May ASKE Milestone 7 Report for AMIDOL

Eric Davis[1], Alec Theriault[1], and Ryan Wright[1]

[1]Galois, Inc

1 Introduction

In this report we discuss recent extensions to AMIDOL's framework which seek to extend AMIDOL's Abstract Knowledge Layer and Formulation and Paletter generation capabilities, and briefly discuss extensions to AMIDOL's ability to work with additional intermediate representations at the Structured Knowledge Layer, and support for additional backends for its Executable Knowledge Layer.

2 Formulations

AMIDOL's Abstract Knowledge Layer focuses on the representation, manipulation, and extension of problem formulations as part of the metamodeling process.

Definition 2.1. Formulation A formulation is a high-level domain model which represents and encodes semantic domain knowledge about a complex system. Formulations are the model-stack equivalent of a high level language and should focus on being accessible to domain scientists. While formal semantics may be implied by a formulation, the primary point of formulating a model is not writing down executable code or mathematical representations. It is writing down assumptions and knowledge which can, at a later stage, be used to infer these properties of a model.

Formulations should be represented in a form that is close to natural language, or natural representations, such as diagrams, used by domain scientists.

Formulations define the Abstract Knowledge Layer of the modeling stack.

The primary mechanism used for formulations in AMIDOL are Visual Domain Specific Ontological Languages (VDSOLs). A VDSOL is defined by a Formulation Palette,

Definition 2.2. Formulation Palette A formulation palette is a set $P = \{p_0, p_1, \ldots\}$ of palette elements.

Definition 2.3. Palette Element A palette element is a type which can be instantiated in a formulation as a concrete occurance of the element. These instance elements must have unique names, and can be connected with arcs to define structured co-spans of instance elements which can be compiled into AMIDOL's intermediate representation.

A palette element is defined by the combination of a palette template T, and a definition in AMIDOL's intermediate representation R into the tuple (T, R)

An instance element is a palette element, combined with a unique identifier N and a set of parameters and constants C, (T, R, N, C).

Definition 2.4. Palette Template A palette template is a category defined by an input set, output set, and measure set, over an unknown or undefined AMIDOL IR. Palette templates are essentially container categories for compatible AMIDOL IR models whose input and output cardinality and typing match, making them interchangable.

A Formulation in AMIDOL is a set of palette element instances composed together by a set of relations, $A \rightarrow B$ such that for the output cardinality of A and the input cardinality of B are the same, and have compatible types. Type relations allow restrictions on model composition which are richer than input/output cardinality; such as that used by all current AMIDOL palettes which states if A is a noun, B must be a verb, and if A is a verb, B must be a noun. Further restrictions could be used to apply types to individual inputs or outputs to restrict their composability to.

2.1 Formulation Paradigm

The process of model formulation in AMIDOL seeks to allow domain experts to construct meta-models in a novel way, using VDSOLs. These VDSOLs utilize an underlying intermediate abstract representation to give formal meaning to the intuitive process diagrams scientists and domain experts normally create. The intention is to remove the burden of having to code models explicitly, and enable domain experts to build models of complex systems which are easier to maintain, validate, and verify, and avoid common pitfalls of monolithic and hand-coded implementations.

While to date these formulations have all been through the use of diagrams, we are extending AMIDOL to support other types of formal languages that are natural for certain domains.

2.2 Formulation Languages

To explore other types of visual languages and diagrams, we are extending AMIDOL to support systems of stochastic chemical equations via standard notations for networks of chemical reactions of the form

$$A + B \xrightarrow{\beta} C$$

which is interpreted as "a molecule of A combines with a molecule of B to form a molecule of C with propensity β". We use the following formal grammar to represent stochastic chemical equations in AMIDOL as a formulation:

```
start   : line+

line    : equation
    | comment

comment   : "#" ANY

equation  : forwardequation
    | backwardequation
    | bothequation

forwardequation : rate "," reactants FORWARD_ARROW reactants
backwardequation: rate "," reactants BACKWARD_ARROW reactants
bothequation  : rates "," reactants BOTH_ARROW reactants

rates   : "(" rates ")"
    | forwardrate "," backwardrate
    | bothrate

```

```
forwardrate : rate
backwardrate  : rate
bothrate  : rate

rate    : "(" rate ")"
    | FLOAT
    | INT

reactants : reactants "+" reactant
    | reactant

reactant  : INT name
    | name

name    : POLAR
    | NONPOLAR

FORWARD_ARROW : /-->|==>|->|=>|>/
BACKWARD_ARROW  : /<--|<==|<-|<=|</
BOTH_ARROW  : /<-->|<==>|<->|<=>|<>/
FLOAT   : /\d+\.\d+/
INT   : /\d+/
NONPOLAR  : /[a-zA-Z_][a-zA-Z0-9_]*/
POLAR   : /[a-zA-Z_][a-zA-Z0-9_]*[+-]?/
WHITESPACE  : /[ \t\n\r\f\v]+/
ANY   : /[\d\w \t\r\f\v]+/

%ignore WHITESPACE
```

Listing 1: Stochastic Chemical Equation Grammar

AMIDOL is capable of reading networks of chemical reactions, like the following:

```
1.0,       A        -> B
2.0,       A + B <--> C + D
3.0, 4.0, A + B <--> C + D
```

Listing 2: Example Stochastic Chemical Equation

and generating an abstract syntax tree like that shown in Figure 1. This abstract syntax tree can then be used to generate a model in the AMIDOL IR, representing each reactant with a state variable, and constructing events from the equations.

While designed to test the expressive capability of AMIDOL's Abstract Knowledge Layer in representing chemical equations, it can also be used to represent other models, like the SIR model, as follows:

```
beta / (S + I + R), S + I --> I
gamma,              I --> R
```

Listing 3: Example Stochastic Chemical Equation

3 Formulations from Models

In order to support a joint demo with GTRI and the University of Arizona, we have also been focusing on developing methods which allow us to automatically generate formulations in a visual language from models at the Structured Knowledge Layer. To support this, we have constructed a

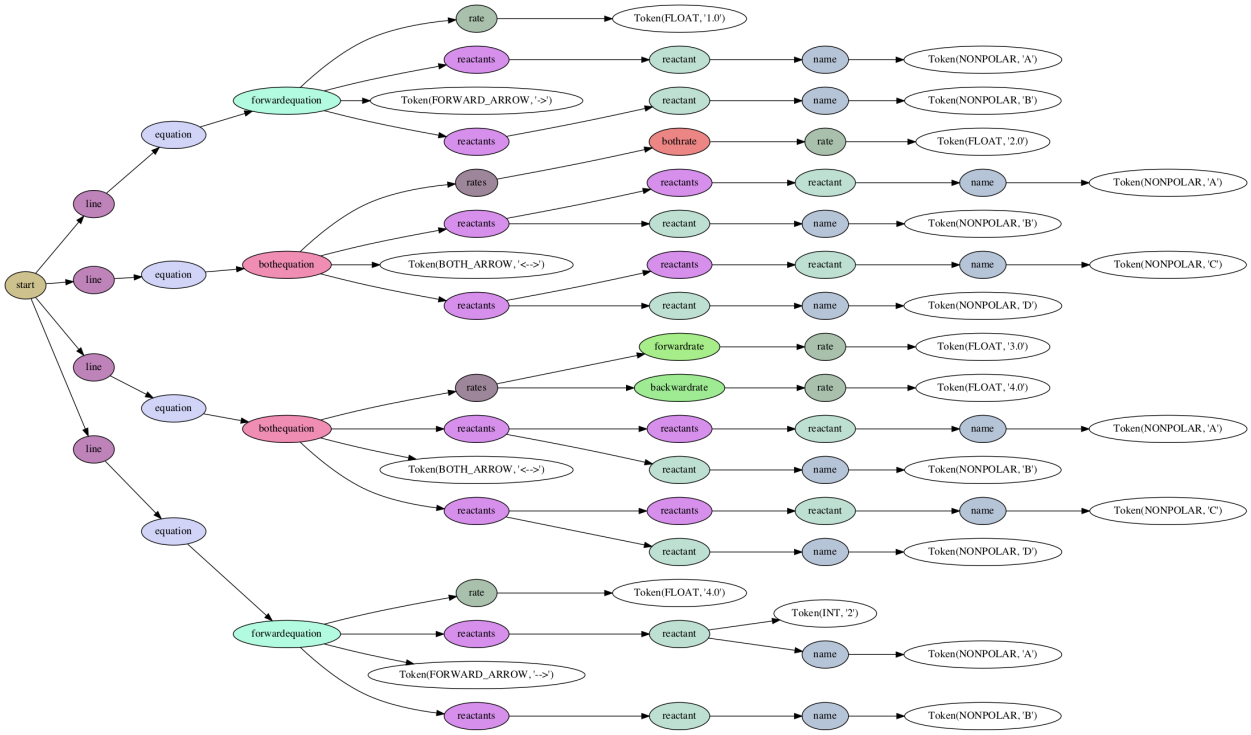

Figure 1: Example Parse Tree from Stochastic Chemical Grammar

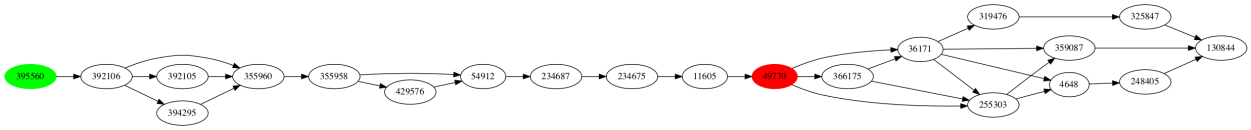

Figure 2: Ontology search for viral template from grounding “Influenza A virus subtype H3N2v”

new module for AMIDOL which is able to translate Julia abstract syntax trees into AMIDOL’s IR. These models are combined with grounding information to perform the novel process of Formulation Inference in AMIDOL.

3.1 Formulation Inference

Formulation Inference is a formal procedure in AMIDOL for automatically generating new palette elements from existing palette templates, using an ontology of domain specific terms.

Figure 2 shows a subset of the SNOMED CT ontology, representing elements of the ontology with nodes labeled with their SCTID. In this example, the green node represents the term “Influenza A virus subtype H3N2v” while the red node represents the term “Virus (organism)”. Formulation inference assumes the existance of an annotated ontology, with nodes containing palette templates as annotations. In this example the “Virus (organism)” node has been annotated with the palette template for a viral infection verb.

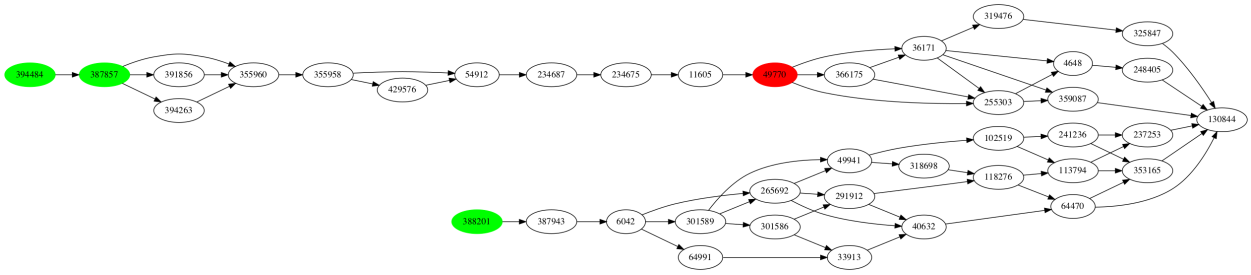

Figure 3: Ontology search for viral template from grounding "Influenza A (H1N1"

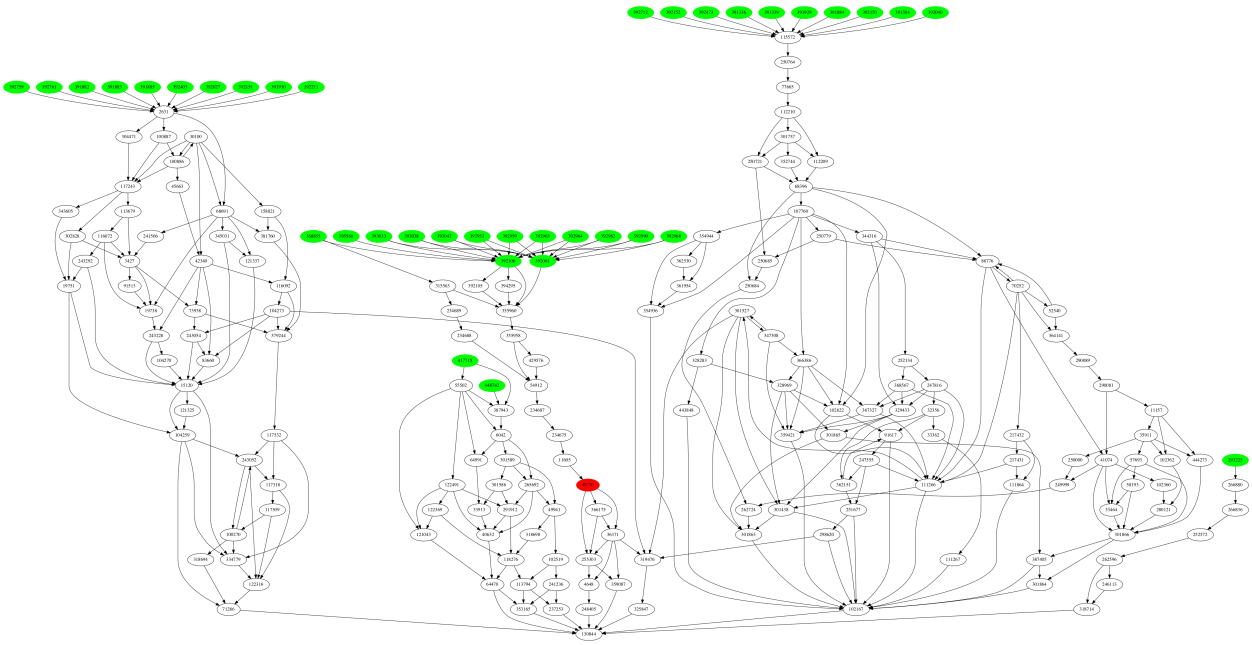

Figure 4: Ontology search for viral template from grounding "H3N2"

4 The SNOMED Ontologies

5 Examples

* Updated Diagram * Formulation vs. Model vs. Implementation * Formulation paradigm * Formulation languages * Julia AST -¿ IR * Snomed Ontology Summary * Palette Generation from Templates * Category Theory * Templating * Snomed and Palette Generation

6 Other Recent AMIDOL Extensions

7 Resources, web sites, etc.

The current AMIDOL source code, including example models and documentation, is available at the AMIDOL Github site `https://github.com/GaloisInc/AMIDOL`.

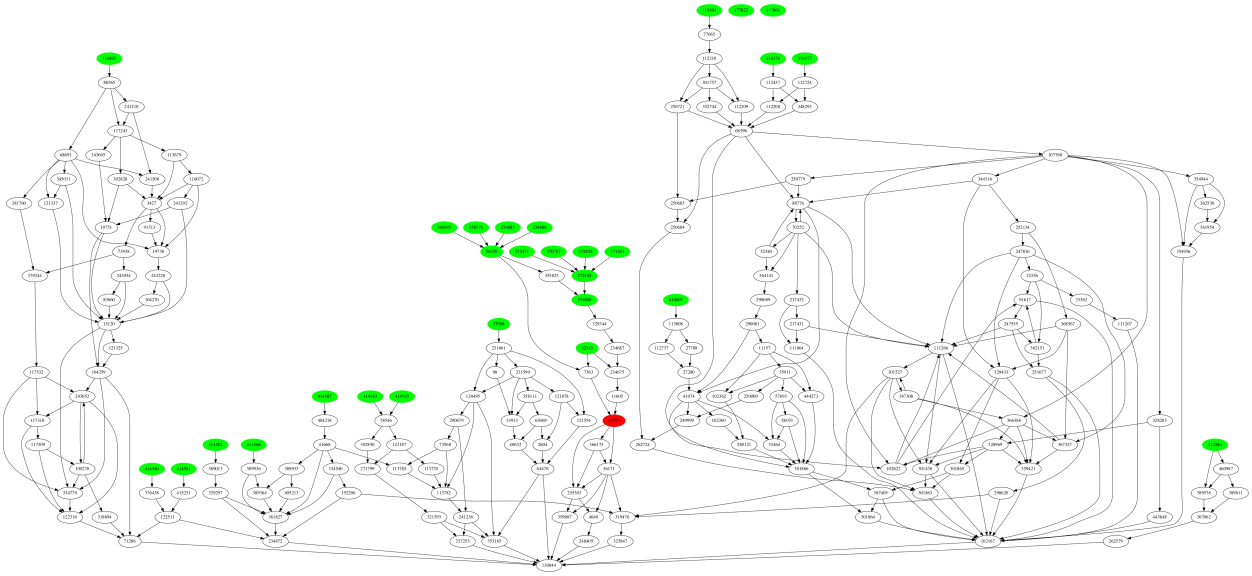

Figure 5: Ontology search for viral template from grounding “Ebola”