Enhancement of Mizar Texts with Transitivity Property of Predicates

Artur Korniłowicz(✉)

Institute of Informatics, University of Białystok,
K. Ciołkowskiego 1M, 15-245 Białystok, Poland
arturk@mizar.org

Abstract. A typical proof step in mathematical reasoning consists of two parts – a formula to be proven and a list of references used to justify the formula. In addition, computer proof-assistants can use specialized procedures and algorithms to strengthen their computational power to verify the correctness of reasonings.

The Mizar system supports several mechanisms to increase automation of some reasoning steps. One of them is registration of chosen properties of predicates and functors when they are defined. We propose strengthening of the Mizar system by processing another common property used in mathematics – transitivity.

1 Introduction

The Mizar system [1–3] is a computer system invented for computer-assisted verification of mathematical papers. It consists of three main components: a language – the Mizar language, a bunch of computer programs including Verifier and a repository of formal texts – Mizar Mathematical Library (MML) – written in the language and automatically verified for their logical correctness. The Mizar language is a declarative formal language designed to write mathematical papers readable for humans and effectively processed by computers. The language is highly structured to ensure producing rigorous and semantically unambiguous texts. Apart from rules for writing traditional mathematical items (e.g. definitions, lemmas, theorems, proof steps, etc.) it also provides syntactic constructions to launch distinguished algorithms for processing particular mechanisms (e.g. term identifications, term reductions [4], flexary connectives [5]) increasing computational power of Verifier. The most interesting mechanism, from the point of view of this research, is the possibility of registering various properties of predicates and functors [6] at the stage of defining new notions. The current version of the Mizar system supports registration of `reflexivity`, `irreflexivity`, `symmetry`, `asymmetry` and `connectedness` for binary predicates; `involutiveness` and `projectivity` for unary operations; and `commutativity` and `idempotence` for binary operations. Table 1 presents how registrations of the properties are used in the MML and how they influence on proofs stored in the library. Consecutive columns show numbers of occurrences

M. Kohlhase et al. (Eds.): CICM 2016, LNAI 9791, pp. 157–162, 2016.
DOI: 10.1007/978-3-319-42547-4_12

Table 1. Properties of predicates and functors

Property	Occurrences	Articles	Errors	Articles with errors
Predicates				
`reflexivity`	138	91	356	44
`irreflexivity`	11	10	9	2
`symmetry`	122	82	498	47
`asymmetry`	6	6	6	4
`connectedness`	4	4	65	4
total	281	119	934	73
Functors				
`involutiveness`	38	32	163	18
`projectivity`	21	18	11	3
`commutativity`	155	86	1423	55
`idempotence`	20	13	155	9
total	234	115	1718	70

of each property, numbers of articles in which the properties were declared, numbers of errors occurring after removing registrations of the properties from texts, and numbers of articles with such errors.[1]

In this paper we propose strengthening of the Mizar system by processing of another common property used in mathematics – `transitivity`. It is described in Sect. 2. In Sect. 3 we present some results of our implementation and describe its potential influence on the MML. In Sect. 4 we indicate several directions of further development of processing properties in Mizar.

2 Transitivity

Transitivity is a very common property of predicates. It is a subject of research in various branches of mathematics. It is used to define, for example, orders, equivalences, etc. Many relations are tested to determine if they are transitive or not. Many mathematical theorems assert the transitivity of various relations.

We propose an enhancement of the Mizar system supporting automatic processing of transitive predicates, where by automatic we mean that some computations during the verification process are executed based on knowledge gathered in the MML which is not explicitly referred to in processed proof steps.[2] To enable such an automation, when a new predicate is defined, if it is transitive, it should be declared as transitive (just like in the case of other properties

[1] Total numbers are not simply sums of columns, because errors occurring after removing different registrations could occur in the same articles.

[2] Other such automations are, for example, processing of adjectives [7] and definitional expansions [8].

supported by the Mizar verifier [6]). Such a declaration has to be done within a definitional block with syntax

```
definition
 let x_1 be θ_1, x_2 be θ_2, ..., x_n be θ_n, y_1,y_2 be θ_{n+1};
 pred π(y_1,y_2) means :ident:
  Φ(x_1,x_2,...,x_n,y_1,y_2);
 transitivity
 proof
  thus for a, b, c being θ_{n+1}
   st Φ(x_1,x_2,...,x_n,a,b) and Φ(x_1,x_2,...,x_n,b,c)
   holds Φ(x_1,x_2,...,x_n,a,c);
 end;
end;
```

The correctness of the definition must be proven according to a special formula expressing the transitivity of the defined predicate. The formula is generated by the system. Having such a definition, whenever Verifier meets a conjunction of formulas $\pi(a, b)$ and $\pi(b, c)$ within an inference, the inference is enlarged by automatically generated formula $\pi(a, c)$ which may help to justify the proof step. For example, when one wants to prove the transitivity of `<=` for real numbers, that is the statement `a<=b & b<=c implies a<=c`, Verifier (as a classical disprover) assumes three premises: `a<=b`, `b<=c` and `a>c`. Then, by transitivity, it knows that `a<=c` which contradicts with `a>c` and finishes the proof.

3 Experiments

The implemented software was tested on Mizar Version 8.1.02 working with the MML Version 5.36.1267.[3]

An important part of the package is a tool (transdet) which detects theorems stored in the MML, that could be rewritten as registrations of the transitivity of some predicates (we will call such theorems *transitivity-like theorems*). In the current version of the library 127 such theorems were found in 90 articles. The Library Committee, who is responsible for the management, developing and revisions of the MML, will analyze all cases and decide which of them would be incorporated into the library. In the case of approval, a refactoring of the MML [9] will be required while maintaining licensing its content [10].

To present some examples detected in the MML[4] let us cite the transitivity of ordering of elements of a semilattice [11]

[3] Computations were carried out at the Computer Center of University of Białystok http://uco.uwb.edu.pl.

[4] The full list is accessible at http://alioth.uwb.edu.pl/~artur/transitivity/th2trans.txt.

```
theorem
for L being join-associative non empty \/-SemiLattStr,
    a, b, c being Element of L holds a [= b & b [= c implies a [= c;
```

and the transitivity of being isomorphic groups [12]

```
theorem
for G, H, I being Group holds
 G,H are_isomorphic & H,I are_isomorphic implies G,I are_isomorphic;
```

An important gain from rewriting detected transitivity-like theorems as declarations of the transitivity of used predicates is decreasing the number of explicit references to the theorems from all proofs collected in the MML. Table 2 presents top 10 most cited such theorems.[5] These numbers mean that 16855 out of all 629048 (2.7 %) references in the entire library can be removed while ensuring that all proofs remain valid.

Table 2. References to transitivity-like theorems

Article	References
XXREAL_0:2	12601
XBOOLE_1:1	3162
ORDERS_2:3	319
ORDINAL1:10	235
LATTICES:7	97
NAT_D:4	70
INT_2:9	44
WELLORD2:15	44
PBOOLE:13	43
BORSUK_6:79	31
...	...
total	16855

The software can be downloaded from http://alioth.uwb.edu.pl/~artur/transitivity.

4 Further Work

A possible direction to continue work is to implement processing of other commonly used in mathematics properties of relations, like, for example, antisymmetry, trichotomy, left- and right- Euclidean. Another topic is to introduce properties which are collections of other properties, like equivalence which is reflexive,

[5] The full list of non-zero numbers of references is accessible at http://alioth.uwb.edu.pl/~artur/transitivity/references.txt.

symmetric and transitive; or preorder which is reflexive and transitive; etc. Of course, one may declare a relation as reflexive, symmetric and transitive, but it would be probably worth to enrich the MIZAR language to make it closer and closer to traditional mathematical vernacular.

In the current stage of the MIZAR system, all properties of predicates can be registered for binary predicates only. So, for example, the theorem [13]

```
theorem :: REWRITE1:16
for R being Relation, a,b,c being object st R reduces a,b & R reduces b,c
 holds R reduces a,c;
```

is not transformable to a registration of transitivity, since the predicate `reduces` is ternary, not binary. But it is seen, that if we fix the value of one argument, the predicate can be understood as binary one, and we may think about its transitivity (or other properties of binary predicates). In general, properties dedicated for binary predicates, can be introduced for n-ary predicates, where $2 \leq n$, with $n - 2$ fixed arguments.

5 Conclusions

In the paper we presented an extension of the MIZAR system by introducing a new word to the MIZAR language (transitivity) and new rules for processing transitive predicates. We detected theorems describing the transitivity of various relations gathered in the MML. It can be concluded that our implementation will have strong impact on the shape of many proofs – many explicit references to the theorems can be removed. It may even result in reorganization of proof steps within entire proofs [14].

As the last observation, it can be said that this new feature of the MIZAR system was anticipated and expected by MIZAR users. Josef Urban, in one of his papers [15], annotated a theorem as follows[6]:

```
:: remove when transitivity implemented
theorem :: OSALG_1:2
for S being non empty non void OverloadedMSSign
 for o,o1,o2 being OperSymbol of S
  holds o ~= o1 & o1 ~= o2 implies o ~= o2;
```

References

1. Trybulec, A.: Mizar. In: Wiedijk, F. (ed.) The Seventeen Provers of the World. LNCS (LNAI), vol. 3600, pp. 20–23. Springer, Heidelberg (2006). doi:10.1007/11542384_4
2. Bancerek, G., et al.: Mizar: state-of-the-art and beyond. In: Kerber, M., Carette, J., Kaliszyk, C., Rabe, F., Sorge, V. (eds.) CICM 2015. LNCS, vol. 9150, pp. 261–279. Springer, Heidelberg (2015). doi:10.1007/978-3-319-20615-8_17

[6] http://mizar.uwb.edu.pl/version/current/html/osalg_1.html.

3. Grabowski, A., Korniłowicz, A., Naumowicz, A.: Four decades of Mizar. J. Autom. Reason. **55**(3), 191–198 (2015). doi:10.1007/s10817-015-9345-1
4. Korniłowicz, A.: On rewriting rules in Mizar. J. Autom. Reason. **50**(2), 203–210 (2013). doi:10.1007/s10817-012-9261-6
5. Korniłowicz, A.: Flexary connectives in Mizar. Comput. Lang. Syst. Struct. **44**, 238–250 (2015). doi:10.1016/j.cl.2015.07.002
6. Naumowicz, A., Byliński, C.: Improving Mizar texts with *properties* and *requirements*. In: Asperti, A., Bancerek, G., Trybulec, A. (eds.) MKM 2004. LNCS, vol. 3119, pp. 290–301. Springer, Heidelberg (2004). doi:10.1007/978-3-540-27818-4_21
7. Naumowicz, A.: Enhanced processing of adjectives in Mizar. In: Grabowski, A., Naumowicz, A. (eds.) Computer Reconstruction of the Body of Mathematics. Studies in Logic, Grammar and Rhetoric, pp. 89–101. University of Białystok, Białystok (2009)
8. Korniłowicz, A.: Definitional expansions in Mizar. J. Autom. Reason. **55**(3), 257–268 (2015). doi:10.1007/s10817-015-9331-7
9. Grabowski, A., Schwarzweller, C.: Revisions as an essential tool to maintain mathematical repositories. In: Kauers, M., Kerber, M., Miner, R., Windsteiger, W. (eds.) MKM/CALCULEMUS 2007. LNCS (LNAI), vol. 4573, pp. 235–249. Springer, Heidelberg (2007). doi:10.1007/978-3-540-73086-6_20
10. Alama, J., Kohlhase, M., Mamane, L., Naumowicz, A., Rudnicki, P., Urban, J.: Licensing the Mizar mathematical library. In: Davenport, J.H., Farmer, W.M., Urban, J., Rabe, F. (eds.) MKM 2011 and Calculemus 2011. LNCS, vol. 6824, pp. 149–163. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22673-1_11
11. Żukowski, S.: Introduction to lattice theory. Formalized Math. **1**(1), 215–222 (1990)
12. Trybulec, W.A., Trybulec, M.J.: Homomorphisms and isomorphisms of groups. Quotient group. Formalized Math. **2**(4), 573–578 (1991)
13. Bancerek, G.: Reduction relations. Formalized Math. **5**(4), 469–478 (1996)
14. Pąk, K.: Improving legibility of formal proofs based on the close reference principle is NP-hard. J. Autom. Reason. **55**(3), 295–306 (2015). doi:10.1007/s10817-015-9337-1
15. Urban, J.: Order sorted algebras. Formalized Math. **10**(3), 179–188 (2002)