Crafting a Repository of Knowledge Based on Transformation Rules

A.D. Rich (1) and D.J. Je rey (2)

[1] 62-3614 Loli'i Way, Kamuela, Hawaii, USA.

- Restrict to domains of validity. Many rules are valid only if their variables are restricted to a certain domain. Conditions on a properly de ned rule must restrict its application to the domain over which it is valid. For example, the transformation $\sqrt{z^2} \rightarrow z$ should only be applied if z is known to be purely imaginary or in the right half of the complex plane (unrestricted versions of this transformation caused the well remembered `square-root bug' in Maple).
- Restrict to simpli cation. To avoid in nite loops, applications of rules must eventually result in an expression that can be made no simpler (i.e. an expression to which no rules applies). The conditions attached to a rule must limit its application to those expressions for which its application results in a simpler expression. For example, if F stands for any trigonometric function and n for a rational number, the goal of transformations of $F(n)$ is to reduce the magnitude of the angle. Thus, speci cally, although $\sin(n) \rightarrow \cos((n$

4 Integration examples

One conspicuous bene t of rule-based integration is the greater simplicity of its results. Simplicity can include not just one integral, but consistent behavior over families of integrals. For example, the following integrals show symmetry between trigonometric and hyperbolic functions:

$$\int \frac{dx}{\sqrt{a+x}\sqrt{b+x}} = 2\,\mathrm{arctanh}\,\frac{\sqrt{a+x}}{\sqrt{b+x}}\,; \tag{1}$$

$$\int \frac{dx}{\sqrt{a+x}\sqrt{b-x}} = 2\arctan\,\frac{\sqrt{b-x}}{\sqrt{a+x}}\,: \tag{2}$$

In contrast, Mathematica and Maple express one integral using arctangent and one using logarithm:

$$\int \frac{dx}{\sqrt{a+x}\sqrt{b+x}} = \ln\left(a+b+2x+2\sqrt{a+x}\sqrt{b+x}\right)\,: \tag{3}$$

5 Platform Requirements

An e cient and reliable software platform is required to build a rule-based repository of knowledge. As a minimum the support platform needs to provide the following services:

- Transformation rules. The platform must make it possible to de ne and recursively apply transformation rules to expressions of a speci ed form. This requires a exible and natural syntax for the patterns used to specify the form of expressions.
- E cient pattern matching. A rule-based system may have thousand of rules, and hundreds of rule applications may be required to simplify an expression. Thus when given an expression to simplify, it is essential that the pattern matcher quickly nd the applicable rule, if any. Thoughts on how to implement an e cient pattern matcher is discussed below.
- Exact and arbitrary precision arithmetic. Numerical routines are required for built-in functions and operators since a rule-based approach is usually not appropriate for numerical computations.
- Programming environment. The platform must provide the ability to input, evaluate and display expressions, as well as provide a suitable environment for testing and debugging the repository.
  Since most modern computer algebra systems provide the above capabilities, they are suitable platforms for crafting a rule-based repository of knowledge.

6 E cient Pattern Matching

A general purpose repository might require thousands or even tens of thousands of rules. Obviously sequentially searching a list of that many rules to nd a match would be unacceptably slow. Even having a separate list of rules for each built-in function or operator is insu cient, since some functions may have a large number of rules associated with it (e.g. our integrator requires over a 1000 rules).

Therefore instead of a list, the software platform supporting a repository should store the rules in the form of a discrimination net based on the tree structure of expressions. Then, for example, all rules applicable to expressions of the form sin(u) will be collected in one branch of the tree, all di erentiation rules in another, all integration rules in another, etc. Then the rules in each branch will be recursively subdivided based on the form of its arguments, etc.

With the rules stored in such a discrimination net, the rule applicable to a given expression can be quickly found by a simple tree walk in $\log(n)$ time, where n is the number of rules in the repository.

7 Advantages

The following summarizes the advantages of storing mathematical knowledge in the form of a repository based on properly de ned transformation rules:

- Human and machine readable. Since rules are de ned using mathematical formulas rather than procedural programming constructs, they express a self-contained mathematical fact that can be attractively displayed in standard two-dimensional mathematical notation.
- Able to show simpli cation steps. The successive application of rules exactly corresponds to the steps required to simplify an expression. Thus when a rule is applied, it can display itself in standard mathematical notation as justi cation for the step, and then suspend further simpli cation so the partially simpli ed result is returned.
- Mechanical rule veri cation. Since the right side of a properly de ned rule is just a mathematical expression, the rules validity can often be mechanically veri ed. For example, the right side of integration rules can be di erentiated to see if they equal the integrand on the left.
- Facilitates program development. The fact that properly de ned rules are inherently self-contained and free of side-e ects makes it easy to test the e ect on the system of selectively adding, modifying or deleting rules. Although collections of rules may be highly recursive, each individual rule must be able to stand on its own, thus making it possible to test it on examples before adding it to the collection.
- Platform independent. Since properly de ned transformation rules consists only of mathematical expressions and pattern matching speci cations, the translation of rules from the syntax of one computer algebra system to another is relatively straight-forward.
- White box transparency. For the most part, computer algebra systems appear as mysterious black boxes to users with little or no explanation given as to how results are obtained. However, if the source le of rules on which a CAS is based were included with the system, it becomes a transparent white box, making it possible for users to modify existing rules and even add new ones.
- Fosters community development. The open source nature of a rule-based repository of knowledge would foster an active community of users. A website blog dedicated to a repository could provide developers the ability to propose new rules and improvements to existing ones. Developers would vie with one another to get credit for adding new rules to the repository. Others would shoot down defective ones. Thus the system would grow and evolve in Darwinian fashion much the same way Wikipedia does.
- An active repository. Encyclopedias and reference manuals, even on-line ones, are inherently passive repositories in the sense that users have to nd the knowledge required to solve a given problem, and then manually apply it. However, given a problem a rule-based repository actively nds and applies the knowledge required to solve it. Thus the knowledge in such repositories is in a much more useful form.

References

1. Abramowitz,M. & Stegun,I., *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. US Government Printing O ce, 1964. 10th Printing December 1972.
2. Einwohner, T.H. & Fateman, Richard J., *Searching techniques for integral tables*, Proceedings ISSAC '95, pp 133-139, ACM Press, 1995.