Well...It isn't Quite That Simple

Robert M. Corless and David J. Jeffrey

Department of Applied Mathematics
The University of Western Ontario
London, Ontario, CANADA, N6A 5B7
rcorless@uwo.ca
djeffrey@uwo.ca
SIGSAM Bulletin Vol 26(3) Issue 101, 1992

Present computer algebra systems base their interactive sessions on a very simple model of mathematical discourse. he user's input to the system is a line containing a mathematical expression (an operation, a formula, a set of equations, etc) and the system's response to the user is an output line which contains a mathematical expression similar to the input. here are many situations, however, in which this is too simple a model of mathematics. Algebra systems should be allowed to reply 'Well ... it isn't quite that simple'.

Consider, as a starting problem, the evaluation of a definite integral. he table of integrals by Gradshteyn and Ryshik contains the entry [1, 3.310]

$$\int_0^\infty e^{-px}\,dx =$$

CAS	*Problem 1*	*Problem 2*
Mathematica	$1/p$	$x = 1$
Maple	$\int_0^\infty \exp(-px)dx$	$x = 1$
Derive	?	$x = 1$
ALJABR	Is p +, -, or zero?	x = 1
heorist	—	$x = 1$ if $k \neq 0$ x undetermined if $k = 0$

able 1: Responses of popular CAS to 2 standard problems.

---

and in spite of them, we find all of the responses in the table unsatisfactory, for reasons we now discuss.

Mathematica's answer for Problem 1 is unsatisfactory because it is incorrect on specialization. If, having obtained it, we later set $p = -2$, then the result is simply wrong. o be fair, the answer is correct if p happens to be positive, and the user may be lucky. he bottom line, nevertheless, is that Mathematica has (to use a popular term nowadays [2,3]) lied to the user: the integral of $\exp(-pt)$ from $t = 0$ to $t =$ is not always $1/p$. Maple's answer is unsatisfactory because it avoids being wrong by avoiding being helpful. In fact, the user may be unaware that Maple can do better, and simply think "

furcation problem with six free parameters, for example: without some physical knowledge of the reasonable domains for the parameters, this problem could be hopeless to solve.

he automatic exploration of conditions or alternative results requires considerable computational resources, and for the sake of speed there is an attraction to picking one 'obvious' answer. his way of proceeding is also attractive because it fits easily into the existing model of interactive exchange. he difficulty is to balance efficiency against correctness. Some users will say 'I don't care *how* quickly a program can get me the wrong answer' while others will say 'I don't care *how* correct the answer will be, if I have to wait a thousand years to see it'.

As a focus for discussion, we can take a specific problem and list some of the possible templates that a CAS could use when presenting its solution. Given the integral

$$\int_0^\infty \exp(a^3 - a)x \; dx, \tag{3}$$

a CAS could return any of the following responses.

(A) An unevaluated integral: $\displaystyle\int_0^\infty \exp(a^3 - a)x \; dx.$

It could be returned unevaluated, but with an indication of why.

(B) $\displaystyle\int_0^\infty \exp(a^3 - a)x \; dx,$ [ADVICE: Need the sign of $a^3 - a$].

An unqualified result could be returned.

(C) $-1/(a^3 - a)$

One value could be returned together with a proviso.

(D) $-1/(a^3 - a),$ [PROVISO: $a^3 - a < 0$].

If only one value is returned, it may not be obvious which value that should be. A qualified result that is equally valid is

(E) , [PROVISO: $a^3 - a \geq 0$].

We could return all possible values, but without analysing the conditions beyond the form in which they naturally arise.

(F)
$$\begin{cases} \ , & \text{if } \ a^3 - a \geq 0, \\ -1/(a^3 - a), & \text{if } \ a^3 - a < 0. \end{cases}$$

he condition on $a^3 - a$ could be subject to further analysis, to obtain conditions on a.

(G)
$$\begin{cases} -1/(a^3 - a), & \text{if } \ a < -1, \\ \ , & \text{if } \ -1 \leq a \leq 0, \\ -1/(a^3 - a), & \text{if } \ 0 < a < 1, \\ \ , & \text{if } \ 1 \leq a. \end{cases}$$

true, then the answer is so-and-so". It is possible that the conditions may be such that they are never true at the same time, but that cannot be a criticism of the approach, because it may be the mathematically correct statement. Without such provisions, the CAS and the user are charging blindly ahead, and the '`simplify`' command should really be re-named '`oversimplify`'.

he above proposal is only one mode of operation. he important issue here is *user control*. In some fashion, the user's expectations about the parameter values must be taken into account or prodded into existence. Facilities should be available for the user to establish a knowledge base of what is known about parameters and variables (assume commands, etc), but also facilities should be available for the system to indicate when that knowledge base needs to be expanded. Only the user knows how this is done best. Only the user knows enough about the problem context to decide what regions of the parameter space are relevant. So the issue for the CA community is how to provide the flexibility to get the desired behaviour. A 'mode switch' would go a long way towards this:

```
provisionprinting := complete;
< Thereafter provisions are automatically printed
and simplified with each result>
or
provisionprinting := partial;
< Thereafter provisions are printed only on request but
simplified as the computation proceeds >
or
provisionprinting := background;
< Thereafter provisions are printed on request, but not simpli
```