Posets and words in Sage-Combinat

Franco V. Saliola
(saliola@gmail.com)

Institut Gaspard Monge
Université de Marne-la-Vallée
France

Sage Days 10
11 October 2008
A *word* is a sequence of elements — called *letters* — from a set $A$. 

Combinatorics on Words
A **word** is a sequence of elements — called **letters** — from a set $A$.

Thus, words arise in several areas of mathematics and the sciences:

- *word problem* in semigroup and group theory;
- *permutations as words* in combinatorics;
- *automatic sequences* in number theory;
- *DNA* in biology;
- *words* in linguistics;
- *etc.*
Combinatorics on Words

Goal: develop tools for studying words.
Combinatorics on Words

Goal: develop tools for studying words.

Examples: want efficient algorithms and data structures for
- searching text;
- pattern recognition;
- inferring combinatorial, probabilistic and statistical properties;
- counting distinct factors;
- storing and retrieving factors;
- factorizations (Lyndon, Crochemore, . . . );
- ....
History

Francois Bergeron is invited to Sage Days 7, but couldn't go.

Me: "Me! Me! I want to go! Tell them to invite me! Pay for my trip! I promise to give a talk about Sage when I get back!"

I go to Sage Days 7, and get really excited about what I see.

I give a very enthusiastic talk about it when I get back.

Others get excited too!
François Bergeron is invited to Sage Days 7, but couldn’t go.
François Bergeron is invited to Sage Days 7, but couldn’t go.

Me: “Me! Me! I want to go! Tell them to invite me! Pay for my trip! I promise to give a talk about Sage when I get back!”
François Bergeron is invited to Sage Days 7, but couldn’t go.

Me: “Me! Me! I want to go! Tell them to invite me! Pay for my trip! I promise to give a talk about Sage when I get back!”

I go to Sage Days 7, and get really excited about what I see.
François Bergeron is invited to Sage Days 7, but couldn’t go.

Me: “Me! Me! I want to go! Tell them to invite me! Pay for my trip! I promise to give a talk about Sage when I get back!”

I go to Sage Days 7, and get really excited about what I see.

I give a very enthusiastic talk about it when I get back.
Françcois Bergeron is invited to Sage Days 7, but couldn’t go.

Me: “Me! Me! I want to go! Tell them to invite me! Pay for my trip! I promise to give a talk about Sage when I get back!”

I go to Sage Days 7, and get really excited about what I see.

I give a very enthusiastic talk about it when I get back.

Others get excited too!
History

- Srečko Brlek wants to base a Combinatorics on Words package on Sage.

May 2008: sage-words is born. Developers: Arnauld Bergeron, Sébastien Labbé, Amy Glen and me.

Sept. 2008: People at MLV also want to get in on the action; we'll be discussing more about what should be included soon.
Srečko Brlek wants to base a Combinatorics on Words package on Sage.

Reasons: algorithms are useful; much software exists, but it is fragmented, not unified, unmaintained; no nice interface.
Srečko Brlek wants to base a Combinatorics on Words package on Sage.

Reasons: algorithms are useful; much software exists, but it is fragmented, not unified, unmaintained; no nice interface.

He puts his (grant) money where his mouth is: hires people to work on it over the summer.
History

- Srečko Brlek wants to base a Combinatorics on Words package on Sage.

- Reasons: algorithms are useful; much software exists, but it is fragmented, not unified, unmaintained; no nice interface.

- He puts his (grant) money where his mouth is: hires people to work on it over the summer.

- May 2008: sage-words is born. Developers: Arnauld Bergeron, Sébastien Labbé, Amy Glen and me.
Srečko Brlek wants to base a Combinatorics on Words package on Sage.

Reasons: algorithms are useful; much software exists, but it is fragmented, not unified, unmaintained; no nice interface.

He puts his (grant) money where his mouth is: hires people to work on it over the summer.

May 2008: sage-words is born. Developers: Arnauld Bergeron, Sébastien Labbé, Amy Glen and me.

Sept. 2008: People at MLV also want to get in on the action; we’ll be discussing more about what should be included soon.
Pre-existing software?

### Applied Combinatorics on Words: Contents

- **Full text (compressed PostScript: 2.5 MB)**
- Last Modification: June 23, 2004

#### Presentation
- Contents and presentation

#### Core algorithms
- **Algorithms on words**
- **Structures for indexes**

#### Natural language processing
- **Symbolic natural language processing**
- **Statistical natural language processing**

#### Bioinformatics
- **Network expression inference**
- **Statistics on words with applications to biological sequences**

#### Algorithms
- **Analytic approach to pattern matching**
- **Periodic structures in words**

#### Mathematics
- **Counting, coding and sampling with words**
- **Words in number theory**

#### References
- **Bibliography and index**

Authors:
- Jean Berstel and Dominique Perrin
- Maxime Crochemore
- Eric Laporte
- Mehryar Mohri
- Marie-France Sagot and Nadia Pisanti
- Gesine Reinert, Sophie Schbath and Michael S. Waterman
- Philippe Jacque
- Wojciech Szpankowski
- Roman Kolpakov and Gregory Kucherov
- Dominique Poulalhon and Gilles Schaeffer
- Jean-Paul Allouche and Valérie Berthé
Pre-existing software?

**Applied Combinatorics on Words: Implementation of algorithms**

**Algorithms on words**

A set of *computer programs in Java* for the algorithms of Chapter 1 is available in a preliminary form. They can be freely copied and used with the mention of their origin. The idea is to present an illustration of a possible effective implementation rather than fine tuned optimal software. No guarantee at all is given for correctness. A *documentation* is in progress.

**Structures for indexes**

*Computer programs in Java and C* for the algorithms of Chapter 2 and for other text processing algorithms are available.

**Statistical natural language processing**

Programs for the algorithms of this chapter are available at:


**Statistics on words with applications to biological sequences**

Computations of words with exceptional frequency in DNA were performed with programs available at: [http://www-mig.jouy.inra.fr/ssb/rmes/](http://www-mig.jouy.inra.fr/ssb/rmes/)

**Periodic structures in words**

Concerning this chapter, principal algorithms have been implemented in the mreps software [http://www.loria.fr/mreps/](http://www.loria.fr/mreps/).
What does sage-words do?

Demo
What’s next for sage-words?

Current version: 0.3. Before 1.0 design may change drastically.
Want to merge into sage-combinat; I want to use CombinatorialClass, Streams, etc.
New features need to be added.
Morphisms should be able to map into other monoids.
Better algorithms need to be implemented.
Need to decide an best backend: suffix trees? other?
Cythonize!
What’s next for sage-words?

- Current version: 0.3. Before 1.0 design may change drastically.
What’s next for sage-words?

- Current version: 0.3. Before 1.0 design may change drastically.

- Want to merge into sage-combinat; I want to use CombinatorialClass, Streams, etc.
What’s next for sage-words?

- Current version: 0.3. Before 1.0 design may change drastically.
- Want to merge into sage-combinat; I want to use CombinatorialClass, Streams, etc.
- New features need to be added.
What’s next for sage-words?

- Current version: 0.3. Before 1.0 design may change drastically.
- Want to merge into sage-combinat; I want to use CombinatorialClass, Streams, etc.
- New features need to be added.
- Morphisms should be able to map into other monoids.
What’s next for sage-words?

- Current version: 0.3. Before 1.0 design may change drastically.
- Want to merge into sage-combinat; I want to use CombinatorialClass, Streams, etc.
- New features need to be added.
- Morphisms should be able to map into other monoids.
- Better algorithms need to be implemented.
What’s next for sage-words?

- Current version: 0.3. Before 1.0 design may change drastically.
- Want to merge into sage-combinat; I want to use CombinatorialClass, Streams, etc.
- New features need to be added.
- Morphisms should be able to map into other monoids.
- Better algorithms need to be implemented.
- Need to decide an best backend: suffix trees? other?

Cythonize!
What’s next for sage-words?

- Current version: 0.3. Before 1.0 design may change drastically.
- Want to merge into sage-combinat; I want to use CombinatorialClass, Streams, etc.
- New features need to be added.
- Morphisms should be able to map into other monoids.
- Better algorithms need to be implemented.
- Need to decide an best backend: suffix trees? other?
- Cythonize!
Posets in sage-combinat

A poset is a set with a partial-order.

Current code is for working with finite posets via the Hasse diagram (that is, it uses DiGraphs as the backend).
Posets in sage-combinat

- A **poset** is a set with a **partial-order**.
A **poset** is a set with a **partial-order**.

Current code is for working with finite posets via the Hasse diagram (that is, it uses DiGraphs as the backend).
Posets in sage-combinat

- A **poset** is a set with a **partial-order**.
- Current code is for working with finite posets via the Hasse diagram (that is, it uses DiGraphs as the backend).

DEMO
Posets in sage-combinat

Lots to do here:
Posets in sage-combinat

Lots to do here:

- Improve the current code: finite poset generator; additional input methods; fix bugs & docstrings; . . . .
Posets in sage-combinat

Lots to do here:

- Improve the current code: finite poset generator; additional input methods; fix bugs & docstrings; . . . .

- Rethink the design.

Want to be able to work with posets without storing the Hasse diagram (with very big, like infinite, posets).

Posets with EL-labellings.

Want to have operations defined on posets: $\ast$, $\oplus$, $\div$, . . . .

Very important: Need to be able to take objects (say, permutations) and turn them into posets easily.
Posets in sage-combinat

Lots to do here:

- Improve the current code: finite poset generator; additional input methods; fix bugs & docstrings; . . . .

- Rethink the design.

- Want to be able to work with posets without storing the Hasse diagram (with very big, like infinite, posets).
Posets in sage-combinat

Lots to do here:

- Improve the current code: finite poset generator; additional input methods; fix bugs & docstrings; . . . .

- Rethink the design.

- Want to be able to work with posets without storing the Hasse diagram (with very big, like infinite, posets).

- Posets with EL-labellings.
Posets in sage-combinat

Lots to do here:

- Improve the current code: finite poset generator; additional input methods; fix bugs & docstrings; . . . .

- Rethink the design.

- Want to be able to work with posets without storing the Hasse diagram (with very big, like infinite, posets).

- Posets with EL-labellings.

- Want to have operations defined on posets: *, +, /, . . . .
Posets in sage-combinat

Lots to do here:

- Improve the current code: finite poset generator; additional input methods; fix bugs & docstrings; . . . .

- Rethink the design.

- Want to be able to work with posets without storing the Hasse diagram (with very big, like infinite, posets).

- Posets with EL-labellings.

- Want to have operations defined on posets: *, +, /, . . . .

- Very important: Need to be able to take objects (say, permutations) and turn them into posets easily.