fKenzo: a user interface for computations in Algebraic Topology

Jónathan Heras Vicente

Departamento de Matemáticas y Computación
Universidad de La Rioja
Spain

December 9, 2009

Table of Contents

Table of Contents

Preliminaries

- Kenzo:
  - Symbolic Computation System devoted to Algebraic Topology
  - Homology groups unreachable by any other means
  - A Common Lisp package:

Preliminaries

- Kenzo:
  - Symbolic Computation System devoted to Algebraic Topology
  - Homology groups unreachable by any other means
  - A Common Lisp package:
    - Advantage: freedom given by Common Lisp remains available
    - Disadvantage: technicalities of Common Lisp remain present as well

Preliminaries

- Kenzo:
  - Symbolic Computation System devoted to Algebraic Topology
  - Homology groups unreachable by any other means
  - A Common Lisp package:
    - Advantage: freedom given by Common Lisp remains available
    - Disadvantage: technicalities of Common Lisp remain present as well
- fKenzo:
  - An extensible user interface:

Preliminaries

- Kenzo:
  - Symbolic Computation System devoted to Algebraic Topology
  - Homology groups unreachable by any other means
  - A Common Lisp package:
    - Advantage: freedom given by Common Lisp remains available
    - Disadvantage: technicalities of Common Lisp remain present as well
- fKenzo:
  - An extensible user interface:
    - **f**riendly access to **Kenzo** (**fKenzo**)
    - connect different Computer Algebra Systems and Theorem Provers

Preliminaries

- Kenzo:
  - Symbolic Computation System devoted to Algebraic Topology
  - Homology groups unreachable by any other means
  - A Common Lisp package:
    - Advantage: freedom given by Common Lisp remains available
    - Disadvantage: technicalities of Common Lisp remain present as well
- fKenzo:
  - An extensible user interface:
    - **f**riendly access to **Kenzo** (**fKenzo**)
    - connect different Computer Algebra Systems and Theorem Provers
  - Not necessary to know Common Lisp

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
  - Introducing the loop space Ω and classifying space B functors

May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
  - Introducing the loop space Ω and classifying space B functors

    May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).
  - There is a symmetry between both functors

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
    - Introducing the loop space Ω and classifying space B functors

      May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).
    - There is a symmetry between both functors
    - are the functors Ω and B inverse of each other?

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
  - Introducing the loop space Ω and classifying space B functors

    May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).
  - There is a symmetry between both functors
  - are the functors Ω and B inverse of each other?
  - Compare with Kenzo and fKenzo S^2 and $B\Omega S^2$ (Computing Homology groups)

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
  - Introducing the loop space Ω and classifying space B functors

    May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).
  - There is a symmetry between both functors
  - are the functors Ω and B inverse of each other?
  - Compare with Kenzo and fKenzo S^2 and $B\Omega S^2$ (Computing Homology groups)
  - Compare with Kenzo and fKenzo ΩS^2 and $\Omega B \Omega S^2$ (Computing Homology groups)

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
  - Introducing the loop space Ω and classifying space B functors

    May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).
  - There is a symmetry between both functors
  - are the functors Ω and B inverse of each other?
  - Compare with Kenzo and fKenzo S^2 and $B\Omega S^2$ (Computing Homology groups)
  - Compare with Kenzo and fKenzo ΩS^2 and $\Omega B\Omega S^2$ (Computing Homology groups)
  - That provides plausibility to the relation $B\Omega = id$

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
  - Introducing the loop space Ω and classifying space B functors

    May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).
  - There is a symmetry between both functors
  - are the functors Ω and B inverse of each other?
  - Compare with Kenzo and fKenzo S^2 and $B\Omega S^2$ (Computing Homology groups)
  - Compare with Kenzo and fKenzo ΩS^2 and $\Omega B\Omega S^2$ (Computing Homology groups)
  - That provides plausibility to the relation $B\Omega = id$
  - Demo

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
  - Introducing the loop space Ω and classifying space B functors

    May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).
  - There is a symmetry between both functors
  - are the functors Ω and B inverse of each other?
  - Compare with Kenzo and fKenzo S^2 and $B\Omega S^2$ (Computing Homology groups)
  - Compare with Kenzo and fKenzo ΩS^2 and $\Omega B\Omega S^2$ (Computing Homology groups)
  - That provides plausibility to the relation $B\Omega = id$
  - Demo
  - All the spaces (except S^2) are not of finite type

Kenzo and fKenzo as learning tools

- Consider a postgraduate course devoted to Algebraic Topology:
  - Introducing the loop space Ω and classifying space B functors

    May J.P., *Simplicial objects in Algebraic Topology*, Van Nostrand Mathematical Studies (11), (1967).
  - There is a symmetry between both functors
  - are the functors Ω and B inverse of each other?
  - Compare with Kenzo and fKenzo S^2 and $B\Omega S^2$ (Computing Homology groups)
  - Compare with Kenzo and fKenzo ΩS^2 and $\Omega B\Omega S^2$ (Computing Homology groups)
  - That provides plausibility to the relation $B\Omega = id$
  - Demo
  - All the spaces (except S^2) are not of finite type
  - Challenging task, but easy with our systems

Table of Contents

fKenzo mediated access

- fKenzo:

  user interface

fKenzo mediated access

- fKenzo:

  user interface + an intermediary layer

fKenzo mediated access

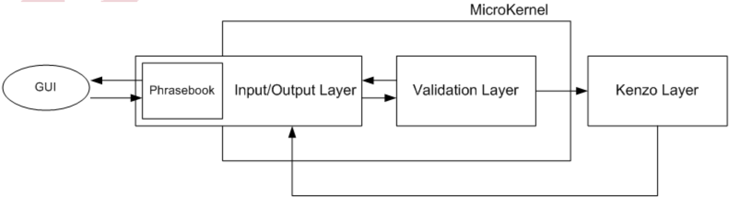
- fKenzo:

  user interface + an intermediary layer + Kenzo

fKenzo mediated access

- fKenzo:

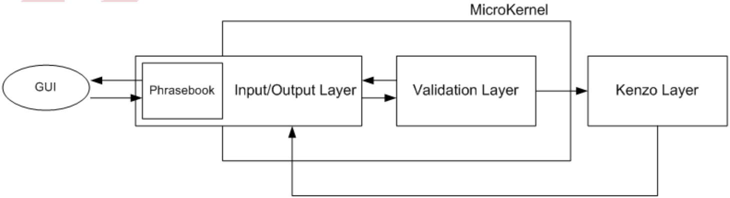
  user interface + an intermediary layer + Kenzo



fKenzo mediated access

- fKenzo:

  user interface + an intermediary layer + Kenzo

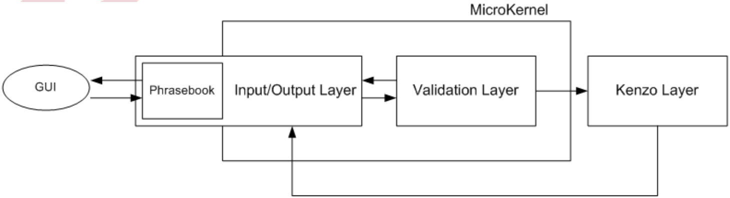


- Intelligent enhancements:
  - Controlling the input specification
  - Avoiding operations that will raise errors
  - Providing new operations

fKenzo mediated access

- fKenzo:

  user interface + an intermediary layer + Kenzo



- Intelligent enhancements:
  - Controlling the input specification
  - Avoiding operations that will raise errors
  - Providing new operations
- Demo

fKenzo as CAS and TP framework

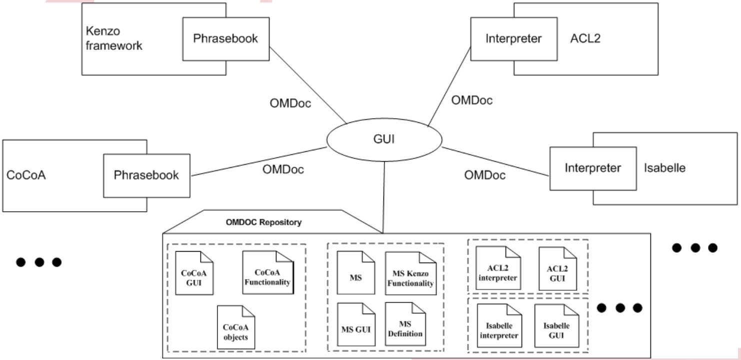
- Main Goal: friendly interaction with Kenzo

fKenzo as CAS and TP framework

- Main Goal: friendly interaction with Kenzo
- Secondary Goal: Framework to interact with Computer Algebra Systems and Theorems Provers related with Algebraic Topology

fKenzo as CAS and TP framework

- Main Goal: friendly interaction with Kenzo
- Secondary Goal: Framework to interact with Computer Algebra Systems and Theorems Provers related with Algebraic Topology

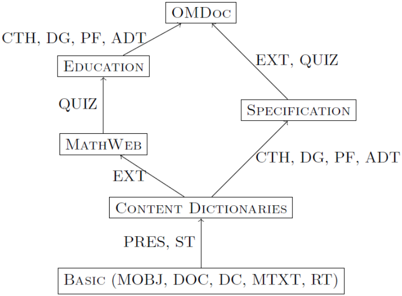


OMDoc Documents

- OMDoc format:
  - mathematical documents + knowledge encapsulate in them
  - three levels of information:
    - formulæ
    - mathematical statements
    - mathematical theories

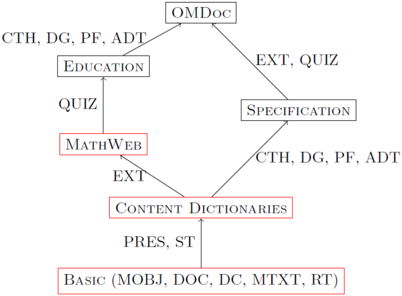
OMDoc Documents

- OMDoc format:
  - mathematical documents + knowledge encapsulate in them
  - three levels of information:
    - formulæ
    - mathematical statements
    - mathematical theories
- Sub-languages:



OMDoc Documents

- OMDoc format:
  - mathematical documents + knowledge encapsulate in them
  - three levels of information:
    - formulæ
    - mathematical statements
    - mathematical theories
- Sub-languages:



OMDoc Documents

- OMDoc format:
  - mathematical documents + knowledge encapsulate in them
  - three levels of information:
    - formulæ
    - mathematical statements
    - mathematical theories
- Sub-languages:
- 3 kinds of OMDoc documents:
  - Definition of Mathematical Structures
  - Functionality of the System
  - Definition of GUI structure

Definition of Mathematical Structures

- Goal:
  - Define the mathematical structures of the Computer Algebra Systems

Definition of Mathematical Structures

- Goal:
  - Define the mathematical structures of the Computer Algebra Systems
- Sub-language:
  - OMDoc Content Dictionaries Sub-language

Definition of Mathematical Structures

- Goal:
  - Define the mathematical structures of the Computer Algebra Systems
- Sub-language:
  - OMDoc Content Dictionaries Sub-language
- OpenMath CDs $\subset$ OMDoc CDs

Functionality of the System

- Defines the functionality of each module
- Several Aims:
  - Interaction with Computer Algebra Systems and Theorem Provers
  - Event handlers
  - Interpreter from OMDoc to Theorem Provers
  - ...

Functionality of the System

- Defines the functionality of each module
- Several Aims:
    - Interaction with Computer Algebra Systems and Theorem Provers
    - Event handlers
    - Interpreter from OMDoc to Theorem Provers
    - ...
- Sub-language:
    - MathWeb sub-language (EXT module)
        - <code> tag
        - Common Lisp code
        - code for different applications

Definition of GUI structure

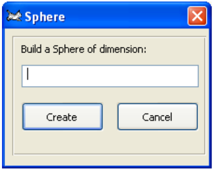
- Goal:
  - Define the structure of our GUI

Definition of GUI structure

- Goal:
  - Define the structure of our GUI
- Sub-language:
  - MathWeb sub-language
    - `<OMForeign>` tag
    - XUL

Definition of GUI structure

- Goal:
  - Define the structure of our GUI
- Sub-language:
  - MathWeb sub-language
    - `<OMForeign>` tag
    - XUL
- XUL:
  - Mozilla's XML-based user interface language
  - To build feature rich cross-platforms




```
<OMForeign>
  <window name="Sphere">
    <groupbox>
      <label value="Build a Sphere of dimension:"/>
      <textbox id="n" type="number" min="1" max="14"/>
      <hbox>
        <button label="Create" onclick="create-sphere-on-click"/>
        <button label="Cancel" onclick="cancel-sphere-on-click"/>
      </hbox>
    </groupbox>
  </window>
</OMForeign>
```

Gathering all the pieces

- Goal:
  - Glue all the parts of a module

Gathering all the pieces

- Goal:
  - Glue all the parts of a module
- Sub-language:
  - Basic OMDoc
    - `<omgroup>` tag

Gathering all the pieces

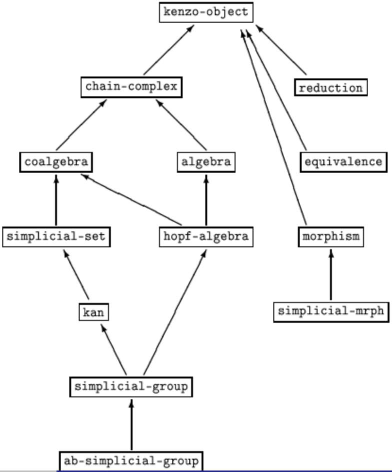
- Goal:
  - Glue all the parts of a module
- Sub-language:
  - Basic OMDoc
    - <omgroup> tag

```
<omdoc id="simplicial-sets.omdoc">
    ...
    <omgroup type="sequence">
        <ref xref="simplicial-sets-logic"/>
        <ref xref="simplicial-sets-presentation"/>
        <ref xref="simplicial-sets-conceptual-model"/>
    </omgroup>
    ...
</omdoc>
```

Table of Contents

Kenzo Content Dictionaries

- Kenzo works with the main mathematical structures used in Simplicial Algebraic Topology



Organization of CDs

- All the mathematical structures Kenzo works with are graded structures.
- Each graded structure is represented in Kenzo by means of the invariant of its underlying set.

```
inv: U nat  -> bool
     x  n   -> True  if x ∈ K^n
               False if x ∉ K^n
```

Specification of a Mathematical Structure $\langle \Sigma, Prop \rangle$ $\rightarrow$ Specification of a Mathematical Structure *Representation* $\langle \Sigma \cup \{inv\}, Prop \cup \{Prop_{inv}\} \rangle$

Organization of CDs

- All the mathematical structures Kenzo works with are graded structures.
- Each graded structure is represented in Kenzo by means of the invariant of its underlying set.

```
inv: U nat  -> bool
     x  n   -> True  if x ∈  K^n
               False if x ∉  K^n
```

Specification of a Mathematical Structure $\langle \Sigma, Prop \rangle$ $\rightarrow$ Specification of a Mathematical Structure *Representation* $\langle \Sigma \cup \{inv\}, Prop \cup \{Prop_{inv}\} \rangle$

- Each OpenMath Representation of a Mathematical Structure has:
  - Signature (in a Signature Dictionary)
  - Properties of the mathematical structure
  - Example
  - Predefined Objects (optional)

ACL2

- ACL2 (A Computational Logic for an Applicative Common Lisp)

ACL2

- ACL2 (A Computational Logic for an Applicative Common Lisp)
- ACL2
  - Programming Language
  - First-Order Logic
  - Theorem Prover

ACL2

- ACL2 (A Computational Logic for an Applicative Common Lisp)
- ACL2
  - Programming Language
  - First-Order Logic
  - Theorem Prover
- Proof techniques:
  - Simplification
  - Induction
  - *"The Method"*

ACL2

- ACL2 (A Computational Logic for an Applicative Common Lisp)
- ACL2
  - Programming Language
  - First-Order Logic
  - Theorem Prover
- Proof techniques:
  - Simplification
  - Induction
  - *"The Method"*
- Encapsulate: to the constrained introduction of new functions
  - Signatures
  - Properties
  - Witness

From a Kenzo CD to an ACL2 encapsulate

- Goal: Integration of Kenzo with ACL2 to increase the reliability of the Kenzo system
- ACL2 axiomatic structures: *encapsulate*

From a Kenzo CD to an ACL2 encapsulate

- Goal: Integration of Kenzo with ACL2 to increase the reliability of the Kenzo system
- ACL2 axiomatic structures: *encapsulate*
- Encapsulate:
  - Signatures
  - Properties
  - Witness

From a Kenzo CD to an ACL2 encapsulate

- Goal: Integration of Kenzo with ACL2 to increase the reliability of the Kenzo system
- ACL2 axiomatic structures: *encapsulate*
- Encapsulate:
  - Signatures
  - Properties
  - Witness
- Interpreter from Kenzo CDs to ACL2 Encapsulates

OMDoc CDs		ACL2 Encapsulates
Signatures	$\longrightarrow$	*Signatures*
Properties	$\longrightarrow$	*Properties*
Example	$\longrightarrow$	*Witness*

A case study: simplicial sets

Definition

A simplicial set K, is a disjoint union $K = \bigcup_{q \geq 0} K^q$, where the K^q are sets, together with functions

$$
\begin{array}{lll}
\partial_i^q : K^q \rightarrow K^{q-1}, & q > 0, & i = 0, \ldots, q, \\
\eta_i^q : K^q \rightarrow K^{q+1}, & q \geq 0, & i = 0, \ldots, q,
\end{array}
$$

subject to relations

$$
\begin{array}{lllr}
\partial_i^{q-1}\partial_j^q & = & \partial_{j-1}^{q-1}\partial_i^q, & i < j \\
\eta_i^{q+1}\eta_j^q & = & \eta_j^{q+1}\eta_{i-1}^q, & i > j \\
\partial_i^{q+1}\eta_j^q & = & \eta_{j-1}^{q-1}\partial_i^q, & i < j \\
\partial_i^{q+1}\eta_i^q & = & \partial_{i+1}^{q+1}\eta_i^q, & \text{identity} \\
\partial_i^{q+1}\eta_j^q & = & \eta_j^{q-1}\partial_{i-1}^q, & i > j+1
\end{array}
$$

A case study: simplicial sets

```
<Signature name="simplicial-set">
    <OMOBJ xmlns="http://www.openmath.org/OpenMath">
        <OMA>
            <OMS name="mapsto" cd="sts"/>
            <OMA id="inv">
                <OMS cd="sts" name="mapsto"/>
                <OMV name="Simplicial-Set-Element"/>
                <OMV name="PositiveInteger"/>
                <OMS cd="setname2" name="boolean"/>
            </OMA>
            <OMA id="face">
                <OMS cd="sts" name="mapsto"/>
                <OMV name="Simplicial-Set-Element"/>
                <OMV name="PositiveInteger"/>
                <OMV name="PositiveInteger"/>
                <OMV name="Simplicial-Set-Element"/>
            </OMA>
            <OMA id="degeneracy">
                <OMS cd="sts" name="mapsto"/>
                <OMV name="Simplicial-Set-Element"/>
                <OMV name="PositiveInteger"/>
                <OMV name="PositiveInteger"/>
                <OMV name="Simplicial-Set-Element"/>
            </OMA>
            <OMV name="Simplicial-Set"/>
        </OMA>
    </OMOBJ>
</Signature>
```

A case study: simplicial sets

```
(((face * * *) => *)
 ((degeneracy * * *) => *)
 ((inv * *) => *))
```

A case study: simplicial sets

```
<CMP> The face operator is well defined </CMP>
<FMP>
...
<OMA>
    <OMS cd="logic1" name="implies"/>
    <OMA>
        <OMV name="inv"/>
        <OMV name="x"/>
        <OMV name="q"/>
    </OMA>
    <OMA>
        <OMV name="inv"/>
        <OMA>
            <OMV name="face"/>
            <OMV name="x"/>
            <OMV name="i"/>
            <OMV name="q"/>
        </OMA>
        <OMA>
            <OMS cd="arith1" name="minus"/>
            <OMV name="q"/>
            <OMI>1</OMI>
        </OMA>
    </OMA>
</OMA>
...
</FMP>
```

A case study: simplicial sets

```
; The face operator is well defined
(defthm prop1
(implies (inv x q) (inv (face x i q) (- q 1))))
```

A case study: simplicial sets

```
<example>
...
    <OMBIND>
        <OMS name="face"/>
        <OMBVAR>
            <OMV name="x"/>
            <OMV name="i"/>
            <OMV name="q"/>
        </OMBVAR>
        <OMS cd="list" name="nil"/>
    </OMBIND>
...
</example>
```

A case study: simplicial sets

```
(local (defun face (x i q)
 (declare (IGNORE x i q))
 nil))
```

A case study: simplicial sets

Demo

GAP, Kenzo and ACL2

- GAP:
  - System for computational discrete algebra
  - Focuses on Group Theory
  - HAP: A Homological Algebra library

GAP, Kenzo and ACL2

- GAP:
  - System for computational discrete algebra
  - Focuses on Group Theory
  - HAP: A Homological Algebra library
- Kenzo:
  - Symbolic Computation System devoted to Algebraic Topology
  - Homology groups unreachable by any other means

GAP, Kenzo and ACL2

- GAP:
  - System for computational discrete algebra
  - Focuses on Group Theory
  - HAP: A Homological Algebra library
- Kenzo:
  - Symbolic Computation System devoted to Algebraic Topology
  - Homology groups unreachable by any other means
- ACL2:
  - Theorem Prover
  - First order logic
  - Based on Common Lisp

Connecting GAP, Kenzo and ACL2 I

- Aim:

  Obtain resolutions of Cyclic Groups from HAP

Connecting GAP, Kenzo and ACL2 I

- Aim:

  Obtain resolutions of Cyclic Groups from HAP
  Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo
  2. Build the cyclic group in GAP

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo
  2. Build the cyclic group in GAP
  3. Build a resolution of the cyclic group using HAP package

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo
  2. Build the cyclic group in GAP
  3. Build a resolution of the cyclic group using HAP package
  4. Export the resolution in a file using OpenMath format from GAP

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo
  2. Build the cyclic group in GAP
  3. Build a resolution of the cyclic group using HAP package
  4. Export the resolution in a file using OpenMath format from GAP
  5. Import the resolution to Kenzo

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo
  2. Build the cyclic group in GAP
  3. Build a resolution of the cyclic group using HAP package
  4. Export the resolution in a file using OpenMath format from GAP
  5. Import the resolution to Kenzo
  6. Build the cyclic group in Kenzo

Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo
  2. Build the cyclic group in GAP
  3. Build a resolution of the cyclic group using HAP package
  4. Export the resolution in a file using OpenMath format from GAP
  5. Import the resolution to Kenzo
  6. Build the cyclic group in Kenzo
  7. Assign the resolution to the correspondent cyclic group in Kenzo

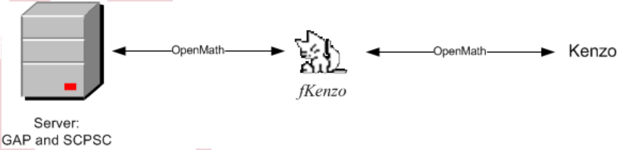
Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo
  2. Build the cyclic group in GAP
  3. Build a resolution of the cyclic group using HAP package
  4. Export the resolution in a file using OpenMath format from GAP
  5. Import the resolution to Kenzo
  6. Build the cyclic group in Kenzo
  7. Assign the resolution to the correspondent cyclic group in Kenzo
  8. Build the space $K(G, 1)$ where G is the cyclic group in Kenzo

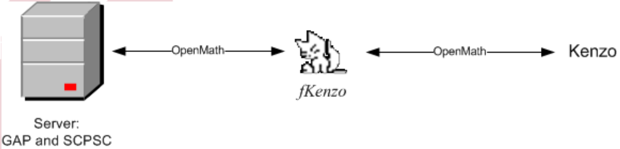
Connecting GAP, Kenzo and ACL2 I

- Aim:
  - Obtain resolutions of Cyclic Groups from HAP
  - Use these resolutions in Kenzo to build Eilenberg-MacLane Spaces and to compute homology groups
  - Certify that the cyclic groups of Kenzo are Abelian Groups
- Working manually:
  1. Load the necessary packages and files in GAP and Kenzo
  2. Build the cyclic group in GAP
  3. Build a resolution of the cyclic group using HAP package
  4. Export the resolution in a file using OpenMath format from GAP
  5. Import the resolution to Kenzo
  6. Build the cyclic group in Kenzo
  7. Assign the resolution to the correspondent cyclic group in Kenzo
  8. Build the space $K(G,1)$ where G is the cyclic group in Kenzo
  9. Compute $H_n(K(G,1))$ in Kenzo

Connecting GAP, Kenzo and ACL2 II

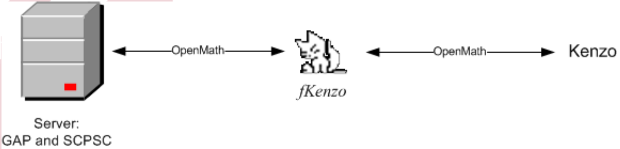


Connecting GAP, Kenzo and ACL2 II



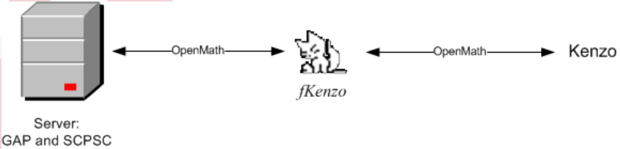
- SCSCP: Remote Procedure Call Framework based on OpenMath Language

Connecting GAP, Kenzo and ACL2 II



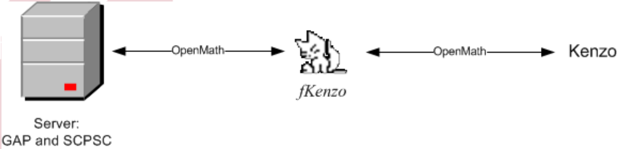
- SCSCP: Remote Procedure Call Framework based on OpenMath Language
- Working in *fKenzo*:

Connecting GAP, Kenzo and ACL2 II



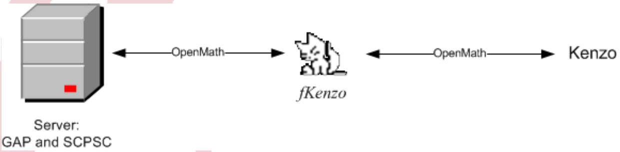
- SCSCP: Remote Procedure Call Framework based on OpenMath Language
- Working in *fKenzo*:
  1. Load the GAP-Kenzo-ACL2 module in our system

Connecting GAP, Kenzo and ACL2 II



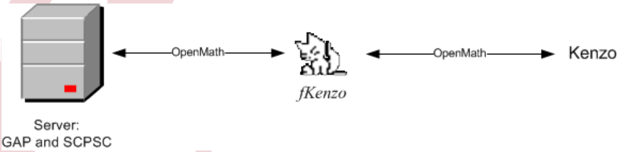
- SCSCP: Remote Procedure Call Framework based on OpenMath Language
- Working in *fKenzo*:
    1. Load the GAP-Kenzo-ACL2 module in our system
    2. Build the cyclic group

Connecting GAP, Kenzo and ACL2 II



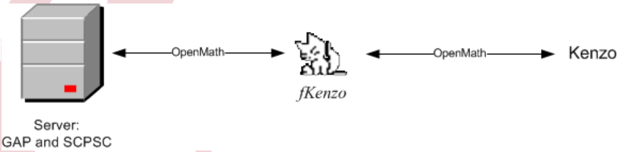
- SCSCP: Remote Procedure Call Framework based on OpenMath Language
- Working in *fKenzo*:
  1. Load the GAP-Kenzo-ACL2 module in our system
  2. Build the cyclic group
  3. Build the space $K(G, 1)$ where G is the cyclic group

Connecting GAP, Kenzo and ACL2 II



- SCSCP: Remote Procedure Call Framework based on OpenMath Language
- Working in *fKenzo*:
  1. Load the GAP-Kenzo-ACL2 module in our system
  2. Build the cyclic group
  3. Build the space $K(G, 1)$ where G is the cyclic group
  4. Compute $H_n(K(G, 1))$

Connecting GAP, Kenzo and ACL2 II



- SCSCP: Remote Procedure Call Framework based on OpenMath Language
- Working in *fKenzo*:
  1. Load the GAP-Kenzo-ACL2 module in our system
  2. Build the cyclic group
  3. Build the space $K(G, 1)$ where G is the cyclic group
  4. Compute $H_n(K(G, 1))$
- Demo

Connecting GAP, Kenzo and ACL2 III

- Goal:
  - Improve the reliability of our system
  - Certification of the cyclic groups implemented in Kenzo

Connecting GAP, Kenzo and ACL2 III

- Goal:
  - Improve the reliability of our system
  - Certification of the cyclic groups implemented in Kenzo
- Steps to certify cyclic groups:

Connecting GAP, Kenzo and ACL2 III

- Goal:
  - Improve the reliability of our system
  - Certification of the cyclic groups implemented in Kenzo
- Steps to certify cyclic groups:
  - Interpreter loads the definition of Abelian Groups from an OMDoc file

Connecting GAP, Kenzo and ACL2 III

- Goal:
  - Improve the reliability of our system
  - Certification of the cyclic groups implemented in Kenzo
- Steps to certify cyclic groups:
  - Interpreter loads the definition of Abelian Groups from an OMDoc file
  - Interpreter loads the definition of Cyclic Groups from an OMDoc file

Connecting GAP, Kenzo and ACL2 III

- Goal:
  - Improve the reliability of our system
  - Certification of the cyclic groups implemented in Kenzo
- Steps to certify cyclic groups:
  - Interpreter loads the definition of Abelian Groups from an OMDoc file
  - Interpreter loads the definition of Cyclic Groups from an OMDoc file
  - An instance of the previous definition is automatically generated for the selected cyclic group

Connecting GAP, Kenzo and ACL2 III

- Goal:
  - Improve the reliability of our system
  - Certification of the cyclic groups implemented in Kenzo
- Steps to certify cyclic groups:
  - Interpreter loads the definition of Abelian Groups from an OMDoc file
  - Interpreter loads the definition of Cyclic Groups from an OMDoc file
  - An instance of the previous definition is automatically generated for the selected cyclic group
  - The cyclic group is certified

Connecting GAP, Kenzo and ACL2 III

- Goal:
  - Improve the reliability of our system
  - Certification of the cyclic groups implemented in Kenzo
- Steps to certify cyclic groups:
  - Interpreter loads the definition of Abelian Groups from an OMDoc file
  - Interpreter loads the definition of Cyclic Groups from an OMDoc file
  - An instance of the previous definition is automatically generated for the selected cyclic group
  - The cyclic group is certified
- Demo

Table of Contents

Conclusions

- Friendly + Mediated access to Kenzo
- Framework to interact with several CAS and TP related with Algebraic Topology
- Underlying lessons to interact with different systems:

Conclusions

- Friendly + Mediated access to Kenzo
- Framework to interact with several CAS and TP related with Algebraic Topology
- Underlying lessons to interact with different systems:
  - Example: Interaction with Mathematica

Conclusions

- Friendly + Mediated access to Kenzo
- Framework to interact with several CAS and TP related with Algebraic Topology
- Underlying lessons to interact with different systems:
  - Example: Interaction with Mathematica
  - Definition of Mathematical Structures: Fractals

Conclusions

- Friendly + Mediated access to Kenzo
- Framework to interact with several CAS and TP related with Algebraic Topology
- Underlying lessons to interact with different systems:
  - Example: Interaction with Mathematica
  - Definition of Mathematical Structures: Fractals
  - Definition of Interaction with Mathematica

Conclusions

- Friendly + Mediated access to Kenzo
- Framework to interact with several CAS and TP related with Algebraic Topology
- Underlying lessons to interact with different systems:
  - Example: Interaction with Mathematica
  - Definition of Mathematical Structures: Fractals
  - Definition of Interaction with Mathematica
  - Definition of GUI structure

Conclusions

- Friendly + Mediated access to Kenzo
- Framework to interact with several CAS and TP related with Algebraic Topology
- Underlying lessons to interact with different systems:
  - Example: Interaction with Mathematica
  - Definition of Mathematical Structures: Fractals
  - Definition of Interaction with Mathematica
  - Definition of GUI structure
  - Demo

`Mathematica.wmv`

Further Work

- Find a suitable way of editing spaces
- Extend fKenzo:
    - Kenzo evolves
    - Connection with other CAS and TP.

fKenzo: a user interface for computations in Algebraic Topology

Jónathan Heras Vicente

Departamento de Matemáticas y Computación
Universidad de La Rioja
Spain

December 9, 2009