# SMT-LIB in XML clothes

Filip Marić
filip@matf.bg.ac.yu
Faculty of Mathematics, University of Belgrade
Studentski trg 16, 11 000 Belgrade, Serbia

Predrag Janičić
janicic@matf.bg.ac.yu
Faculty of Mathematics, University of Belgrade
Studentski trg 16, 11 000 Belgrade, Serbia

**Abstract**

In this paper we propose how to further improve SMT-LIB standards (for the field of decision procedures). We propose using XML for all SMT-LIB standards and we believe that this promising, already widely used framework, would also make SMT-LIB easier to use, more powerful, and more popular. All that should hopefully help in advancing the field of decision procedures.

## 1 Introduction

After three decades of research in use of decision procedures in automated reasoning, this (sub)field now has a plenty of both theoretical and practical results and is already rather well-established. There are two dominating approaches for combining decision procedures — Nelson/Oppen's scheme and Shostak's scheme [4, 6] — and a number of their rational reconstructions and extensions. There are also techniques for augmenting decision procedures by additional hypotheses, including the influential Boyer/Moore's scheme [1]. Most (if not all) of the state-of-the-art proving systems have some support for decision procedures and for some of the above schemes. Nowadays, researchers in the field of decision procedures (and potential users) need ways for easier exchanging of ideas, benchmarks, test results, implementations, abstract representation of algorithms etc. That would help in advancing the field, by collecting different sources of conjectures in a standard form, by uniform representation of different techniques, and by deeper understanding of variants of different approaches. The SMT-LIB initiative aims at these goals and already gathers many (if not most) researchers in this field [5]. In this paper we propose how to further improve SMT-LIB standards and make them more expressive and easier to use. Namely, we suggest that SMT-LIB should move into the XML framework. We provide arguments for this suggestion and also present the support for SMT-LIB in XML form. This support has been implemented within our ARGO-LIB project [3].

## 2 SMT-LIB

The main goal of the SMT-LIB initiative [5], supported by a growing number of researchers world-wide is to produce a library of benchmarks for satisfiability modulo theories and all required standards and notational conventions. Such a library will facilitate the evaluation and the comparison of different approaches for using decision procedures and advance the state of the art in the field. The progress that has been made so far supports these expectations. In SMT-LIB the background logic is first order classical logic with equality (while it also allows sorted logic to more easily express benchmarks). The existing SMT-LIB standard proposes a LISP-like syntax for describing benchmarks and theories. An example benchmark is shown in the left column of Figure 3.

## 3 XML and MathML

XML is the Extensible Markup Language. It is designed to improve the functionality of the Web by providing more flexible and adaptable information identification. It is called extensible because it is not a fixed format like HTML (a single, predefined markup language). Instead, XML is actually a "metalanguage" — a language for describing other languages, which lets one design his/her own customized markup languages for limitless different types of documents. XML is intended "to make it easy and straightforward to define document types, easy to author and manage documents, and easy to transmit and share them across the Web". However, XML is not just for Web pages: it can be used to store any kind of structured information, and to enclose or encapsulate information in order to pass it between different computing systems. An XML document can carry both presentation (i.e., plausible visualisation) and content information.

XML is a project of the World Wide Web Consortium (W3C) [7], and the development of the specification is being supervised by their XML Working Group. XML is a public format — it is not a proprietary development of any company. Almost all browsers that are currently in use support XML natively.

A DTD is a formal description in XML Declaration Syntax of a particular type of document (i.e., of its syntactical restrictions). It sets out what names are to be used for the different types of element, where they may occur, and how they all fit together. This formal description enables automatic verification ("validation") of whether a given document meets the given syntactical restrictions. The alternative to a DTD is a Schema, which is written in Instance Syntax and provides much more extensive validation facilities.

XML comes with the XSLT document processing language that is used to transform the input XML documents i.e., the input files to the desired output documents. An XSLT style-sheet declares a set of rules (templates) for an XSLT processor to use when interpreting the contents of an input XML document. These rules tell the XSLT processor how that data should be presented – as an XML document, as an HTML document, as plain text, or in some other form.

MathML is the Mathematical Markup Language [2]. It is an XML application for describing mathematical notation and capturing both its structure and content. MathML is a product of the W3C Math working group. One of the goals of MathML is to enable mathematics to be presented and and processed on the Web, just as HTML has

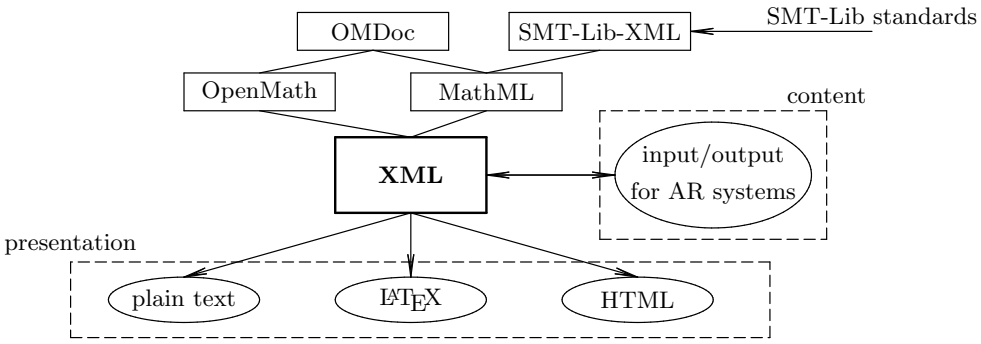


Figure 1: Logical organization of XML systems

enabled this functionality for text. It provides a much needed foundation for the inclusion of mathematical expressions in Web pages. Many implementations of MathML are available (browsers and authoring tools), many of which are open source software. MathML is supported by the current versions of MS Internet Explorer by special plug-ins (e.g. MathType), and natively by MathML-enabled versions of the open source browsers like Mozilla and Amaya.

However, MathML is not only used in Web pages but also for storing general purpose mathematical documents. Several markup schemes for mathematical documents have been developed. For example, OMDoc [8] is a scheme for describing various mathematical documents including articles, textbooks, interactive books and courses. OMDoc uses MathML, and a similar scheme called OpenMath to describe mathematical formulae (see Figure 1).

## 4 SMT-LIB-XML

We believe that the syntax of smt-lib benchmarks and theory descriptions should be adapted and put into the xml framework. The right column of the Figure 3 shows the benchmark written in the proposed new syntax.

All (or almost all) restrictions stated in the smt-lib specifications can be formally represented in a dtd document, saying what syntactical constraints each benchmark has to meet. We also propose using a subset of MathML for describing the formulas themselves in the benchmarks. The main reason for this is the fact that MathML is becoming a standard markup-scheme for mathematical formulae, and the fact that it has wide support in existing software which is constantly growing. Figure 2 shows a dtd for the smt-lib benchmarks. The general MathML can be restricted with respect to smt-lib requirements. This dtd can then be used, in conjunction with the generic xml validation mechanism, for verifying whether a given benchmark is legal. The same applies to representation of theories (or other operational content, such as rewrite rules).

By means of xslt, xml representation of benchmarks and theories can be easily transformed into html format that is convenient for human-readable display in browsers. It can also be transformed into any specific input/output representation, required by any automated reasoning system (see Figure 1). We have also developed a style-sheet that converts xml benchmarks back into standard smt-lib format.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- SMT-LIB-XML 1.0 DTD  -->
<!ENTITY % mathmldtd PUBLIC "-//W3C//DTD MathML 2.0//EN"
           "http://www.w3.org/Math/DTD/mathml2/mathml2.dtd">
%mathmldtd;

<!-- Description of the root element smt-lib-benchmark-set-->
<!ELEMENT smt-lib-benchmark-set
    (name, theories, language, extra_signature?, benchmarks)>

<!-- Name of benchmark set -->
<!ELEMENT name (#PCDATA)>

<!-- Decription of theories -->
<!ELEMENT theories (theory+)>
<!ELEMENT theory (#PCDATA)>

<!-- Description of used language -->
<!ELEMENT language (#PCDATA)>

<!--Decription of extra signature. Extra signature consists
of extra-functions and extra-predicates. -->
<!ELEMENT extra_signature
       (function_symbol | predicate_symbol)+>
<!ELEMENT function_symbol (argument*)>
<!ATTLIST function_symbol
    name CDATA #REQUIRED
    sort (Real | Rat | Int | Nat) #REQUIRED
>
<!ELEMENT predicate_symbol (argument*)>
<!ATTLIST predicate_symbol
    name CDATA #REQUIRED>
<!ELEMENT argument EMPTY>
<!ATTLIST argument
    sort (Real | Rat | Int | Nat) #REQUIRED>

<!-- Benchmarks -->
<!ELEMENT benchmarks (benchmark+)>
<!ELEMENT benchmark (hypotheses*, formula)>

<!-- Formulas -->
<!ELEMENT formula (math)>
<!ATTLIST formula
    status (valid|satisfiable|unsatisfiable|invalid) #REQUIRED>

<!-- Hypotheses -->
<!ELEMENT hypotheses (hypothesis+)>
<!ELEMENT hypothesis (math)>
<!ATTLIST hypothesis
    type (formula | rewrite-rule) #REQUIRED>
```

Figure 2: SMT-LIB dtd

```
(:name "Boyer/Moore's example"
 :theory "PRA"
 :theory "FOLeq"
 :language "Universally quantified"
 :extra_funs
 ( (delta Int Int Int Int)
   (maxint Int)
 )

 :benchmarks
   (:hypotheses
      (:formula
         (<=
            (delta !x !y !z)
            !y
         )
      )
    )

   (:formula
      (:forall (?lp Int)
      (:forall (?lt Int)
      (:forall (?i Int)
      (:forall (?c Int)
         (:impl
            (:and
               (<=
                  (+ ?lp ?lt)
                  maxint
               )
               (<=
                  ?i
                  ?lt
               )
            )
            (<=
               (+ ?i
                  (delta ?lt ?lp ?c)
               )
               maxint
            )
         )
       ))))
      :status :valid
    )
)
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE smt-lib-benchmark-set
          SYSTEM "smt-lib-xml.dtd">
<smt-lib-benchmark-set>
  <name>Boyer-Moore's example</name>
  <theories>
    <theory>FOLeq</theory>
    <theory>PRA</theory>
  </theories>
  <language>
     Universally quantified
  </language>
  <extra_signature>
    <function_symbol name="delta" sort="Int">
      <argument sort="Int"/>
      <argument sort="Int"/>
      <argument sort="Int"/>
      <argument sort="Int"/>
    </function_symbol>
    <function_symbol name="maxint" sort="Int"/>
  </extra_signature>
  <benchmarks>
    <benchmark>
      <hypotheses>
        <hypothesis type="rewrite-rule">
          <math xmlns="http://www.w3.org/
                              1998/Math/MathML">
            <apply><leq/>
              <apply><fn><ci>delta</ci></fn>
                <ci>!x</ci>
                <ci>!y</ci>
                <ci>!z</ci>
              </apply>
              <ci>!y</ci>
            </apply>
          </math>
        </hypothesis>
      </hypotheses>
      <formula status="valid">
        <math xmlns="http://www.w3.org/
                              1998/Math/MathML">
           ...
        </math>
      </formula>
    </benchmark>
  </benchmarks>
</smt-lib-benchmark-set>
```

Figure 3: Example benchmark written in slightly extended SMT-LIB syntax (left) and in XML syntax (right; due to the lack of space, the formula is replaced by ellipsis.)

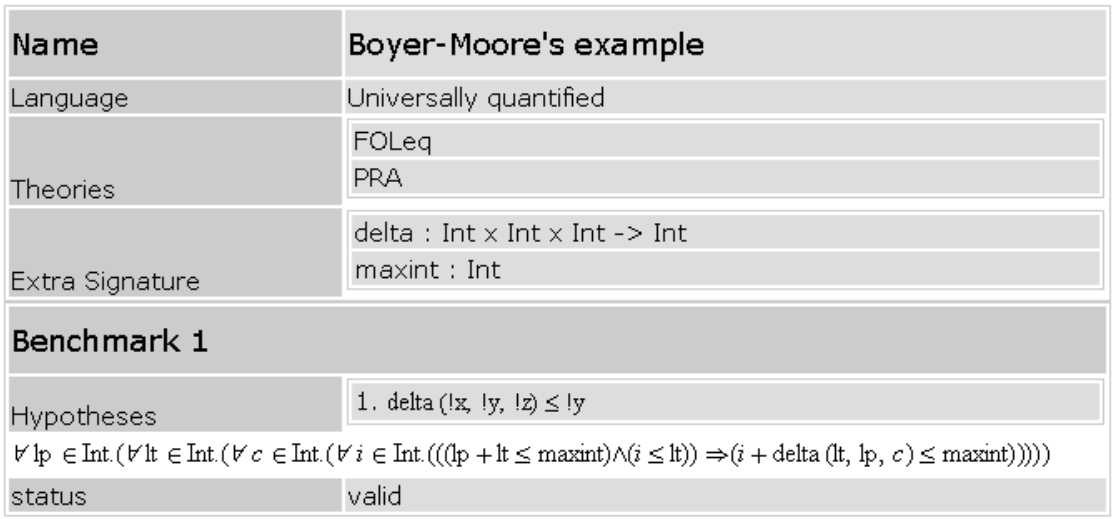


Figure 4: HTML representation of the benchmark shown in Figure 3.

## 5 ARGO-LIB

In this section we give a brief description of ARGO-LIB, a system that already supports the standards we propose.

ARGO-LIB[1] is being developed as a flexible, modular, and efficient generic platform for using decision procedures aimed at *realistic* use both in academia and in industry [3]. It provides support for a range of decision procedures (for a range of theories) and also for different techniques for using decision procedures. ARGO-LIB is implemented in the standard C++ programming language. It can work stand-alone but can also be simply integrated into some other tool (e.g., a theorem prover, constraint solver, model checking system etc.).

ARGO-LIB has support for a library of theories and conjectures and support for benchmarking. In this sense, ARGO-LIB builds on motivations, ideas, and standards promoted by the SMT-LIB initiative. ARGO-LIB uses the SMT-LIB format as its native input format (for theory representations and benchmark representations). Most of the requirements of the ARGO-LIB system are met by the SMT-LIB standard, however there were still several additions to SMT-LIB. These are additions to SMT-LIB format that ARGO-LIB implements and uses: the slot for additional *hypotheses* in the benchmark representation; one benchmark can rely on *several theories*; the slot for an *extra signature* in the benchmark representation (the extra signature is being updated and can include, for instance, Skolem constants); *generic sorts* (useful, for instance, in describing the theory of lists over a generic sort); representation for *rewrite rules*.

[1]The web page for ARGO-LIB, including source files and documentation is at: `www.matf.bg.ac.yu/~janicic/argo`. There is also available a suite of tools for converting from and to XML from different formats.

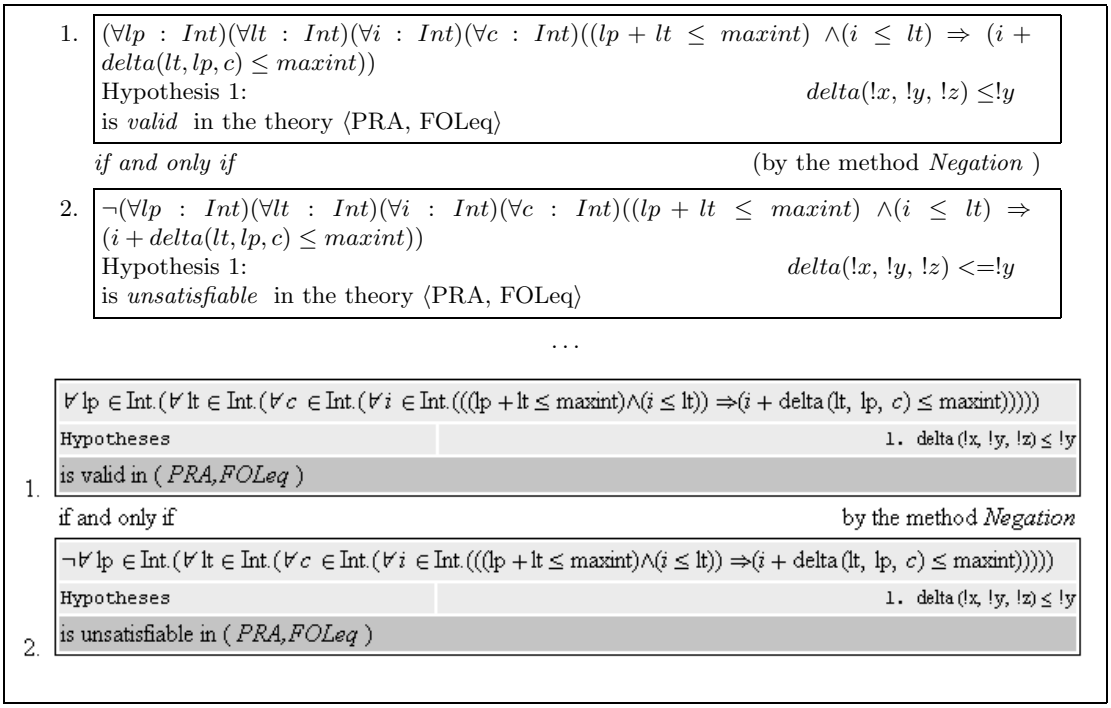


Figure 5: A part of an ARGO trace presented in LaTeX and HTML format

## 6 ARGO-LIB Traces

ARGO-LIB does not produce object-level proofs, but only traces with information of higher-level methods used. Other reasoning systems can also have specific output traces. Output traces of the systems that produce object level proofs, could use MATHML to describe a proof. Given the mechanism of XML validation, we can focus on defining a DTD for (specific) proof traces, making strict restrictions that any trace has to meet (for instance, all branches must be closed, only methods from a fixed can set be used etc). Any generated trace can then be verified automatically using the DTD and the system of XML validation. This, of course, is not full verification of the output, but it can be an important help.

## 7 Conclusions and Future Work

In this paper we proposed and advocated the use of XML in SMT-LIB. We gave a brief description of SMT-LIB, XML and MATHML, and why the use of XML would be beneficial for SMT-LIB. Some of the main advantages of using XML and MATHML would be:

- instead of raw, plain text representation, SMT-LIB files will be stored in strictly structured files; these files will be easy to parse, process, and convert into different forms and formats;
- input/output tasks will be supported by generic, external tools and different automated reasoning systems will communicate easily;

- easier communication and exchange of material with the rest of mathematical/computer science community;
- wide and growing support for XML;
- different sorts of presentation (text form, LaTeX form, HTML) easily enabled;
- strict content validation of documents with respect to given restrictions.

Within the ARGO-LIB system we have already built a prototype support for SMT-LIB files in XML form and we are planning to further improve it. We are planning to use only XML files (for theories, benchmarks, rewrite rules, etc.) both for input and for output, while we will use XSLT for converting output files into LaTeX format, text format, or HTML format. This would make the whole system very flexible and all presentation issues would be subject to changes in external (XSLT) files and not in the source code itself. At the same time, different sorts of validations of content will be possible. We will also work on further potentials of the validation mechanism and try also to verify some semantic conditions.

We hope that this support would be useful and helpful for everyone interested in the SMT-LIB initiative and in pragmatics of using decision procedures in automated reasoning. By this paper we advocate moving all SMT-LIB standards to the XML framework and we also call upon the SMT-LIB interest group to support that move.

## References


[1] R. S. Boyer and J S. Moore. Integrating Decision Procedures into Heuristic Theorem Provers: A Case Study of Linear Arithmetic. *Machine Intelligence 11*, 1988.

[2] World Wide Web Math Working Group. Mathematical Markup Language. on-line at: http://www.w3.org/Math.

[3] Filip Marić and Predrag Janičić. ARGO-LIB: A generic platform for decision procedures. In *The 2nd International Joint Conference on Automated Reasoning (IJCAR-2004)*, LNAI, Springer, 2004.

[4] G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, October 1979.

[5] Silvio Ranise and Cesare Tinelli. The SMT-LIB Format: An Initial Proposal. 2003. on-line at: http://goedel.cs.uiowa.edu/smt-lib/.

[6] R. E. Shostak. Deciding combinations of theories. *Journal of the ACM*, 31(1):1–12, January 1984.

[7] World Wide Web Consortium (W3C). XML. on-line at: http://www.w3.org.

[8] OMDoc: An Open Markup format for Mathematical Documents on-line at: http://www.mathweb.org/omdoc/