Report on: FR package by L. Bartholdi.

This package provides functionality for functionally recursive groups, i.e., groups defined by automata. Examples of such groups are the famous self-similar Grigorchuk and Gupta–Sidki groups.

Typically such groups are not finitely presentable. The package thus makes a new large class of interesting groups accessible to computer calculations. Semigroups and algebras defined by automata are also supported.

The package is relatively large, consisting of almost 30000 lines of GAP code and a manual of 140 pages. We checked a cross section of the commands provided, their documentation and accompanying examples for usability, correctness and consistency.

By and large, the package is expertly programmed and well documented. The manual is mostly clear and helpful. The package is well structured, as is reflected in the breakdown of the manual into chapters and the corresponding GAP code files.

However, there are some parts of the package which obviously are work in progress or have not (yet) been documented in the manual, most obviously when there are question marks in the manual. We read the manual version dated December 20, 2007, but the online version also contains them (see below).

One aspect of the package seems a bit peculiar. The package introduces and uses a new boolean value `maybe`, which is neither true or false. It is not clear to us what is the advantage of this newly introduced boolean value over the existing value `fail`, and feel that a package should not introduce such a value without further justification.

The package also comes with a large collection of useful example groups, monoids, and algebras as documented in Chapter 10 of the manual. It might be worth considering a separation of these examples from the library of GAP functions into a data library, and to introduce a new mechanism which accesses named examples in a way that represents the name of the object as a string rather than a global GAP function.

In summary, the package is of a very high standard. It is well-documented and thus easy to use, it produces correct results and provides good implementations of interesting data structures. We recommend that the package be accepted as a GAP package subject to

- a thorough clean-up which identifies and removes the experimental parts from the public version of the package,
- a justification or removal of the `maybe` object,
- the list of specific remarks below,

- and possibly a reorganization of the collection of examples.

Specific Remarks

Page numbers refer to the printed version of the manual on A4 paper.

Section 2.1 The definition of a *Mealy machine* on p. 11 is not the same as that o p. 16

3rd last paragraph: W missing before “is contracting. . . ”

Section 2.2 The part “. . . , initial state . . . ” mentioned in the manual does not appear in reality in the 6th and 8th example boxes.

There is a bracket missing in the 4th command in the 8th example box.

The 13th example box contains some typo.

Chapter 3 Intro: insert ’a’ in the 2nd sentence

Sections 3.x.x outputs like “< FR machine with alphabet [1 .. 2] . . . ” as stated in the manual actually produce “. . . [1,2]”

Section 3.3.2 Replace *machine* by m

Section 3.4.2 Delete ’s’ from “groups” and “semigroups” and replace “of” by “or” in the 3rd paragraph.

Section 4.1.2 An example with more than one *name* would be helpful.

Section 4.1.4 2nd paragraph: “elements” not “element”

The last output has square brackets rather than round ones.

Section 4.2.4 2nd paragraph fix “. . . between *Portrait* and *Portrait*”

Section 4.2.10 The output of FixedState is different in reality.

Section 4.2.13 2nd paragraph replace “in” by “is” in “In that case, the result . . . ”.

Section 5.1.4 Capitalise “mealy” and insert “machine” after it

Section 5.2.23 The function CofinalityClasses(e) does not exist.

Section 5.2.27 It does not become clear what this function does or is supposed to do.

Section 6.1.2 Fix the reference thing.

Section 7.2.12 Example missing.

Section 7.3.4 Example missing.

Section 7.3.5 “An object” not “A object”

Section 8.1.1 Another reference missing.

Section 8.1.2 “with” not “vith”

Section 9.1.1 Example missing.

Section 9.1.3 Example missing.

Section 9.1.5 Calling the function without arguments does not seem to work.

Section 9.2.1 Example missing.

Section 9.2.2 Example missing.

Section 10.1.4 IsRecurrent does not exist!

Section 10.1.8 The output of GrigorchukGroups is different in reality.

Section 10.1.10 Delete one “will”.

Section 10.1.19 Machines and groups have been mixed up here.

Section 10.1.21 The group should be renamed if possible. There are too many other spinal groups for this to make any sense.

Section 10.1.28 Insert “is” after “If G is an abelian group, CG”.

Section 10.4 Introduction missing and examples all missing.