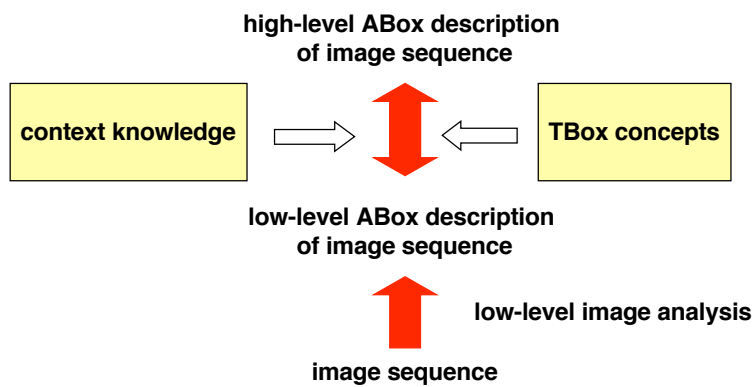


Using Description Logics for Scene Interpretation

1

Basic Structure for Scene Interpretation with a DL System



2

Perceptual Primitives

Perceptual primitives are geometrical and photometrical attributes which can be immediately determined from a GSD.

For object configurations:

- objects provide reference features in terms of
 - locations (center of gravity, corners, surface markings, etc.)
 - lines (edges, surface markings, axes of minimal inertia, etc.)
 - orientations (inate, motion, viewer)
- perceptual primitives are measurements between reference features:
 - distance
 - angle
 - temporal derivatives thereof

3

Qualitative Primitives

Qualitative primitives are predicates over perceptual primitives constant over some time interval.

- qualitatively constant values
e.g. constant orientation, constant distance
- values within a certain range
e.g. topological relations, degrees of nearness, typical speeds
- values smaller or larger than a threshold
e.g. increase of distance, slowing down

4


Meeting Basic Representational Requirements with a DL System

- object oriented representations
yes, but needs user interface
- n-ary relations
no, only binary relations
- taxonomies
yes, automatically constructed from conceptdefinitions
- partonomies
yes, can be represented by roles
- spatial and temporal relations
can be computed from quantitative data via concrete domain extensions
- qualitative predicates
can be computed from quantitative data via concrete domain extensions

5

Representing N-ary as Binary Relations

Reification:

(BETWEEN A B C)  (INSTANCE BETW1 BETWEEN)
(BETWEEN-ARG1 BETW1 A)
(BETWEEN-ARG2 BETW1 B)
(BETWEEN-ARG3 BETW1 C)

(OVERTAKE VEH1 VEH2 23 46)  (INSTANCE OT1 OVERTAKE)
(OVERTAKER OT1 VEH1)
(OVERTAKEE OT1 VEH2)
(TBEG OT1 23)
(TEND OT1 42)

6

Concrete Domain Concepts in RACER

CDC → (a AN) (an AN)
 (no AN)
 (min AN integer)
 (max AN integer)
 (equal AN integer)
 (> aexpr aexpr)
 (>= aexpr aexpr)
 (< aexpr aexpr)
 (<= aexpr aexpr)
 (= aexpr aexpr)

aexpr → AN
 real
 (+ aexpr1 aexpr1*)
 aexpr1

aexpr1 → AN
 real
 (* real AN)

Example:
 Quantitative constraints on the size
 of an object

(and (min size 13) (max size 20))



integer-valued attribute "size"
 receives values from low-level vision

7

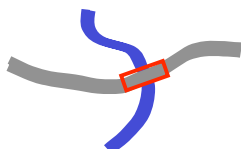
DL Concept for a Cover

```
(equivalent cover
  (and configuration
    (exactly 1 cv-pl plate)
    (exactly 1 cv-sc (and saucer (some near plate)))
    (exactly 1 cv-cp (and cup (some on saucer)))
    (subset cv-pl (compose cv-sc near))
    (subset cv-sc (compose cv-cp on))))
```

- parts are expressed as qualified fillers of specific roles
 e.g. cv-pl, cv-sc, cv-cp
- sameness (or distinctness) of parts and properties of parts are expressed by the subset construct
- spatial constraints are modelled as primitive predicates
 e.g. near, on

8

Example: DL Model for a Bridge



Assumptions:

Image analysis computes bottom-up

- strips (= lengthy regions)
- colours
- spatial relations (touch, contain)

TBox:

```
(equivalent bridge
  (and strip-section
    (some has-road road)
    (some has-river1 river)
    (some has-river2 river)
    (subset has-road o contain)
    (subset has-river1 o touch)
    (subset has-river2 o touch)))

(equivalent strip-section
  (and (some within strip)
    (= has-width within o has-width)))

(equivalent road
  (and strip
    (some has-colour road-colour)))

(equivalent river
  (and strip
    (some has-colour river-colour)))
```

Example ABox:

```
(instance strip1 strip)      (related strip1 blue has-colour)  (related strip1 strip3 touch)
(instance strip2 strip)      (related strip2 blue has-colour)  (related strip2 strip3 touch)
(instance strip3 strip)      (related strip3 greyhas- colour)  (related strip3 strip1 touch)
...                          ...                                     (related strip3 strip2 touch)
...                          ...                                     ...
```

Problem: Generating instances of strip-section

9

Simplified DL Concept for Placing a Cover

```
(equivalent place-cover
  (and agent-activity
    (exactly 1 pc-tp1 (and transport (some tp-obj plate)))
    (exactly 1 pc-tp2 (and transport
      (some tp-obj saucer)
      (some before (and transport (some tp-obj cup))))))
    (exactly 1 pc-tp3 (and transport (some tp-obj cup)))
    (subset pc-tp3 (compose pc-tp2 before))))
```

Severe disadvantage of purely symbolic spatial and temporal constraints:

Pairwise constraints must be computed bottom-up by low-level vision procedures irrespective of high-level concepts!



Express spatial and temporal constraints as predicates over concrete-domain elements

10

Quantitative Spatial and Temporal Constraints

```
(equivalent place-cover
  (and agent-activity
    (exactly 1 pc-tp1 (and transport (some tp-obj plate))
      (exactly 1 pc-tp2 (and transport (some tp-obj saucer))
        (exactly 1 pc-tp3 (and transport (some tp-obj cup))
          (<= pc-tp2 o tp-end pc-tp3 o tp-end)
          (= pc-beg (minim pc-tp1 o tp-beg pc-tp2 o tp-beg pc-tp3 o tp-beg))
          (= pc-end (maxim pc-tp1 o tp-end pc-tp2 o tp-end pc-tp3 o tp-end))
          (<= (- pc-end pc-beg) max-duration))))))
```

- Equality and inequality as concrete domain predicates
- Specific constraints for each concept
- Incremental constraint computation required for prediction!

Example: (and (= cv-sc o sc-loc cv-cp o cp-loc))

Known saucer position restricts expected cup positions

11

General Structure for Aggregate Definitions

```
(equivalent <concept-name>
  (and <parent-concept1> ... <parent-conceptN>
    (<number-restriction1> <role-name1> <part-concept1>)
    ...
    (<number-restrictionK> <role-nameK> <part-conceptK>)
    <constraints between parts>))
```

Summary of DL constructs required for aggregates: ALCF(D)

=> aggregates can in principle be represented in RACER, however,
not all syntax features are currently available

12

Image Interpretation as Deduction?

The classifier of a description logic carries out classifications automatically:

evidence \Rightarrow class (concept) membership

Problems:

- partial evidence must be sufficient
- deduction of all possible partial interpretations
- no goal-oriented analysis
- no comparative evaluation of conflicting interpretations

Support of hypothesize-and-test cycle is required !

13

Hypothesizing Possible Concept Specializations

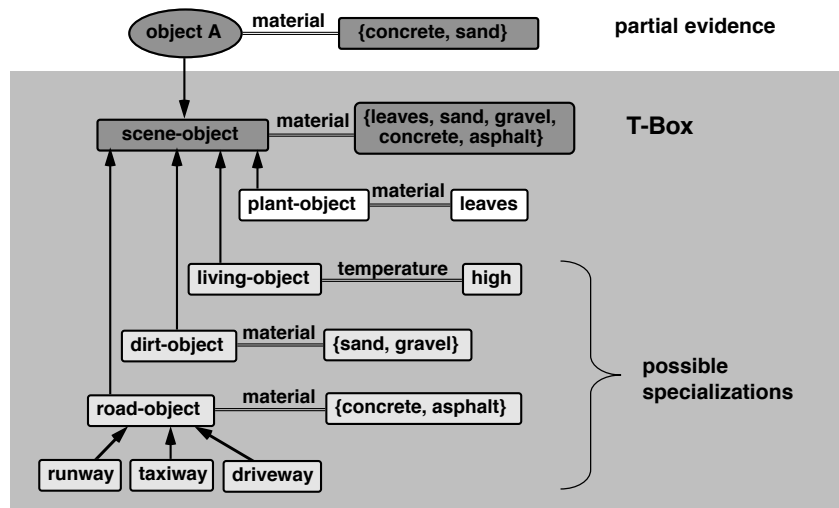
Extension of description logic reasoning service for hypothesis generation:

- Which concept hypotheses can be specialized further consistent with existing evidence?
- Which additional evidence is required for specialization?

1. partial evidence \Rightarrow consistent concepts
2. partial evidence + concepts \Rightarrow missing evidence

14

Example for Possible Concept Specializations

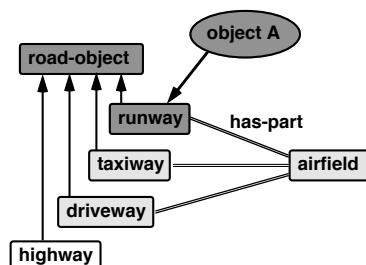


15

Hypothesizing Possible Aggregats (1)

For which concepts (aggregats) are roll fillers (parts) available?

- Provide concepts which are consistent with existing role fillers
- Which roles provide decisive evidence?
- Criteria for ranking hypotheses



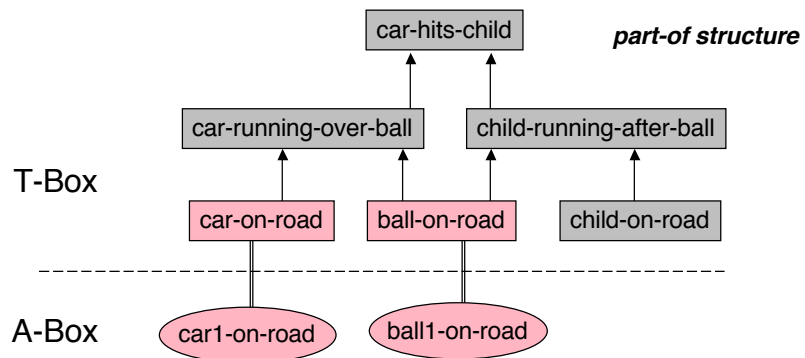
Existing instance runway is evidence for airfield and its further parts taxiway and driveway.

16

Hypothesizing Possible Aggregats (2)

For which concepts (aggregats) are roll fillers (parts) available?

Generating temporal and spatial expectations:

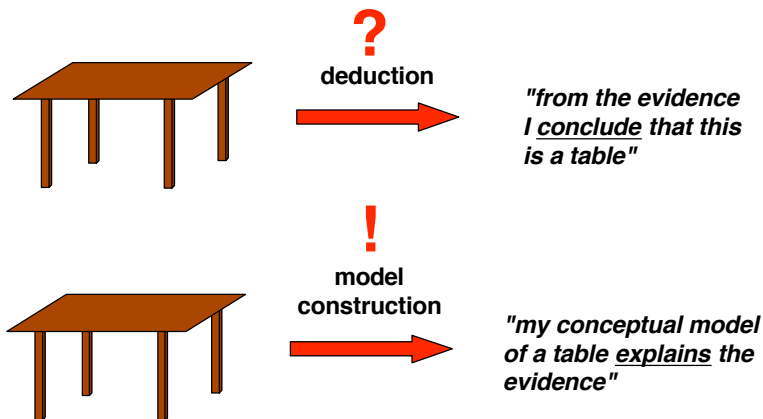


17

Logics of Image Interpretation

18

Describing Image Interpretation in Logical Terms



Reiter & Mackworth 87, Matsuyama 90, Schröder 99

19

Image Interpretation as (Logical) Model Construction

An interpretation $I = [D, \varphi, \pi]$ of a logical language maps

- constant symbols of the language into elements of a real-world domain D
- predicate symbols of the language into predicate functions over D

A model of some clauses is an interpretation where all predicates are true.

Image interpretation as model construction:

- establish mapping φ by assigning segmentation results to constant symbols
- establish mapping π by assigning computational procedures to predicate symbols
- find clauses for which predicates are true

Deciding whether a model exists is undecidable in FOPC!
There may be infinitely many models!

20

Finite Model Construction (Reiter & Mackworth 87)

- an image consists of regions and chains (edges)
- the image elements constitute all constant symbols of an interpretation (domain closure assumption)
- different constant symbols denote different image elements and vice versa (unique name assumption)

➔ Problem can be expressed in Propositional Calculus and solved as a constraint satisfaction problem (CSP)

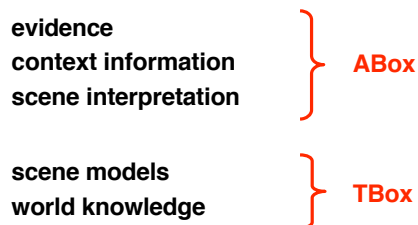
For MAPSEE, scene interpretation amounts to finding a mapping π for predicates *road, river, shore, land, water*.

21

So what is a Scene Interpretation?

Intuitively:

A scene interpretation is a scene description in terms of instantiated scene models consistent with evidence, context information and world knowledge.



22

Practical Requirements for Partial Logical Models

- **Task-dependent scope and abstraction level**
 - **no need for checking all predicates**
e.g. propositions outside a space and time frame may be uninteresting
 - **no need for maximal specialization**
e.g. geometrical shape of "thing" suffices for obstacle avoidance
- **Partial model may not have consistent completion**
 - **uncertain propositions due to inherent ambiguity**
 - **predictions may be falsified**
- **Real-world agents need single "best" scene interpretation**
 - **uncertainty rating for propositions**
 - **preference measure for scene interpretations**



Logical model property provides only loose frame for possible scene interpretations

23

Stepwise Construction of Partial Models

Four kinds of interpretation steps for constructing interpretations consistent with evidence:

Aggregate instantiation

Inferring an aggregate from (not necessarily all) parts

Instance specialization

Refinements along specialization hierarchy or in terms of aggregate parts

Instance expansion

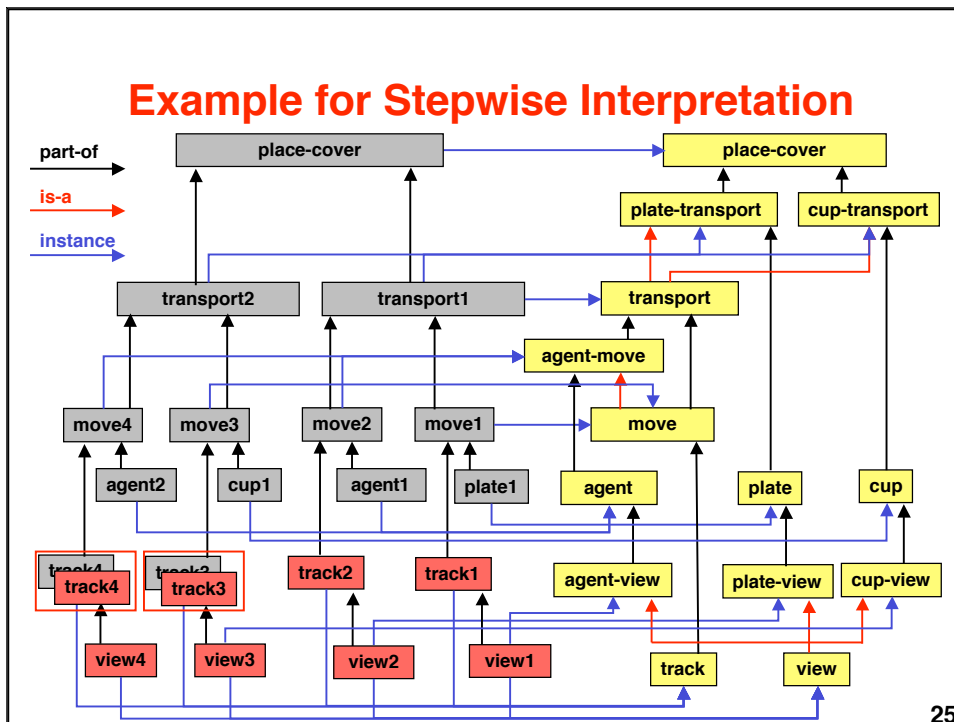
Instantiating parts of an instantiated aggregate

Instance merging

Merging identical instances constructed by different interpretation steps

Repertoire of interpretation steps allow flexible interpretation strategies
e.g. mixed bottom-up and top-down, context-dependent, task-oriented

24



DL Reasoning Services

ABox consistency checking is at the heart of all reasoning services
Model construction is the method of choice for many DL reasoners

- Concept satisfiability
- Concept subsumption
- Concept disjointness
- Concept classification
- TBox coherence
- **ABox consistency w.r.t. a TBox**
- Instance checking
- Most-specific atomic concepts of which an individual is an instance
- Instances of a concept
- Role fillers for a specified individual
- Pairs of individuals related by a specified role
- Conjunctive queries

26

DL Reasoning Support for Scene Interpretation

- **Maintaining a coherent knowledge base**
Scene interpretation may require extensive common-sense knowledge, intuitive knowledge representation is doomed
- **Maintaining consistent scene interpretations**
A consistent ABox is a (partial) model and hence formally a (partial) scene interpretation => ABox consistency checking ensures consistent scene interpretations

ABox realization (computing most specific concepts for individuals) cannot be used in general:

- **scene interpretations cannot be deduced**
- **high-level individuals must be hypothesized before consistency check**

27

DL Support for Interpretation Steps

Aggregate instantiation

Determine aggregates for which an individual is a role filler
=> RACER query language

Instance specialization

Retrieve all specializations of a given concept
=> use specialization hierarchy

Instance expansion

Instantiate parts of an aggregate instance
=> easy service by looking up the aggregate definition

Instance merging

Determine whether it is consistent to unify two individual descriptions
=> unification by recursive specialization can be supported

Important missing service:

Preference measure for choosing "promising" alternatives

28