Qualitative Constraint Language

Travaux dirigés du cours *Contraintes* (HAI910I)

Michael Sioutis

9 Oct 2023

1 Recap on Relations

Let us (re-)familiarize ourselves with relations and operations on them. We recall the following definition from basic math courses:

Definition 1. *We call R a* relation *over sets $X_1, X_2, \ldots, X_\epsilon$ if*

$$R \subseteq X_1 \times X_2 \times \ldots \times X_\epsilon$$

and we call ϵ the arity of R.

We sometimes write $R(x_1, x_2, \ldots, x_\epsilon)$ meaning $(x_1, x_2, \ldots, x_\epsilon) \in R$; for a binary relation R we may also write $x\ R\ y$ meaning $(x, y) \in R$. The relation $U = X_1 \times X_2 \times \ldots \times X_\epsilon$ is called the *universal relation* over $X_1 \times X_2 \times \ldots \times X_\epsilon$.

Since relations are subsets of U, we can perform computations using the usual set operations:

$$\begin{aligned} R^C &= \{(x,y) \in U \mid (x,y) \notin R\} \\ R \cap S &= \{(x,y) \in U \mid (x,y) \in R \wedge (x,y) \in S\} \\ R \cup S &= \{(x,y) \in U \mid (x,y) \in R \vee (x,y) \in S\} \end{aligned}$$

We recall the definition of converse ($^{-1}$) and composition ($\circ$):

$$\begin{aligned} R^{-1}(x,y) &= R(y,x) \\ (R \circ S)(x,y) &= \exists z \text{ such that } R(x,z) \wedge S(z,y) \end{aligned}$$

There are some commonly considered properties of a binary relation, we recall them as follows:

Definition 2. *A binary relation R over a set U is called:*

- *reflexive iff,* $\forall x \in U, R(x,x)$
- *symmetric iff,* $\forall x, y \in U, (R(x,y) \rightarrow R(y,x))$
- *asymmetric iff,* $\forall x, y \in U, (R(x,y) \rightarrow \neg R(y,x))$

- *antisymmetric iff,* $\forall x, y \in U, (R(x,y) \land R(y,x) \to x = y)$[1]
- *transitive iff,* $\forall x, y, z \in U, (R(x,y) \land R(y,z) \to R(x,z))$
- *strict partial order iff R is asymmetric and transitive*

2 Exercices

Exercice 1. *Let relations R and S be defined as the following binary relations over $\{a, b, c, d, e\}$:*

$$\begin{aligned} R &= \{(a,a),(a,b),(b,c),(c,e)\} \\ S &= \{(a,a),(b,a),(e,b),(c,e)\} \end{aligned}$$

Compute $(R^{-1} \circ S) \cap R$

Exercice 2. *Consider a linear flow of time, and that events are being represented by real numbers.*

1. *Design a set of binary relations to represent the temporal order of time points.*
2. *For the relations you considered, which relational properties (e.g., reflexive, transitive) are satisfied?*
3. *Try to give an axiomatisation of the relations over time points in first-order logic. Do you run into any problems with specifying natural semantics in first-order logic?*

Exercice 3. *Compute the compositions between all Point Algebra relations; as a reminder, the set of Point Algebra base relations is* $\mathsf{B} = \{<, =, >\}$.

$\circ$	$\emptyset$	$<$	$=$	$>$	$\leq$	$\geq$	$\neq$	B
$\emptyset$								
$<$		$<$	$<$	B				
$=$		$<$	$=$	$>$				
$>$		B	$>$	$>$				
$\leq$								
$\geq$								
$\neq$								
B								

[1] Note that this definition is not pure first-order logic, as it assumes '=' to mean equal.

1. *How does computation of composition differ when considering relations over finite and infinite sets respectively?*
2. *Can we avoid reasoning about each of the* $8 \times 8 = 64$ *entries, e.g., can be exploit some properties?*
3. *Give the definition of* weak *composition* ($\diamond$)*, and explain how do composition* ($\circ$) *and* weak *composition* ($\diamond$) *differ in the case of Point Algebra.*

Exercice 4. *Consider Interval Algebra from the course, a reminder of its base relations is provided in the figure below.*

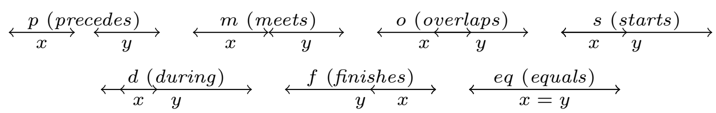

Figure 1: The 13 base relations of Interval Algebra; inverses are omitted

1. *Give the definitions of relations* m *and* o *(using the endpoint representation of intervals).*
2. *Compute the compositions* $m \diamond di$, $o \diamond b$, *and* $\{m, o\} \diamond \{di, b\}$.

3 Project (Optional)

Throughout the course we will develop a reasoning system capable of handling arbitrary algebras of qualitative relations; we will accomplish this by implementing reasoning methods independent of the underlying domain, be it points, intervals, or regions. You may choose any universal programming language you feel comfortable with.

Consider a relation algebra 2^{B} generated by a finite set B of JEPD relations over an infinite domain D.

1. How could a data type (e.g., class, data structure) be designed to represent elements of 2^{B} in a way that implements the set-theoretic operations as efficient as possible?
2. Relating to the first point, design an abstract data type to represent relations for an arbitrary set of JEPD base relations B that is given as a set of symbols. In other words, your program should ideally not depend on hard-coding a set of symbols like $<$, $=$, $>$, before, after, etc., but be usable with different sets of relations.
3. Implement the data type and the operations $\cap, \cup, ^C$ of a set algebra.
4. Describe the complexity of your operations using big-O-notation with respect to the number of base relations $n = |\mathsf{B}|$.

Next, we will extend the implementation by providing composition and converse operations; you may already anticipate that in your design choices.