Qualitative Constraint Network: Basic Notions

Travaux dirigés du cours *Contraintes* (HAI910I)

Michael Sioutis

20 Nov 2023

1 Reminder of Basic Notions

Spatial or temporal information for a set of entities can be represented by a qualitative constraint network (QCN).

Definition 1. *A* qualitative constraint network (QCN) *of some qualitative constraint language is a tuple* (V, C) *where:*

- V *is a set of variables over the infinite domain* D *of the language;*
- *and* C *is a mapping* $C : V \times V \rightarrow 2^{\mathsf{B}}$ *associating a relation (set of base relations) of the language with each pair of variables in* V*.*

Typically, we require that, $\forall v \in V$, $C(v, v) = \{\mathsf{Id}\}$, and that, $\forall v, v' \in V$, $C(v, v') = (C(v', v))^{-1}$. You can view an example QCN in Figure 1; for conciseness, converse relations or $\{\mathsf{Id}\}$ loops are not shown in the figure.

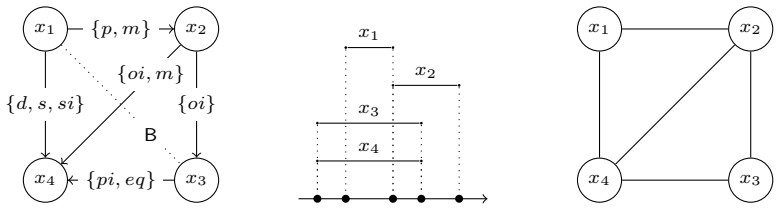

Figure 1: A QCN of Interval Algebra along with a solution, and a graph G

Let us also recall the definition of $\overset{\diamond}{}_G$-consistency:

Definition 2. *Given a* QCN $\mathcal{N} = (V, C)$ *and a graph* $G = (V, E)$*,* $\mathcal{N}$ *is* $\overset{\diamond}{}_G$-consistent *iff,* $\forall \{v_i, v_j\}, \{v_i, v_k\}, \{v_k, v_j\} \in E$, $C(v_i, v_j) \subseteq C(v_i, v_k) \diamond C(v_k, v_j)$

Intuitively, $\overset{\diamond}{}_G$-consistency checks if all triples of variables in $\mathcal{N}$ that correspond to triangles in G are closed under weak composition.

Remember also that $\diamond$-consistency is $\overset{\diamond}{}_G$-consistency where G is a complete graph. Hence, $\diamond$-consistency can be viewed as a special case of $\overset{\diamond}{}_G$-consistency.

2 Exercices

Exercice 1. *Construct a* QCN *of Interval Algebra for representing the temporal information of the following piece of text:*

> *"In the morning, I have breakfast and power on my laptop to answer some emails. In general, my productive work takes place in the afternoon, and I usually have a lunch break around 13h00. In the evening, I like to prepare a nice dinner, and sometimes meet up with friends for a walk either before or after my meal."*

Advice: *Depending on what assumption(s) you make, you might get a different* QCN *in the end, and that is OK!*

Exercice 2. *Provide a formal proof for Proposition 1:*

Proposition 1. *The satisfiability checking problem and the minimal labeling problem of a* QCN *are equivalent under polynomial-time Turing reductions, i.e., the former is polynomial-time Turing reducible to the latter, and vice versa.*

Let us first show that the satisfiability checking problem is polynomial-time Turing reducible to the minimal labeling problem. (This means that we have an oracle for solving the minimal labeling problem and we can use it a polynomial number of times to solve the satisfiability checking problem.) Let $\mathcal{N} = (V, C)$ be a QCN, and $u, v \in V$ any two of its variables. Then, for each base relation $b \in C(u, v)$ we construct a new QCN $\mathcal{N}_b = (V, C_b)$ such that:

- $C_b(u, v) = \{b\}$;
- $C_b(v, u) = \{b\}^{-1}$;
- and, $\forall (u', v') \in (V \times V) \setminus \{(u, v), (v, u)\}$, $C_b(u', v') = C(u', v')$.

In other words, each QCN $\mathcal{N}_b$ is the original QCN $\mathcal{N}$ where the base relation b has been assigned to the constraint $C(u, v)$. For each such $\mathcal{N}_b$, we ask our orable to decide whether $C_b(u, v)$ contains unfeasible/impossible base relations (i.e., base relations that cannot be satisfied by any solution of $\mathcal{N}_b$). If the oracle returns NO for some $\mathcal{N}_b$, then the original QCN $\mathcal{N}$ is satisfiable; otherwise, if the oracle returns YES for every $\mathcal{N}_b$, then the original QCN $\mathcal{N}$ is unsatisfiable.

Next, we need to show that the minimal labeling problem is polynomial-time Turing reducible to the satisfiability checking problem. We follow the same construction of the set of QCNs $\mathcal{N}_b$ as before, and we use the (satisfiability checking) oracle as follows. If the oracle returns NO for some $\mathcal{N}_b$, then the constraint $C(u, v)$ (and the original QCN $\mathcal{N}$) is not minimal, i.e., it contains unfeasible base relations; otherwise, if the oracle returns YES for every $\mathcal{N}_b$, then the constraint $C(u, v)$ is minimal (but we do not know whether the entire original QCN $\mathcal{N}$ is minimal too; see the next exercise for an answer to this question).

Exercice 3. *We say that a* QCN $\mathcal{N} = (V, C)$ *is* minimal *if and only if,* $\forall u, v \in V$ *and* $\forall b \in C(u, v)$, b *is a feasible base relation, i.e., a base relation that appears in some scenario of* $\mathcal{N}$. *Let* $\mathcal{N}_{\mathsf{min}}$ *denote the unique equivalent minimal sub-*QCN *of a* QCN $\mathcal{N}$*:*

1. *Design an algorithm to compute* $\mathcal{N}_{\mathsf{min}}$ *for a given* QCN $\mathcal{N}$, *provided that you have an oracle for solving the satisfiability checking problem of* $\mathcal{N}$.

   Hint: *Exploit your proof of Proposition 1.*

2. *Study the computational time complexity of your algorithm, provided that the aforementioned oracle runs in* α *time.*

1. As hinted, to compute $\mathcal{N}_{\mathsf{min}}$ we can perform the steps of the second part of the proof of Exercise 2, for *every* constraint in $\mathcal{N}$. Every time that the satisfiability checking oracle returns NO for the QCN that results by assigning a base relation to a constraint, we do not include that base relation in $\mathcal{N}_{\mathsf{min}}$. In the end, it should be clear that $\mathcal{N}_{\mathsf{min}}$ will only contain base relations (in every constraint) that are feasible.

2. We need to call our oracle $O(|\mathsf{B}| \cdot |V|^2)$ times, as we have $O(|V|^2)$ constraints in the QCN and each of these contains $O(|\mathsf{B}|)$ base relations. Since the oracle runs in in α time, the total runtime is $O(\alpha \cdot |\mathsf{B}| \cdot |V|^2)$.

Exercice 4. *Given two* QCN*s* $\mathcal{N}_1 = (V, C_1)$ *and* $\mathcal{N}_2 = (V, C_2)$, *let* $\mathcal{N}_2 \cup \mathcal{N}_2$ *denote the* QCN (V, C) *where,* $\forall u, v \in V$, $C(u, v) = C_1(u, v) \cup C_2(u, v)$. *Provide a formal proof for Proposition 2:*

Proposition 2. *Let* $\mathcal{N}_1 = (V, C_1)$ *and* $\mathcal{N}_2 = (V, C_2)$ *be any two* QCN*s, and* $G = (V, E)$ *any graph. Then, if* $\mathcal{N}_1$ *and* $\mathcal{N}_2$ *are* $\diamond_G$*-consistent, so is* $\mathcal{N}_1 \cup \mathcal{N}_2$.

Exercice 5. *Explain how the result of Proposition 2 can be used to assert that, for any* QCN $\mathcal{N} = (V, C)$ *and any graph* $G = (V, E)$, *there exists a unique largest* $\diamond_G$*-consistent sub-*QCN *of* $\mathcal{N}$, *i.e., a closure of* $\mathcal{N}$ *under* $\diamond_G$*-consistency (in the course we denoted this closude by* $\diamond_G(\mathcal{N})$*; see again the* dominance *property).*

Exercice 6. *Consider the* QCN *of Interval Algebra and the graph* G *in Figure 1, and answer the following questions:*

1. *Check whether the* QCN *is* $\diamond_G$*-consistent and detail your steps; if it is not* $\diamond_G$*-consistent, apply algebraic closure under* $\diamond_G$*-consistency to make it* $\diamond_G$*-consistent and detail your steps.*

2. *Check whether the* QCN *is* $\diamond$*-consistent and detail your steps; if it is not* $\diamond$*-consistent, apply algebraic closure under* $\diamond$*-consistency to make it* $\diamond$*-consistent and detail your steps.*

   Subquestion: *Can we exploit the outcome of Question* 1, *and to what extent?*

3. *Replace edge* $\{x_2, x_4\}$ *with edge* $\{x_1, x_3\}$ *in graph* G, *and repeat Question* 1.

4. Try to obtain a scenario of the QCN *and detail your steps.*

1. To check whether the QCN is $\overset{\diamond}{}_G$-consistent or not, we just follow the definition of $\overset{\diamond}{}_G$-consistency (see the course slides). In sum, we need to check if all triples of variables of the QCN that correspond to triangles in the given graph G are closed under (weak) composition. In this case, we have two such triples of variables, viz., $\{x_1, x_2, x_4\}$ and $\{x_1, x_3, x_4\}$. Please note that for each such triple we need to perform a closure check for each of its three constraints. For example, let us start with $\{x_1, x_2, x_4\}$. Then, we need to perform the three following closure checks:

- $C(x_1, x_2) \subseteq C(x_1, x_4) \diamond C(x_4, x_2)$;
- $C(x_1, x_4) \subseteq C(x_1, x_2) \diamond C(x_2, x_4)$;
- and $C(x_2, x_4) \subseteq C(x_2, x_1) \diamond C(x_1, x_4)$.

Please note that it does not matter if we choose a constraint $C(u, v)$, or if we choose its converse $C(v, u)$, for a closure check, as we are dealing with relation algebras, which come with certain nice properties (see the course slides). Let us perform the closure check for $C(x_1, x_4) \subseteq C(x_1, x_2) \diamond C(x_2, x_4)$. We have that $C(x_1, x_2) \diamond C(x_2, x_4) = \{p \diamond oi\} \cup \{p \diamond m\} \cup \{m \diamond oi\} \cup \{m \diamond m\} = \{p, d, s, m, o\} \cup \{p\} \cup \{d, s, o\} \cup \{p\} = \{p, d, s, m, o\} \not\supseteq C(x_1, x_4) = \{d, s, si\}$. Thus, the QCN is *not* $\overset{\diamond}{}_G$-consistent, as a closure check failed for one of our triples of variables.

To apply $\overset{\diamond}{}_G$-consistency, we just iteratively perform the following operation for all triples of variables of the QCN that correspond to triangles in the given graph G until a fixed state is reached:

$$\forall \{v_i, v_j\}, \{v_i, v_k\}, \{v_k, v_j\} \in E, C(v_i, v_j) \leftarrow C(v_i, v_j) \cap ((v_i, v_k) \diamond C(v_k, v_j))$$

For example, earlier we saw that $C(x_1, x_4) = \{d, s, si\}$, but, on the other hand, $C(x_1, x_2) \diamond C(x_2, x_4) = \{p, d, s, m, o\}$. So, $C(x_1, x_4)$ should be updated to $\{d, s, si\} \cap \{p, d, s, m, o\} = \{d, s\}$. Please note that every time a constraint gets updated, we need to *propagate* the changes until we reach a fixed state where no more propagation is possible. To this end, there is a naive and a state-of-the-art algorithm, as explained in the course, and you may choose either of them for applying $\overset{\diamond}{}_G$-consistency. For instance, if we follow the state-of-the art approach, since $C(x_1, x_4)$ got updated, we need to check whether the neighboring constraints $C(x_1, x_2)$ and $C(x_2, x_4)$ are affected. Indeed, $C(x_2, x_4)$ is affected, because $C(x_2, x_1) \diamond C(x_1, x_4) = \{d, f, mi, oi, pi\} \not\supseteq C(x_2, x_4) = \{oi, m\}$. So, $C(x_2, x_4)$ should be updated to $\{d, f, mi, oi, pi\} \cap \{oi, m\} = \{oi\}$. Repeating these operations until a fixed state is reached, we will also have that $C(x_3, x_4)$ gets updated to $\{eq\}$. At this point, the QCN will have become $\overset{\diamond}{}_G$-consistent.

2. As explained in the course, $\diamond$-consistency is $\overset{\diamond}{}_G$-consistency where G is the complete graph on the set of variables of a given QCN. With respect to our QCN, this means that we must also take into account the edge $\{x_1, x_3\}$ in our $\overset{\diamond}{}_G$-consistency definition. It is clear that we can use the refined, $\overset{\diamond}{}_G$-consistent QCN of the previous step as a starting point, and refine it further (if necessary).

Indeed, we can see that the QCN is not $\diamond$-consistent and it needs to be refined further. Specifically, we have that $C(x_1, x_4) \diamond C(x_4, x_3) = \{d, s\} \not\supseteq C(x_1, x_3) = \mathsf{B}$ (remember that B is the set of all base relations, viz., the universal relation). So, $C(x_1, x_3)$ should be updated to $\{d, s\} \cap \mathsf{B} = \{d, s\}$. At this point we can verify that the QCN is $\diamond$-consistent as no more propagation is possible.

Exercice 7. *Consider Point Algebra, where* $\mathsf{B} = \{<, =, >\}$*, and find* $r, s, t \in 2^{\mathsf{B}}$ *such that (weak) composition does not distribute over non-empty intersection, i.e.,* $r \diamond (s \cap t) \neq (r \diamond s) \cap (r \diamond t)$*; before exploring cases, argue first whether it is possible to find such* r, s, t *or not.*

3 Project (Optional)

Extend your implementation from the previous TD to include operations for converse and (weak) composition. Specifically, build upon your abstract data type for representing relations for an arbitrary set of JEPD base relations B, and implement methods to compute converse and (weak) composition for such relations. At the end of this assignment you should have implemented methods that allow computations like $(\{<, =\})^{-1} \diamond \{>, <\}$ to be performed.

Representing QCNs Further extend your implementation to represent QCNs of some qualitative constraint language. Specifically, you should be able to read a QCN definition from a file with the following format:

```
n #description
x1 x2 rel1
x1 x3 rel2
.
```

Some details follow:

- `n` represents the total number of variables, `description` is for documentation only.
- `x1`, `x2`, ... are non-negative integers in the range $[0, n-1]$, representing the n variables.
- `rel1, rel2`, ... are relations represented as whitespace-separated lists inside parentheses, e.g., `( < = )`.
- All constraints not defined are assumed to be the universal relation, converses *may* appear.

Supporting $\diamond$-consistency Finally, implement a method for checking the $\diamond$-consistency of an input QCN.