Mathematical Computations for Linked Data Applications with OpenMath

Ken Wenzel and Heiner Reinhardt

Fraunhofer-Institute for Machine Tools and Forming Technology IWU,
Chemnitz, Germany
`ken.wenzel@iwu.fraunhofer.de`

Abstract

Linked data can be used to connect information stemming from usually disparate sources. The Semantic Web combines various kinds of logical reasoning for the inference of additional knowledge from existing data and for consistency checking of data sets. Although mathematics is crucial for most areas where Semantic Web technologies can be applied, support for them is lacking from related standards and tools. This paper presents an approach for integrating OpenMath with RDF data for the representation of mathematical relationships and the integration of mathematical computations into reasoning systems.

1 Introduction

Mathematical relationships are ubiquitous within data sets of elements with numerical properties. The data can be as simple as a list of persons with age and height data or as complex as a mathematical model that uses equations to describe a physical process. Both cases can benefit from using linked data principles [2] to combine multiple data sets or to integrate additional information from external data sources. Besides, there is also already a considerable amount of data available as Linked Open Data[1] that could benefit from mathematical support, e.g. for statistical computations [6].

In our case, we are using linked data principles to connect models of production systems and processes with operating data, e.g. for analyzing the energy performance of production facilities [14]. The related performance indicators (like average energy consumption or energy consumption per part) should be automatically computed under consideration of different calculation models depending on the machine and product types. Another use case are distributed component libraries for process planning of manufacturing operations [4], where descriptions of machines and their components are directly fetched from the web (e.g. from a vendors website), instead of manually building local repositories. The latter is especially interesting if vendors are enabled to supply formulas within their product descriptions to compute properties like power consumption or life-cycle costs for a given operating context.

Although an integration of mathematical computations into linked data environments seems promising, there are only few approaches that try to combine both worlds.

A simple approach was proposed by Sánchez-Macián et al. [11] for extending SWRL with functions for the evaluation of mathematical expressions. They give a rough overview on how OpenMath XML expressions can be embedded into ontologies and used by custom SWRL functions for reasoning. These functions rely on SWRL to collect the parameters required for evaluating the mathematical expressions.

[1]`http://linkeddata.org/`

A comparable solution could also be realized with SPIN (SPARQL Inferencing Notation)[2] that allows to use SPARQL[3] queries for reasoning and constraint checking over RDF data. The upcoming SPARQL 1.1[4] also supports aggregates, which can be used for, e.g. computing the sum or average of a set of elements. However, SPARQL only support simple mathematical operations and therefore an extension with custom functions would be necessary. Furthermore, expressing rules using SPIN may get complicated for domain experts since mathematical terms have to be nested into SPARQL queries.

Another approach is followed by OntoCAPE [8] and OntoModel [13]. These methods define means for sharing mathematical models in a collaborative environment by using ontologies. While OntoCAPE provides vocabulary for describing equation systems with OWL and RDF [9], OntoModel rather focuses on amending a system of equations expressed in Content MathML with additional meta data. The two approaches are purpose built for representing and solving equation systems with a fixed set of input variables whose values need to be determined, e.g. by a user or a software agent, before running the computation.

A general review on representation of mathematical knowledge on the web, which covers a wide range of possible representation formats and systems, is given by Lange [7]. In conclusion of his review, there are no existing systems available that combine Semantic Web technologies and computer algebra systems for mathematical computations or proof.

Our approach realizes a tight integration between OpenMath and RDF[5] for consuming linked data from computer algebra systems. Complemented by an RDF representation for mathematical expressions it not only allows to publish formulas as linked data but also to extend reasoning systems with inferencing capabilities based on mathematical computations.

In Section 2 an OpenMath content dictionary for retrieval and representation of RDF data and corresponding human-friendly notation are introduced. Section 3 describes an OWL ontology for OpenMath objects that enables serialization as RDF. Based on the presented formalisms a method and architecture for math-enhanced inference is introduced in Section 4.

If not stated otherwise then all examples concerning OpenMath objects are given in the Popcorn[6] [5] OpenMath representation. For reasons of simplicity, we use the publicly accessible Friend of a Friend (FOAF)[7] vocabulary to provide linked data examples in RDF that are presented in this paper by using the Turtle syntax[8].

2 Query RDF data from OpenMath

Existing methods for representing and solving equation systems with RDF and OWL [8, 13] as well as the integration of OpenMath with SWRL [11] use a two-step approach where first the required input values are retrieved and then the computation is explicitly executed using them as parameters. By directly integrating RDF query support into OpenMath, the description of mathematical relationships over linked data and also the reuse of related formulas can be simplified. For this reason, we have created an OpenMath content dictionary for RDF[9] that defines symbols to access linked data from computer algebra systems. The main purpose of this

[2]`http://spinrdf.org/`
[3]`http://www.w3.org/TR/rdf-sparql-query/`
[4]`http://www.w3.org/TR/sparql11-query/`
[5]`http://www.w3.org/TR/rdf-concepts/`
[6]`http://java.symcomp.org/FormalPopcorn.html`
[7]`http://xmlns.com/foaf/spec/`
[8]`http://www.w3.org/TR/turtle/`
[9]`http://www.openmath.org/cd/contrib/cd/rdf.xhtml`

CD is the retrieval of RDF resources and their associated properties along with a representation for RDF resources and literals.

2.1 Symbols for RDF retrieval

The following four symbols are provided for retrieving data from an RDF store:

resourceset A unary construction function that takes one string argument that is a Manchester Syntax description[10]. These expressions are a subset of the Manchester Syntax for the definition of OWL classes and restrictions. The result of applying the resourceset function is a set of resource objects.

Example: `rdf.resourceset("foaf:Person and foaf:age some xsd:int[>18]")`

resource An unary construction function with one string argument representing an IRI reference as defined by SPARQL[11]. Possible values are prefixed names in the form `"prefix:resourceName"` or absolute IRIs in the form `"<IRI>"`.

Example: `rdf.resource("<http://example.org/p#Alice>")`

valueset A function to retrieve a set of property values for a specific RDF resource. It takes two arguments where the first argument denotes an RDF property as IRI reference and the second argument an OM object that corresponds to an RDF resource. The result of this function is a set of RDF resources or literal values.

Example:

```
$alice := rdf.resource("<http://example.org/p#Alice>");
rdf.valueset("foaf:knows", $alice)
```

value Works like the valueset function but expects and returns exactly one value. The application of this function results in an error if none or more than one value exists.

Example:

```
rdf.value("foaf:age", rdf.resource("<http://example.org/p#Alice>"))
```

Although it would be possible to directly embed SPARQL queries into OpenMath objects, we haven chosen Manchester Syntax descriptions for querying RDF resources. The main advantage of this approach is type-safety. While SPARQL `SELECT` queries may result in tuples of different cardinality and type (either resource or literal), the result of a Manchester Syntax description is always a set of resources.

Since Manchester Syntax is a concise language for OWL ontologies, it essentially is based on description logic (DL). This is beneficial for fast query evaluation, which can also be directly executed by a DL reasoner, but does not allow the use of variables and hence has limited expressiveness in comparison to SPARQL. Please note that Manchester Syntax descriptions can also be directly translated to SPARQL queries [12] and consequently do not require the use of OWL ontologies for the description of datasets.

We also propose to reuse the vocabulary of the `set1` content dictionary to handle instances of `rdf.resourceset` and `rdf.valueset`. This allows for applying existing `set1` functions like `union`, `intersect`, `in` or `size` for basic set operations.

[10]http://www.w3.org/TR/owl2-manchester-syntax/#Descriptions
[11]http://www.w3.org/TR/sparql11-query/#rIRIref

2.2 Representation of literal values

The representation of RDF literals in OpenMath is straightforward. Typed RDF literals are mapped according to their datatype to basic OpenMath objects of type integer, IEEE floating point number, character string or byte array. If the literal's type is not directly convertible or should be preserved in OpenMath then the *literal_ type* attribute can be used to annotate the corresponding OpenMath object. Plain literals are represented as strings and can be annotated with a language tag by using the attribute *literal_ lang*.

2.3 Shortening of IRI references

The functions introduced in Section 2.1 allow the specification of IRI references in a shortened form as so-called prefixed names. For example, the expression

```
rdf.resource("foaf:Person")
```

uses the prefixed name *foaf:Person* that expands to a full IRI by concatenating the namespace IRI referred to by the prefix `"foaf"` with the local part `"Person"` resulting in the full IRI `<http://xmlns.com/foaf/0.1/Person>`.

Mappings between prefixes and their corresponding namespaces can be specified by using the *prefixes* attribute in combination with the *prefix* symbol:

```
rdf.resourceset("foaf:Person and foaf:age some rdfs:Literal"){
  rdf.prefixes -> {
    rdf.prefix("rdfs", "http://www.w3.org/2000/01/rdf-schema#"),
    rdf.prefix("foaf", "http://xmlns.com/foaf/0.1/")
  }
}
```

These prefix declarations are scoped to the annotated OpenMath object including its descendants. Please note that annotations in OpenMath do not automatically propagate to children and therefore an implementor has to take care of the correct propagation of prefix declarations.

2.4 User-friendly input with Popcorn

We have extended the Popcorn notation with additional symbols for concise integration of RDF into OpenMath. These RDF expressions have the form:

```
'@' '@'? PROPERTY_REF? ('(' OM_OBJECT ')' | '[' CLASS_REF ']')
```

Basically, the character `@` introduces references to RDF data. A single `@` as prefix indicates that the result is also only a single value whereas `@@` indicates that the result is a set of zero or more values.

The following examples demonstrate the mapping of symbols from the content dictionary for RDF to compact Popcorn expressions:

```
rdf.resourceset("foaf:Person")              → @@[foaf:Person]
rdf.resource("<http://example.org/Alice>")  → @(<http://example.org/Alice>)
$alice := rdf.resource("example:Alice")     → $alice := @(example:Alice)
rdf.valueset("foaf:knows", $alice)          → @@foaf:knows($alice)
rdf.value("foaf:age", $alice)               → @foaf:age($alice)
```

2.5 Integration into computer algebra systems

As with every new content dictionary for OpenMath, the related phrase-books for computer algebra systems (CAS) need to be extended with additional translation rules. Since, to the authors' knowledge, no current CAS supports querying of RDF stores, additional effort for extending existing systems is required.

The implementation of the functions `resource`, `valueset` and `value` is rather simple. The symbol `resource` just defines a special constant and the latter execute basic retrieval operations against an RDF store.

Implementing the symbol `resourceset` needs considerably more work since parsing of Manchester Syntax and transformation to SPARQL (or OWL for DL reasoning) is required. Possible solutions may use extended SPARQL dialects like Terp [12]. Another solution that directly converts the Manchester Syntax descriptions to OWL for leveraging existing DL reasoners is proposed in the following section.

In order to support the `set1` content dictionary, a simple approach would be to first retrieve the data objects and then execute the set operation within the computer algebra system.

However, we suggest a mapping to compound SPARQL expressions that efficiently execute the `set1` operations `union`, `intersect`, `in` or `size` for large-scale RDF data.

3 Representing OpenMath objects as linked data

The content dictionary for RDF as introduced by Section 2 provides sufficient support for integrating linked data with OpenMath. However, a slight weakness of this representation stems from the usage of string constants for IRI references. These make it hard to track referenced classes or properties and to update them if the associated RDF resources are renamed.

A plain RDF encoding of OpenMath objects overcomes these shortcomings and bears additional advantages. Thus, established mechanisms for annotating and linking of RDF data can be used with OpenMath, and furthermore, IRI references and Manchester Syntax descriptions can be directly encoded as RDF (Section 3.2).

3.1 An ontology for OpenMath

The ontology denoted by Figure 1[12] is basically a verbatim mapping of the OpenMath standard to OWL. In contrast to the approach proposed by Robbins [10] for the encoding of Strict Content MathML in RDF, we decided to stay as close as possible to the OpenMath XML encoding to simplify the conversion between both representation formats.

The only difference in comparison to OpenMath XML is that we have removed the `OMR` element that encodes explicit references to other OpenMath objects. These references are unnecessary since RDF itself provides the means for cross-referencing between resources.

3.2 Referencing of RDF data

We propose to use a special encoding for functions from the content dictionary for RDF (Section 2.1). Normally, these functions take string arguments for IRI references or Manchester Syntax descriptions. But when using the RDF encoding of OpenMath objects these arguments can directly be represented as resource IRIs or `owl:Restrictions`.

For example, the OpenMath object

[12]The ontology is available at `http://numerateweb.org/vocab/math`.

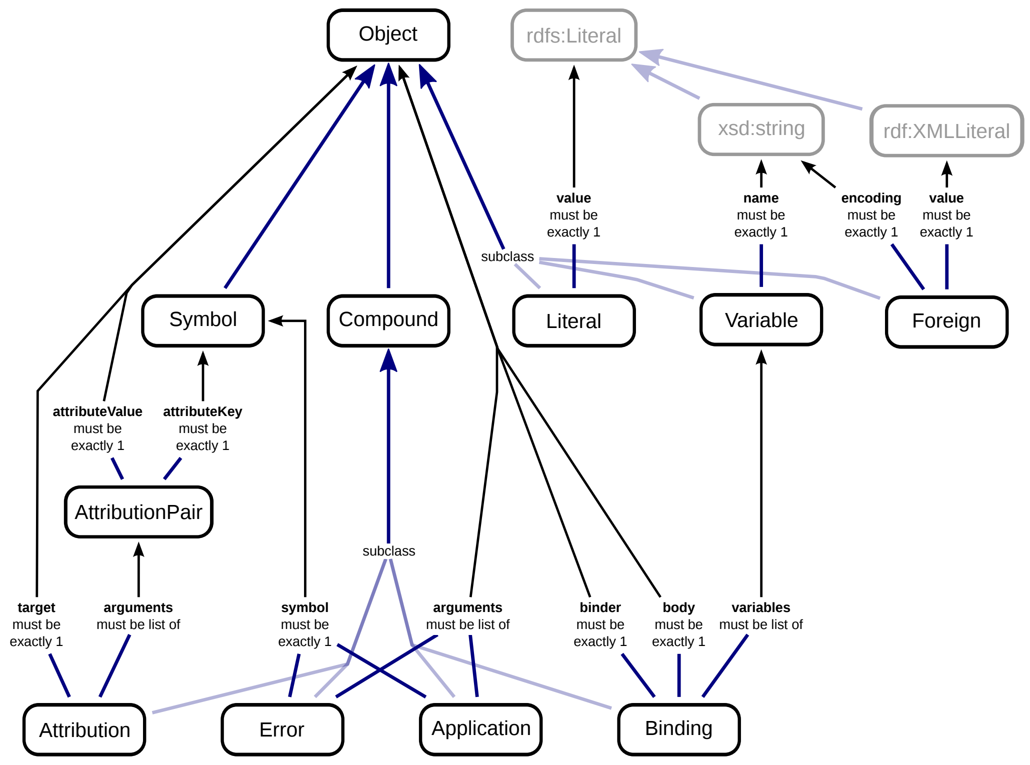

Figure 1: Ontology for OpenMath objects.

```
rdf.resourceset("foaf:Person and foaf:age some xsd:int[>18]")
```

can be encoded in RDF as

```
[ a om:Application; om:symbol <http://www.openmath.org/cd/rdf#resourceset>;
  om:arguments ("foaf:Person and foaf:age some xsd:int[>18]") ] .
```

or when using the native RDF encoding for arguments as

```
[ a om:Application; om:symbol <http://www.openmath.org/cd/rdf#resourceset>;
  om:arguments ([ owl:intersectionOf (
    foaf:Person
    [ a owl:Restriction; owl:onProperty foaf:age;
      owl:someValuesFrom [ a rdfs:DataType; owl:onDataType xsd:int;
        owl:withRestrictions (xsd:minExclusive "18"^^xsd:int) ]
    ]
  )])
] .
```

The latter form represents the Manchester Syntax expression directly as an `owl:Class` description. This encoding allows to use OWL reasoners for determining the contents of the `rdf.resourceset`.

3.3 Extending OpenMath by using RDF and OWL

The ontology as defined in Section 3 enables the representation of OpenMath objects as RDF for sharing mathematical expressions as linked data and for their interoperability with other information serialized as RDF (like `owl:Restrictions`). Together with the content dictionary for RDF as introduced by Section 3.2, it enables the creation of links between OpenMath objects and RDF resources and therefore solves the linking issue mentioned by [6].

Besides this, the ontology does not yet capture the vocabulary for the definition of content dictionaries with symbols and their properties (roles, CMPs, FMPs, etc.). But nonetheless, we can use the ontology to define new OpenMath symbols and relate them to existing ones as discussed by [3]. For example, the following RDF statements define the symbol `asymp1.Landauin` as a subclass of `set1.in`:

```
<http://www.openmath.org/cd/asymp1#Landauin>
  a om:Symbol; rdfs:subClassOf <http://www.openmath.org/cd/set1#in> .
```

In this case, the metamodeling capabilities of OWL 2[13] allow us to treat `asymp1.Landauin`[14] and `set1.in` as OpenMath symbols and as OWL classes at the same time.

The OpenMath ontology can also be extended by additional subclasses of `om:Symbol` (e.g. BinderSymbol, AttributionSymbol, ApplicationSymbol) to enable modeling of symbol roles. These roles can then be applied to symbols by using `rdf:type`:

```
asymp1:Landauin rdf:type om:ApplicationSymbol .
```

If additionally `om:hasCMP` and `om:hasFMP` are introduced as properties of `om:Symbol` then the definition of OpenMath content dictionaries in a distributed way (as linked data) gets possible:

```
set1:emptyset a om:ConstantSymbol;
  om:hasCMP "The intersection of A with the emptyset is the emptyset";
  om:hasFMP [ a om:Application; om:symbol relation1:eq;
    om:arguments (
      # omitted for simplification
    )
  ] .
```

4 Reasoning

Based on the content dictionary for RDF and the OpenMath RDF encoding this section describes how mathematical relationships can be expressed and embedded into OWL ontologies. Furthermore an architecture for respective rule processing is proposed.

4.1 Rule definition

For a human-readable definition of mathematical relationships in an ontology we use the Popcorn syntax and propose the following extension:

```
[Class] : @Property = Expression
```

[13]`http://www.w3.org/TR/owl2-new-features/#Simple_metamodeling_capabilities`
[14]The symbol was introduced as an example by [3].

`Class` specifies a set of individuals in the same manner as an application of `rdf.resourceset` does. Together with `@Property` (essentially an application of `rdf.value`) it defines the domain of the mathematical relationship. `Expression` is any valid Popcorn term including the extensions described in Section 2.4. The rule is to be understood in that way that any instance of `Class` has a `Property` whose value can be calculated by `Expression`.

In order to exemplary describe the reasoning process we use the FOAF vocabulary and extend it with the custom properties `e:mass` and `e:height`. While the former is a measurement for the weight of a person, the latter is used for representing the height of a human being. Further properties are computed using the following rules:

```
[foaf:Person] :
@e:bmi = @e:mass / @e:height ^ 2

[foaf:Group] :
@e:aBMI = sum(@@foaf:member,lambda[$x->@e:bmi($x)]) / set1.size(@@foaf:member)
```

The first rule describes the calculation of the body mass index (`e:bmi`) for any instance of `foaf:Person`. Each individual has to define values for the properties `e:mass` and `e:height`. This can either be done explicitly or by using property values computed by logical reasoning. If we, for example, consider an instance of `foaf:Person` that has a value for `medical:weight` instead of `e:mass` we could use an ontology alignment to infer the required property value:

```
medical:weight rdfs:subPropertyOf e:mass
```

The second rule is used for computing the average body mass index (`e:aBMI`) across a specific set of people. Therefore the sum of respective body-mass-index property values is divided by the cardinality.

4.2 Rule processing

For rule processing we propose the architecture as illustrated in Figure 2. It involves a computer algebra system that has been extended as described in Section 2.5. Thus, this system can handle both, complex mathematics and retrieval of RDF data. In a current implementation of this architecture we make use of Symja[15].

The prototype relies on a simple ontology[16] that is used to represent mathematical rules. It defines the class `mathrl:Constraint` for the representation of such relationships. Instances of `mathrl:Constraint` have the attributes `mathrl:onProperty` for the target RDF property and `mathrl:expression` for the related formula represented by our OpenMath ontology (Section 3). Objects of type `mathrl:Constraint` are assigned to OWL classes by using the property `mathrl:constraint`. The following listing illustrates the serialization of the first rule given in section 4.1.

[15]`http://code.google.com/p/symja/`

[16]The ontology is available at `http://numerateweb.org/vocab/math/rules`.

```
foaf:Person mathrl:constraint [
  mathrl:onProperty e:bmi;
  mathrl:expression [
    a om:Application; om:symbol <http://www.openmath.org/cd/arith1#divide>;
    om:arguments (
      [ a om:Application; om:symbol <http://www.openmath.org/cd/rdf#value>;
        om:arguments (e:mass)]
      [ a om:Application;
        om:symbol <http://www.openmath.org/cd/arith1.xhtml#power>;
        om:arguments ([ a om:Application;
          om:symbol <http://www.openmath.org/cd/rdf#value>;
          om:arguments (e:height)] "2"^^xsd:long)]
    )]
] .
```

During the reasoning process, which can be triggered by the user or on data change, our reasoner iterates over all instances of classes that are related to one or more mathematical rules. Within an iteration it computes the properties given by the `mathrl:onProperty` attribute of the correspondent instances of `mathrl:Constraint` using the mathematical term that is referenced by `mathrl:expression`. Therefore the reasoner uses a phrase-book to transform the mathematical expression for feeding the computer algebra system. The result of the calculation is saved explicitly into the triple store. If, during computation, some of the referenced property values cannot be retrieved then an error is logged. The described process is applicable to backward and forward chaining. Furthermore, it relies on a closed world assumption like SPARQL [1] and thus also SPIN.

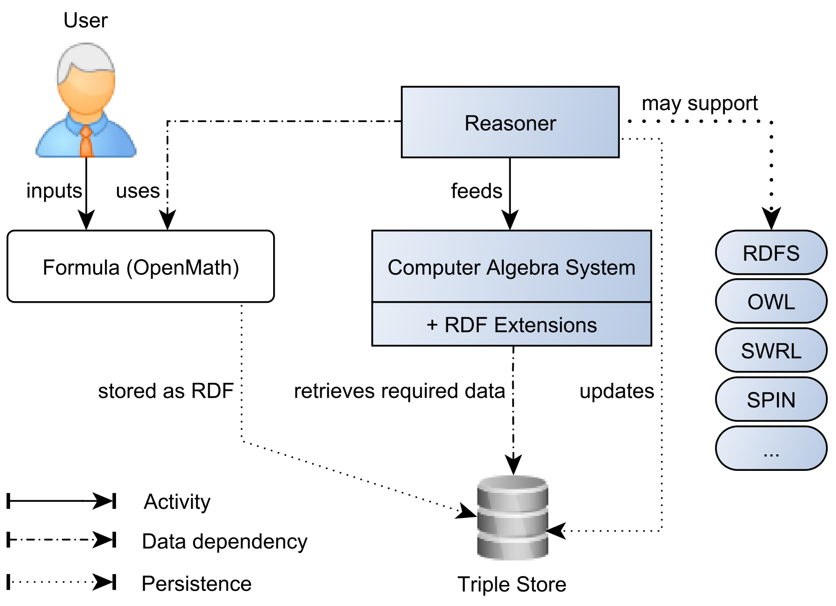

Figure 2: Reasoning architecture for math-enhanced inference.

The examples show that rules may interfere with each other. If the reasoner encounters a rule that is dependent on an individual with a property whose value is retrieved from another rule, then, if not already existent, these respective values are calculated first. Possible strategies to overcome cycles involve the reasoning *until no more changes occur* or *a certain processing limit, e.g. timeout, is reached* or *executing every rule for every individual only once*. So far we can only handle acyclic dependencies.

We additionally support RDFS and OWL-based reasoning, which relies on an open world assumption. The combination with further description logic and rule languages like SPIN or SWRL is possible, but could be a source for additional dependency issues.

5 Conclusion and future work

We have introduced a method for integrating RDF data retrieval into OpenMath to enable the consumption of linked data by computer algebra systems. In combination with our proposed RDF serialization for OpenMath objects it enables the definition of mathematical relationships within OWL ontologies. The related rules are used by our software architecture to execute mathematically enhanced inference over linked data sets.

Future work may investigate how the integration between OpenMath and RDF can be further improved, particularly concerning the representation of RDF statements within OpenMath and the alignment of OWL's set logic with OpenMath's `set1` symbols. The development of methods for partial and distributed calculation could solve scalability issues when applying mathematically enhanced inference on huge data sets.

More sophisticated methods to overcome circular dependencies need to be applied, especially when combining our mathematically enhanced inference approach with reasoning based on description logic or further rule languages.

Finally, we think that, besides our use cases for energy performance analysis and process planning, the integration between OpenMath and linked data has a high potential for many applications, e.g. for open government data where methods for statistical calculations should be transparent. Especially the combination with existing technologies for knowledge representation and logical reasoning may provide improved support for today's engineering tasks (e.g. integration of multiple data sources for the development of sustainable products).

Acknowledgements

The work presented in this paper is co-funded within the Cluster of Excellence Energy-Efficient Product and Process Innovation in Production Engineering (eniPROD®) by the European Union (European Regional Development Fund) and the Free State of Saxony.

References

[1] M. Arenas and J. Pérez. Querying semantic web data with sparql. In *Proceedings of the 30th symposium on Principles of database systems of data*, pages 305–316. ACM, 2011.

[2] Tim Berners-Lee. Linked Data, 2006. `http://www.w3.org/DesignIssues/LinkedData.html`.

[3] James H. Davenport and Paul Libbrecht. The Freedom to Extend OpenMath and its Utility. *Mathematics in Computer Science*, 2(2):253–277, 2008.

[4] B. Denkena, M. Shpitalni, P. Kowalski, G. Molcho, and Y. Zipori. Knowledge management in process planning. *CIRP Annals-Manufacturing Technology*, 56(1):175–180, 2007.

[5] P. Horn and D. Roozemond. OpenMath in SCIEnce: SCSCP and POPCORN. *Intelligent Computer Mathematics*, pages 474–479, 2009.

[6] Christoph Lange. Towards OpenMath Content Dictionaries as Linked Data. In *23rd OpenMath Workshop*, 2010.

[7] Christoph Lange. Ontologies and languages for representing mathematical knowledge on the Semantic Web. *Semantic Web*, 2011.

[8] Jan Morbach, Andreas Wiesner, and Wolfgang Marquardt. OntoCAPE – A (re)usable ontology for computer-aided process engineering. *Computers & Chemical Engineering*, 33(10):1546 – 1556, 2009. Selected Papers from the 18th European Symposium on Computer Aided Process Engineering (ESCAPE-18).

[9] Jan Morbach, Aidong Yang, and Wolfgang Marquardt. OntoCAPE 2.0 – Mathematical Models. Technical Report (LPT-2008-28). Technical report, Lehrstuhl für Prozesstechnik, RWTH Aachen University, 2008. `http://www.avt.rwth-aachen.de/AVT/fileadmin/files/Service_software/Software_Simulation/Mathematical_Models.pdf`.

[10] Andrew Robbins. Semantic MathML, 2009. `http://straymindcough.blogspot.de/2009/06/semantic-mathml.html`.

[11] A. Sánchez-Macián, E. Pastor, J.E.L. De Vergara, and D. López. Extending SWRL to enhance mathematical support. In *Proceedings of the 1st international conference on Web reasoning and rule systems*, pages 358–360. Springer-Verlag, 2007.

[12] E. Sirin, B. Bulka, and M. Smith. Terp: Syntax for OWL-friendly SPARQL queries. In *Proceedings of OWLED*, 2010.

[13] Pradeep Suresh, Girish Joglekar, Shuohuan Hsu, Pavan Akkisetty, Leaelaf Hailemariam, Ankur Jain, Gintaras Reklaitis, Venkat Venkatasubramanian, Bertrand Braunschweig, and Xavier Joulia. OntoMODEL: Ontological mathematical modeling knowledge management. In *18th European Symposium on Computer Aided Process Engineering*, volume Volume 25, pages 985–990. Elsevier, 2008.

[14] Ken Wenzel, Jörg Riegel, Andreas Schlegel, and Matthias Putz. Semantic Web Based Dynamic Energy Analysis and Forecasts in Manufacturing Engineering. In *Glocalized Solutions for Sustainability in Manufacturing*, Life cycle engineering, pages 507–512. Springer-Verlag, Berlin, Heidelberg, 2011.