Omni-directional Hyper Logic Programs in SILK and RIF*

Benjamin Grosof[1], Carl Andersen[2], Mike Dean[2], and Michael Kifer[3]

[1] Vulcan Inc., Seattle, WA, USA, benjaming@vulcan.com
[2] Raytheon BBN Technologies, Arlington, VA, USA, {canderse, mdean}@bbn.com
[3] Stony Brook University, Stony Brook, NY, USA, kifer@cs.stonybrook.edu

Abstract. SILK is an expressive Semantic Web rule language and system equipped with scalable reactive higher-order defaults. We present two of its latest novel features. *Omni rules* are prioritized defeasible clauses that provide a higher-level expressive abstraction for users by making rules omni-directional. They handle multi-way conflicts. And they are soundly interoperable with FOL and W3C OWL as well as with RIF-BLD, SPARQL, and RDF. *RIF-SILK* is a new RIF dialect that extends RIF-BLD via RIF-FLD.

1 Introduction to SILK

SILK[4] (Semantic Inferencing for Large Knowledge) is an expressive Semantic Web rule language and system equipped with scalable reactive higher-order defaults. The system includes capabilities for reasoning, knowledge interchange, and user interface (UI). Part of Project Halo[5], sponsored by Vulcan Inc., the SILK research program addresses fundamental knowledge representation (KR) requirements for scaling the Semantic Web to widely-authored Very Large Knowledge Bases (VLKBs) in business and science that answer questions, proactively supply info, and reason powerfully. The SILK effort has over 15 contributing institutions, including Vulcan, Stony Brook University, Raytheon BBN Technologies, Cycorp, and SRI International.

SILK pushes the frontier of KR by combining expressiveness plus semantics plus scalability. It targets defeasibility, higher-order, and actions — including to support reasoning about complex processes that are described in terms of causality, hierarchical structure, and/or hypothetical scenarios. Longer-term, SILK targets widely collaborative KA by subject matter experts (SMEs), such as science students/teachers or business people, not just knowledge engineers (KEs) or programmers. SILK's techniques apply to many commercial domains, e.g., for policies and regulations.

SILK has a new fundamental KR: *hyper* logic programs, which extends normal declarative logic programs (LP). Hyper LP is the first to tightly *combine*

* This work is part of the SILK project sponsored by Vulcan Inc.

[4] http://silk.semwebcentral.org

[5] http://projecthalo.com

several key advanced expressive features: *defaults*, with strong negation and priorities, cf. courteous LP [1] with argumentation theories [2]; (quasi) *higher-order* syntax, reification, and meta-reasoning, cf. HiLog [3] and Common Logic; and procedural attachments to *external* actions (side-effectful), queries (to built-ins, web sources or services), and events (knowledge update flows), cf. situated/production LP [1] (and similar to production rules). Other advanced hyper LP expressive features include: webized syntax and interchange cf. W3C Rule Interchange Format (RIF) [4] (and the earlier RDF, RuleML [5], SPARQL, and OWL); frame syntax (object-oriented style) cf. F-Logic [6]; and semantically clean negation-as-failure (NAF), cf. well-founded [7].

The SILK language includes an ASCII presentation syntax, an abstract syntax, and a RIF dialect (RIF-SILK) which includes markup syntax. The hyper LP semantics includes a model theory and proof theory. Essentially, its implementation can be viewed as *transforming knowledge* from higher to lower abstraction levels. Higher is good for knowledge acquisition (human cognition); lower is good for reasoning (code reuse and optimization). Despite hyper LP's expressive power, inferencing in it is highly scalable computationally. It is tractable under non-onerous restrictions, having practically the same computational complexity as Horn LP (similar to Horn first-order logic (FOL)). Hyper LP retains the pragmatic quality of LP: it is *intuitionistic*, i.e., conclusions drawn are definite, and lacks general "reasoning by cases".

SILK knowledge interchange can be via load (import), or query (external, during inferencing), or event (update received). KR languages supported for interchange include: SPARQL and RDF(S); SQL and ODBC (e.g., Excel spreadsheets); SILK, RIF (-BLD and -SILK), and OWL (-RL); Cyc; and AURA [8].

2 Omni Rules: Omni-directional LP Defaults

We have developed a novel major expressive extension in hyper LP, called *omni rules*, and implemented support for it in SILK. Omni rules are prioritized defeasible axioms that provide a yet higher-level expressive abstraction for users. Omni rules handle multi-way conflicts. And omni rules are soundly interoperable with FOL and W3C OWL as well as with RIF-BLD, SPARQL, and RDF.

SILK syntax: A SILK rule has the form "`@[tag->G] H :- B ;`". Here, "`:-`" is the usual LP rule implication connective, i.e., informally "if". `H` and `B` are called the rule head and body, respectively; and "`;`" is the statement ending delimiter. ":- B" may be absent, i.e., the body may be empty (true). `a # c` means that `a` is an instance of class `c`. Skolems are prefixed by the underscore character ("_"). A courteous rule label term, used for prioritization, is called a *tag*. SILK has recently been extended to incorporate rule *annotations*. An explicit tag is specified in a rule via an annotation which optionally begins the rule: `@[tag->G] H :- B ;` specifies a rule with tag `G`. The tag (`G`) may be strict (`silk:strict`), i.e., non-defeasible; if so, the "`tag->silk:`" may be omitted. An annotation can also be used to specify an explicit unique ID for the rule and/or other processing directives.

Clausal case: Omni rules generalize the permitted form of rule head formula. The basic case is a *clausal* omni rule; this has the form `@[tag->G] F ;` . Here, the body is empty. The head F is a disjunction of literals. Each such head literal is an atom or the `neg` of an atom; F is NAF-free (as usual in LP). E.g.:

```
@[tag->hi] wet(?Loc,today) or neg occur(rain,?Loc,yesterday) ;
```

The tag is optional. Outer universal quantification can be left implicit.

The semantics of omni rules is defined (and implemented currently) via a transformation. A clausal omni rule:

`@[tag->G]` L_1 `or` L_2 `or ... or` L_k `;`

— where each L_i is an atom or the neg of an atom, and $k > 0$ — is transformed into a set of k *directional* rules, one for each choice of head literal — i.e., one for each "contrapositive variant":

`@[tag->G]` L_1 `:- neg` L_2 `and ... and neg` L_k `;`

...

`@[tag->G]` L_i `:- neg` L_1 `and ... and neg` L_{i-1} `and`
`neg` L_{i+1} `and ... and neg` L_k `;`

...

`@[tag->G]` L_k `:- neg` L_1 `and ... and neg` L_{k-1} `;`

(Here, `neg neg` A is treated as equivalent to A, for any atom A.) This is called the set of *directional variant* rules, a.k.a. the *variant set*, for that clausal omni. The transformation of a clause into this set of directional rules is called its *directionalization*. E.g., the variant set for the omni above about wet and rain is:

```
@[tag->hi] wet(?Loc,today) :- occur(rain,?Loc,yesterday) ;
@[tag->hi] neg occur(rain,?Loc,yesterday) :- wet(?Loc,today) ;
```

General case: More generally, a omni rule may have a body and it may have a *universal FOL-formula* head, i.e., take the form

```
@[tag->G] HU :- B ;
```

Here, `B` is any hyper LP body (same as before the omni rules feature was introduced). The head `HU` is any syntactically FOL-looking formula that is outermost universally quantified. The quantifier-free subformula of `HU` may be formed syntactically from atoms using `neg`, `and`, and `or` freely composed. The *strong implication* connective `<==` is also permitted as a syntactic and conceptual convenience. `D <== E` is defined as equivalent to `D or neg E`. Likewise, `==>` and `<==>` are permitted as well. E.g., the rule above about wet and rain is equivalently reformulated as:

```
@[tag->hi] wet(?Loc,today) <== occur(rain,?Loc,yesterday) ;
```

Note that the head formula may use skolemization so as to approximately represent existentials, and thus be "nearly" full FOL in form. Moreover, the head formula may use the higher-order (HiLog) and frame syntax/features of hyper LP. Any universal formula `HU` is straightforwardly transformed (reduced) to a conjunctive set of clauses as in FOL [9]; let us write those as `HC1,...,HCn`. Accordingly, a omni rule of the above form is transformed in two steps. The first step transforms it into a set of clausal-head omni rules:

```
@[tag->G] HC1 :- B ;
...
```

```
@[tag->G] HCn :- B ;
```

In the second step, each of these clausal-head omni rules is transformed into its variant set, as described earlier except that `B` is also added to the body of each directional variant rule in each variant set. I.e., the i'th rule of the variant set shown above becomes:

@[tag->G] L_i :- neg L_1 and ... and neg L_{i-1} and
neg L_{i+1} and ... and neg L_k and B ;

When the omni rule's head is a single literal, both steps of the transformation reduce to identity. Thus the previous directional kind of LP rules are simply a special case of omni rules, in hyper LP. *One can thus view every rule in hyper LP as an omni rule.*

We have further developed a sophisticated version of Courteous, in which the second/clausal step of the directionalization transformation also outputs an *exclusion* statement used by the argumentation theory [2]. This Courteous version more powerfully utilizes prioritization in resolving multi-way (not just 2-way) conflicts. The exclusion statement generalizes the "mutex" aspect of the previous form of Courteous. However, to describe these deeper aspects of omni rules is beyond the scope of this paper.

Advantages for Authoring of rules: The courteous defaults feature of hyper LP enables users to author rules that mainly employ strong negation (`neg`) and priorities rather than negation-as-failure (`naf`). The omni-directional feature of hyper LP further enables users to author rules that mainly employ an "omni-directional" form of implication (`<==`) rather than the directional form of implication (`:-`) of LP rules. In this sense, omni rules abstract away the usual (i.e., previous in the LP literature) directionality of LP rules. They enable users to author FOL-style formulas, which behave omni-directionally but are nevertheless defeasible. Omni rules also allow logically-inclined users to use complex FOL formulas to represent rules in a compact, easily readable form. In this respect, hyper rules confer benefits similar to Lloyd-Topor transformations.

Figure 1 shows, in a SILK GUI [11], an example of directionalization: for the fictional business policy rule, "Only color full-page ads appear on the last page of the front section of an issue of the New York Times". This universal omni rule transforms into two clausal omni rules — one for the color ad literal and one for the full page size literal — that each in turn transform into a seven-member variant set of directional rules.

No "Reasoning by Cases": Despite permitting disjunction within the head, omni rules do NOT permit or perform "reasoning by cases" as FOL and Answer Set Programs (ASP) do. Omni rules thus do not incur the far greater computational complexity FOL and ASP have. Rather, the variant set in this regard behaves similarly to the unit/linear resolution strategy in FOL.

Sound Interoperability with FOL via Hypermonotonic Mapping: We have developed a novel approach for sound interchange with FOL, generalizing the method used in Description Logic Programs [10] and OWL-RL, based on the analysis that hyper LP has a tight relationship to FOL, akin to that for Horn LP.

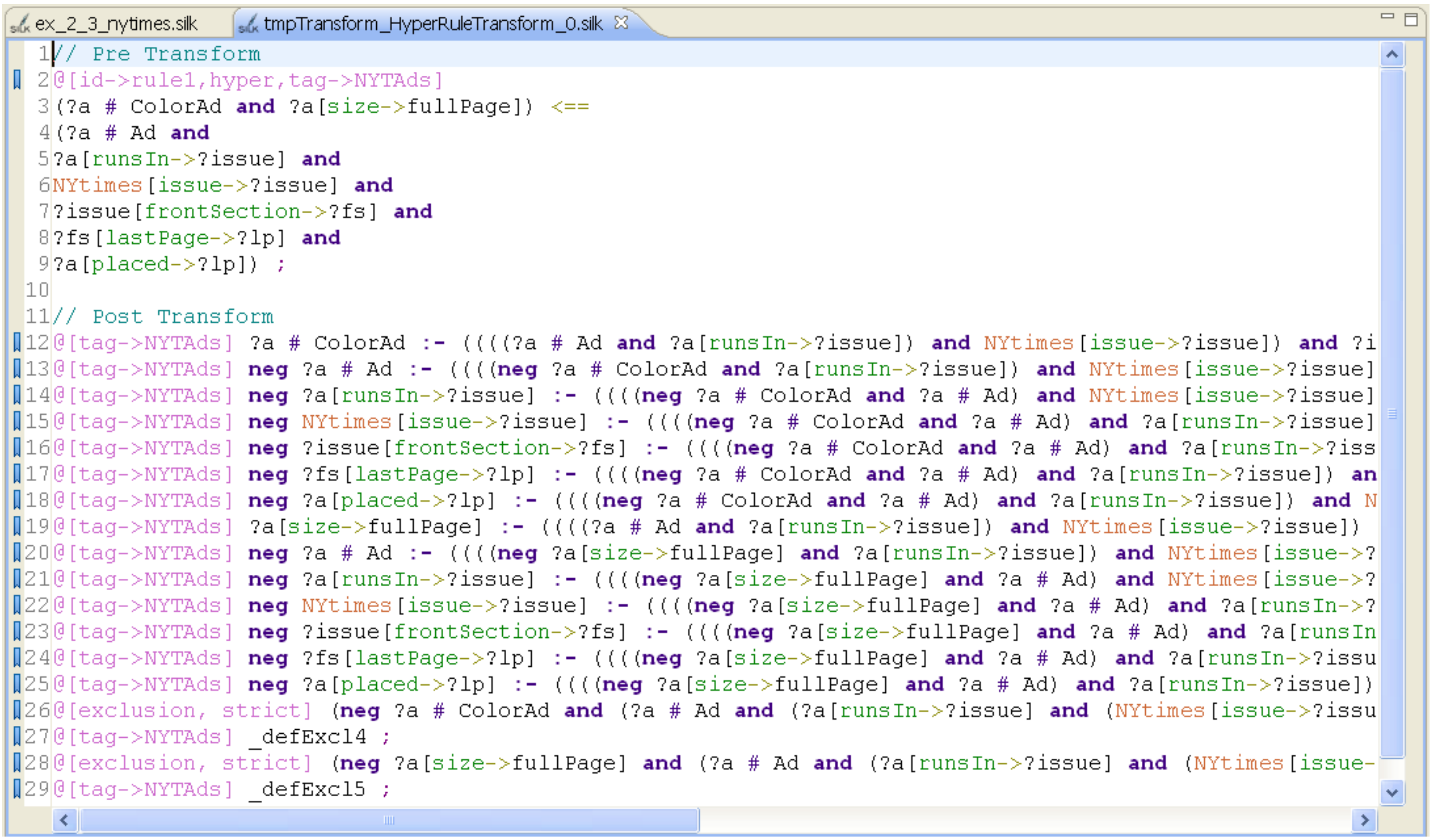

Fig. 1. Example of directionalization in a SILK GUI. Lines 12–18 and lines 19–26 are the two variant sets.

We can define this relationship via a *hypermonotonic* mapping T that consists of a pair of mappings $(T1, T2)$, one for each interchange direction. $T1$ maps a hyper LP rule into a universal FOL axiom. It replaces `:-` by `<==`, and ignores the tag. I.e., the hyper LP rule `@[tag->G] H :- B ;` is mapped into the FOL axiom `H <== B ;`. Certain non-onerous syntactic restrictions apply, since some expressive constructs in hyper LP have no correspondent in FOL. Thus the hyper LP rule must not contain `naf` nor the other body-only constructs that expressively essentially rely upon `naf`, including: aggregates, Lloyd-Topor forall, and Lloyd-Topor implies. And it must not contain external actions (effector literals). $T1$ maps a (true) hyper LP conclusion into a FOL axiom with the same formula.

$T2$ maps a universal FOL axiom `F;` into the hyper LP omni rule `F;` that has the same formula as head and an empty body.

Interestingly, from a FOL viewpoint, entailment in hyper LP is *sound* — even though hyper LP is *nonmonotonic!* — and is *incomplete*, e.g., it lacks "reasoning by cases". Thus we say (restricted) hyper LP is *FOL-sound* with respect to the interchange mapping T. *When there is conflict, the incompleteness is desirable!* Hyper LP entailment/reasoning is then usefully selective, unlike FOL which entails all sentences.

One usage for interchange is to import universal FOL (e.g., clauses) into hyper LP. One can assign prioritization to the imported rules, e.g., based on source authority, recency, or reliability. A second usage for interchange is to import hyper LP conclusions into FOL, e.g., in the conflict-free case.

A well-known special case of such interchange is between definite Horn LP and definite Horn FOL, especially when function-free. In this case, in the associated variant set exactly one directional clause has a positive head; the other rules in the variant set can-be/are essentially ignored. And in this case, there is completeness, not just soundness, of the LP with respect to the set of entailed ground atoms. This special case relationship is the KR foundation used in most implementations of Horn *FOL* reasoning — the typically used techniques essentially do *LP* reasoning — e.g., currently for OWL-RL, RDFS, RIF-BLD whose semantics are all based essentially on Horn FOL.

The hypermonotonic mapping T greatly generalizes that approach. It handles strong negation (`neg`), and any clause not just Horn. When skolemization is employed, it can in effect expressively cover much of FOL — thus much of OWL, Common Logic [12], and SBVR [13].

Overcoming FOL's Brittleness: Hyper LP can handle conflict robustly, always producing a consistent set of conclusions. By contrast, FOL is a "bubble" — it is *perfectly* brittle semantically in the face of contradictions from quality problems or merging conflicts. Any contradiction is totally contagious — the conclusions all become garbage. FOL's extreme sensitivity to conflict limits its scalability in the number of axioms and the number of merges. The expressive opportunity arises for conflict essentially as soon as strong negation (`neg`) is introduced into a focal subset of FOL. For example, OWL beyond -RL and Common Logic both suffer from this problem.

A Very Large KB (VLKB) with a million or billion axioms formed by merging from multiple web sources is unlikely to have *zero* KB/KA conflicts from: human knowledge entry/editing; implicit context and cross-source ontology interpretation; updating cross-source; and source trustworthiness. Hyper LP's approach provides a *critical* advantage for KB scalability — *semantically* as well as computationally.

3 RIF SILK Dialect

Using RIF's Framework for Logic Dialects (FLD), [14], we have defined a RIF dialect, called RIF-SILK [15], to facilitate the exchange of rules employing SILK's advanced expressive features. RIF-SILK is a strict superset of RIF Basic Logic Dialect (BLD) that adds higher-order (HiLog, reification), prioritized defaults (Courteous, neg, omni-directionality), well founded NAF, reactiveness (external actions), and some other (expressive) features. RIF-SILK also shares a number of features with other RIF dialects. These include frame/object-oriented syntax as in F-logic [6], internationalized resource identifiers (IRIs) [16] as identifiers for concepts, and XML Schema datatypes [17]. Our implementation of RIF-SILK is as an import/export module to the SILK engine.

Like RIF Production Rule Dialect (PRD), RIF-SILK provides procedural attachments for external actions, etc. Unlike RIF-PRD, RIF-SILK provides a declarative semantics — and further adds higher-order, defaults, and a number of other features.

The RIF Core Answer Set Programming Dialect (RIF-CASPD) [18] was proposed recently. Like RIF-SILK, RIF-CASPD can express both NAF and strong negation (neg) — and its treatment of NAF coincides with RIF-SILK's for the stratified case. Unlike RIF-SILK, RIF-CASPD supports "reasoning by cases" (head disjunction) — thus is far more computationally complex — but lacks higher-order, priorities, functions, and equality.

4 Conclusions

Contributions of this work were summarized in the Abstract, the end of section 2, and the beginning of section 3.

Next, we discuss the broad potential applicability of SILK's core KR technology approach, and give future directions for the core technology itself.

SILK inherits a variety of use cases from previous work on courteous LP defaults, situated/production LP actions and events, and webized information integration — including in RuleML and RIF. There are many potential applications in business and government for SILK's reactive higher-order defaults expressiveness. Horizontally, these include: policies and policy-based workflows (monitoring, trust); and ontology mapping/mediation and knowledge integration. Vertically, these include: e-commerce (shopping & advertising, contracts, customer care); defense (intelligence, operations); finance (reporting, regulatory compliance); biomed (pharma, e-science, clinical records and guidance, insurance); and mobile (personalization).

Directions for future KR work include more interchange languages, probabilistic knowledge, and parallelization of inferencing.

Acknowledgements: Thanks to all of the SILK team, including Paul Haley for suggesting the example newspaper ad policy rule and Brett Benyo for help implementing it.

References

1. Grosof, B.N.: Representing E-Commerce Rules via Situated Courteous Logic Programs in RuleML. Electronic Commerce Research and Applications **3**(1) (2004)
2. Wan, H., Grosof, B., Kifer, M., et al.: Logic Programming with Defaults and Argumentation Theories. In: Proc. 25th Intl. Conf. on Logic Programming. (2009)
3. Chen, W., Kifer, M., Warren, D.: HiLog: A foundation for higher-order logic programming. Journal of Logic Programming **15**(3) (February 1993) 187–230
4. Boley, H., Kifer, M.: RIF Basic Logic Dialect. W3C Recommendation 22 June 2010, `http://www.w3.org/TR/2010/REC-rif-bld-20100622/`
5. The Rule Markup Initiative: Rule Markup Language (RuleML). `http://ruleml.org/`
6. Kifer, M., Lausen, G., Wu, J.: Logical Foundations of Object-Oriented and Frame-Based Languages. Journal of ACM **42** (July 1995) 741–843
7. Przymusinski, T.: Well-founded and Stationary Models of Logic Programs. Annals of Mathematics and Artificial Intelligence **12**(3) (1994) 141–187
8. Gunning, D., Chaudhri, V.K., et al.: Project Halo Update: Progress Toward Digital Aristotle. AI Magazine to appear, Fall 2010.
9. Tseitin, G.: On the complexity of derivation in propositional calculus. Studies in constructive mathematics and mathematical logic **2**(115-125) (1968) 10–13
10. Grosof, B.N., Horrocks, I., Decker, S., Volz, R.: Description Logic Programs: Combining Logic Programs with Description Logic. In: Proc. WWW-2003 Conf. (2003)
11. Grosof, B., Burstein, M., Dean, M., Andersen, C., Benyo, B., Ferguson, W., Inclezan, D., Shapiro, R.: A SILK Graphical UI for Defeasible Reasoning, with a Biology Causal Process Example. In: Proc. 4th Intl. Web Rule Symp. (RuleML). (2010)
12. ISO: ISO/IEC 24707:2007 Information technoogy - Common Logic (CL): a framework for a family of logic-based languages (2007)
13. Object Management Group: Semantics of Business Vocabulary and Business Rules (SBVR) (January 2008) `http://www.omg.org/spec/SBVR/1.0/`.
14. Boley, H., Kifer, M.: RIF Framework for Logic Dialects. W3C Recommendation 22 June 2010, `http://www.w3.org/TR/2010/REC-rif-fld-20100622/`
15. Sherman, J., Dean, M.: RIF SILK Dialect. `http://silk.semwebcentral.org/RIF-SILK.html`
16. Duerst, M., Suignard, M.: Internationalized Resource Identifiers (IRIs). IETF RFC 3987, `http://www.ietf.org/rfc/rfc3987`
17. Biron, P.V., Malhotra, A.: XML Schema Part 2: Datatypes Second Edition. W3C Recommendation 28 October 2004, `http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/`
18. Heymans, S., Kifer, M.: RIF Core Answer Set Programming Dialect. `http://ruleml.org/rif/RIF-CASPD.html`