

Data structures for Partially Ordered Sets

Preparatory work for the master thesis

June 7, 2018

Hakim Boulahya
supervised by
Emmanuel Filiot, Guillermo A. Pérez

Département d'Informatique
Université Libre de Bruxelles
Belgium

ULB

1. **Introduction** Subject and motivations
2. **Definitions** Poset and antichains
3. **Existing implementation**
4. **Objective** Requirements and overview

Subject

- ▶ Implement data structures to represent partially ordered sets
- ▶ Main focus on antichain-based algorithms in automata theory

Motivations

- ▶ There exists new algorithms that uses antichains:
 - ▶ Model checking
 - ▶ Synthesis problem
 - ▶ Language universality
- ▶ There is a need for an efficient implementation of antichains

Partial order

A partial order \preceq on a set S is a binary relation $\preceq \subseteq S \times S$ that is reflexive, transitive and antisymmetric.

Partially ordered set (or poset)

A partially ordered set is a pair $\langle S, \preceq \rangle$ where S is a set, and \preceq a partial order on S .

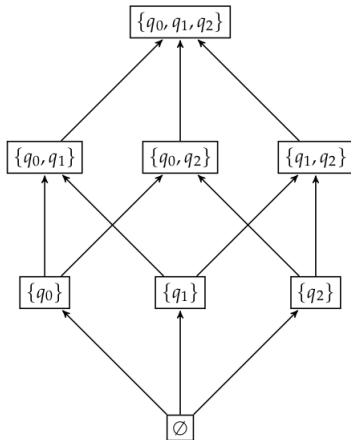
Antichains

An antichain α is a set of incomparable elements of poset $\langle S, \preceq \rangle$ w.r.t. to \preceq .

Interesting operations

- ▶ **Lower closure** of an antichain α on S , denoted $\downarrow\alpha$, is the set of all elements of S that are smaller or equal to an element of α
- ▶ **Appartenance** $s \in \downarrow\alpha_1$
- ▶ **Inclusion** $\downarrow\alpha_1 \subseteq \downarrow\alpha_2$
- ▶ **Union** $\downarrow\alpha_1 \cup \downarrow\alpha_2 = \downarrow[\alpha_1 \cup \alpha_2]$
- ▶ **Intersection** $\downarrow\alpha_1 \cap \downarrow\alpha_2$

Example



- Poset is $\langle 2^Q, \subseteq \rangle$, where $Q = \{q_0, q_1, q_2\}$
- $\alpha = \{\{q_0, q_2\}, \{q_1, q_2\}\}$ is an antichain
- Can retrieve all subset of cardinality 1 using the closure on the antichain: $\downarrow \alpha$
- $\downarrow \alpha = \{\{q_0\}, \{q_1\}, \{q_2\}, \{q_0, q_2\}, \{q_1, q_2\}\}$

Do implementation already exists ?

Yes, 2 for antichains:

1. **AaPAL**: Antichains implementation in *C*
 - ▶ More of an API: user must implement comparison and intersection
2. Antichains for the Dedekind number problem
 - ▶ Bitarray representation (a bit is a subset)
 - ▶ Can only be applied to natural numbers

Requirements

- ▶ Owl library: an automata library
- ▶ But symbolic antichain-based algorithm are missing (including antichain data structures)
- ▶ Implementation in Java 10

What to do ?

- ▶ Provide an API and implement it against `owl`
- ▶ Provide some implementation for antichains depending on the universe of the sets
- ▶ Define the algorithms to test our antichains implementation against
- ▶ Study the performance of those implementations

Thank you!

ULB