

SEMANTICS

1

Retrieval by Meaning

Query:
"Accident of a Mercedes"

Retrieved image:



Methods for retrieval by meaning:

- high-level image understanding
beyond state-of-the-art except easy cases
- natural language annotation
keywords, thesaurus, free text
- formal language annotation
ontologies, concept language, semantic web

2

What are "Meaningful" Scene Descriptions?

Description of images in meaningful terms:

Naming

- objects
- object configurations
- situations
- events, ongoing activities

according to human understanding



"garbage collection"

"mailman distributes mail"

3

High-level Scene Understanding

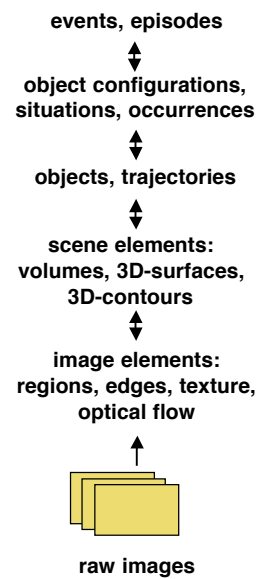
High-level scene understanding is silent-movie understanding



comprehensive real-world knowledge required ("AI-complete")!

State-of-the-art allows high-level scene understanding for restricted domains, e.g.

- traffic scenes
- parking lot supervision
- animal behaviour
- landscapes



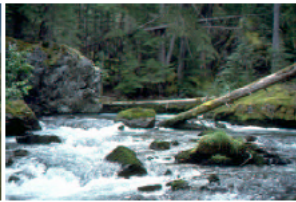
4

State-of-the-art of Semantic Scene Interpretation

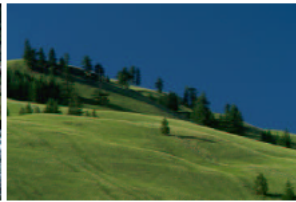
From: J. Vogel & B. Schiele, A Semantic Typicality Measure for Natural Scene Categorization
In: Pattern Recognition Symposium DAGM 2004, Tübingen, Germany, September 2004



mountains



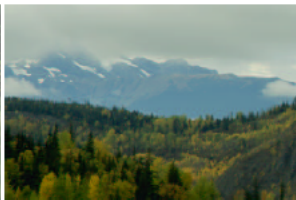
rivers/lakes



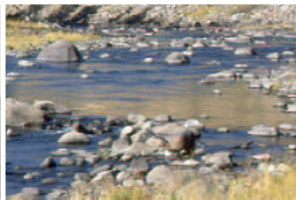
plains



forest (instead of rivers/lakes)



mountains (instead of forest)



coasts (instead of rivers/lakes)

5

Annotation Using a Formal Language


Describing the meaning of an image is a knowledge representation task

➔ use of a knowledge representation language

Query:

name:		name:	p1
class:	accident	class:	mercedes
has-parts:	p1	colour:	

Image description in database:

	name:	collision371	name:	q1	name:	q2
	class:	collision	class:	mercedes-e-type	class:	bus
	has-parts:	q1, q2	colour:	blue	colour:	yellow

6

Requirements for Formal Annotation

Intended semantics:

- Query specifies class of scenes
- Retrieval returns all images which contain instances of the query class

Basic requirements for representation language:

- classes vs. instances
- object-oriented
- taxonomical relations
- compositional relations
- spatial relations

7

Description Logics for Formal Knowledge Representation

DLs are a family of knowledge-representation formalisms

- object-centered, roles and features (binary relations)
- necessary vs. sufficient attributes
- inference services
 - subsumption check
 - consistency check
 - classification
 - instance retrieval ← free benefit from a DL system
 - abstraction
 - default reasoning
 - spatial and temporal reasoning
- guaranteed correctness, completeness, decidability and complexity properties
- highly optimized implementations (e.g. RACER)

8

Development of Description Logics

There exist several commercial and experimental developments of DLs, among them

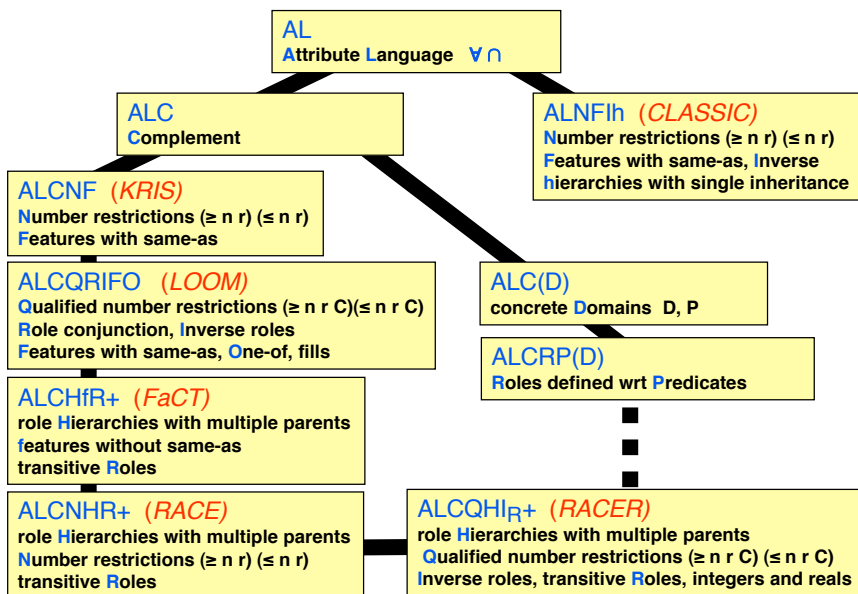
- KL-ONE first conception of a DL (1985)
- CLASSIC commercial implementation by AT&T
- LOOM experimental system at USC
- FaCT experimental and commercial system (Horrocks, Manchester)
- RACER experimental system in Hamburg and Montreal (Haarslev & Moeller)

There is active research on DLs:

- extending the expressivity of concept languages
- decidability and tractability of inference services
- integration of predicates over concrete domains (e.g. numbers)
- highly optimized implementations
- developing new inference services (e.g. for scene interpretation)

9

Family of Description Logics



10

RACER Concept Language

<p><i>C</i> concept term <i>CN</i> concept name <i>R</i> role term <i>RN</i> role name</p> <p><i>C</i> -> <i>CN</i> *top* *bottom* (not <i>C</i>) (and <i>C1 ... Cn</i>) (or <i>C1 ... Cn</i>) (some <i>R C</i>) (all <i>R C</i>) (at-least <i>n R</i>) (at-most <i>n R</i>) (exactly <i>n R</i>) (at-least <i>n R C</i>) (at-most <i>n R C</i>) (exactly <i>n R C</i>) <i>CDC</i></p>	<p><i>concept definition</i> (equivalent <i>CN C</i>)</p> <p><i>concept axioms</i> (implies <i>CN C</i>) (implies <i>C1 C2</i>) (equivalent <i>C1 C2</i>) (disjoint <i>C1 ... Cn</i>)</p> <p><i>roles</i> <i>R</i> -> <i>RN</i> (inv <i>RN</i>)</p>	<p><i>concrete-domain concepts</i> <i>AN</i> attribute name</p> <p><i>CDC</i> -> (a <i>AN</i>) (an <i>AN</i>) (no <i>AN</i>) (min <i>AN integer</i>) (max <i>AN integer</i>) (> <i>aexpr aexpr</i>) (>= <i>aexpr aexpr</i>) (< <i>aexpr aexpr</i>) (<= <i>aexpr aexpr</i>) (= <i>aexpr aexpr</i>)</p> <p><i>aexpr</i> -> <i>AN</i> <i>real</i> (+ <i>aexpr1 aexpr1*</i>) <i>aexpr1</i></p> <p><i>aexpr1</i> -> <i>real</i> <i>AN</i> (* <i>real AN</i>)</p>
--	---	---

11

Primitive and Defined Concepts

Concept expressions of a DL describe sets of entities within terms of properties (unary relations) and the roles (binary relations).

The main building blocks are primitive oder defined concepts.

Primitive concepts: concept \Rightarrow satisfied properties and relations

satisfied properties and relations are necessary conditions for an object to belong to a class

Defined concepts: concept \Leftrightarrow satisfied properties and relations

satisfied properties and relations are necessary and sufficient conditions for an object to belong to a classt

Primitive concept "person":
(implies person (and mammal (some has-gender (or female male))))

Defined concept "parent":
(equivalent parent (and person (some has-child person)))

12

Example of a TBox

```
(signature :atomic-concepts (person human female male woman man parent
                             mother father grandmother aunt uncle sister brother)
:roles ((has-child :parent has-descendant)
        (has-descendant :transitive t)
        (has-sibling)
        (has-sister :parent has-sibling)
        (has-brother :parent has-sibling)
        (has-gender :feature t)))
```

Signature of TBox

```
(implies *top* (all has-child person))
(implies (some has-child *top*) parent)
(implies (some has-sibling *top*) (or brother sister))
(implies *top* (all has-sibling (or sister brother)))
(implies *top* (all has-sister (some has-gender female)))
(implies *top* (all has-brother (some has-gender male)))
```

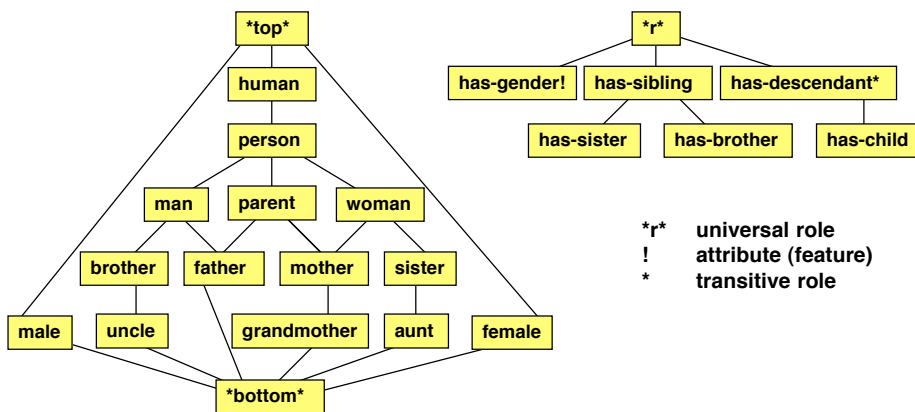
domain and range
restrictions for
roles

```
(implies person (and human (some has-gender (or female male))))
(disjoint female male)
(implies woman (and person (some has-gender female)))
(implies man (and person (some has-gender male)))
(equivalent parent (and person (some has-child person)))
(equivalent mother (and woman parent))
(equivalent father (and man parent))
(equivalent grandmother (and mother (some has-child (some has-child person))))
(equivalent aunt (and woman (some has-sibling parent)))
(equivalent uncle (and man (some has-sibling parent)))
(equivalent brother (and man (some has-sibling person)))
(equivalent sister (and woman (some has-sibling person)))
```

concepts

13

Concept and Role Hierarchies Implied by TBox



14

TBox Inferences

A DL system offers several inference services. At the core is a consistency test:

$$C \stackrel{?}{\models} \text{*bottom*} \text{ (the empty concept)}$$

Example: (and (at-least 1 has-child) (at-most 0 has-child)) \models *bottom*

Consistency checking is the basis for several other inference services:

- **subsumption**
(implies $C1 \sqsubseteq C2$) \Leftrightarrow (and $C1$ (not $C2$)) \models *bottom*
- **classification of a concept expression**
searches the existing concept hierarchy for the most special concept which subsumes the concept expression

15

ABox of a Description Logic System

TBox = terminological knowledge (concepts and roles)

ABox = assertional knowledge (facts)

An ABBox contains:

- concept assertions (instance $IN \ C$)
individual IN is instance of a concept expression C
- role assertions (related $IN_1 \ IN_2 \ RN$)
individual IN_1 is related to IN_2 by role RN
- An ABBox always refers to a particular TBox.
- An ABBox requires unique names
- ABBox facts are assumed to be incomplete (OWA).
 - OWA = Open World Assumption
(new facts may be added, hence inferences are restricted)
 - CWA = Closed World Assumption
(no facts may be added)

16

ABox Inferences

ABox inferences = inferring facts about ABox individuals

Typical queries:

- consistency *is ABox consistent?*
- retrieval *which individuals satisfy a concept expression?*
- classification *what are the most special concept names which describe an individual?*

ABox consistency checking is in general more complicated than TBox consistency checking

ABox consistent \Leftrightarrow there exists a "model" for ABox and TBox

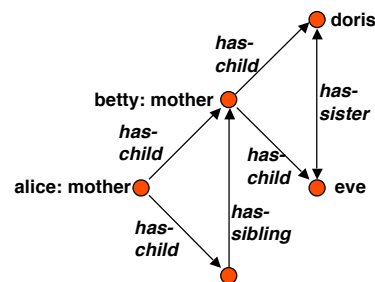
All ABox inferences are based on the ABox consistency check.

17

Example of ABox Queries

Contents of ABox

(instance alice mother)
 (related alice betty has-child)
 (related alice charles has-child)
 (instance betty mother)
 (related betty doris has-child)
 (related betty eve has-child)
 (instance charles brother)
 (related charles betty has-sibling)
 (instance charles (at-most 1 has-sibling))
 (related doris eve has-sister)
 (related eve doris has-sister)



charles: (and brother (at-most 1 has-sibling))

Questions and answers

(individual-instance? doris woman)

Is doris instance of the concept woman?

T

(individual-types eve)

Of which concept names is eve an instance?

((sister) (woman) (person) (human) (*top*))

(individual-fillers alice has-descendant)

What are the descendants of eve?

(doris eve charles betty)

(concept-instances sister)

Which instances has the concept sister?

(doris betty eve)

18

Abstraction with Description Logics

Abstraction = omission of properties or relations, extending a concept, generalization

Examples:

- **Superordinate concept name of a concept expression**
(= concept classification)
(and person (some has-size tall)) → person
- **Generalization of concept expressions**
(and (some has-occupation professor) (at-least 3 has-child))
↓
(and (some has-occupation civil-servant) (at-least 1 has-child))
- **Concept expression which subsumes several individuals**
 1. classify individuals
 2. determine least common subsumer (LCS)
 - for RACER: trivial solution in terms of (OR $C_1 \dots C_n$)
 - for DLs without OR: special abstraction operator LCS

19

Example: Table-top Scene Description

TBox (excerpt):

(implies plate dish)
(implies saucer dish)
(implies cup dish)
(implies napkin cloth)
(equivalent cover
 (and configuration
 (exactly 1 has-part plate)
 (exactly 1 has-part (and saucer (some near plate)))
 (exactly 1 has-part (and cup (some on saucer))))

ABox (excerpt):

(instance plate1 plate)
(instance saucer1 saucer)
(instance saucer2 saucer)
(instance cup1 cup)
(instance cup2 cup)
(instance napkin1 napkin)
(instance cover1 cover)
(related saucer1 plate1 near)
((related cup1 saucer1 on)
(related napkin1 plate1 on)



20

Example: Query for Table-top Scene

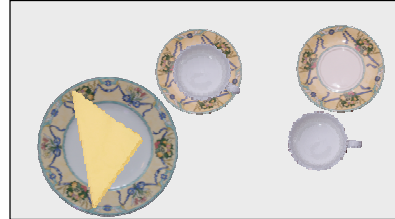
Queries:

(concept-instances cover)
⇒(cover1)

(concept-instances (some on dish))
⇒(cup1 napkin1)

(concept-instances (and cloth (some on plate))
⇒(napkin1)

(concept-instances (not (some on saucer)))
⇒() *for OWA - a fact (related (cup2 saucer3)) could be added*
⇒(cup2) *for CWA*



21

RACER Query Language

Interface language for retrieving patterns from an ABox

Basic retrieval command:

(retrieve <list-of-objects> <query-body>)

Example:

```
(retrieve (?x ?y ?z) (and(?x plate)
                        (?y saucer)
                        (?z cup)
                        (?x ?y near)
                        (?z ?y on)))
```

➔ (((?x plate1) (?y saucer1) (?z cup1))
 ((?x plate2) (?y saucer2) (?z cup2)))

Note: Query language retrieval commands allow to retrieve patterns for which no individuals have been introduced.

22

DL-Based Information Retrieval

TV assistant selects program items based on conceptual description of user preferences - prototype developed by LKI

