Resolving deictic references with fuzzy CSPs

Thies Pfeiffer

A.I. Group, Faculty of Technology, Bielefeld University

Problem

Introduction

When humans communicate, we use **deictic** expressions to refer to other entities, such as places, events or persons.

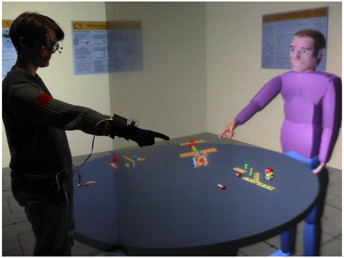


Figure: *“put the red bolt in this block”*

Problem

Introduction

"put the red bolt in this block"
This is an expression which can be easily understood by a human interlocutor, given the right context, i.e. if both interlocutors are **situated** in the same environment and perceive each other and their surroundings.

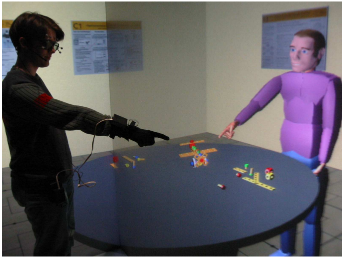


Figure: *"put the red bolt in this block"*

Problem

Introduction

“put the red bolt in this block”
How can this expression be interpreted by a computer system?

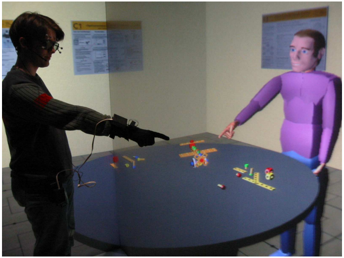


Figure: *“put the red bolt in this block”*

Problem

Expression

"put the red bolt in this block"

Problem description revised

The situation is defined by the **world** W. The speaker utters a deictic expression to discriminate the **topic** T from all possible subsets of W.

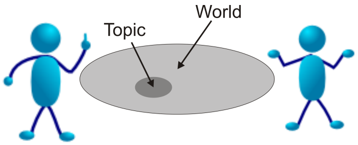


Figure: Communicating a topic

Problem

Expression

"put the red bolt in this block"

Problem statement

- Find the topic T in W so that the instances in T satisfy the deictic expression.
- The deictic expression formulates constraints on W to discriminate T.
- $\rightarrow$ constraint satisfaction problem

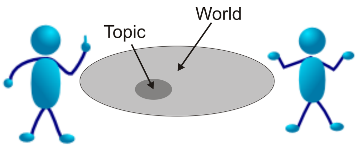


Figure: Communicating a topic

Content

Constraint Satisfaction Problems

Definition

Definition

A **constraint satisfaction problem (CSP)** is a tuple (X, P). X is the set $\{x_i | x_i \in D_i\}$ of **variables** of the CSP, each with an individual domain D_i. P is the set of **predicates** over the variables: $p_k(x_{k1}, \ldots, x_{kn}) : D_{k1} \times \cdots \times D_{kn} \rightarrow \{true, false\}$.

Grade of a CSP

Predicates can include any number of variables, the maximum count for an individual predicate defines the **grade** of the CSP. CSPs of any grade can be transformed into CSPs of grade 2 (Bacchus & van Beek, 1998). Hence, algorithms concentrate on solving CSPs of grade 2. In addition, these CSPs can be visualized as graphs.

Solving CSPs

Search

- **Generate-and-Test** (pro: finds all solutions, con: not efficient)
- **Backtracking** (pro: finds all solutions, con: naïve algorithm may take even longer than Generate-and-Test)

optimizing backtracking

- **intelligent backtracking** returns directly to conflicting variable
- **consistency checks** test early in the processing effects on other variables (Mackworth, 1977; Mackworth & Freuder, 1985)

Types of CSPs

Types of CSPs

- **weighted CSPs** associates costs with assignments and tries to minimize the overall cost of the solution
- **probabilistic CSPs** associate probabilities to predicates
- **fuzzy CSPs** associate a value in $[0 \ldots 1]$ with an assignment and provide the assignment with the maximum minimal assignment to an individual variable as a solution

Example

Expression

"put the red bolt in this block"

as CSP

```
(var "?object-1" BOLT)
(var "?object-2" BLOCK)
(has-color "?object-1" RED)
(fits "?object-1" "?object-2")
```

Example

Expression

“put the red bolt in this block”

as CSP

```
(var "?object-1" BOLT)
(var "?object-2" BLOCK)
(has-color "?object-1" RED)
(fits "?object-1" "?object-2")
```

Open Questions

Expression

"put the red bolt in this block"

Open Questions

- How to test for different shades of red?
- How to express that a constraint might be there or not, e.g. a possible pointing gesture accompanying *this*?
- How to differentiate between alternative solutions? Can heuristics be included? E.g. objects recently been used should be preferred.

Fuzzy Logic

Origin

- Fuzzy-sets have been developed by Lotfi A. Zadeh (1965).
- Good introductions can be found, e.g. in Pal & Mitra (1999).

Definition

Fuzzy-sets

A **fuzzy-set** is a pair (A, μ_A). A is a subset of a set $R = \{r\}$ characterized by the **membership function** $\mu_A(r)$. $\mu_A : R \to [0, 1]$ represents the **grade** of membership of r regarding A.

Support

The **support** S of A is defined as $S(A) = \{r | r \in R \wedge \mu_A(r) > 0\}$.

Definition

Generic membership functions (A)

$$\mu_A(r;\alpha,\beta,\gamma) = \begin{cases} 0 & : \quad r \leq \alpha \\ 2(\frac{r-\alpha}{\gamma-\alpha})^2 & : \quad \alpha < r \leq \beta \\ 1 - 2(\frac{r-\gamma}{\gamma-\alpha})^2 & : \quad \beta < r \leq \gamma \\ 1 & : \quad \gamma < r \end{cases}$$

with crossover-point $\beta = (\alpha + \gamma)/2$, i.e. the point where μ_A is 0.5

Definition

Generic membership functions (B)

$$\pi(r;\gamma,\lambda) = \begin{cases} \mu_A(r;\gamma-\lambda,\gamma-\frac{\lambda}{2},\gamma) & : \quad r \leq \gamma \\ 1-\mu_A(r;\gamma,\gamma+\frac{\lambda}{2},\gamma+\lambda) & : \quad r > \gamma \end{cases}$$

with the bandwidth λ and the center γ

Properties

Properties of fuzzy-sets

$$
\begin{aligned}
A = B &: \quad \mu_A(r) = \mu_B(r) \\
A = \bar{B} &: \quad \mu_A(r) = \mu_{\bar{B}}(r) = 1 - \mu_B(r) \\
A \subseteq B &: \quad \mu_A(r) \leq \mu_B(r) \\
A \cup B &: \quad \mu_{A \cup B}(r) = \max(\mu_A(r), \mu_B(r)) \\
A \cap B &: \quad \mu_{A \cap B}(r) = \min(\mu_A(r), \mu_B(r))
\end{aligned}
$$

Finetuning

Finetuning (1/2)

The contrast of a membership function can be increased with the following function:

$$\mu_{INT(A)}(r) = \begin{cases} 2(\mu_A(r))^2 & : \quad 0 \leq \mu_A(r) \leq 0.5 \\ 1 - 2(1 - \mu_A(r))^2 & : \quad \textit{otherwise} \end{cases}$$

Finetuning

Finetuning (2/2)

Other modifying functions are:

$$\begin{aligned}
\mu_{\text{not small}} &= 1 - \mu_{\text{small}} \\
\mu_{\text{very small}} &= (\mu_{\text{small}})^2 \\
\mu_{\text{not very small}} &= 1 - \mu_{\text{very small}} \\
\mu_{\text{more or less small}} &= (\mu_{\text{small}})^{0.5}
\end{aligned}$$

Norms

Relevant for using fuzzy set theory in the context of constraint satisfaction problems are two **norms**, **T-norm (T)** and **T-conorm (S)**. Think of them as generalized versions of **AND** and **OR**.

Norms

$$T, S : [0, 1] \times [0, 1] \rightarrow [0, 1]$$

Properties

commutativity	$X(a, b)$	$=$	$X(b, a)$	
associativity	$X(X(a, b), c)$	$=$	$X(a, X(b, c))$	
monotonicity	$X(a, b)$	$\geq$	$X(c, d)$	$a \geq c \wedge b \geq d$
borders	$T(a, 1)$	$=$	a	
	$S(a, 0)$	$=$	a	

Norms

T-norm (AND)

Different variants of the T-norm:

$$
\begin{array}{llcl}
\text{minimum} & T^m(a,b) & = & \min(a,b) \\
\text{product} & T^p(a,b) & = & a * b \\
\text{quasilinear} & T^q(a,b) & = & \max(0, a+b-1) \\
\text{whoever} & T(a,b) & = & \begin{cases} a & : \quad b = 1 \\ b & : \quad a = 1 \\ 0 & : \quad \textit{otherwise} \end{cases} \\
\text{Yager} & T_p(a,b) & = & 1 - \min(1, ((1-a)^p + (1-b)^p)^{\frac{1}{2}}) \\
 & & & \text{with} \quad p \geq 0
\end{array}
$$

Norms

T-conorm (OR)

maximum	$S^m(a,b)$	$=$	$\max(a,b)$
prob. sum	$S^p(a,b)$	$=$	$a+b-ab$
quasilinear	$S^q(a,b)$	$=$	$\min(1, a+b)$
Yager	$S_p(a,b)$	$=$	$\min(1, (a^p+b^p)^{\frac{1}{2}})$
			with $p \geq 0$

Example

Expression

"put the red bolt in this block"

as fCSP

```
(var "?object-1" BOLT)
(prefer (recent-object "?object-1"))
(var "?object-2" BLOCK)
(prefer (recent-object "?object-2"))
(very (has-color "?object-1" RED))
(very (fits "?object-1" "?object-2"))
(maybe (pointed-to "?object-2"))
```

Example

Expression

"put the red bolt in this block"

as fCSP

```
(var "?object-1" BOLT)
(prefer (recent-object "?object-1"))
(var "?object-2" BLOCK)
(prefer (recent-object "?object-2"))
(very (has-color "?object-1" RED))
(very (fits "?object-1" "?object-2"))
(maybe (pointed-to "?object-2"))
```

Distributed Ontologybased Object Reference Resolution System

Introduction

DOORS

- Distributed Ontologybased Object Reference Resolution System
- developed by Pfeiffer (2003) as Diploma Thesis
- targets the Virtual Constructor
- resolves multimodal references
- based on fCSPs and hierarchical CSPs

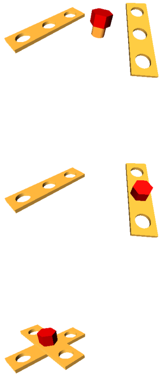
Application Example

Instruction One

I: *"put the red bolt in the middle of the three-hole-bar"*
K: *"OK"*

Instruction Two

I: *"now put the airscrew perpendicular into the middle of this three-hole-bar"*
K: *"OK"*



Constraints

Basic Constraints

- has-name
- has-color
- has-type
- has-attribute-value
- distinct

Spatial Constraints

- is-target-of-pointing-gesture
- has-position
- has-position-near-to-user
- has-size
- have-close-positions

Constraints

Connection Constraints

- port-is-free
- port-has-anchor-position
- port-has-relative-position
- is-connected-to
- part-of-same-aggregate
- is-part-of
- ports-are-connected
- ports-fit

Ontology

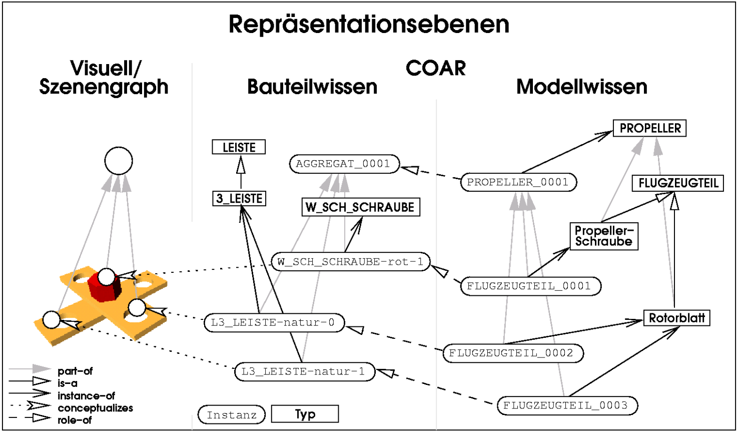


Figure: Knowledge about domain stored in an ontology on constructions

Ontology

Constraints

- has-type
- has-role
- is-connected-to
- part-of-same-aggregate

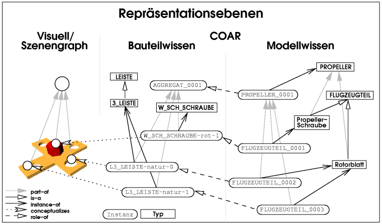


Figure: Knowledge about domain stored in ontology

Architecture

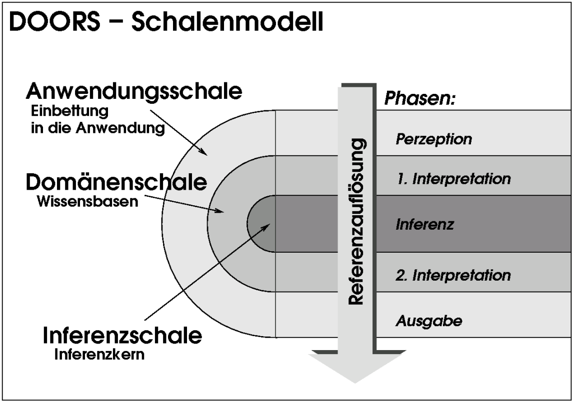


Figure: Shell-Architecture of DOORS

Architecture

Features

- Ontology for constraints (formalism is UML)
- Basic constraints can be inherited by specialized constraints
- Fuzzy CSP to cope with uncertainty in multimodal input
- . . . and with defaults (e.g. prefer objects closer together)
- Hierarchical CSP to speed up processing (e.g. simple symbolic constraints first, expensive constraints last)
- Distributed CSP leaves the data where it is (e.g. geometric constraints are evaluated in the scenegraph, symbolic constraints in the knowledge ontology)

Summary

Benefits

- CSPs are an intuitive problem representation
- fuzzy CSPs cope with the ambiguities of the real world
- combination of explicit and implicit constraints possible (also: defaults)
- fast

Drawbacks

- transformation from expression to CSP representation a problem of its own (e.g. rule-based sytem)

Bibliography

Bacchus, F. & van Beek, P. (1998). On the Conversion between Non-Binary and Binary Constraint Satisfaction Problems. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98) and of the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI-98)*, (pp. 311–318)., Menlo Park. AAAI Press.

Mackworth, A. (1977). Consistency in networks of relations [AC1-3]. *Artificial Intelligence*, *8*, 99–118.

Mackworth, A. & Freuder, E. (1985). The complexity of some polynomial network consistency algorithms for constraint satisfaction problems [AC1-3]. *Artificial Intelligence*, *25*, 65–74.

Pal, S. K. & Mitra, S. (1999). *Neuro-Fuzzy Pattern Recognition*. New York: John Wiley & Sons, Inc.

Pfeiffer, T. (2003). Eine Referenzauflösung für die dynamische Anwendung in Konstruktionssituationen in der Virtuellen Realität. Master's thesis, Faculty of Technology, Bielefeld University.

Bibliography

Zadeh, L. A. (1965). *Information and Control*, volume 8, chapter Fuzzy Sets, (pp. 338–353). ACM.