Hap Programming – An experimental framework for objectifying the data structures of Hap

( development version of 10.02.2020 )

Marc Röder

Marc Röder  Email: marc.roeder(at)nuigalway.ie

Address: Marc Röder, Department of Mathematics, NUI Galway, Ireland
Abstract

This extension does not change the behaviour of Hap and is fully backwards-compatible. It is not a part of Hap and there is no guarantee that it will at any point be supported by Hap. Use at your own risk.

Copyright

© 2007 Marc Röder.

This package is distributed under the terms of the GNU General Public License version 2 or later (at your convenience). See the file LICENSE or http://www.gnu.org/copyleft/gpl.html

Acknowledgements

This work was supported by Marie Curie Grant No. MTKD-CT-2006-042685
# Contents

1 **Resolutions in Hap**  
   1.1 The Standard Representation HapResolutionRep  
   1.2 The HapLargeGroupResolutionRep Representation  
   
2 **Accessing and Manipulating Resolutions**  
   2.1 Representation-Independent Access Methods  
   2.2 Converting Between Representations  
   2.3 Special Methods for HapResolutionRep  
   2.4 The HapLargeGroupResolutionRep Representation  
   
3 **Contracting Homotopies**  
   3.1 The PartialContractingHomotopy Data Type  

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolutions in Hap</td>
<td>4</td>
</tr>
<tr>
<td>The Standard Representation HapResolutionRep</td>
<td>4</td>
</tr>
<tr>
<td>The HapLargeGroupResolutionRep Representation</td>
<td>5</td>
</tr>
<tr>
<td>Accessing and Manipulating Resolutions</td>
<td>6</td>
</tr>
<tr>
<td>Representation-Independent Access Methods</td>
<td>6</td>
</tr>
<tr>
<td>Converting Between Representations</td>
<td>8</td>
</tr>
<tr>
<td>Special Methods for HapResolutionRep</td>
<td>9</td>
</tr>
<tr>
<td>The HapLargeGroupResolutionRep Representation</td>
<td>10</td>
</tr>
<tr>
<td>Contracting Homotopies</td>
<td>13</td>
</tr>
<tr>
<td>The PartialContractingHomotopy Data Type</td>
<td>13</td>
</tr>
</tbody>
</table>
Chapter 1

Resolutions in Hap

This document is only concerned with the representation of resolutions in Hap. Note that it is not a part of Hap. The framework provided here is just an extension of Hap data types used in HAPcryst and HAPprime.

From now on, let \( G \) be a group and \( \cdots \rightarrow M_n \rightarrow M_{n-1} \rightarrow \cdots \rightarrow M_1 \rightarrow M_0 \rightarrow \mathbb{Z} \) be a resolution with free \( \mathbb{Z}G \) modules \( M_i \).

The elements of the modules \( M_i \) can be represented in different ways. This is what makes different representations for resolutions desirable. First, we will look at the standard representation (HapResolutionRep) as it is defined in Hap. After that, we will present another representation for infinite groups. Note that all non-standard representations must be sub-representations of the standard representation to ensure compatibility with Hap.

1.1 The Standard Representation HapResolutionRep

For every \( M_i \) we fix a basis and number its elements. Furthermore, it is assumed that we have a (partial) enumeration of the group of a resolution. In practice this is done by generating a lookup table on the fly.

In standard representation, the elements of the modules \( M_i \) are represented by lists -"words"- of pairs of integers. A letter \([i,g]\) of such a word consists of the number of a basis element \( i \) or \(-i\) for its additive inverse and a number \( g \) representing a group element.

A HapResolution in HapResolutionRep representation is a component object with the components

- group, a group of arbitrary type.
- elts, a (partial) list of (possibly duplicate) elements in \( G \). This list provides the "enumeration" of the group. Note that there are functions in Hap which assume that elts[1] is the identity element of \( G \).
- appendToElts\( (g) \) a function that appends the group element \( g \) to .elts. This is not documented in Hap 1.8.6 but seems to be required for infinite groups. This requirement might vanish in some later version of Hap [G. Ellis, private communication].
- dimension\( (k) \), a function which returns the \( \mathbb{Z}G \)-rank of the Module \( M_k \).
• boundary(k,j), a function which returns the image in $M_{k−1}$ of the $j$th free generator of $M_k$. Note that negative $j$ are valid as input as well. In this case the additive inverse of the boundary of the $j$th generator is returned.

• homotopy(k,[i,g]) a function which returns the image in $M_{k+1}$, under a contracting homotopy $M_k \rightarrow M_{k+1}$, of the element $[[i,g]]$ in $M_k$. The value of this might be fail. However, currently (version 1.8.4) some Hap functions assume that homotopy is a function without testing.

• properties, a list of pairs ["name","value"] "name" is a string and value is anything (boolean, number, string...). Every HapResolution (regardless of representation) has to have ["type","resolution"], ["length",length] where length is the length of the resolution and ["characteristic",char]. Currently (Hap 1.8.6), length must not be infinity. The values of these properties can be tested using the Hap function EvaluateProperty(resolution,propertyname).

Note that making HapResolution immutable will make the .elts component immutable. As this lookup table might change during calculations, we do not recommend using immutable resolutions (in any representation).

1.2 The HapLargeGroupResolutionRep Representation

In this sub-representation of the standard representation, the module elements in this resolution are lists of groupring elements. So the lookup table .elts is not used as long as no conversion to standard representation takes place. In addition to the components of a HapResolution, a resolution in large group representation has the following components:

• boundary2(resolution,term,gen), a function that returns the boundary of the gen$\text{th}$ generator of the term$\text{th}$ module.

• groupring the group ring of the resolution resolution.

• dimension2(resolution,term) a function that returns the dimension of the term$\text{th}$ module of the resolution resolution.

The effort of having two versions of boundary and dimension is necessary to keep the structure compatible with the usual Hap resolution.
Chapter 2

Accessing and Manipulating Resolutions

2.1 Representation-Independent Access Methods

All methods listed below take a `HapResolution` in any representation. If the other arguments are compatible with the representation of the resolution, the returned value will be in the form defined by this representation. If the other arguments are in a different representation, GAPs method selection is used via `TryNextMethod()` to find an applicable method (a suitable representation).

The idea behind this is that the results of computations have the same form as the input. And as all representations are sub-representations of the `HapResolutionRep` representation, input which is compatible with the `HapResolutionRep` representation is always valid.

Every new representation must support the functions of this section.

2.1.1 StrongestValidRepresentationForLetter

\[ \text{StrongestValidRepresentationForLetter}(\text{resolution}, \text{term}, \text{letter}) \]

**Returns:** filter

Finds the sub-representation of `HapResolutionRep` for which `letter` is a valid letter of the `term`th module of `resolution`. Note that `resolution` automatically is in some sub-representation of `HapResolutionRep`. This is mainly meant for debugging.

2.1.2 StrongestValidRepresentationForWord

\[ \text{StrongestValidRepresentationForWord}(\text{resolution}, \text{term}, \text{word}) \]

**Returns:** filter

Finds the sub-representation of `HapResolutionRep` for which `word` is a valid word of the `term`th module of `resolution`. Note that `resolution` automatically is in some sub-representation of `HapResolutionRep`. This is mainly meant for debugging.

2.1.3 PositionInGroupOfResolution

\[ \text{PositionInGroupOfResolution}(\text{resolution}, g) \]

\[ \text{PositionInGroupOfResolutionNC}(\text{resolution}, g) \]

**Returns:** positive integer

This returns the position of the group element `g` in the enumeration of the group of `resolution`. The NC version does not check if `g` really is an element of the group of `resolution`. 
2.1.4 IsValidGroupInt

▷ IsValidGroupInt(resolution, n)

Returns: boolean

Returns true if the nth element of the group of resolution is known.

2.1.5 GroupElementFromPosition

▷ GroupElementFromPosition(resolution, n)

Returns: group element or fail

Returns nth element of the group of resolution. If the nth element is not known, fail is returned.

2.1.6 MultiplyGroupElt

▷ MultiplyGroupElt(resolution, x, y)

Returns: positive integer or group element, depending on the type of x and y

If x and y are given in standard representation (i.e. as integers), this returns the position of the product of the group elements represented by the positive integers x and y.

If x and y are given in any other representation, the returned group element will also be represented in this way.

2.1.7 MultiplyFreeZGLetterWithGroupElt

▷ MultiplyFreeZGLetterWithGroupElt(resolution, letter, g)

Returns: A letter

Multiplies the letter letter with the group element g and returns the result. If resolution is in standard representation, g has to be an integer and letter has to be a pair of integer. If resolution is in any other representation, letter and g can be in a form compatible with that representation or in the standard form (in the latter case, the returned value will also have standard form).

2.1.8 MultiplyFreeZGWordWithGroupElt

▷ MultiplyFreeZGWordWithGroupElt(resolution, word, g)

Returns: A word

Multiplies the word word with the group element g and returns the result. If resolution is in standard representation, g has to be an integer and word has to be a list of pairs of integers. If resolution is in any other representation, word and g can be in a form compatible with that representation or in the standard form (in the latter case, the returned value will also have standard form).

2.1.9 BoundaryOfFreeZGLetter

▷ BoundaryOfFreeZGLetter(resolution, term, letter)

Returns: free ZG word (in the same representation as letter)

Calculates the boundary of the letter (word of length 1) letter of the termth module of resolution.

The returned value is a word of the term-1st module and comes in the same representation as letter.
2.1.10 BoundaryOfFreeZGWord

BoundaryOfFreeZGWord(resolution, term, word) (method)

Returns: free ZG word (in the same representation as letter)
Calculates the boundary of the word word of the term-th module of resolution.
The returned value is a word of the term-1st module and comes in the same representation as word.

2.2 Converting Between Representations

Four methods are provided to convert letters and words from standard representation to any other representation and back again.

2.2.1 ConvertStandardLetter

ConvertStandardLetter(resolution, term, letter) (method)
ConvertStandardLetterNC(resolution, term, letter) (method)

Returns: letter in the representation of resolution
Converts the letter letter in standard representation to the representation of resolution. The NC version does not check whether letter really is a letter in standard representation.

2.2.2 ConvertStandardWord

ConvertStandardWord(resolution, term, word) (method)
ConvertStandardWordNC(resolution, term, word) (method)

Returns: word in the representation of resolution
Converts the word word in standard representation to the representation of resolution. The NC version does not check whether word is a valid word in standard representation.

2.2.3 ConvertLetterToStandardRep

ConvertLetterToStandardRep(resolution, term, letter) (method)
ConvertLetterToStandardRepNC(resolution, term, letter) (method)

Returns: letter in standard representation
Converts the letter letter in the representation of resolution to the standard representation. The NC version does not check whether letter is a valid letter of resolution.

2.2.4 ConvertWordToStandardRep

ConvertWordToStandardRep(resolution, term, word) (method)
ConvertWordToStandardRepNC(resolution, term, word) (method)

Returns: word in standard representation
Converts the word word in the representation of resolution to the standard representation. The NC version does not check whether word is a valid word of resolution.
2.3 Special Methods for HapResolutionRep

Some methods explicitly require the input to be in the standard representation (HapResolutionRep). Two of these test if a word/letter is really in standard representation, the other ones are non-check versions of the universal methods.

2.3.1 IsFreeZGLetter

\[
\text{IsFreeZGLetter}(\text{resolution}, \text{term}, \text{letter})
\]

Returns: boolean
Checks if \text{letter} is an valid letter (word of length 1) in standard representation of the \text{term}th module of \text{resolution}.

2.3.2 IsFreeZGWord

\[
\text{IsFreeZGWord}(\text{resolution}, \text{term}, \text{word})
\]

Returns: boolean
Check if \text{word} is a valid word in large standard representation of the \text{term}th module in \text{resolution}.

2.3.3 MultiplyGroupEltsNC

\[
\text{MultiplyGroupEltsNC}(\text{resolution}, x, y)
\]

Returns: positive integer
Given positive integers \(x\) and \(y\), this returns the position of the product of the group elements represented by the positive integers \(x\) and \(x\). This assumes that all input is in standard representation and does not check the input.

2.3.4 MultiplyFreeZGLetterWithGroupEltNC

\[
\text{MultiplyFreeZGLetterWithGroupEltNC}(\text{resolution}, \text{letter}, g)
\]

Returns: A letter in standard representation
Multiplies the letter \text{letter} with the group element represented by the positive integer \(g\) and returns the result. The input is assumed to be in HapResolutionRep and is not checked.

2.3.5 MultiplyFreeZGWordWithGroupEltNC

\[
\text{MultiplyFreeZGWordWithGroupEltNC}(\text{resolution}, \text{word}, g)
\]

Returns: A letter in standard representation
Multiplies the word \text{word} with the group element represented by the positive integer \(g\) and returns the result. The input is assumed to be in HapResolutionRep and is not checked.

2.3.6 BoundaryOfFreeZGLetterNC

\[
\text{BoundaryOfFreeZGLetterNC}(\text{resolution}, \text{term}, \text{letter})
\]

Returns: free ZG word in standard representation
Calculates the boundary of the letter (word of length 1) \text{letter} of the \text{term}th module of \text{resolution}. The input is assumed to be in standard representation and not checked.
2.3.7 **BoundaryOfFreeZGWordNC**

- **BoundaryOfFreeZGWordNC**(resolution, term, word)  
  **Returns:** free ZG word in standard representation
  Calculates the boundary of the word word of the termth module of resolution. The input is assumed to be in standard representation and not checked.

2.4 **The HapLargeGroupResolutionRep Representation**

The large group representation has one additional component called groupring. Elements of the modules in a resolution are represented by lists of group ring elements. The length of the list corresponds to the dimension of the free module.

All methods for the generic representation do also work for the large group representation. Some of them are wrappers for special methods which do only work for this representation. The following list only contains the methods which are not already present in the generic representation.

Note that the input or the output of these functions does not comply with the standard representation.

2.4.1 **GroupRingOfResolution**

- **GroupRingOfResolution**(resolution)  
  **Returns:** group ring
  This returns the group ring of resolution. Note that by the way that group rings are handled in GAP, this is not equal to GroupRing(R,GroupOfResolution(resolution)) where R is the ring of the resolution.

2.4.2 **MultiplyGroupElts_LargeGroupRep**

- **MultiplyGroupElts_LargeGroupRep**(resolution, x, y)  
  **MultiplyGroupEltsNC_LargeGroupRep**(resolution, x, y)  
  **Returns:** group element
  Returns the product of x and y. The NC version does not check whether x and y are actually elements of the group of resolution.

2.4.3 **IsFreeZGLetterNoTermCheck_LargeGroupRep**

- **IsFreeZGLetterNoTermCheck_LargeGroupRep**(resolution, letter)  
  **Returns:** boolean
  Returns true if letter has the form of a letter (a module element with exactly one non-zero entry which has exactly one non-zero coefficient) a module of resolution in the HapLargeGroupResolution representation. Note that it is not tested if letter actually is a letter in any term of resolution.

2.4.4 **IsFreeZGWordNoTermCheck_LargeGroupRep**

- **IsFreeZGWordNoTermCheck_LargeGroupRep**(resolution, word)  
  **Returns:** boolean
Hap Programming – An experimental framework for objectifying the data structures of Hap

Returns true if word has the form of a word of a module of resolution in the HapLargeGroupResolution representation. Note that it is not tested if word actually is a word in any term of resolution.

2.4.5 IsFreeZGLetter_LargeGroupRep

\[ \text{IsFreeZGLetter}_\text{LargeGroupRep}(\text{resolution}, \text{term}, \text{letter}) \]

Returns: boolean

Returns true if and only if letter is a letter (a word of length 1) of the termth module of resolution in the hapLargeGroupResolution representation. I.e. it tests if letter is a module element with exactly one non-zero entry which has exactly one non-zero coefficient.

2.4.6 IsFreeZGWord_LargeGroupRep

\[ \text{IsFreeZGWord}_\text{LargeGroupRep}(\text{resolution}, \text{term}, \text{word}) \]

Tests if word is an element of the termth module of resolution.

2.4.7 MultiplyFreeZGLetterWithGroupElt_LargeGroupRep

\[ \text{MultiplyFreeZGLetterWithGroupElt}_\text{LargeGroupRep}(\text{resolution}, \text{letter}, \text{g}) \]

\[ \text{MultiplyFreeZGLetterWithGroupEltNC}_\text{LargeGroupRep}(\text{resolution}, \text{letter}, \text{g}) \]

Returns: free ZG letter in large group representation

Multiplies the letter letter with the group element g and returns the result. The NC version does not check whether g is an element of the group of resolution and letter can be a letter.

2.4.8 MultiplyFreeZGWordWithGroupElt_LargeGroupRep

\[ \text{MultiplyFreeZGWordWithGroupElt}_\text{LargeGroupRep}(\text{resolution}, \text{word}, \text{g}) \]

\[ \text{MultiplyFreeZGWordWithGroupEltNC}_\text{LargeGroupRep}(\text{resolution}, \text{word}, \text{g}) \]

Returns: free ZG word in large group representation

Multiplies the word word with the group element g and returns the result. The NC version does not check whether g is an element of the group of resolution and word can be a word.

2.4.9 GeneratorsOfModuleOfResolution_LargeGroupRep

\[ \text{GeneratorsOfModuleOfResolution}_\text{LargeGroupRep}(\text{resolution}, \text{term}) \]

Returns: list of letters/words in large group representation

Returns a set of generators for the termth module of resolution. The returned value is a list of vectors of group ring elements.

2.4.10 BoundaryOfGenerator_LargeGroupRep

\[ \text{BoundaryOfGenerator}_\text{LargeGroupRep}(\text{resolution}, \text{term}, \text{n}) \]

\[ \text{BoundaryOfGeneratorNC}_\text{LargeGroupRep}(\text{resolution}, \text{term}, \text{n}) \]

Returns: free ZG word in the large group representation
Returns the boundary of the $n$th generator of the $\text{term}^\text{th}$ module of $\text{resolution}$ as a word in the $n-1$st module (in large group representation). The NC version does not check whether there is a $\text{term}^\text{th}$ module and if it has at least $n$ generators.

### 2.4.11 BoundaryOfFreeZGLetterNC_LargeGroupRep

- **BoundaryOfFreeZGLetterNC_LargeGroupRep($\text{resolution, term, letter}$)** (method)

**Returns:** free ZG word in large group representation

Calculates the boundary of the letter $\text{letter}$ of the $\text{term}^\text{th}$ module of $\text{resolution}$ in large group representation. The NC version does not check whether $\text{letter}$ actually is a letter in the $\text{term}^\text{th}$ module.

### 2.4.12 BoundaryOfFreeZGWord_LargeGroupRep

- **BoundaryOfFreeZGWord_LargeGroupRep($\text{resolution, term, word}$)** (method)

**Returns:** free ZG word in large group representation

Calculates the boundary of the element $\text{word}$ of the $\text{term}^\text{th}$ module of $\text{resolution}$ in large group representation. The NC version does not check whether $\text{word}$ actually is a word in the $\text{term}^\text{th}$ module.
Chapter 3

Contracting Homotopies

3.1 The PartialContractingHomotopy Data Type

A partial contracting homotopy is a component object that knows the values of a contracting homotopy on some subspace of a resolution. It has two mandatory components:

- .resolution a HapResolution on which the contraction is defined.
- .knownPartOfHomotopy a list of Records with components .space and .map.

Let \( h \) be a contracting homotopy. The lookup table .knownPartOfHomotopy has one entry for each term of the resolution \( h\.resolution \) (that is, one more than \( \text{Length}(h\.resolution) \)).

The \( i \)th element of .knownPartOfHomotopy contains a record with components .space and .map where .space is a FreeZGWord of the \( i-1 \)st term of the resolution. The component .map is a list of length \( \text{Dimension}(h\.resolution)(i-1) \). The entries of this list are pairs \([g, im]\) where \( g \) represents a group element and \( im \) represents the image of the contraction. So the entry \([g, im]\) in the \( k \)th component of the list .map means that the \( k \)th free generator of the corresponding module multiplied with the group element represented by \( g \) is mapped to \( im \) under the partial contracting homotopy. Note that the data type of \( g \) or \( im \) are not fixed at this level. They must be specified by the sub representations. Also, \( im \) need not represent the actual image under a contracting homotopy. It is possible to just store a bit of information that is then used to generate the actual image.

As this is a very general data type, it has very few methods.

3.1.1 ResolutionOfContractingHomotopy

\[ \text{ResolutionOfContractingHomotopy}(homotopy) \]

\textbf{Returns:} A HapResolution

This returns the resolution of the homotopy \( homotopy \) (the component \( homotopy!.resolution \)).

3.1.2 PartialContractingHomotopyLookup

\[ \text{PartialContractingHomotopyLookup}(homotopy, \text{term}, \text{generator}, \text{groupel}) \]

\textbf{Returns:} The entry \( im \) of the corresponding lookup table
Looks up the known part of the contracting homotopy \textit{homotopy} and returns the corresponding image. More precisely, it returns the image of the \textit{generator}th generator times the group element represented by \textit{groupl} in term \textit{term} under the partial homotopy. The data type of this image depends on the representation of \textit{homotopy}.

\textit{term} has to be an integer and \textit{generator} a positive integer. \textit{groupl} only has to be an \texttt{Object}.

The NC version does not do any checks on the input. The other version checks if \textit{term} and \textit{generator} are sensible. It does not check \textit{groupl}. 