The ReducedContextCompletion Algorithm

Updating the reduced context of a lattice in linear time and linear memory

The Galactic Organization <contact@thegalactic.org>

2018-2022 [1]

2022-03-11

The ReducedContextCompletion Algorithm

The ReducedContextCompletion Algorithm
Updating the reduced context of a lattice in linear time and linear memory
The Galactic Organization <contact@thegalactic.org>
2018-2022 [1]
[1] © 2018-2022 the Galactic Organization. This document is licensed under CC-by-nc-nd (https://creativecommons.org/licenses/by-nc-nd/4.0/deed.en)

Motivation

Exporting the FCA outside of its inner self

The FCA is a powerfull tool to deal with complex data, but the output can be hard to hard for non-It users. The following work will be a first step to:

- Allow non-It users to uses FCA algorithms.
- Put the data-scientist in the center of the process.

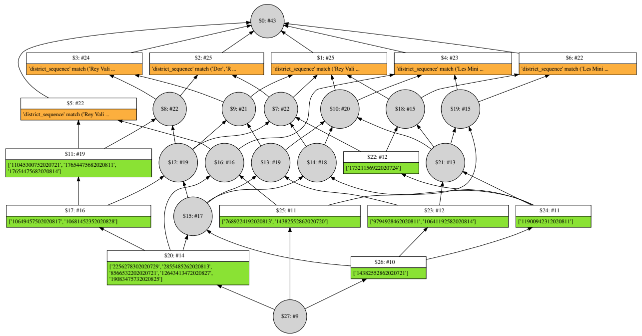
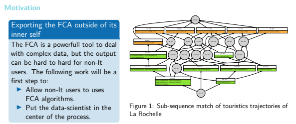


Figure 1: Sub-sequence match of touristics trajectories of La Rochelle

2022-03-11

The REDUCEDCONTEXTCOMPLETION Algorithm
└Motivation
 └Motivation
  └Motivation



Motivation

Exporting the FCA outside of its inner self

- Build in an iterative and intuitive way a lattice of concepts.
- Allow the user to change the strategy during the building of the lattice.

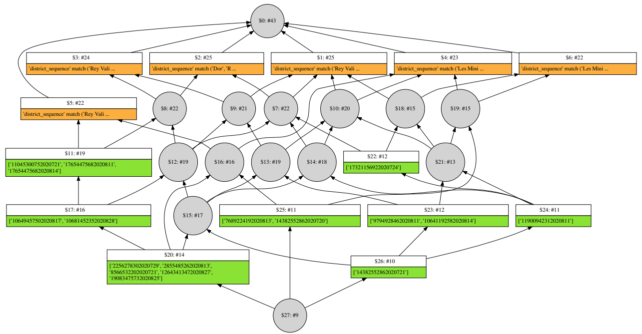


Figure 2: Sub-sequence match of touristics trajectories of La Rochelle

Motivation
Exporting the FCA outside of its inner self
- Build in an iterative and intuitive way a lattice of concepts.
- Allow the user to change the strategy during the building of the lattice.

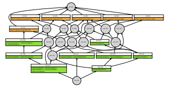
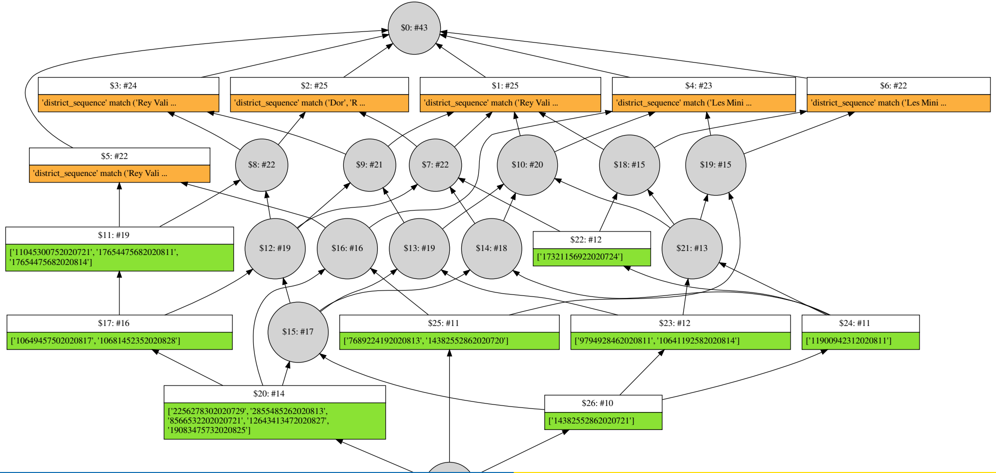


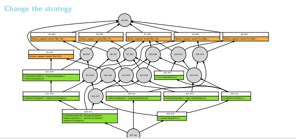
Figure 2: Sub-sequence match of touristics trajectories of La Rochelle

Change the strategy

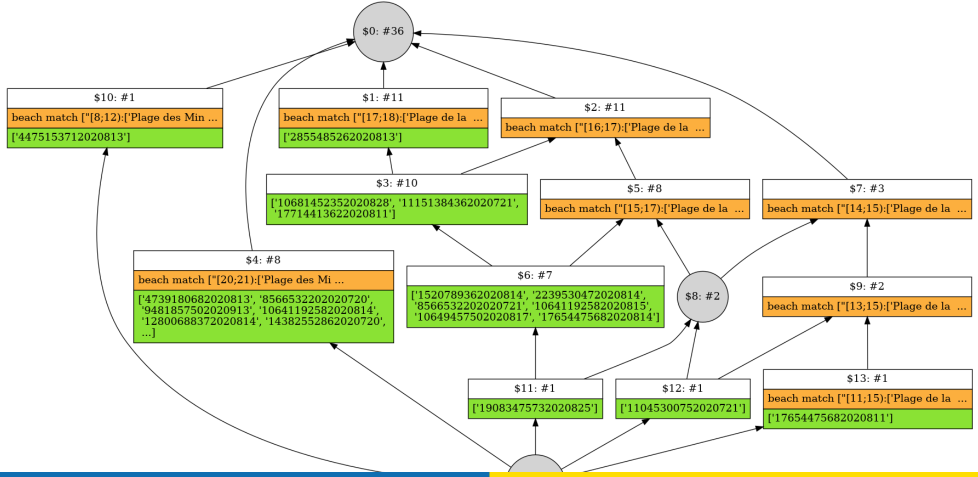


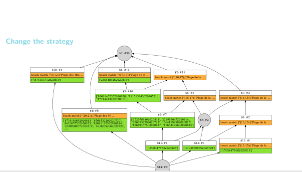
2022-03-11

The ReducedContextCompletion Algorithm
└Motivation
└Motivation
└Change the strategy



Change the strategy





Change the strategy

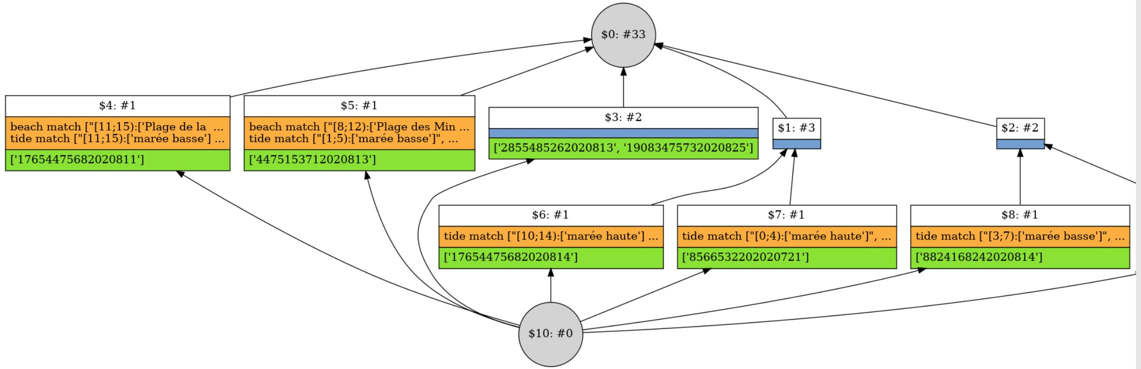


Figure 4

A first step to the conceptual navigation into lattice-shaped data-structure :

The conceptual navigation

- Achieve a proper interactive navigation into complex data structures such as lattices.
- Put the data-scientist at the center of the analysis, because only him know the semantic of its data
- Giving the possibility to change strategies “on the road” (Allowed by the NextPriorityConcept algorithm)

2022-03-11

The ReducedContextCompletion Algorithm
└Motivation
└Interractive navigation
└A first step to the conceptual navigation into lattice-shaped data-structure :

A first step to the conceptual navigation into lattice-shaped data-structure :
The conceptual navigation
- Achieve a proper interactive navigation into complex data structures such as lattices.
- Put the data-scientist at the center of the analysis, because only him know the semantic of its data
- Giving the possibility to change strategies “on the road” (Allowed by the NextPriorityConcept algorithm)

A first step to the conceptual navigation into lattice-shaped data-structure :

The conceptual navigation

- Achieve a proper interactive navigation into complex data structures such as lattices.
- Put the data-scientist at the center of the analysis, because only him know the semantic of its data
- Giving the possibility to change strategies "on the road" (Allowed by the NextPriorityConcept algorithm)

Optimization of the solution

Rather than rebuilding the lattice (that can be huge) :

- Maintain a condensed form (the reduced context) of a lattice.
- Update this reduced context according to the changes made by the analysis.

A first step to the conceptual navigation into lattice-shaped data-structure :
The conceptual navigation
Achieve a proper interactive navigation into complex data structures such as lattices.
Put the data-scientist at the center of the analysis, because only him know the semantic of its data
Giving the possibility to change strategies "on the road" (Allowed by the NextPriorityConcept algorithm)
Optimization of the solution
Rather than rebuilding the lattice (that can be huge) :
Maintain a condensed form (the reduced context) of a lattice.
Update this reduced context according to the changes made by the analysis.

prime factor lattice

Definition of a prime factor lattice

In the prime factor lattice, elements are tuple of prime number divisor. The meet operation between two points seeks the greatest common multiple of those two numbers, while the join operation return the greatest common divisor such as shown in figure 5.

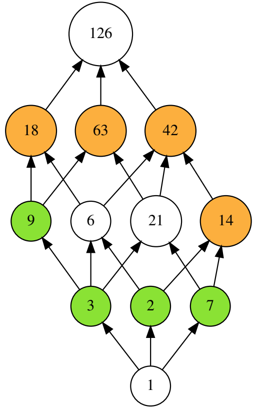
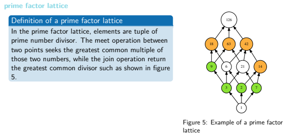


Figure 5: Example of a prime factor

2022-03-11

The ReducedContextCompletion Algorithm
└─Motivation
  └─Example (Application on integer lattice)
    └─prime factor lattice



$x_1 = 2$

Figure 6: Initial lattice

Irreducibles

$J^{\curlywedge}$	$M^{\curlywedge}$
$\emptyset$	$\emptyset$

$x_2 = 3$

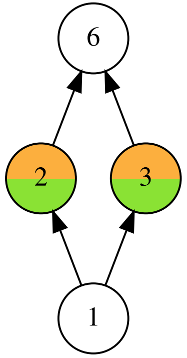


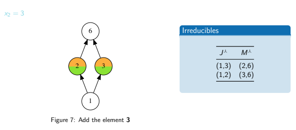
Figure 7: Add the element **3**

Irreducibles

$J^{\curlywedge}$	$M^{\curlywedge}$
(1,3)	(2,6)
(1,2)	(3,6)

2022-03-11

The ReducedContextCompletion Algorithm
└Motivation
└Exemple of the reduced context completion
└$x_2 = 3$



$x_3 = 7$

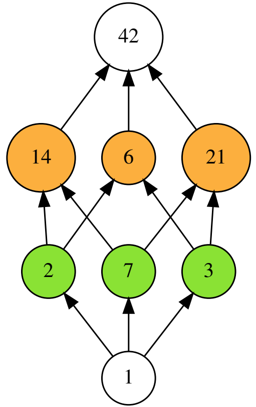
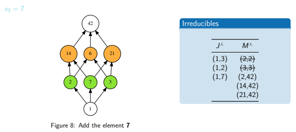


Figure 8: Add the element **7**

Irreducibles

$J^{\curlywedge}$	$M^{\curlywedge}$
(1,3)	$\cancel{(2,2)}$
(1,2)	$\cancel{(3,3)}$
(1,7)	(2,42)
	(14,42)
	(21,42)



$x_4 = 9$

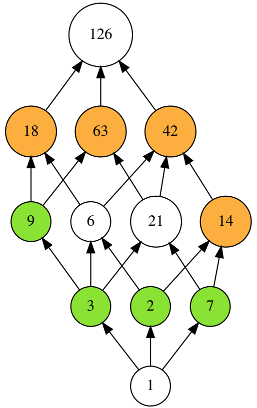


Figure 9: Add the element **9**

Irreducibles

$J^{\curlywedge}$	$M^{\curlywedge}$
(1,3)	~~(6,6)~~
(1,2)	(42,14)
(1,7)	~~(21,21)~~
(1,9)	(18,126)
	(63,126)
	(42,126)

2022-03-11
The ReducedContextCompletion Algorithm
Motivation
Exemple of the reduced context completion
$x_4 = 9$

Relation order

Ordinal definition

$\leq$ is a binary relation on the set S which satisfy three properties such as :

- $\leq$ is **reflexive** : for all $x \in S, x \leq x$
- $\leq$ is **antisymmetric** : for all $x, y \in S$ if $x \leq y$ and $y \leq x$ then $x = y$
- $\leq$ is **transitive** : with $x, y, z \in S$, if we have $x \leq y$ and $y \leq z$ then $x \leq z$.

Relation order
Ordinal definition
$\leq$ is a binary relation on the set S which satisfy three properties such as :
$\leq$ is **reflexive** : for all $x \in S, x \leq x$
$\leq$ is **antisymmetric** : for all $x, y \in S$ if $x \leq y$ and $y \leq x$ then $x = y$
$\leq$ is **transitive** : with $x, y, z \in S$, if we have $x \leq y$ and $y \leq z$ then $x \leq z$.

The meet and join operator

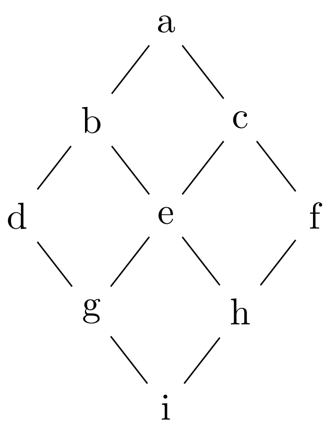
Ordinal definition

- The greatest lower bound of two elements (also named a join) x, y is noted $x \vee y$ refers to the greatest element of the predecessors of both x and y such that $z \leq x$ and $z \leq y$, $z \in S$.
- The least upper bound of two elements (also named a meet) x, y is noted $x \wedge y$ refers to the smallest element of the successor of both x and y such that $x \geq z$ and $y \geq z$, $z \in S$.
- $\vee$ and $\wedge$ can be used to defines the partial order relation $\leq$ on S in the context of a lattice structure.

Definition of a lattice

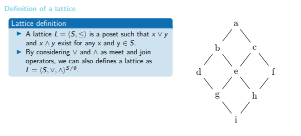
Lattice definition

- A lattice $L = \langle S, \leq \rangle$ is a poset such that $x \vee y$ and $x \wedge y$ exist for any x and y $\in S$.
- By considering $\vee$ and $\wedge$ as meet and join operators, we can also defines a lattice as $L = \langle S, \vee, \wedge \rangle^{S \neq \emptyset}$.



2022-03-11

The ReducedContextCompletion Algorithm
└Introduction
└Basic definition
└Definition of a lattice



The irreducible

Reduced context definition

- An element $j \in S$ is called join-irreducible if for any subset X of S, $j \neq \wedge X$. We noted j a join-irreducible on the lattice L, j_L. The set of the join-irreducible of L is noted J_L. In a lattice, a join-irreducible element only has one immediate predecessor noted j^-.
- An element $m \in S$ is called meet-irreducible if for any subset X of S, $m \neq \vee X$. We noted m a join-irreducible on the lattice L, m_L. The set of the meet-irreducible of L is noted M_L. In a lattice, a meet-irreducible element only has one immediate successor noted m^+.

The irreducible
Reduced context definition
- An element $j \in S$ is called join-irreducible if for any subset X of S, $j \neq \wedge X$. We noted j a join-irreducible on the lattice L, j_L. The set of the join-irreducible of L is noted J_L. In a lattice, a join-irreducible element only has one immediate predecessor noted j^-.
- An element $m \in S$ is called meet-irreducible if for any subset X of S, $m \neq \vee X$. We noted m a join-irreducible on the lattice L, m_L. The set of the meet-irreducible of L is noted M_L. In a lattice, a meet-irreducible element only has one immediate successor noted m^+.

The irreducible

Lattice definition

Irreducible elements of a lattice can't be obtain by $\vee$ operator (for join irreducible) or $\wedge$ (for meet irreducible), they represent the very structure of a lattice.

- We can't have more join irreducible elements than the number of individuals in our data.
- We can't have more meet irreducible elements than the number of different attributes in all our individuals.

J_L	M_L

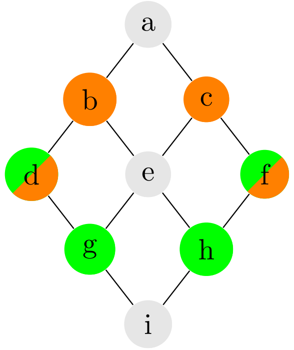
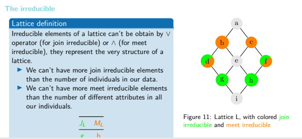


Figure 11: Lattice L, with colored join irreducible and meet irreducible



The reduced context definition

Reduced context definition

- The reduced context (Also called the table) of a lattice is defined as both all elements in J_L and M_L and the relational order $\leq$ on S such that $R_L = \langle J_L, M_L, \leq \rangle$.
- Based on the fundamental theorem, any lattice L is isomorphic to the concept lattice of its reduced context, we can then rebuilt any lattice L based on its irreducible.

The reduced context definition
Reduced context definition
- The reduced context (Also called the table) of a lattice is defined as both all elements in J_L and M_L and the relational order $\leq$ on S such that $R_L = \langle J_L, M_L, \leq \rangle$.
- Based on the fundamental theorem, any lattice L is isomorphic to the concept lattice of its reduced context, we can then rebuilt any lattice L based on its irreducible.

The reduced context definition

Maintaining the reduced context

- As we can rebuild a lattice L from its reduced context (And as the reduced context only contains irreducible elements and binaries operators of L), we only need to stock the structure R.
- By maintaining and working only on R_L, we drastically reduce the number of elements to take into account during the process and the storage.

The reduced context definition
Maintaining the reduced context
- As we can rebuild a lattice L from its reduced context (And as the reduced context only contains irreducible elements and binaries operators of L), we only need to stock the structure R.
- By maintaining and working only on R_L, we drastically reduce the number of elements to take into account during the process and the storage.

Definition : joins and immediate predecessors

Covering relation between joins and immediate predecessors

For the purpose of the algorithm, we introduce a covering relation between joins irreducible and theirs immediate predecessor such as :

$$^{L}\prec_{|J} = \{(j^{-}, j) | j \in J\}$$

Where we note $J_L^{\curlywedge} = {}^{L}\prec_{|J}$ such that $J_L^{\curlywedge}$ is a list of couple (j^{-}, j).

Definition : joins and immediate predecessors
Covering relation between joins and immediate predecessors
For the purpose of the algorithm, we introduce a covering relation between joins irreducible and theirs immediate predecessor such as :
$^{L}\prec_{|J} = \{(j^{-}, j) | j \in J\}$
Where we note $J_L^{\curlywedge} = {}^{L}\prec_{|J}$ such that $J_L^{\curlywedge}$ is a list of couple (j^{-}, j).

Definition : meets and immediate successors

Covering relation between meets and immediate successors

For the purpose of the algorithm, we introduce a covering relation between meets irreducible and theirs immediate successors such as :

$$^{L}\prec_{|M} = \{(m, m^{+}) | m \in M\}$$

Where we note $M_L^{\curlywedge} = {}^{L}\prec_{|M}$ such that $M_L^{\curlywedge}$ is a list of couple (m, m^{+}).

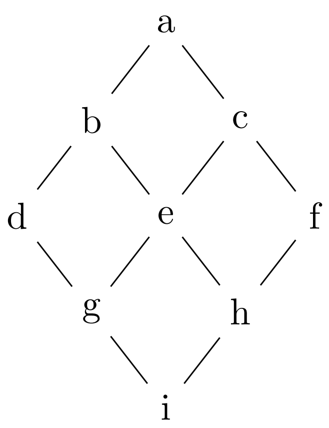
Definition : meets and immediate successors
Covering relation between meets and immediate successors
For the purpose of the algorithm, we introduce a covering relation between meets irreducible and theirs immediate successors such as :
$^{L}\prec_{|M} = \{(m, m^{+}) | m \in M\}$
Where we note $M_L^{\curlywedge} = {}^{L}\prec_{|M}$ such that $M_L^{\curlywedge}$ is a list of couple (m, m^{+}).

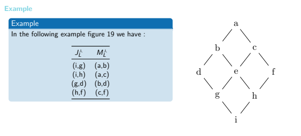
Example

Example

In the following example figure 19 we have :

$J_L^{\curlywedge}$	$M_L^{\curlywedge}$
(i,g)	(a,b)
(i,h)	(a,c)
(g,d)	(b,d)
(h,f)	(c,f)





Problematic

Two sets of couple

In this work, we will uses a special reduced context of L where $R_L^{\curlywedge} = \langle J_L^{\curlywedge}, M_L^{\curlywedge}, \wedge, \vee \rangle$.

Two sets of couple

Based on $R_L^{\curlywedge}$, the reduced context of L, and X, $X \subseteq S$, find the reduced context $R_{L_X}^{\curlywedge}$ of the sublattice L_X, $L_X \subseteq L$, the smallest lattice containing X.

2022-03-11
The ReducedContextCompletion Algorithm
The reduced context completion
Problematic Covering relation meet/successor and join/predecessor
Problematic

The reduce context completion function

Foreword

- Let L be a finite lattice, with a bottom element noted $\bot_L$ and a top element noted $\top_L$
- With $\mathcal{CL}$ an application that build a the concept lattice, where $L = \mathcal{CL}(\langle M_X^{\curlywedge}, J_X^{\curlywedge}, \top_X, \bot_X, \wedge, \vee \rangle)$
- Let $L_X = \mathcal{CL}(\langle M_X^{\curlywedge}, J_X^{\curlywedge}, \top_X, \bot_X, \wedge, \vee \rangle)$, the smallest lattice containing $X \subseteq S$.

The λ case

We introduce λ, a particular case where $\lambda^x = \mathcal{CL}(\langle \emptyset, \emptyset, x, x, \wedge, \vee, \rangle)$ is a lattice that only contains an element $\{x\}$ and the knowledge of join and meet operator of L.

The reduce context completion function
Foreword
Let L be a finite lattice, with a bottom element noted $\bot_L$ and a top element noted $\top_L$
With $\mathcal{CL}$ an application that build a the concept lattice, where $L = \mathcal{CL}(\langle M_X^{\curlywedge}, J_X^{\curlywedge}, \top_X, \bot_X, \wedge, \vee \rangle)$
Let $L_X = \mathcal{CL}(\langle M_X^{\curlywedge}, J_X^{\curlywedge}, \top_X, \bot_X, \wedge, \vee \rangle)$, the smallest lattice containing $X \subseteq S$.
The λ case
We introduce λ, a particular case where $\lambda^x = \mathcal{CL}(\langle \emptyset, \emptyset, x, x, \wedge, \vee, \rangle)$ is a lattice that only contains an element $\{x\}$ and the knowledge of join and meet operator of L.

The reduce context completion function

The reduce context completion function

κ is our reduced context completion function, returning the smallest sub-lattice of L with the element $\{x\}$. Where the function κ^x find $J_x^{\curlywedge}$, $M_x^{\curlywedge}$ from the reduced context of L, the minimal number of irreducible from L required to build L_x, the smallest lattice containing the element $\{x\}$; based on $\wedge$ and $\vee$ operator of L.
With $X = \{x_1, x_2..x_n\}$ we note :

$$L_X = \mathcal{CL}(\kappa^{x_1}(\kappa^{x_2}(\ldots\lambda^{x_n}))) = \mathcal{CL}(\langle M_X^{\curlywedge}, J_X^{\curlywedge}, \top_X, \bot_X, \wedge, \vee\rangle)$$

The reduce context completion function
The reduce context completion function
κ is our reduced context completion function, returning the smallest sub-lattice of L with the element $\{x\}$. Where the function κ^x find $J_x^{\curlywedge}$, $M_x^{\curlywedge}$ from the reduced context of L, the minimal number of irreducible from L required to build L_x, the smallest lattice containing the element $\{x\}$; based on $\wedge$ and $\vee$ operator of L.
With $X = \{x_1, x_2..x_n\}$ we note :
$$L_X = \mathcal{CL}(\kappa^{x_1}(\kappa^{x_2}(\ldots\lambda^{x_n}))) = \mathcal{CL}(\langle M_X^{\curlywedge}, J_X^{\curlywedge}, \top_X, \bot_X, \wedge, \vee\rangle)$$

Example

Exemple of a reduced context completion

Let L be a lattice $L = \langle S, \leq \rangle$. Let compute the reduced context completion with a subset of S, $X = \{b, e, d, c\}$ and the goal is to find L_X such that :

$$L_X = \mathcal{CL}(\langle M_X^{\curlywedge}, J_X^{\curlywedge}, \top_X, \bot_X, \wedge, \vee \rangle) = \mathcal{CL}(\kappa^c(\kappa^d(\kappa^e(\lambda^b))))$$

Example
Exemple of a reduced context completion
Let L be a lattice $L = \langle S, \leq \rangle$. Let compute the reduced context completion with a subset of S, $X = \{b, e, d, c\}$ and the goal is to find L_X such that :
$L_X = \mathcal{CL}(\langle M_X^{\curlywedge}, J_X^{\curlywedge}, \top_X, \bot_X, \wedge, \vee \rangle) = \mathcal{CL}(\kappa^c(\kappa^d(\kappa^e(\lambda^b))))$

Initialisation

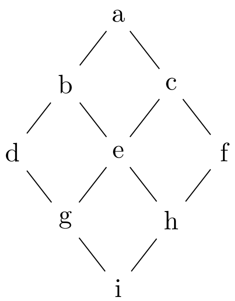


Figure 13: Lattice L $< S, \vee, \wedge >$

The original lattice $L = \langle S, \leq \rangle$,
$S = \{a, b, c, d, e, f, g, h, i\}$

Initialisation
The original lattice $L = \langle S, \leq \rangle$,
$S = \{a, b, c, d, e, f, g, h, i\}$
Figure 13: Lattice L $< S, \vee, \wedge >$

n=1

Figure 14: λ^b

We create the first sub-lattice λ^b where
$\lambda^b = \mathcal{CL}(\langle \emptyset, \emptyset, b, b, \wedge, \vee, \rangle)$

2022-03-11
The ReducedContextCompletion Algorithm
└The reduced context completion
└Example
└n=1

n=1
We create the first sub-lattice λ^b where
$\lambda^b = \mathcal{CL}(\langle \emptyset, \emptyset, b, b, \wedge, \vee, \rangle)$
Figure 14: λ^b

n=2

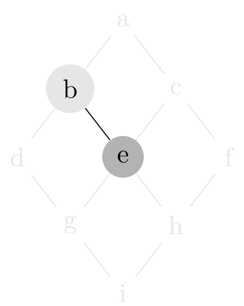
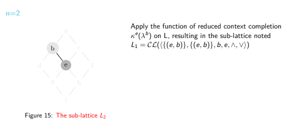


Figure 15: The sub-lattice L_2

Apply the function of reduced context completion $\kappa^e(\lambda^b)$ on L, resulting in the sub-lattice noted $L_1 = \mathcal{CL}(\langle\{(e,b)\},\{(e,b)\},b,e,\wedge,\vee\rangle)$



n=3

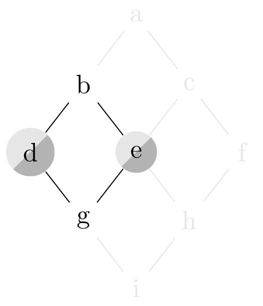


Figure 16: The sub-lattice L_3

Apply the function of reduced context completion $\kappa^d(L_1)$ on L, resulting in the sub-lattice noted $L_2 = \mathcal{CL}(\langle\{(g,d),(g,e)\},\{(d,b),(e,b)\},b,g,\wedge,\vee\rangle)$

2022-03-11

The ReducedContextCompletion Algorithm
└The reduced context completion
└Example
└n=3

n=4

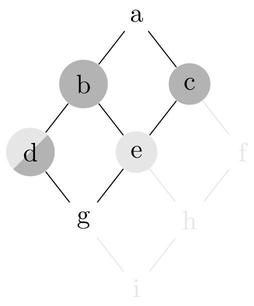


Figure 17: The sub-lattice L_3

Apply the function of reduced context completion $\kappa^c(L_2)$ on L, resulting in the sub-lattice $\kappa^n(\lambda^b)$ noted $L_3 = \mathcal{CL}(\langle\{(g,d),(g,e)\},\{(d,b),(c,a),(b,a)\},a,g,\wedge,\vee\rangle)$

2022-03-11

The ReducedContextCompletion Algorithm
└The reduced context completion
└Example
└n=4

n=4

Apply the function of reduced context completion $\kappa^c(L_2)$ on L, resulting in the sub-lattice $\kappa^n(\lambda^b)$ noted $L_3 = \mathcal{CL}(\langle\{(g,d),(g,e)\},\{(d,b),(c,a),(b,a)\},a,g,\wedge,\vee\rangle)$

Figure 17: The sub-lattice L_3

Theorem

Theorem

Given a lattice $\mathcal{L} = (S, \wedge, \vee)$, then the algorithm reduced completion computes the minimal sublattice of $\mathcal{L}$ containing a subset $X \subseteq S$.

2022-03-11
The ReducedContextCompletion Algorithm
└Theorem
└Theorem
└Theorem

Theorem

Theorem
Given a lattice $\mathcal{L} = (S, \wedge, \vee)$, then the algorithm reduced completion computes the minimal sublattice of $\mathcal{L}$ containing a subset $X \subseteq S$.

Lattice of sublattices

The lattice of sublattices

Let us consider each sublattice of $\mathcal{L}$ given by its set of elements, and the relation between these sublattices by inclusion on these sets of elements. The Moore family of all the sublattices of $\mathcal{L}$ equipped with this inclusion relation, and an emptyset $\emptyset$ forms a lattice. **The lattice of sublattice**.

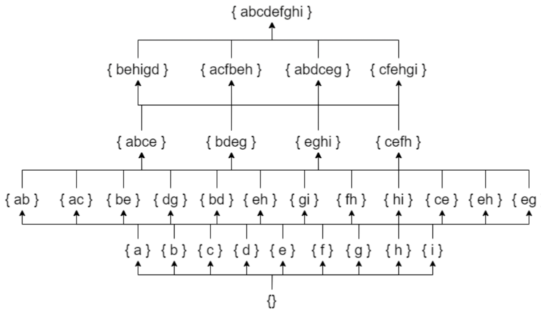
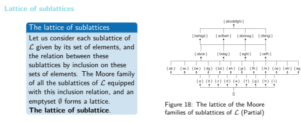


Figure 18: The lattice of the Moore families of sublattices of $\mathcal{L}$ (Partial)



Lattice of sublattices

A lattice L

Lattice of sublattices
A lattice L

Lattice of sublattices

Theorem

With φ as closure operator. The smallest sublattice of $\mathcal{L}$ containing X is then uniquely defined as $\mathcal{CL}(\varphi(X))$ in this lattice of sublattices.

Lattice of sublattices
Theorem
With φ as closure operator. The smallest sublattice of $\mathcal{L}$ containing X is then uniquely defined as $\mathcal{CL}(\varphi(X))$ in this lattice of sublattices.

Lattice of sublattices

Theorem

With φ as closure operator. The smallest sublattice of $\mathcal{L}$ containing X is then uniquely defined as $\mathcal{CL}(\varphi(X))$ in this lattice of sublattices.

Closure operator

A closure operator is a function on a set S that satisfy :

- $X \subset \varphi(X)$: **Extensive**
- $X \subset Y \rightarrow \varphi(X) \subset \varphi(Y)$: **Isotone**
- $\varphi(\varphi(X)) = \varphi(X)$: **Idempotent**

Closure operator are determined by their **closed set** (Or Moore Families . . .). The closure $\varphi(X)$ of a set X is the smallest set containing X.

2022-03-11

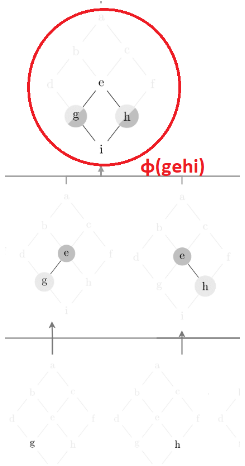
The REDUCEDCONTEXTCOMPLETION Algorithm
└Theorem
 └Proof
  └Lattice of sublattices

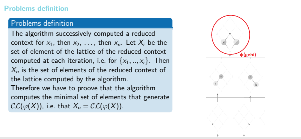
Lattice of sublattices
Theorem
With φ as closure operator. The smallest sublattice of $\mathcal{L}$ containing X is then uniquely defined as $\mathcal{CL}(\varphi(X))$ in this lattice of sublattices.
Closure operator
A closure operator is a function on a set S that satisfy :
- $X \subset \varphi(X)$: **Extensive**
- $X \subset Y \rightarrow \varphi(X) \subset \varphi(Y)$: **Isotone**
- $\varphi(\varphi(X)) = \varphi(X)$: **Idempotent**
Closure operator are determined by their **closed set** (Or Moore Families . . .). The closure $\varphi(X)$ of a set X is the smallest set containing X.

Problems definition

Problems definition

The algorithm successively computed a reduced context for x_1, then x_2, ..., then x_n. Let X_i be the set of element of the lattice of the reduced context computed at each iteration, i.e. for $\{x_1, .., x_i\}$. Then X_n is the set of elements of the reduced context of the lattice computed by the algorithm.
Therefore we have to proove that the algorithm computes the minimal set of elements that generate $\mathcal{CL}(\varphi(X))$, i.e. that $X_n = \mathcal{CL}(\varphi(X))$.





Proof

Proof

Let us assume that we have the reduced context of the smallest context containing $\{x_1, .., x_{i-1}\}$, then $X_{i-1} = \mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\}))$.
κ computes the reduced context of the smallest sublattice containing $X_{i-1} \cup \{x_i\} \subseteq X_i$.
Then X_i is the smallest lattice containing $\mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\} \cup \{x_i\}))$. Therefore $X_i = \mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\}))$, and by recursion, $X_n = \mathcal{CL}(\varphi(X))$.

Proof
Proof
Let us assume that we have the reduced context of the smallest context containing $\{x_1, .., x_{i-1}\}$, then $X_{i-1} = \mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\}))$.
κ computes the reduced context of the smallest sublattice containing $X_{i-1} \cup \{x_i\} \subseteq X_i$.
Then X_i is the smallest lattice containing $\mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\} \cup \{x_i\}))$. Therefore $X_i = \mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\}))$, and by recursion, $X_n = \mathcal{CL}(\varphi(X))$.

Proof

Proof

Let us assume that we have the reduced context of the smallest context containing $\{x_1, .., x_{i-1}\}$, then $X_{i-1} = \mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\}))$.
κ computes the reduced context of the smallest sublattice containing $X_{i-1} \cup \{x_i\} \subseteq X_i$.
Then X_i is the smallest lattice containing $\mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\} \cup \{x_i\}))$. Therefore $X_i = \mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\}))$, and by recursion, $X_n = \mathcal{CL}(\varphi(X))$.

Proof

X_i is composed of these updated irreducible elements, together with join and meet issued from these irreducible elements.

Proof

Proof

Let us assume that we have the reduced context of the smallest context containing $\{x_1, .., x_{i-1}\}$, then $X_{i-1} = \mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\}))$.
κ computes the reduced context of the smallest sublattice containing $X_{i-1} \cup \{x_i\} \subseteq X_i$.
Then X_i is the smallest lattice containing $\mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\} \cup \{x_i\}))$. Therefore $X_i = \mathcal{CL}(\varphi(\{x_1, .., x_{i-1}\}))$, and by recursion, $X_n = \mathcal{CL}(\varphi(X))$.

Proof

X_i is composed of these updated irreducible elements, together with join and meet issued from these irreducible elements.

Proof

κ is a closure operator

- This computation is extensive since $X \subseteq X_n = \mathcal{CL}(\varphi(X))$.
- Moreover, if κ is applied twice, then no element is added $\mathcal{CL}(\varphi(\varphi(X))) = \mathcal{CL}(\varphi(X))$: idempotent.
- And this computation add new irreducible elements to generate new elements, then it is isotone: if $X \subseteq Y$ then $\mathcal{CL}(\varphi(X)) \subseteq \mathcal{CL}(\varphi(Y))$.

Proof
κ is a closure operator
- This computation is extensive since $X \subseteq X_n = \mathcal{CL}(\varphi(X))$.
- Moreover, if κ is applied twice, then no element is added $\mathcal{CL}(\varphi(\varphi(X))) = \mathcal{CL}(\varphi(X))$: idempotent.
- And this computation add new irreducible elements to generate new elements, then it is isotone: if $X \subseteq Y$ then $\mathcal{CL}(\varphi(X)) \subseteq \mathcal{CL}(\varphi(Y))$.

Input of the algorithme

Input

- The reduced context of a lattice L
- An element $x \in S$

Output

- the reduced context of the sublattice L_X, $L_X \subseteq L$

2022-03-11
The ReducedContextCompletion Algorithm
└The algorithm of ReducedContextCompletion
└Input of the algorithme
└Input of the algorithme

Input of the algorithme
Input
The reduced context of a lattice L
An element $x \in S$
Output
the reduced context of the sublattice L_X, $L_X \subseteq L$

The algorithm of ReducedContextCompletion

Input

- $M^{\curlywedge}$: Set of couples (m, m^{+})
- $J^{\curlywedge}$: Set of couples (j^{-}, j)
- $\top$: The top element of the current lattice
- $\bot$: The bottom element of the current lattice
- $\wedge$: The join operator of L
- $\vee$: the meet operator of L
- X : Subset of S

Three step

- Startup Check
- Maximum and minimum updating
- Irreducibles updating

2022-03-11

The ReducedContextCompletion Algorithm
└The algorithm of ReducedContextCompletion
└Input of the algorithme
└The algorithm of ReducedContextCompletion

The algorithm of ReducedContextCompletion
Input
- $M^{\curlywedge}$: Set of couples (m, m^{+})
- $J^{\curlywedge}$: Set of couples (j^{-}, j)
- $\top$: The top element of the current lattice
- $\bot$: The bottom element of the current lattice
- $\wedge$: The join operator of L
- $\vee$: the meet operator of L
- X : Subset of S
Three step
- Startup Check
- Maximum and minimum updating
- Irreducibles updating

Startup check of ReducedContextCompletion

Startup check

- if $(., x)$ in $J^{\curlywedge}$: continue
- if $(x, .)$ in $M^{\curlywedge}$: continue
- if x is $\top$: continue
- if x is $\perp$: continue

Startup check of ReducedContextCompletion

Startup check

- if $(., x)$ in $J^{\curlywedge}$: continue
- if $(x, .)$ in $M^{\curlywedge}$: continue
- if x is $\top$: continue
- if x is $\bot$: continue

Maximum and minimum redefinition

- if $x > \top$: $\textsc{AddMaximum}(x)$
- if $x < \bot$: $\textsc{addMinimum}(x)$
- if $x \not\leq \top$ nor $\not\geq \top$: $\textsc{ExtendMaximum}(x)$
- if $x \not\leq \bot$ nor $\not\geq \bot$: $\textsc{ExtendMinimum}(x)$

2022-03-11

The $\textsc{ReducedContextCompletion}$ Algorithm
└The algorithm of ReducedContextCompletion
└Startup check
└Startup check of ReducedContextCompletion

Add new maximum

AddMaximum(x)

- $add(\top, x)$ in $M^{\curlywedge}$ // add x as successor of $\top$
- $add(x, \top)$ in $J^{\curlywedge}$ // add $\top$ as predecessor of x
- $\top \leftarrow x$ // x become the new top element
- continue

Add new maximum
AddMaximum(x)
- $add(\top, x)$ in $M^{\curlywedge}$ // add x as successor of $\top$
- $add(x, \top)$ in $J^{\curlywedge}$ // add $\top$ as predecessor of x
- $\top \leftarrow x$ // x become the new top element
- continue

Add new minimum

addMinimum(x)

- $add(\perp, x)$ in $J^{\curlywedge}$ // add x as predecessor of $\perp$
- $add(x, \perp)$ in $M^{\curlywedge}$ // add $\perp$ as successor of x
- $\perp \leftarrow x$ // x become the bottom element
- continue

Add new minimum
addMinimum(x)
- $add(\perp, x)$ in $J^{\curlywedge}$ // add x as predecessor of $\perp$
- $add(x, \perp)$ in $M^{\curlywedge}$ // add $\perp$ as successor of x
- $\perp \leftarrow x$ // x become the bottom element
- continue

Extend a new maximum or new minimum

ExtendMinimum(x)

- add$((\bot \wedge x), \bot)$ in $J^{\curlywedge}$ // Add the bottom element of κ^x into the set of join irreducibles
- $\bot \leftarrow (\bot \wedge x)$ // Update bottom element

ExtendMaximum(x)

- add$(\top, (\top \vee x))$ in $M^{\curlywedge}$ // Add the top element of x into the set of meet irreducibles
- $\top \leftarrow (\top \vee x)$ // Update top element

2022-03-11

The ReducedContextCompletion Algorithm
└The algorithm of ReducedContextCompletion
└Extend maximum or\and minimum
└Extend a new maximum or new minimum

Classical case : the main execution

Setting immediate successor and immediate predecessor of x

At this step, x is between $\top$ and $\bot$. We initialize $\bot$ as potential immediate predecessor of $x \in J^{\curlywedge}$, and top as potential immediate successor of $x \in M^{\curlywedge}$, and then we maintain the irreducible.

- $add(\bot, x)$ in $J^{\curlywedge}$
- $add(x, \top)$ in $M^{\curlywedge}$

Classical case : the main execution
Setting immediate successor and immediate predecessor of x
At this step, x is between $\top$ and $\bot$. We initialize $\bot$ as potential immediate predecessor of $x \in J^{\curlywedge}$, and top as potential immediate successor of $x \in M^{\curlywedge}$, and then we maintain the irreducible.
$add(\bot, x)$ in $J^{\curlywedge}$
$add(x, \top)$ in $M^{\curlywedge}$

Classical case : the main execution

Setting immediate successor and immediate predecessor of x

At this step, x is between $\top$ and $\bot$. We initialize $\bot$ as potential immediate predecessor of $x \in J^{\curlywedge}$, and top as potential immediate successor of $x \in M^{\curlywedge}$, and then we maintain the irreducible.

- $add(\bot, x)$ in $J^{\curlywedge}$
- $add(x, \top)$ in $M^{\curlywedge}$

Entering into the main execution

- $\textsc{insertJoinIrreducible}(x)$
- $\textsc{insertMeetIrreducible}(x)$

Classical case : the main execution
Setting immediate successor and immediate predecessor of x
At this step, x is between $\top$ and $\bot$. We initialize $\bot$ as potential immediate predecessor of $x \in J^{\curlywedge}$, and top as potential immediate successor of $x \in M^{\curlywedge}$, and then we maintain the irreducible.
$add(\bot, x)$ in $J^{\curlywedge}$
$add(x, \top)$ in $M^{\curlywedge}$
Entering into the main execution
$\textsc{insertJoinIrreducible}(x)$
$\textsc{insertMeetIrreducible}(x)$

Maintening the join and meet operator

Maintening the join and meet operator

Currently, we are working on finite sub-lattice and therefore we maintain the join and meet operator. In case of a join-semi-lattice, we can only maintain the join operator (And only focus on join irreducible) and the same is also true for a meet-semi-lattice, where we only seeks for the meet irreducible. Future works will address this question directly, but for now, we are looking for both the join and meet irreducible.

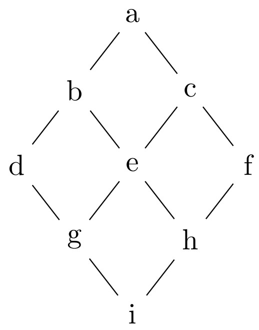


Figure 20: A lattice L

Maintening the join and meet operator

Maintening the join and meet operator

Currently, we are working on finite sub-lattice and therefore we maintain the join and meet operator. In case of a join-semi-la-ttice, we can only maintain the join operator (And only focus on join irreducible) and the same is also true for a meet-semi-lattice, where we only seeks for the meet irreducible. Future works will address this question directly, but for now, we are looking for both the join and meet irreducible.

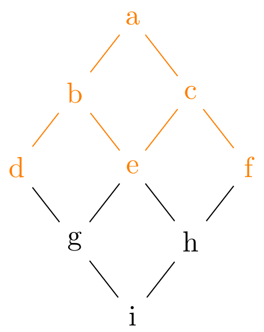


Figure 21: A join-semi-lattice from L

Maintening the join and meet operator

Maintening the join and meet operator

Currently, we are working on finite sub-lattice and therefore we maintain the join and meet operator. In case of a join-semi-lattice, we can only maintain the join operator (And only focus on join irreducible) and the same is also true for a meet-semi-lattice, where we only seeks for the meet irreducible. Future works will address this question directly, but for now, we are looking for both the join and meet irreducible.

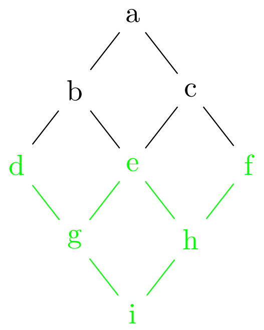


Figure 22: A meet-semi-lattice from L

Maintening the join and meet operator
Maintening the join and meet operator
Currently, we are working on finite sub-lattice and therefore we maintain the join and meet operator. In case of a join-semi-lattice, we can only maintain the join operator (And only focus on join irreducible) and the same is also true for a meet-semi-lattice, where we only seeks for the meet irreducible. Future works will address this question directly, but for now, we are looking for both the join and meet irreducible.
Figure 22: A meet-semi-lattice from L

insertJoinIrreducible(x)

insertJoinIrreducible(x)

for (j^-, j) *in* $J^\curlywedge$ **do**
- $r = j \wedge x$
- $r^- = (j^- \wedge x) \vee (j \wedge x^-)$
- $add(r-, r)$ in $J^\curlywedge$
- $x^- \vee = r$
- $j^- \vee = r$

end

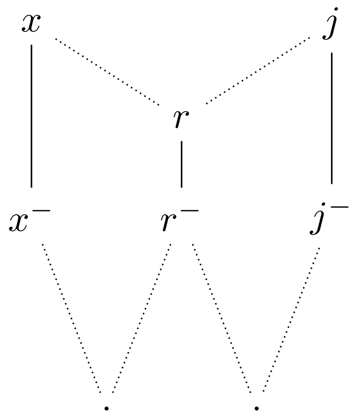
insertJoinIrreducible(x)

for (j^-, j) *in* $J^\curlywedge$ **do**
- **if** $j^- == j$ **then**
  - $remove(j^-, j)$ from $J^\curlywedge$
- **end**

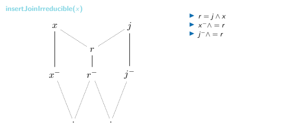
end

2022-03-11
The ReducedContextCompletion Algorithm
└The algorithm of ReducedContextCompletion
└The main execution
└insertJoinIrreducible(x)

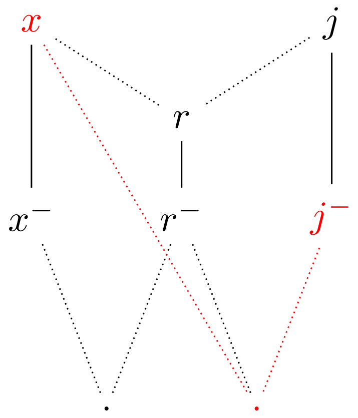
insertJoinIrreducible(x)



- $r = j \wedge x$
- $x^- \wedge = r$
- $j^- \wedge = r$



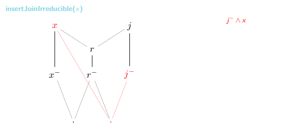
insertJoinIrreducible(x)



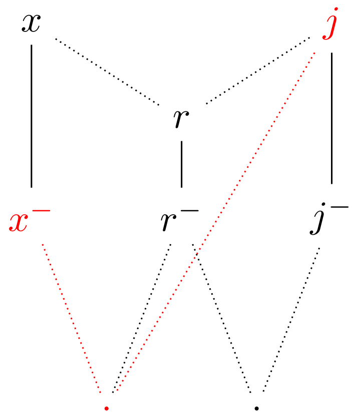
$j^- \wedge x$

2022-03-11

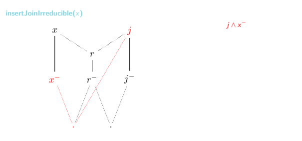
The ReducedContextCompletion Algorithm
└The algorithm of ReducedContextCompletion
└The main execution
└insertJoinIrreducible(x)



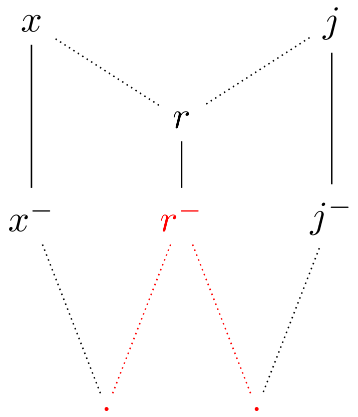
insertJoinIrreducible(x)



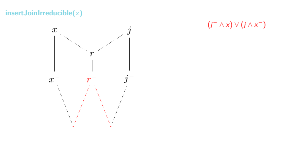
$j \wedge x^-$



insertJoinIrreducible(x)



$(j^- \wedge x) \vee (j \wedge x^-)$



insertMeetIrreducible(x)

insertMeetIrreducible(x)

for (m, m^+) *in* $M^\curlywedge$ **do**
  $r = m \vee x$
  $r^+ = (m^+ \vee x) \wedge (m \vee x^+)$
  $add(r, r+)$ in $M^\curlywedge$
  $x^+ \vee = r$
  $m^+ \vee = r$
end

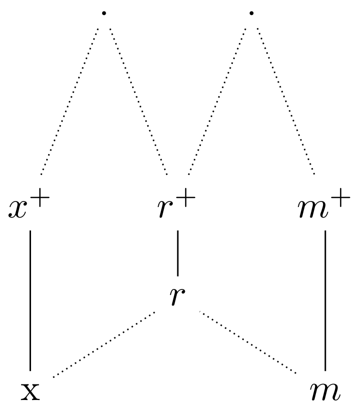
insertMeetIrreducible(x)

for (m, m^+) *in* $M^\curlywedge$ **do**
  if $m^+ == m$ **then**
    $remove(m, m+)$ from $M^\curlywedge$
  end
end

2022-03-11

The ReducedContextCompletion Algorithm
└The algorithm of ReducedContextCompletion
  └The main execution
    └insertMeetIrreducible(x)

insertMeetIrreducible(x)



- $r = m \vee x$
- $x^+ \vee = r$
- $m^+ \vee = r$

2022-03-11

The ReducedContextCompletion Algorithm
└The algorithm of ReducedContextCompletion
└The main execution
└insertMeetIrreducible(x)

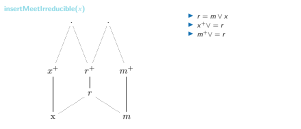
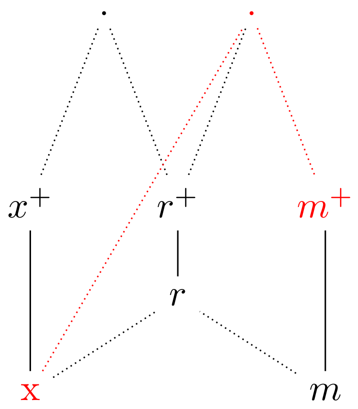
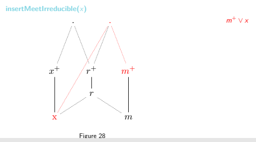


Figure 27

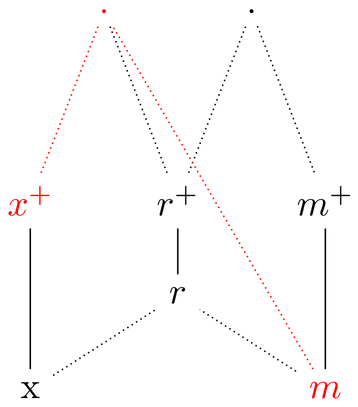
insertMeetIrreducible(x)



$m^+ \vee x$



insertMeetIrreducible(x)



$m \vee x^+$

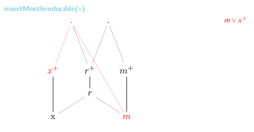
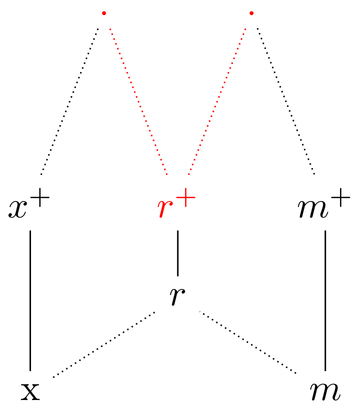


Figure 29

insertMeetIrreducible(x)



$r^+ = (m^+ \vee x) \wedge (m \vee x^+)$

2022-03-11

The REDUCEDCONTEXTCOMPLETION Algorithm
└The algorithm of ReducedContextCompletion
  └The main execution
    └insertMeetIrreducible(x)

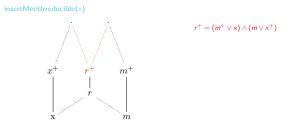


Figure 30

Theorem

Theorem

At the end of the algorithm $\langle J^{\curlywedge}, M^{\curlywedge}, \top, \bot, \vee, \wedge \rangle$ is the reduced context of L_X.

Theorem
Theorem
At the end of the algorithm $\langle J^{\curlywedge}, M^{\curlywedge}, \top, \bot, \vee, \wedge \rangle$ is the reduced context of L_X.

Conclusion

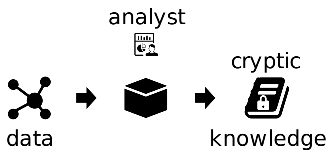


Figure 31: Deep learning

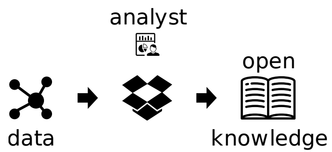


Figure 32: FCA

Conclusion

Future works

The reduced context completion, is a first step in data navigation, but not only. With this algorithm, we are able to only work on reduced contexts of lattices, drastically reducing the number of elements to take into account.