Quantitative vs. qualitative

Spatio-temporal configurations can be described quantitatively by specifying the coordinates of the relevant objects:

**Example:** At time point 10.0 object A is at position (11.0, 1.0, 23.7), at time point 11.0 at position (15.2, 3.5, 23.7). From time point 0.0 to 11.0, object B is at position (15.2, 3.5, 23.7). Object C is at time point 11.0 at position (300.9, 25.6, 200.0) and at time point 35.0 at (11.0, 1.0, 23.7).

Often, however, a qualitative description (using a finite vocabulary) is more adequate:

**Example:** Object A hit object B. Afterwards, object C arrived.

Sometimes we want to reason with such descriptions, e.g.:

*Object C was not close to object A when it hit object B.*
Representation of qualitative knowledge

**Intention:** Description of configurations using a finite vocabulary and reasoning about these descriptions

- Specification of a **vocabulary:** usually a finite set of relations (often binary) that are pairwise disjoint and exhaustive
- Specification of a **language:** often sets of atomic formulae (constraint networks), perhaps restricted disjunction
- Specification of a formal **semantics**
- Analysis of computational properties and design of **reasoning methods** (often constraint propagation)
- Perhaps, specification of operational **semantics** for verifying whether a relation holds in a given quantitative configuration

Applications in ...

- Natural language processing
- Specification of abstract spatio-temporal configurations
- Query languages for spatio-temporal information systems
- Layout descriptions of documents (and learning of such layouts)
- Action planning
- ... 

Qualitative temporal relations: **Point Calculus**

We want to talk about time instants (points) and binary relations over them.

- **Vocabulary:**
  - $X$ equals $Y$: $X = Y$
  - $X$ before $Y$: $X < Y$
  - $X$ after $Y$: $X > Y$

- **Language:**
  - Allow for **disjunctions** of basic relations to express indefinite information. Use set of relations to express that. For instance, $\{<,=\}$ expresses $\leq$.
  - $2^3$ different relations (including the **impossible** and the **universal** relation)
  - Use sets of atomic formulae with these relations to describe configurations. For example: $\{x\{=\}y, y\{<,\}z\}$

- **Semantics:** Interpret the time point symbols and relation symbols over the rational (or real) numbers.

Some reasoning problems

\[ \{x\{<,=\}y, y\{<,=\}z, v\{<,=\}y, w\{>,=\}y, z\{<,=\}x\} \]

- **Satisfiability:** Are there values for all time points such that all formulae are satisfied?
- **Satisfiability with** $v\{=\}w$?
- Finding a satisfying **instantiation** of all time points
- **Deduction:** Does $x\{=\}y$ logically follow?
  Does $v\{<,=\}w$ follow?
- Finding a **minimal description:** What are the most constrained relations that describe the same set of instantiations?
2 Constraint Satisfaction Problems

From a logical point of view …

In general, qualitatively described configurations are simple logical theories:
- Only sets of atomic formulae to describe the configuration
- Only existentially quantified variables (or constants)
- A fixed background theory that describes the semantics of the relations (e.g., dense linear orders)
- We are interested in satisfiability, model finding, and deduction
- Constraint Satisfaction Problems

CSP – Definition

Definition
A constraint satisfaction problem (CSP) is given by
- a set $V$ of $n$ variables $\{v_1, \ldots, v_n\}$,
- for each $v_i$, a value domain $D_i$
- constraints (relations over subsets of the variables)

Tasks:
Find one (or all) solution(s), i.e., tuples

$$(d_1, \ldots, d_n) \in D_1 \times \cdots \times D_n$$

such that the assignment $v_i \mapsto d_i$ $(1 \leq i \leq n)$ satisfies all constraints.

Our example: Point relations

- Our point relation CSP is a binary CSP with infinite domains.
- It can be represented as a constraint graph:
CSP – Example

\textit{k-colorability}: Can we color the nodes of a graph with \( k \) colors in a way such that all nodes connected by an edge have different colors?

- The node set is the set of variables
- The domain of each variable is \( \{1, \ldots, k\} \)
- The constraints are that nodes connected by an edge must have a different value

Note: This CSP has a particular restricted form:

- Only binary constraints
- The domains are finite

Other examples: Many problems (e.g. cross-word puzzle, \( n \)-queens problem, configuration, \( \ldots \)) can be cast as a CSP (and solved this way)

Computational complexity

\textbf{Theorem}

It is NP-hard to decide solvability of CSPs, even binary CSPs.

\textbf{Proof.}

Since \( k \)-colorability is NP-complete (even for fixed \( k \geq 3 \)), solvability of CSPs in general must be NP-hard.

Question: Is CSP solvability in NP?

3 Constraint Solving Methods

- \textbf{Enumeration} of all assignments and testing
  \( \leadsto \ldots \) too costly

- \textbf{Backtracking} search
  \( \leadsto 1001 \) different strategies, often “dead” search paths are explored extensively

- \textbf{Constraint propagation}: elimination of obviously impossible values followed by backtracking search Interleaving backtracking and constraint propagation

- Many other search methods, e.g., local search, stochastic search, etc.
  \( \leadsto \) How do we solve CSP with infinite domains?
General assumptions

- Only at most binary constraints (i.e., we can use constraint graph)
- Uniform domain $D$ for all variables
- Unary constraints $D_i$ and binary constraints $R_{ij}$ are sets of values or sets of pairs of values, resp.
- We assume that for all nodes $i,j$:
  
  $$(x, y) \in R_{ij} \Rightarrow (y, x) \in R_{ji}$$

Local consistency

- A CSP is **locally consistent** if for particular subsets of the variables, solutions of the restricted CSP can be extended to solutions of a larger set of variables.
- Methods to transform a CSP into a tighter, but “equivalent” problem.

**Definition**

A binary CSP $\langle V, D, C \rangle$ is arc-consistent (or 2-consistent) if for all nodes $1 \leq i, j \leq n$,

$$x \in D_i \Rightarrow \exists y \in D_j \text{ s.t. } (x, y) \in R_{ij}$$

- When a CSP is arc-consistent, each one variable assignment $\{v_i\} \rightarrow D_i$ that satisfies all (unary) constraints in $v_i$, i.e., $D_i$ can be extended to a two variable assignment $\{v_i, v_j\} \rightarrow D$ that satisfies all unary/binary constraints in these variables, i.e., $D_i, D_j, \text{and } R_{ij}$.

Arc consistency

**EnforceArcConsistency** ($C$):

*Input:* a (binary) CSP $C = \langle V, D, C \rangle$

*Output:* an equivalent, but arc-consistent CSP $C'$

**repeat**

- for each arc $(v_i, v_j)$ with $R_{ij} \in C$
  
  $D_i := D_i \cap \{x \in D : \exists y \in D_j \text{ s.t. } (x, y) \in R_{ij}\}$

**endfor**

**until** no domain is changed

- Terminates in time $O(n^3 \cdot k^3)$ if we have finite domains (where $k$ is the number of values)
- There exist different (more efficient) algorithms for enforcing arc consistency.

Lemma

- Enforcing arc consistency yields an arc consistent CSP.
- Enforcing arc consistency is solution invariant, i.e. it does not change the set of solutions.

- Arc-consistent CSPs need not be consistent, and vice versa.
Arc consistency – Example

\[
\begin{align*}
D_1 &= \{1, 2, 3\} \\
D_2 &= \{2, 3\} \\
D_3 &= \{2\} \\
R_{ij} &= "\neq" \text{ for } i \neq j
\end{align*}
\]

Since all unary constraints are singletons, this is a solution of the CSP. This is a unique solution of the original CSP.

Local consistency (2): Path consistency

Definition

A binary CSP \(\langle V, D, C \rangle\) is said to be path-consistent (or 3-consistent) if for all nodes \(1 \leq i, j, k \leq n\),
\[
\exists z \in D_k \text{ s.t. } (x, z) \in R_{ik} \text{ and } (y, z) \in R_{jk}
\]

\(\Rightarrow\) When a CSP is path-consistent, each two variable assignment \(\{v_i, v_j\} \rightarrow D\) satisfying all constraints in \(v_i\) and \(v_j\) can be extended to any three variable assignment \(\{v_i, v_j, v_k\} \rightarrow D\) such that all constraints in these variables are satisfied.

Path consistency

**EnforcePathConsistency** \((C)\):

*Input:* a (binary) CSP \(C = \langle V, D, C \rangle\) of size \(n\)

*Output:* an equivalent, but path-consistent CSP \(C'\)

repeat

for all \(1 \leq i, j, k \leq n\)

\[
R_{ij} := R_{ij} \cap \{ (x, y) : \text{ex. } z \in D_k \text{ s.t. } (x, z) \in R_{ik} \text{ and } (y, z) \in R_{jk} \}
\]

endfor

until no binary constraint is changed

\(\Rightarrow\) Terminates in time \(O(n^5 \cdot k^5)\) if we have finite domains

(\(k\) is the number of values)

\(\Rightarrow\) Enforcing path consistency is solution invariant.
Local consistency (2): Path consistency

Definition
A binary CSP \( \langle V, D, C \rangle \) is said to be path-consistent (or 3-consistent) if for all nodes \( 1 \leq i, j, k \leq n \),

\[
\begin{align*}
x \in D_i, y \in D_j, (x, y) \in R_{ij} \Rightarrow \\
\exists z \in D_k \text{ s.t. } (x, z) \in R_{ik} \text{ and } (y, z) \in R_{jk}
\end{align*}
\]

When a CSP is path-consistent, each two variable assignment \( \{v_i, v_j\} \rightarrow D \) satisfying all constraints in \( v_i \) and \( v_j \) can be extended to any three variable assignment \( \{v_i, v_j, v_k\} \rightarrow D \) such that all constraints in these variables are satisfied.

Local consistency (3): \( k \)-consistency and strong \( k \)-consistency

Definition
- A binary CSP \( \langle V, D, C \rangle \) is \( k \)-consistent if, given variables \( x_1, \ldots, x_k \) and an assignment \( a : \{x_1, \ldots, x_{k-1}\} \rightarrow D \) that satisfies all constraint in these variables, \( a \) can be extended to an assignment \( a' : \{x_1, \ldots, x_k\} \rightarrow D \) that satisfies all constraints in these \( k \) variables.
- A binary CSP \( \langle V, D, C \rangle \) is strongly \( k \)-consistent if it is \( k' \)-consistent for each \( k' \leq k \).
- A binary CSP \( \langle V, D, C \rangle \) is globally consistent if it is strongly \( n \)-consistent where \( n \) is the size of \( V \).

Local consistency (3)

- \( k \)-consistency: The computation costs grow exponentially with \( k \).
- If a CSP is globally consistent, then
  - a solution can be constructed in polynomial time,
  - its constraints are minimal,
  - and it has a solution iff there is no empty constraint.
- \( k \)-consistent \( \nRightarrow \) \( k - 1 \)-consistent

4 Qualitative Constraint Satisfaction Problems
Qualitative reasoning with CSP

If we want to use CSPs for qualitative reasoning, we have

- infinite domains
- mostly only finitely many relations (basic relations and their unions)
- arc-consistent CSPs (usually)

Questions:

- How do we achieve k-consistency (for some fixed k)?
- Is k-consistency (for some fixed k) enough to guarantee global consistency?

Operations on binary relations

Composition:

\[ R_1 \circ R_2 = \{ (x, y) \in D^2 : \exists z \in D \text{ s.t. } (x, z) \in R_1 \text{ and } (z, y) \in R_2 \} \]

Converse:

\[ R^{-1} = \{ (x, y) \in D^2 : (y, x) \in R \} \]

Intersection:

\[ R_1 \cap R_2 = \{ (x, y) \in D^2 : (x, y) \in R_1 \text{ and } (x, y) \in R_2 \} \]

Union:

\[ R_1 \cup R_2 = \{ (x, y) \in D^2 : (x, y) \in R_1 \text{ or } (x, y) \in R_2 \} \]

Complement:

\[ \overline{R} = \{ (x, y) \in D^2 : (x, y) \notin R \} \]

Computing operations on relations

Let A be a relation system over the set of base relations B that satisfies the conditions spelled out above.

- We may write relations as sets of base relations:
  \[ B_1 \cup \cdots \cup B_n \sim \{ B_1, \ldots, B_n \} \]

Then the operations on the relations can be computed as follows:

**Composition:**

\[ \{ B_1, \ldots, B_n \} \circ \{ B'_1, \ldots, B'_m \} = \bigcup_{i=1}^{n} \bigcup_{j=1}^{m} (B_i \circ B'_j) \]

**Converse:**

\[ \{ B_1, \ldots, B_n \}^{-1} = \{ B_{1}^{-1}, \ldots, B_{n}^{-1} \} \]

**Complement:**

\[ \{ B_1, \ldots, B_n \}^{-} = \{ B \in B : B \neq B_i \text{ for each } 1 \leq i \leq n \} \]

**Intersection and union** are defined set-theoretically.
Reasoning problems

Given a qualitative CSP:

- **CSP-Satisfiability (CSAT):**
  - Is the CSP satisfiable/solvable?

- **CSP-Entailment (CENT):**
  - Given in addition \( xRy \): Is \( xRy \) satisfied in each solution of the CSP?

- **Computation of an equivalent minimal CSPs (CMIN):**
  - Compute for each pair \( x, y \) the strongest constrained (minimal) relation entailed by the CSP.

\( \rightarrow \) These problems are equivalent under Turing reductions.

Reductions between CSP problems

**Theorem**

CSAT, CENT and CMIN are equivalent under polynomial Turing reductions.

**Proof.**

CSAT \( \leq_T \) CENT and CENT \( \leq_T \) CMIN are obvious.

CENT \( \leq_T \) CSAT: We solve CENT \( (CSP \models xRy?) \) by testing satisfiability of the CSP extended by \( x \{ B \} y \) where \( B \) ranges over all base relations.

Let \( B_1, \ldots, B_k \) be the relations for which we get a positive answer.

Then \( x \{ B_1, \ldots, B_k \} y \) is entailed by the CSP.

CMIN \( \leq_T \) CENT: We use entailment for computing the minimal constraint for each pair. Starting with the universal relation, we remove one base relation until we have a minimal relation that is still entailed.

Path consistency for qualitative CSPs

Given a qualitative CSP with \( R_{ij} = R_{ji}^{-1} \). Then path consistency can be enforced by doing the following:

\[
R_{ij} := R_{ij} \cap (R_{ik} \circ R_{kj}).
\]

Path consistency guarantees . . .

- sometimes minimality
- sometimes satisfiability
- however sometimes the CSP is not satisfiable, even if the CSP contains only base relations

\( \rightarrow \) All this depends on the vocabulary.

Example: Point relations

**Composition table:**

\[
\begin{array}{ccc}
< & = & > \\
< & < & <,=,> \\
= & < & = \\
> & <,=,> & > \\
\end{array}
\]

**Figure:** Composition table for the point algebra. For example:

\[
\{<\} \circ \{=\} = \{<\}
\]

- \( \{<,=\} \circ \{<\} = \{<\} \)
- \( \{<,>\} \circ \{<\} = \{<,=,>\} \)
- \( \{<,=\}^{-1} = \{>,=\} \)
- \( \{<,=\} \cap \{>,=\} = \{=\} \)
Qualitative reasoning with CSP – Example

```
R_{vw} = R_{vw} \cap R_{vy} \circ R_{yw}
= R_{vw} \cap ((< \circ <) \cup (\equiv \circ <))
= \{<, =, >\} \cap \{<\} \cup \{<\}
= \{<, =, >\}
```

R_{vx} = R_{vx} \cap R_{vy} \circ R_{yx}
= R_{vx} \cap ((< \circ =) \cup (< >) \cup (\equiv \circ =) \cup (\equiv \circ >))

= \{<, =, >\}

Some properties of the point relations

**Theorem**

* A path-consistent CSP over the point relations is consistent.

**Corollary**

* CSAT, CENT and CMIN are polynomial problems for the point relations.

**Theorem**

* A path-consistent CSP over all point relations without \{<, >\} is minimal.

Proofs later ...

A pathological relation system

Let \(e, d, i\) be (self-converse) base relations between points on a circle:

- \(e\): Rotation by 72 degrees (left or right)
- \(d\): Rotation by 144 degrees (left or right)
- \(i\): Identity

The following CSP is path-consistent and contains only base relations, but it is not satisfiable:

Composition table:

\[
\begin{align*}
e \circ e &= \{i, d\} \\
d \circ d &= \{i, e\} \\
e \circ d &= \{e, d\} \\
d \circ e &= \{e, d\}
\end{align*}
\]
Qualitative representation and reasoning usually starts with a finite vocabulary (a finite set of relations).

Qualitative descriptions are usually simply logical theories consisting of sets of atomic formulae (and some background theory).

Reasoning problems are (as usual) satisfiability, model finding, and deduction.

Can be addressed with CSP methods (but note: infinite domains).

Path consistency is the basic reasoning step...sometimes this is enough.

Usually, path-consistent atomic CSPs are satisfiable. However, there exist some pathological relation systems.

Can be taken further → relation algebra

Literature

Alan K. Mackworth.
Constraint satisfaction.

Alan K. Mackworth.
Consistency in networks of relations.

Peter B. Ladkin and Roger Maddux.
On binary constraint networks.

Ugo Montanari.
Networks of constraints: fundamental properties and applications to picture processing.
R. Hirsch.
Tractable approximations for temporal constraint handling.
(Contains a pathological set of relations.)