

# GALACTIC architecture

The **GALACTIC** Organization <contact@thegalactic.org>

2018-2023



---

<sup>1</sup>© 2018-2023 the **GALACTIC** Organization. This document is licensed under CC-by-nc-nd (<https://creativecommons.org/licenses/by-nc-nd/4.0/deed.en>)

## Acronym

GALACTIC stands for

**G**Alois  
**L**Attices,  
**C**oncept  
**T**heory,  
**I**mplicational systems and  
**C**losures.



*E. Galois*



## Purpose

### GALACTIC framework

Develop a framework on:

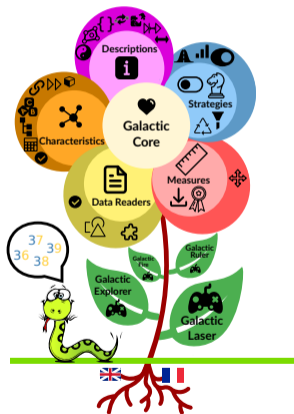
- ▶ **Lattice** theory<sup>a</sup>
- ▶ **Formal Concept Analysis**<sup>b</sup>.

---

<sup>a</sup>BARBUT, Marc et MONJARDET, Bernard. Ordre et classification, vols. 1 and 2. Hachette, Paris, France, 1970.

<sup>b</sup>GANTER, Bernhard et WILLE, Rudolf. Formal concept analysis: mathematical foundations. Springer Science & Business Media, 1999.

## Architecture

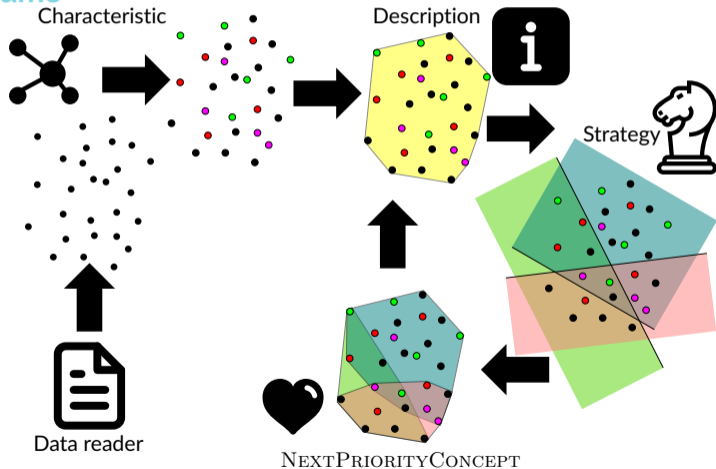


## Written in python, Fully extensible

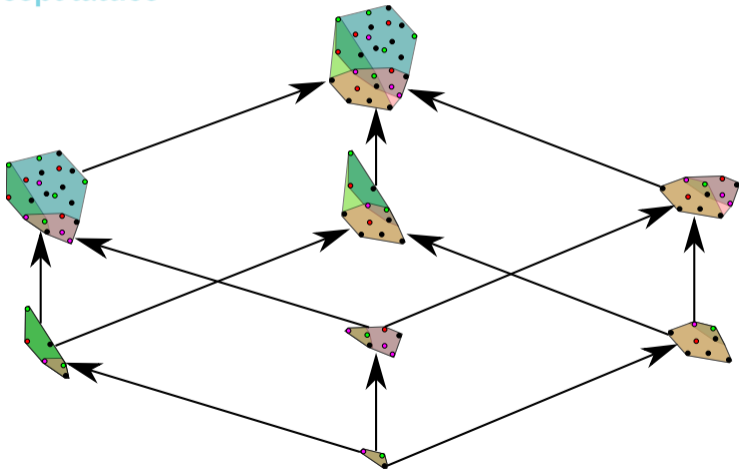
The **GALACTIC** framework is architecturally designed with:

- ▶ a core library
- ▶ applications
- ▶ characteristic plugins
- ▶ description plugins
- ▶ strategy plugins
- ▶ measure plugins
- ▶ data reader plugins
- ▶ localization plugins

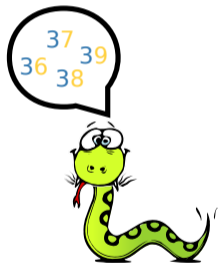
## Resume



## Concept lattice



## Architecture



### Core

The **GALACTIC core** defines the core library, it contains the basic operations and data structures and it implements the new generation algorithm (NEXTPRIORITYCONCEPT) inspired from pattern structures.

## Characteristic Plugins

### Definition

**Characteristic** plugins define characteristics such as numerical characteristics, boolean characteristics.



Existing characteristic plugins:

- ▶  *Boolean* characteristics;
- ▶  *Numerical* characteristics;
- ▶  *Categorical* characteristics;
- ▶  *String* characteristics;






## Characteristic Plugins

### Definition

**Characteristic** plugins define characteristics such as numerical characteristics, boolean characteristics.



Existing characteristic plugins:

- ▶  *Chain* characteristics;
- ▶  *Sequence* characteristics.
- ▶  *Triadic* characteristics.

In preparation:

- ▶  *Graph* characteristics.

## Description Plugins

### Definition

**Description** plugins define predicates and description spaces used to represent and to define data precisely.



Existing description plugins:

- ▶ ✓ *Boolean* descriptions;
- ▶ ☯ *Logical* descriptions;
- ▶ {} *Categorical* descriptions;
- ▶ 🕸 *Numerical* descriptions;
- ▶ ↻ *String* descriptions using regex;
- ▶ ↔ *String* descriptions using distances;





## Description Plugins

### Definition


**Description** plugins define predicates and description spaces used to represent and to define data precisely.



Existing description plugins:

- ▶  *Chain* descriptions;
- ▶  *Sequence* descriptions;
- ▶  *Sequence* descriptions using distances;
- ▶  *Triadic* descriptions.

In preparation:

- ▶  *Graph* descriptions.

## Strategy Plugins

### Definition

**Strategy** plugins define the way used to explore data, it uses descriptions to generate predecessors for each concept in the lattice.



Existing strategy plugins:

- ▶  *Boolean* strategy;
- ▶  *Logical* strategy;
- ▶  *Categorical* strategy;
- ▶  *Numerical basic* strategy;
- ▶  *Numerical quantile* strategy;
- ▶  *String* strategy;
- ▶  *String* strategy using distances;





## Strategy Plugins

### Definition

**Strategy** plugins define the way used to explore data, it uses descriptions to generate predecessors for each concept in the lattice.



Existing strategy plugins:

- ▶  *Chain* strategy;
- ▶  *Sequence* strategy;
- ▶  *Sequence* strategy using distances;
- ▶  *Triadic* strategy.

In preparation:

- ▶  *Graph* strategy.

## Strategy Plugins

### Definition

**Strategy** plugins define the way used to explore data, it uses descriptions to generate predecessors for each node in the lattice.



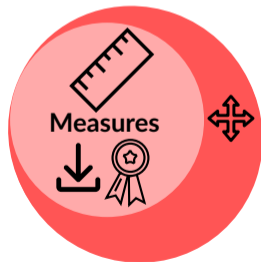
There are 3 ♻️ meta-strategies in the core library:

- ▶️ ⚠️ *Limit filter* which limits the predecessors to those whose measure does not exceed the limit;
- ▶️ ⚠️ *Selection filter* which selects the best or the worst predecessors;
- ▶️ ⚙️ *Conditioned strategy* which triggers the execution of inner strategies when some conditions are met.

## Measure Plugins

### Definition

**Measure** plugins are parameters of the *filter strategies* predefined in the core library.



There are 3 measures in the core library:

- ▶  $\Downarrow$  predecessor *Cardinality*;
- ▶  $\Uparrow$  successor *Cardinality*;
- ▶  $\text{🏆}$  *Confidence*.

One measure plugin has been developed:

- ▶  $\text{🏆}$  *Entropy* of the predecessor relatively to the successor.

## Data Reader Plugins

### Definition

**Data readers** plugins are used to read different types of data files. The *core* engine detects the file type using its extension.



Existing data reader plugins are:


- ▶ ⚙️ *YAML*
- ▶ ⚙️ *JSON*
- ▶ 📄 *CSV*
- ▶ 📄 *TOML*
- ▶ 📄 *INI*
- ▶ ✓ *TXT*
- ▶ ✓ *SLF*
- ▶ ✓ *DAT*
- ▶ ✓ *CXT*



## Localization Plugins

### Definition

**Localization** plugins are used for translating the applications to other languages. The basic language is English.

- ▶  French translation of the **GALACTIC** applications.

## Applications

### Definition

**Applications** are developed for using the library; they are the interface of the user.



Existing applications are:

- ▶ **GALACTIC Laser**: for constructing the lattice and exploring data;
- ▶ **GALACTIC Explorer**: for exploring interactively the constructed lattice;
- ▶ **GALACTIC Ruler**: for extracting implication rules;
- ▶ **GALACTIC Fire**: for executing a system of rules.

## Collaborative version control



# git

git

The library is developed using the collaboration tool `git`, in the `gitlab` of the university. We are using

- ▶ ***pylint*** and ***flake8*** (with plugins) for testing code quality;
- ▶ ***tox*** for generating tests.

## Collaborative version control



### gitlab-runners

Using *gitlab-runners*, the code is automatically recompiled and rebuilt and tests are ran.

- ▶ **core**: 80 python files; 11949 python lines; 8187 comment lines; 4194 blank lines; 8% unit test coverage;
- ▶ **plugins**: 136 python files; 7451 python lines; 6634 comment lines; 2523 blank lines; 17% unit test coverage;
- ▶ **6 guides** (installation, user, practice, experiments, developer, continous integration/deployment)

## Conclusion

- ▶ the version 0.4 was published on January 8th, 2022;
  - ▶ <https://galactic.univ-lr.fr>
  - ▶ <https://ml.univ-lr.fr/sympa/info/galactic>
- ▶ the **GALACTIC** applications, the various manuals and documentation guides are available under certain conditions.