} g4$, but will have a partial negative effect on *Facilitate control (g5)*, $t2 \xmapsto{-} g5$. In fact, it is useful to allow a stronger degree of influence, e.g., developing the booking system *In house (t1)* has a strong positive influence on *Facilitate control (g5)*, $t1 \xmapsto{++} g5$, and a strong negative effect on *Maximize profit (g4)*, $t1 \xmapsto{--} g4$.

Model analysis. A goal model communicates information between and amongst stakeholders, analysts, and developers. To gain a better understanding of it, a model can be used for simulation: exploring "what-if"-scenarios, where certain tasks/goals are assumed to be satisfied (or possibly denied), and one wants to see what effect this has on top-level goals. This is called "forward propagation/reasoning". For example, in Figure 1, what if the hotel rented a system and did not produce one in house? Would their goals be sufficiently "satisfied" or "supported"? In this simple model, the influences of our decisions are easy to trace by hand, but in most i^*models – typically quite large – systematic reasoning supported by language semantics is needed.

[1]This is not to say that previous research on the i^*-family has not used KR&R resources, such as SAT-solvers; only that the specific problems we look at can use previously unmentioned resources.

978-1-4799-6354-6/14/$31.00 © 2014 IEEE

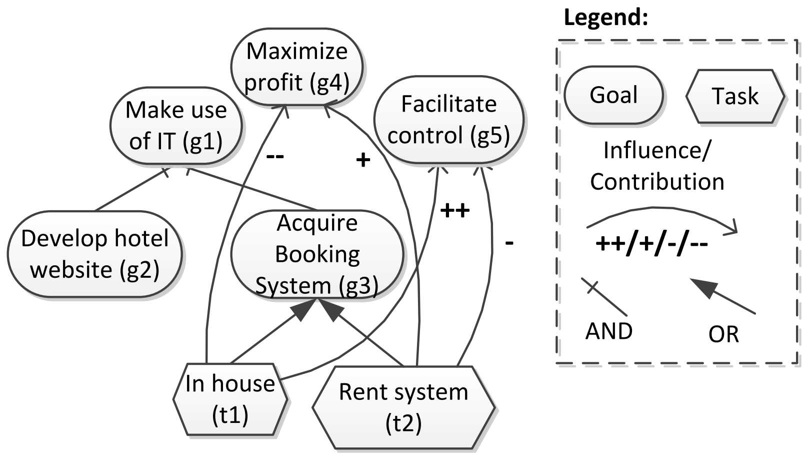

Fig. 1. i^*-CORE example for a Hotel, adapted from [11]

Alternatively, an engineer may want to solve the requirements problem by looking for a set of tasks necessary to achieve some goal(s). For example, is there a set of tasks for the hotel which, if implemented, would satisfy both *Maximize profit* and *Facilitate control*? This is the so-called "backward propagation" problem. Considering a simpler example, if the users wanted *Make use of IT (g1)* to be fully achieved, tracing "backwards" through the links, both *Develop hotel website (g2)* and *Acquire Booking System (g3)* must also be fully achieved, as these goals are linked together via AND refinement. The goal *g2* is not further refined, and must be achieved by user input. In order for *g3* to be achieved, one of *In house (t1)* or *Rent System (t2)* must be achieved, as these tasks are connected to *g3* via OR refinement. Finally, in order to achieve *g1*, we arrive at the solution where *g2*, (*t1* or *t2*) must be achieved. In this paper we will focus on finding "subset minimal" solutions (i.e., ones where there are no "useless" tasks, that can be removed while still achieving the goals, e.g., achieving either *t1* or *t2*, not both).

Because of the presence of positive and negative influences to the same goal, a significant question will be how to evaluate/combine the "satisfaction" of (level of support for) a node, and how to accumulate evidence for this from both refinement and influence information. Essentially, such decisions will specify the *semantics* of the notation, based on which one can prove the correctness of the techniques for answering questions about models.

There are a variety of approaches for specifying the semantics, including ones based on planning, formal logic, and even "procedural semantics" – what an implementation does. (See [13] for a review.) Here we focus on three techniques which represent the i^*-CORE semantics using formal logics, the first of which is taken from the literature.

III. Approaches Based on KR&R

In the sub-category of logical approaches, which are a central topic of interest for AI KR&R, the presence of potentially conflicting influences, which need to co-exist, leads one to initially consider "para-consistent logics", which allow certain propositions to be both true and false, without causing *all* propositions to be both true and false (*ex falso quodlibet*), which is the case in classical logic. Although there are many such logics (e.g., Belnap's 4-valued logic, in which one can assign to a proposition a subset of $\{T,F\}$; in this case *Maximize profit* being assigned $\{T,F\}$ can be interpreted as having evidence both for and against the proposition), the problem with them is that usually even experts find it hard to *intuit* the logical consequences (what can be deduced) from a set of formulas. This is in contrast with logics that have standard Tarski-style semantics, such as FOL or description logics (DL), which appear to have relatively intuitive notions of logical consequence, shared by well-educated computer scientists. For this reason, we will focus on formalizations of our goal model using such KR&R formalisms. They will all share the following features, inspired by the ground-breaking work of Giorgini et al. [8]: (i) associate with every goal a way of accumulating different kinds of evidence – in our case FullSupport (FS), PartialSupport (PS), PartialDenial (PD), FullDenial (FD); (ii) there is no built-in logical mechanism producing conflict/inconsistency between these, the only constraint being that FS implies PS, and FD implies PD. This approach leaves to the users the decision of what to do when the same goal *g* accumulates conflicting evidence (e.g., when both FS and PD hold of *g*).

The three formalizations we offer below are in fact intended to be equivalent in the sense that given a specific i^*-CORE model, the solutions to the forward and backward reasoning problems will be the same. The reason we investigate three alternatives is that they offer different benefits, and also give us an opportunity to demonstrate the variety of different ideas that we can find in the AI KR&R arsenal to support RE analysis over goal models.

A. Formalization in Propositional Logic

Language formalization and specific model representation. A seminal paper providing a formal semantics for i^*-like models, [8], used propositional logic, offering *formula schemas* that are to be instantiated for every model component. Therefore there is no separate theory describing the semantics of the notation. Instead, every specific model is translated into a set of formulas. To begin with, for every goal *g*, 4 propositional symbols FS_*g*, PS_*g*, PD_*g*, FD_*g* are introduced, plus axioms FS_*g* $\rightarrow$ PS_*g* and FD_*g* $\rightarrow$ PD_*g*. These axioms capture the meaning of "partial/full" that if there is full support/denial then there must partial support/denial.

Then, for $h \xmapsto{-} g$ one adds 2 axioms: PD_*h* $\rightarrow$ PS_*g* and PS_*h* $\rightarrow$ PD_*g*. For example, $t2 \xmapsto{-} g5$ in our Hotel example results in axioms PD_*Rent-system* $\rightarrow$ PS_*Facilitate-control* and PS_*Rent-system* $\rightarrow$ PD_*Facilitate-control*.

The axioms for AND-decomposition { *g3:DevelopWebSite*, *g2:AcquireBookingSystem* } $\xmapsto{AND}$ *g1:MakeUseOfIT* are more complex. First one adds 2 axioms for positive evidence propagation, then 4 axioms for negative evidence propagation:

$$
\begin{array}{ll}
\text{FS\_}g3 \wedge \text{FS\_}g2 \rightarrow \text{FS\_}g1 & \\
\text{PS\_}g3 \wedge \text{PS\_}g2 \rightarrow \text{PS\_}g1 & \\
\text{FD\_}g2 \rightarrow \text{FD\_}g1 & \text{PD\_}g2 \rightarrow \text{PD\_}g1 \\
\text{FD\_}g3 \rightarrow \text{FD\_}g1 & \text{PD\_}g3 \rightarrow \text{PD\_}g1
\end{array}
$$

Hopefully this gives the reader an intuition for the translation of the remaining language constructs – the original table indicating the translations has 26 implications, and, on the average, a construct translates to 4 axioms.

Model analysis. To do forward-reasoning, one adds to the above theory propositions for tasks assumed to be carried out (e.g., FS_*t1*) and then checks which of the 4 propositions about a top-level goal *g4* (e.g., FS_*g4*) are derivable.

Interestingly, the above theory is "definite Horn" (every formula is an implication with right side an atom), and it is known [6] that satisfiability and logical consequence of atoms can be computed in *linear time* for such theories. Note that if, as part of the forward propagation problem, we also want to say that certain tasks are not carried out (i.e., $\neg$FS_*t2*), or are mutually exclusive (FS_*t1* $\wedge$ FS_*t2* $\rightarrow$ $False$), the theory is just "Horn", not "definite Horn". Luckily, in this case the same linear-time algorithm works.

Although [8], [9] formalized the problem of qualitative reasoning using logic, as indicated above, they reformulate it using separate accumulators for $supported$ and $denied$, and end up writing their own forward-propagation algorithm, which they prove correct, rather than taking the ready-made solution offered by AI KR&R research (This may be due to the fact that they also have algorithms for non-qualitative reasoning later.)

For backward propagation, we turn to a different kind of reasoning, called *abduction*, which is defined as: given theory $\mathcal{T}$, a set of atoms $\mathcal{H}$, and a desired consequence φ, find a (minimal) subset S of $\mathcal{H}$, such that $\mathcal{T} \cup S$ is consistent and entails φ.[2]

In our case, $\mathcal{T}$ is the logical encoding of the specific i^*-CORE model plus the propagation axioms, as indicated above; $\mathcal{H}$ are the available tasks (in the form of propositions of the form FS_*t* for tasks *t*), and φ is a proposition expressing the desired level of satisfaction/denial of a top-level goal; S will be the solution of the requirements problem – the subset of tasks that are needed to ensure φ.

Because $\mathcal{T}$ is "definite Horn", there is in fact a polynomial time algorithm to solve the backwards reasoning problem, as defined above. Interestingly, if we want to add to $\mathcal{T}$ some additional desiderata that are Horn rules but non-definite (e.g., that some goal is *not* to have strongly denied evidence: $\neg$FD_*g4*; or that a task *t* should not be both denied and supported since it does not make sense to both run it and not run it: PS_*t* $\wedge$ PD_*t* $\rightarrow$ $False$), the abduction problem becomes NP-complete. And if the constraint cannot be expressed in Horn rules, then worst-case complexity of abduction is even higher (second level of the polynomial hierarchy)!

There is in fact a well known AI tool – de Kleer's Augmented Truth Maintenance System (ATMS) [5], that can be used to solve abduction for propositional Horn systems; it has actually been applied in RE research [7], because it supports well incremental changes to a theory.

It is interesting to compare the above approach to solving the "backward reasoning" problem to the one in [15], which is based on satisfiability checking. That paper augments $\mathcal{T}$ with so called Clark-completion axioms, which essentially say that "the only way to achieve goal g is via one of the refinements or influences in the theory"; e.g., if the only implications with head FS_*g4* are FS_*t1* $\rightarrow$FS_*g4* and FS_*t2* $\rightarrow$FS_*g4*, then add FS_*g4* $\rightarrow$(FS_*t1* $\vee$ FS_*t2*). By adding to the theory a desired top goal atom, such as FS_*g4*, together with axioms requiring that every task be either run or not, one can then use any SAT algorithm to find some assignment of True or False to every propositional symbol, thus obtaining some set S of tasks whose performance assures *g4*.

It is interesting to note that if the original problem is (general) Horn, then both this approach and the one based on abduction are NP-complete, but if the requirements problem can be expressed in definite Horn, then the abduction solution can be found in P-time, while the satisfiability-based approach is still NP-complete (because of the added Clark-completion axioms). This shows that even in one formalism (propositional logic) different ways of expressing the problem has different computational consequences.

B. Formalization in FOL

As noted above, the propositional formalization does not have a separate sub-theory giving the general language semantics, and is also extremely verbose. The following approach treats goals and evidence values as constants, and uses a few

[2]For a survey of complexity results on propositional abduction, some used below, see [4].

predicates and quantifiers to alleviate this problem to a great extent.

Model representation.

Given a specific model, we add atoms $goal(g)$ for every goal node g in it. For each of the constructs, we introduce a new predicate, so that binary AND-decomposition $\{g1, g2\} \xmapsto{AND} g3$ is represented as $\text{AND\_REFINE}(g1, g2, g3)$; positive influence $g \xmapsto{++} h$ is represented as $\text{INFL++}(g, h)$; etc. So the theory for Example 1 will contain atomic formulas such as $\text{AND\_REFINE}(developHotelWebsite, aquireBookingAystem, makeuseofIT)$ and $\text{INFL-}(rentSystem, facilitateControl)$.

Note that this representation is isomorphic to the graphical model, and hence much easier to understand and modify even at the logical level.

Language formalization.

The basis of our approach is to treat goals and evidence values as individuals, and introduce a binary predicate $evid(g, val)$ that associates with goal g evidence value val. So the proposition FS_g will now be represented as $evid(g, \text{FS})$. It will be useful to introduce unary predicates $goal$ and $value$, so we can distinguish goal individuals (e.g., $goal(g4)$ will be asserted) from evidence values (e.g., $value(\text{FS})$ will be asserted).

The relationship between FS and PS can now be captured by a single axiom

$$\forall x.goal(x) \rightarrow (evid(x, \text{FS}) \rightarrow evid(x, \text{PS}))$$

in contrast to the propositional case, where this had to be repeated for every goal x. Positive influence edges simply propagate the values of the respective goals, so we can also reduce the number of axioms needed for positive influence, like $h \xmapsto{++} g$, from 4×(# of goals) to 1 by writing[3]

$$value(v) \wedge \text{INFL++}(h, g) \rightarrow (evid(h, v) \rightarrow evid(g, v))$$

For negative influence edges, we observe that they cause the evidence value to be "complemented" (e.g., if $\text{INFL- -}(h, g)$ and h is *supported*, then g will be *denied*, and conversely. This can be conveniently captured by introducing binary predicate $complement$ which holds of (FS,FD), (FD,FS), (PS,PD), and (PD,PS), and then capture more cleanly the semantics of $g \xmapsto{--} h$, for example, through the axiom

$$\begin{aligned} complement(val, \quad & complVal) \wedge \text{INFL- -}(g, h) \\ & \rightarrow (evid(g, val) \rightarrow evid(h, complVal)) \end{aligned}$$

If we add auxilliary predicates $posVal$, true of FS and PS, as well as $negVal$, true of FD and PD, AND decomposition is formalized by 3 rules, which again capture more cleanly the semantics of value propagation:

$$\begin{aligned} \text{AND\_REFINE}(g2, g3, h) \wedge \quad & evid(g2, v) \wedge evid(g3, v) \wedge \\ & posVal(v) \rightarrow evid(h, v) \\ \text{AND\_REFINE}(g2, g3, h) \wedge \quad & evid(g2, v) \wedge \\ & negVal(v) \rightarrow evid(h, v) \\ \text{AND\_REFINE}(g2, g3, h) \wedge \quad & evid(g3, v) \wedge \\ & negVal(v) \rightarrow evid(h, v) \end{aligned}$$

Model analysis. Because all values in the domain are named constants, the quantifiers could in fact be eliminated, and we would obtain a theory much like the propositional one earlier. Let us instead stay with the current FOL formalization. Because $a \rightarrow (b \rightarrow c)$ is logically equivalent to $a \wedge b \rightarrow c$ and all variables are universally quantified at the beginning, the theory generated above consists in fact of predicate logic Horn clauses, where each clause has at least one positive atom ("definite Horn theory"). It is well known that finding all the atoms entailed by such a theory can be done in polynomial time (the problem is actually PTIME-complete) and the unoptimized algorithm for it can be found in any database textbook chapter on Datalog, while optimized implementations are available as Datalog implementations.[4] Note that the price of the cleaner and more succinct representation of the model and its semantics is a jump in formal complexity from linear time up to polynomial time.

As far as solving the backwards reasoning problem, we now rely on First Order Horn-logic abduction, which is again intractable, although there are algorithms in the literature, such as those in [14], which have been empirically evaluated and seem to run effectively.

C. Formalization in Description Logics

Description logics (DLs) have been an important KR&R formalism since the 1980's, and have received intensive attention over the past decade, especially because they are used to define "ontologies", and the OWL DL [17] has been chosen by W3C as the official knowledge representation language for the Semantic Web.

They are therefore of interest as a basis for the semantics of i^*-CORE because, among others, this gives one the opportunity to publish i^*-CORE models on the Semantic Web, as well as taking advantage, for the purposes of RE analysis, of the intensive effort devoted to analyses and tools dealing with OWL and DLs in general.

DLs view the world as populated by individuals, related to each other by binary relations, called *properties*, and grouped into *classes/concepts*. They are distinguished by the fact that in addition to primitive classes, one can *define* composite concepts using concept-constructors, in the manner of composite noun-phrases with nested relative clauses. For example, the "existential restriction" concept constructor $\exists p.C$ denotes objects that are related to at least one instance of C by the p

[3]For brevity, we henceforth leave out the initial universal quantifiers for all the variables.

[4]Please be aware that in KR&R the term Datalog is sometimes applied to the much more powerful Answer Set Programming (ASP) paradigm, rather than just positive Horn clauses.

property, while $\sqcap$ acts like intersection on classes. Thus the concept *PERSON* $\sqcap$ $\exists$ *hasChildren.*($\forall$ *hasPets.DOG)* denotes *"Persons who have some child (who has only pets (that are Dogs))"*.

Classes can be used to specify *subsumption/subclass axioms* of the form *C* $\sqsubseteq$ *D*, indicating that all instances of *C* must also be instances of *D*. For example,

PERSON $\sqcap$ $\exists$ *hasPets.DOGS* $\sqsubseteq$ *VET_VISITOR* (1)

When two concepts subsume each other, and one is an atomic name we consider such an axiom a definition:

MARRIED $\equiv$ *PERSON* $\sqcap$ $\exists$ *hasSpouse.PERSON*

A significant point about DLs is that the subsumption hierarchy can be inferred: given *DOGS* $\sqsubseteq$ *ANIMALS*, the logic/system can then infer $\exists$ *hasPets.DOGS* $\sqsubseteq$ $\exists$ *hasPets.ANIMALS*. In fact, one of the services provided by DL-reasoners is automatic classification of all named concepts in the subsumption hierarchy. This can provide an advantage to requirements model builders, if they wish to use not just atomic names for goals (e.g., have a complex ontology of goals).

It has long been known that DL axioms can be (mechanically) translated into FOL. For example, subsumption (1) above corresponds to the formula

$$\begin{array}{l}\forall x.(PERSON(x) \wedge (\exists y.hasPets(x,y) \wedge DOG(y)) \\ \rightarrow VET\_VISITOR(x)\end{array}.$$

The point is that DLs can express different sets of concepts depending on the constructors available, and much of the research concerns finding sets of constructors for which subsumption is at least decidable (unlike full FOL).

In fact, we have already published a formal semantics in OWL of an extended version of i^*-CORE as part of the semantics for the BIM business intelligence model [10]. However, since then certain subsets of OWL have been declared by W3C as official "profiles". These subsets are less expressive than the full language, but in exchange they have much better computational properties. For example, the OWL2 EL profile has polynomially time concept reasoning, while full OWL2 is double-exponential time complete [18]. We show below how to represent sufficient aspects of the semantics of i^*-CORE using only two concept constructors: conjunction and qualified existential restriction – the $\mathcal{EL}$ description logic.

Model representation. We begin with a class *GOAL*, and introduce primitive subclasses FS, PS, PD, and FD. In this first approach, each specific goal/task in the model will also be a primitive subclass of *GOAL*.[5] The idea is that if goal *g4* will accumulate evidence PS and PD, we will be able to derive

(g4: Maximize profit) $\sqsubseteq$ PS$\sqcap$ PD

Note that there is only one entity in the model, the class *g4*, for the goal g4, in contrast to the propositional encoding, which has 4: FS_*g4*, PS_*g4*,FD_*g4*,PD_*g4*. Influence links will be represented using properties INFLBY++, INFLBY+, INFLBY-, INFLBY- -, so that $t2 \overset{+}{\longmapsto} g4$ will be captured by the axiom

$\exists$ INFLBY+.*(t2: RentSystem)* $\sqsubseteq$ *(g4: MaximizeProfit)*

The reason for using the rather unintuitive inverse relation INFLBY++ instead of INFL++ is that we will need to accumulate evidence *onto g4* from *t2*. (See below.)

Binary refinement of goals will be represented using two properties REFBY$_1$ and REFBY$_2$, so that $\{g2, g3\} \xrightarrow{AND} g1$ will be represented by the axiom

(g1: MakeUseofIT) $\sqsubseteq$
AND_GOAL $\sqcap$
$\exists$ REFBY$_1$.*(g2: DevelopWebsite)* $\sqcap$
$\exists$ REFBY$_2$.*(g3: AcquireBookingSystem)*

where *AND_GOAL* is a primitive subclass of *GOAL*. OR-decompositions such as $\{t1, t2\} \xrightarrow{OR} g3$ will be represented by the axiom

g3 $\sqsubseteq$
OR_GOAL $\sqcap$
$\exists$ REFBY$_1$.*t1* $\sqcap$
$\exists$ REFBY$_2$.*t2*

Language formalization. First, we relate full and partial evidence by two axioms:

FS $\sqsubseteq$ PS FD $\sqsubseteq$ PD

Next, we need axioms for propagating different kinds of evidence across influence links. Consider $\overset{++}{\longmapsto}$; this link propagates the values straightforwardly, which can be captured by 4 axioms such as $\exists$ INFLBY++.FS $\sqsubseteq$ FS and $\exists$ INFLBY++.PS $\sqsubseteq$ PS. The other axioms for evidence propagation across influence also mirror the ones in the propositional case, except that we do not need a separate copy for each individual goal in the model.

For propagating evidence across refinements, consider the case of AND decomposition. Propagating positive evidence from the components of an AND node (values of REFBY$_1$ and REFBY$_2$) is done by axioms such as

(*AND_GOAL* $\sqcap$ $\exists$ REFBY$_1$.FS $\sqcap$ $\exists$ REFBY$_2$.FS) $\sqsubseteq$ FS

Propagating negative evidence in the manner of the propositional case in Giorgini et al., is done by pairs of axioms such as

(*AND_GOAL* $\sqcap$ $\exists$ REFBY$_1$.FD) $\sqsubseteq$ FD
(*AND_GOAL* $\sqcap$ $\exists$ REFBY$_2$.FD) $\sqsubseteq$ FD

Model Analysis. For forward propagation, in case one wants to assume that *t1* was executed, one needs to add the axiom *t1* $\sqsubseteq$ FS. To see if there is partial positive evidence for *g4* as a result of the assumptions, one simply checks whether *g4* is subsumed by PS as a consequence of the theory consisting of the assumptions, domain model and language formalization axioms. In most DL reasoners, the concepts are organized according to the inferred subsumption relationships in a single pass, so checking if *g4* $\sqsubseteq$ PS, can be read of the subclass diagram. The complexity of subsumption in $\mathcal{EL}$ is quadratic according to [2].

[5] This representation is close to the propositional logic one, and utilizes only class-level ("T-Box") reasoning in DLs. A more natural use of DLs would be to make specific goals, such as *g4*, be *instances* of class *GOAL*, and do reasoning at the individual level – so-called "ABox reasoning". We leave an explanation of the effects of this option to future work.

TABLE I

SUMMARY OF BENEFITS AND DRAWBACKS OF CHOICES IN FORMAL GOAL MODEL SEMANTICS

Approach	Complexity Forw.	Complexity Backw.	# Axioms	Lucidity	Existing Tools	Extensibility	Semantic Web
Prop. Logic (Horn)	Liner Time	NP Complete	Many	reasonable	Many (HornSAT, ATMS)	Poor	No
FOL (Horn)	PTIME Complete	NP Hard	Very Few	good	Many (dbDatalog, [3])	Reasonable	No
OWL	2EXP-TIME complete		Few	So-so	Many	Good	Yes
$\mathcal{EL}$ DL	Polynomial O(n^2)	NP-complete	Few	Poor	Some (ELK)	Poor	Yes

Interestingly, a variant of abduction appropriate for concept subsumption has also been studied by Bienvenu [3]. The paper shows that the propositional abduction results for definite Horn theories transfer to $\mathcal{EL}$.

IV. LANGUAGE CHANGES AND EXTENSIONS

We briefly look at a few changes and extensions to i^*-CORE, as a way of seeing how the various formalizations cope with them.

- Sebastiani et al. [15] give different meaning for $\{g1, g2\} \xmapsto{AND} h$ than Giorgini et al.[8], in the case of negative evidence: Sebastiani et al. use (PD_*g1* $\wedge$ PD_*g2*) $\rightarrow$PD_*h* as opposed to the pair { PD_*g1* $\rightarrow$PD_*h*, PD_*g2* $\rightarrow$PD_*h* } given in Section III-A. The point is that the above difference is easy to see and understand above, in the familiar notation of propositional logic, and would have been so in FOL too, but would have been harder to understand in DL, because that notation $\exists$ REFBY_1.PD is less familiar.
- To support goal modeling in practice, we want to introduce a variant of AND/OR which allows n children, for $n \geq 2$. Translation schemas, as in the propositional case, handle this easily. But it is quite painful in the current FOL, which needs to introduce separate (n+1)-ary predicate for every n to represent them. (The same applies for the OWL EL approach in Section III-C, where one needs to introduce addition roles REFBY_i, for $i \geq 2$.) However, the FOL approach can be modified as follows using a reified object to represent the link: add unary predicate AND and binary predicates $anteced$ and $conseq$, so that $\{g1, g2, g3\} \xmapsto{AND} h$, for example, is represented by the 5 atomic assertions $AND(r)$, $anteced(r, g1)$, $anteced(r, g2)$, $anteced(r, g3)$, $conseq(r, h)$. Propagation of FS would then be expressed by the axiom

$$AND(r) \wedge conseq(r, h) \\ \wedge[\forall x.anteced(r, x) \rightarrow evid(x, \text{FS})] \\ \rightarrow evid(h, \text{FS})$$

  which works for any number of antecedents.

  Note that in FOL there was a solution to the problem of having n-ary AND decomposition that was not "brute force", even though we had to change the representation. This was not the case in $\mathcal{EL}$, where the only solution seems to be adding possibly an unbounded number roles. One could take this to indicate that more expressive languages, such as FOL, have a greater ability to accommodate changes.
- Inspired by the OWL concept constructor $\geq k\ p.C$, which represents objects that have at least k values of property p in class C, we could add to i^*-CORE a generalization of AND/OR where at least k out of n refinements need to be satisfied; AND would then be n-of-n, while OR would be 1-of-n. Of course, this is easy to express in OWL (though not OWL EL), but requires many, possibly long, axioms in all the others, including standard FOL. So in this case OWL is superior to FOL, despite FOL's greater expressiveness.

V. DISCUSSION AND CONCLUSIONS

In this paper we have examined three approaches to capturing the formal semantics of an i^*-like goal model that enables the specification of forward and backward RE problem exploration. We had already published a fourth approach using the OWL DL description logic. We summarize some of the benefits and drawbacks of the various approaches in Table I.

The first formalization, in propositional logic, is from [8], but we pointed out that since the theory is propositional definite Horn logic, there were existing solutions in the literature for both forward and backward reasoning.

The other formalizations are new, though in the same spirit. The one in FOL is most lucid, and turns out to be in the Horn clause subset, therefore providing more efficient reasoning solutions than full FOL would have. The one in the OWL2 EL actually uses only the $\mathcal{EL}$ subset, and thus in principle is quadratic time, though all actual implementations handle additional features of OWL2 EL.

For all cases, we showed that the connection of "backward propagation" to abduction can be the source of solutions from KR&R. Note that we have not presented a formal proof of the *equivalence* of the three formalizations – something one would do in KR&R.

We should point out that although in some cases the formal worst-case complexity results are discouraging, experience with specially optimized implementations (e.g., OWL ontology reasoners, full Horn abduction using ATMS, SAT) indicates that on real-world problems performance is often acceptable. In general, an interesting question we have left open is a comparative empirical evaluation of the various tools suggested above on realistic goal models.

In general, there are considerable benefits to be gained by using KR&R formalisms that are continuing to be the object of intense scrutiny in KR&R

Clearly, continued improvements in the effectiveness of SAT testing would benefit any approach that expressed semantics in the non-Horn subset of propositional logic, and relied on SAT-testing for solving some RE problem.

In a different direction, there are numerous results considering the problem of modularizing DL theories: partitioning them into sub-theories that are logically independent in some sense [16]. These results could be applied to RE to break up very large requirements into smaller, independently understandable parts.

Our overall message to the RE community is that those looking to provide precise semantics for goal models, or other languages/notations, and building tools to reason with them, would benefit from trying first one of the *standard* KR&R schemes mentioned above, looking for existing results and tools in the KR&R literature. (Aside: we should however state that we are not dogmatic about using standard logics: some of the authors of this paper themselves are looking at the large literature on para-consistent logics used in RE, to see relative benefits and ways to compare them.) This is relevant because a recent survey has found 22 papers within the last 10 years mapping goal models to some sort of formal language [12]. This proliferation is undesirable: it is not clear how many of these correspond to different semantics, and if so, usually no argument is made why a new one is chosen.

In summary, when formalizing language semantics, language developers must make three key decisions:

- 1) Which formalism? There are benefits to looking at alternative formalisms to express the semantics, since these can have different advantages;
- 2) Which representation? Even within the same formalism, there are different ways of expressing the semantics, with different effects on complexity.
- 3) What semantics? One can explore or at least make clear the effect of different semantics within one formalism, especially if the formalism is familiar to the user and the axiomatization is lucid;

In this paper, we have focused primarily on the first decision, giving examples touching on the latter two points only in Sections IV and V. Often, when providing the semantics for an RE language, these decisions are made implicitly, with little justification for the selection of formalism, representation and semantics. It is our hope that the description and comparison in this paper has demonstrated some of the considerations in making such decisions, particularly when deciding on an appropriate formalism, taking full advantage of existing advances in the KR&R literature.

REFERENCES

[1] (2014) Citeulike iStar Library. [Online]. Available: http://www.citeulike.org/group/14571

[2] F. Baader, "Terminological cycles in a description logic with existential restrictions," in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, ser. IJCAI'03. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, pp. 325–330. [Online]. Available: http://dl.acm.org/citation.cfm?id=1630659.1630707

[3] M. Bienvenu, "Complexity of abduction in the el family of lightweight description logics," in *Proc. KR'08*, 2008, pp. 220–230.

[4] N. Creignou and B. Zanuttini, "A complete classification of the complexity of propositional abduction," *SIAM Journal on Computing*, vol. 36, no. 1, pp. 207–229, 2006.

[5] J. de Kleer, "An assumption-based TMS," *Artificial Intelligence*, no. 28, p. 127162, 1986.

[6] W. Dowling and J. H. Gallier, "Linear-time algorithms for testing the satisfiability of propositional horn formulae," *J. Log. Program.*, vol. 1, no. 3, pp. 267–284, 1984.

[7] N. A. Ernst, A. Borgida, and I. Jureta, "Finding incremental solutions for evolving requirements," in *RE*, 2011, pp. 15–24.

[8] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, "Reasoning with Goal Models," in *Proc. of ER'02*, ser. LNCS, vol. 3084, 2002, pp. 167–181.

[9] ——, "Formal Reasoning Techniques for Goal Models," *Journal on Data Semantics*, vol. 2800, no. 1, pp. 1–20, 2004. [Online]. Available: http://www.springerlink.com/content/nfe4tm8u9vt5j12d

[10] J. Horkoff, A. Borgida, J. Mylopoulos, D. Barone, L. Jiang, E. Yu, and D. Amyot, "Making data meaningful: The business intelligence model and its formal semantics in description logics," in *On the Move to Meaningful Internet Systems: OTM 2012*, ser. Lecture Notes in Computer Science. Springer, 2012, vol. 7566, pp. 700–717.

[11] J. Horkoff, B. Fatma, F.-L. Li, T. Li, and J. Mylopoulos, "Evaluating the expressiveness and effectiveness of a modeling language: A requirements modeling language example," in *Proceedings of the 33rd International Conference on Conceptual Modeling*, ser. ER'14. Springer, 2014.

[12] J. Horkoff, T. Li, F.-L. Li, J. Pimentel, M. Salnitri, E. Cardoso, P. Giorgini, and J. Mylopoulos, "Taking goal models downstream: A systematic roadmap," in *Proceedings of the IEEE 8th International Conference on Research Challenges in Information Science*, ser. RCIS'14. IEEE Press, 2014.

[13] J. Horkoff and E. Yu, "Analyzing Goal Models: Different Approaches and How to Choose among Them," in *Proc. of SAC'11*, 2011, pp. 675–682.

[14] H. T. Ng and R. J. Mooney, "An efficient first-order horn-clause abduction system based on the atms," in *Proc. AAAI'91*, ser. AAAI'91. AAAI Press, 1991, pp. 494–499.

[15] R. Sebastiani, P. Giorgini, and J. Mylopoulos, "Simple and minimum-cost satisfiability for goal models," in *Proc. of CAiSE'04*, 2004, pp. 20–35.

[16] H. Stuckenschmidt, C. Parent, and S. Spaccapietra, Eds., *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, ser. Lecture Notes in Computer Science. Springer, 2009, vol. 5445.

[17] W3C. (2012) OWL 2 Web Ontology Language Document Overview (Second Edition). [Online]. Available: http://www.w3.org/TR/owl2-overview/

[18] ——. (2012) OWL2 EL Profile. [Online]. Available: http://www.w3.org/TR/owl2-profiles/#OWL_2_EL

[19] E. Yu, "Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering," in *Proc. of RE'97*, 1997, pp. 226–235.