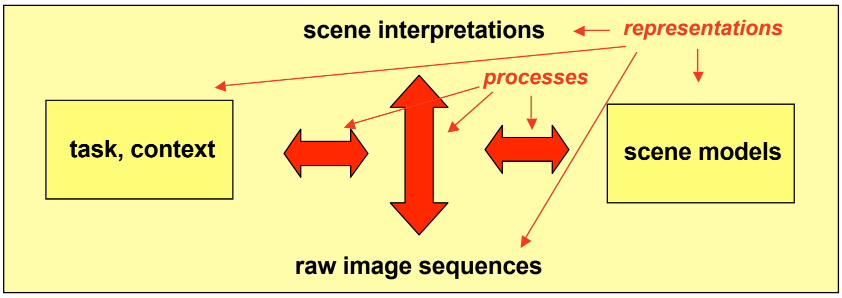
Representations and Processes in Knowledge-based Systems



Characteristics of ideal knowledge-based systems:

- Problems are specified by background and task knowledge using a declarative knowledge representation language
- Problems are solved using standard inference procedures

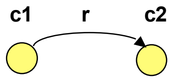
Knowledge representation formalisms must support representations and processes (inferences)!

Basic Knowledge Representation Formalisms

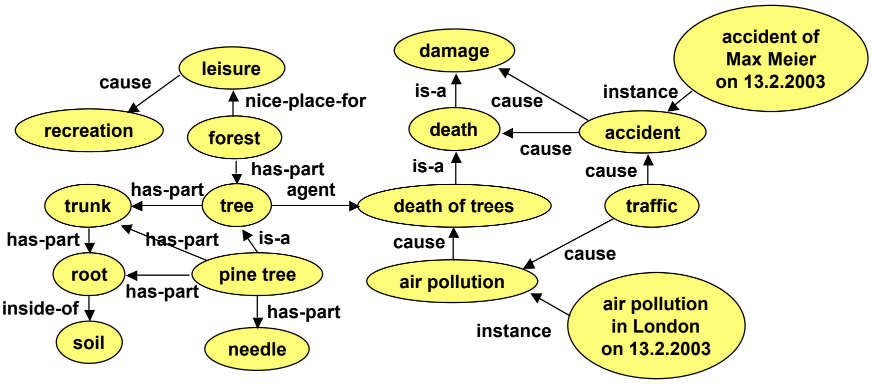
- Semantic Networks
- Frames
- Constraints
- Relational Structures

Semantic Networks

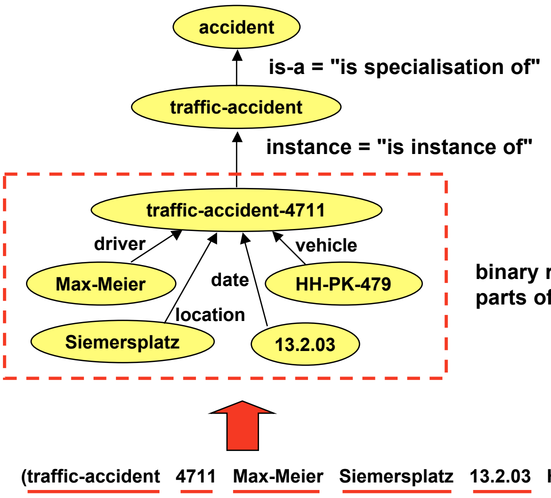
Graphical representation of binary relations:
labelled nodes = concepts
directed labelled edges = binary relations



Semantic networks were originally developed to model associations between concepts in the human mind.



Basic Relations in Semantic Networks



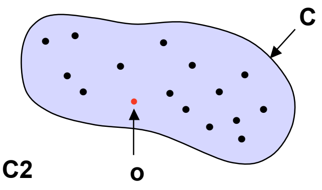
binary relations linking to
parts of a reified n-ary relation

Concepts and Instances

Nodes of a semantic network describe concepts and individuals.

A concept denotes a <u>set of objects</u>.

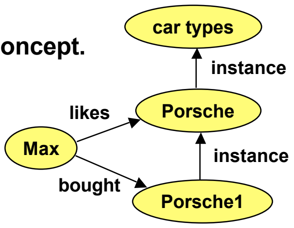
An individual denotes a <u>single object</u>.



C1 is-a C2 specifies that C1 is a subset of C2

o instance C specifies that o is a member of C

A node may represent both, an individual and a concept.
<u>Example</u>:

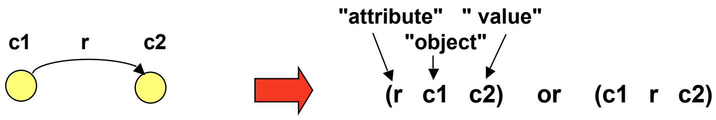


Max likes a Porsche.

Max bought a Porsche at the car dealer.

Attribute-Object-Value Triplets

In knowledge representation languages and programming languages, a semantic network can be represented by a set of triplets:



(r c1 c2) or (c1 r c2)

The accident example:

(is-a traffic-accident accident)
(instance traffic-accident-4711 traffic-accident)
(driver traffic-accident-4711 Max-Meier)
(location traffic-accident-4711 Siemersplatz)
(date traffic-accident-4711 13.2.03)
(vehicle traffic-accident-4711 HH-PK-479)

<u>Note</u>:

- notions of attribute, object and value do not always seem fitting
- notation is not object centered

Physical Object Descriptions

Scene interpretations are based on information about physical objects. Hence concepts and relations about the "physical reality" are important.

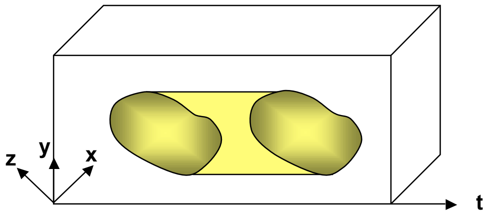
Characteristics of physical object descriptions:

1) Individuals
- **A description is valid for an absolute time point or period**
- **Nodes denote individual physical objects (or object parts)**
- **Objects have a spatial extent**
- **Objects often have a shape and appearance**
- **Objects are often described in terms of location and orientation**
- **Objects obey physical laws**

2) Concepts
- **Concepts define equivalent classes by abstracting from invidual properties**
- **Abstractions may be defined in terms of qualitative properties**
- **Abstractions may involve relations to other objects**

Objects in Space-Time

An epistemologically well-founded way of defining an individual physical object is in terms of a subspace of 4-dimensional space-time.



Example: The potatoe lying on the Max Meier´s table from 11:45 until 12:10 on August 8, 2003.

Individual physical objects which keep their identity over time constitute a common kind of abstractions.

Example: Max Meier is considered an individual in spite of his changes over time.

Frames

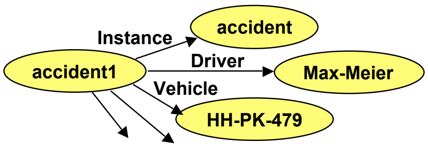
Frames have been proposed as knowledge representation structures for representing interrelated knowledge in larger units.

Marvin Minsky: "A Framework for Representing Knowledge", 1975

Simple frame structure for the individual "accident1":

ID:	accident1
Instance:	accident
Driver:	Max-Meier
Vehicle:	HH-PK-479
Location:	Siemersplatz
Date:	13.2.03
Damage:	5000-EUR
Police Report:	HH-2003-AX4711
Witness:	Karl-Kruse

- Slots represent binary relations:



- Slot fillers may be primitives or frames
- Inheritance and other inference services may be provided

Frame Representation Language FRL

- Facet names specify different slot filler "metatypes":

$DATA	normal data
$DEFAULT	default values
$IF-ADDED	write-access triggers specified demon procedures
$IF-NEEDED	read-access triggers specified demon procedures
$REQUIRE	demon procedures check conditions which must be met by slot fillers

- Built-in inference services enriched by demon procedures

Example:

ID:	($DATA Person007)
Is-a:	($DATA Person)
Name:	($DATA Max-Meier)
Age:	($REQUIRE Agetest) ($DATA 27)
Nationality:	($DEFAULT German)
Hobbies:	($DATA Eating, Sleeping, Singing) ($IF-ADDED Singing Notify-Uni-Choir)
Phone:	($IF-NEEDED Directory-Retrieval-Service)
Address:	($DATA Address4711

Values are retrieved

1. from $DATA facet
2. by inheritance from parent $DATA facets
3. from $DEFAULT facet
4. by inheritance from parent $DEFAULT facets
5. by $IF-NEEDED demon procedures

Constraints

Constraints express restrictions on the values of variables.

Given variables and constraints, a constraint satisfaction problem (CSP) is the task of assigning values to the variables such the constraints are satisfied.

Constraints are useful for knowledge representation and inferencing:

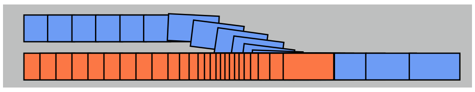
- **Constraints may provide a compact representation for n-ary relations**

  Example: sum(action1.duration, action2.duration, action3.duration) ≤ 120 sec

- **Spatial and temporal constraints are important for scene interpretation**

Is this a cover?

Is this a forced brake ("Ausbremsen")?



- **There exist efficient algorithms for solving special CSPs**
- **Constraints support flexible interpretation strategies**

Constraints for Scene Interpretation (1)

Constraints may model conceptual knowledge:

properly-parked-car-group
parts: f-car is-a car
b-car is-a car
r-car is-a car
constraints: behind (b-car, f-car)
behind (r-car, b-car)
distance (f-car, b-car) ≥ 30cm
distance (r-car, b-car) ≥ 30cm

Constraints may express concrete knowledge about a scene:

length(car1) ≤ 300cm
behind (car1, car2)
behind (car3, car2)
distance (car1, car2) = 42cm
distance (car3, car2) ≥ 400cm

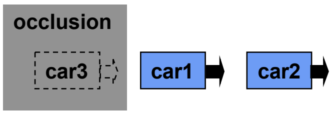
Constraints may express inferred knowledge about a scene:

behind (car3, car1)
distance (car3, car1) ≥ 100cm

Constraints for Scene Interpretation (2)

Constraints may restrict hypotheses:

properly-parked-car-group1
parts: car2 is-a car
car1 is-a car
car3 is-a car
constraints: behind (car1, car2)
behind (car3, car1)
distance (car1, car2) = 42cm
distance (car3, car1) ≥ 30cm



car3 is not visible but may be hypothesized

Constraints may help to focus processing:

behind (car3, car1)

scene analysis may be focussed on "behind"-region

Hard and Soft Constraints

- Hard constraints must be satisfied. A violated constraint prohibits a solution. The CSP is a satisfiability problem.
- Soft constraints should be satisfied. A violated constraint impairs the quality of a solution. The CSP is an optimization problem.

Constraints relevant for scene interpretation may have different origin:

Constraints arising from logics
Examples:
- to be "relatives" persons must have a common ancestor
- "same-object-as" requires that two objects are identical
- "touches" implies "near"

Constraints arising from physical laws
Examples:
- an object may not be at different places at the same time
- different solid objects may not occupy the same place at the same time
- "holding" requires that the holder is physically connected to the held object

Constraints arising from conventions or goal-directed behavior
Examples:
- spatial constraints for a "cover" on a table
- temporal constraints for a typical "overtake"
- actions for inserting a CD into a CD-player

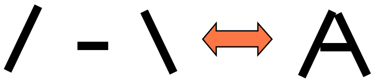
Relational Models

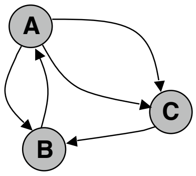
Relational models describe objects (object classes) based on parts (components) and relations between the parts

Relational model can be represented as structure with nodes and edges:

<u>nodes</u>: parts with properties

e.g.





<u>edges</u>: relations between parts

e.g.

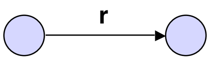
- **- obtuse-angle**
- **- 2cm-distance**
- **- touches**
- **- surrounds**
- **- left-of**
- **- after**

Relations between Components

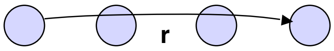
unary relation:	**property**
n-ary relation:	**relation, constraint**

<u>Graphical representation</u>

binary relation:



n-ary relation:



"hypergraph"

Relational Models for High-level Vision

Relational models describe objects (object classes) based on parts (components) and relations between the parts

A relational model can be represented as a structure with nodes and edges:

Nodes: parts with properties

A
is-a person
state running

B
is-a person
state jumping

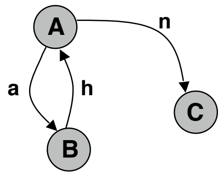
C
is-a ball
colour black

Edges: relations between parts

approaches A B

nearby B A

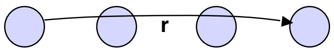
holds B C



Representing N-ary Relations

Awkward graphical representation:

"hypergraph"



N-ary relations can be transformed into binary relations:

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

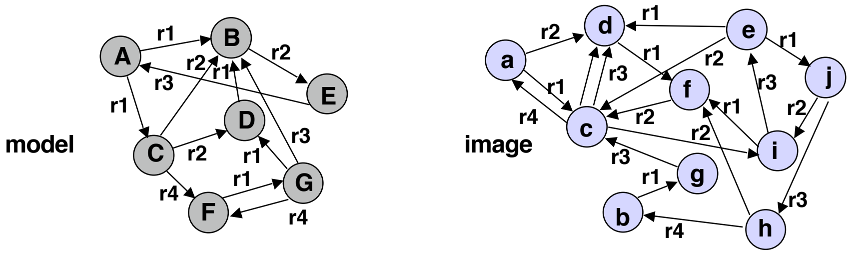
(OVERTAKE VEH1 VEH2 23 46) ➡ (INSTANCE OT1 OVERTAKE)
(OVERTAKER OT1 VEH1)
(OVERTAKEE OT1 VEH2)
(TBEG OT1 23)
(TEND OT1 42)

Transforming a relation into an object is called reification.

Recognition by Relational Matching

Principle:

- construct relational model(s) for object class(es)
- construct relational image description
- compute morphism (best partial match) between image and model(s)



Compatibility of Relational Structures

Different from graphs, nodes and edges of relational structures may represent entities with rich distinctive descriptions.

Example: **nodes = image regions with diverse properties**
edges = spatial relations

1. Compatibility of nodes

An image node is compatible with a model node, if the properties of the nodes match.

2. Compatibility of edges

An image edge is compatible with a model edge, if the edge types match.

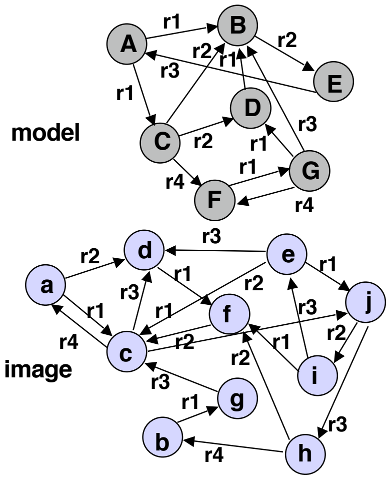
3. Compatibility of structures

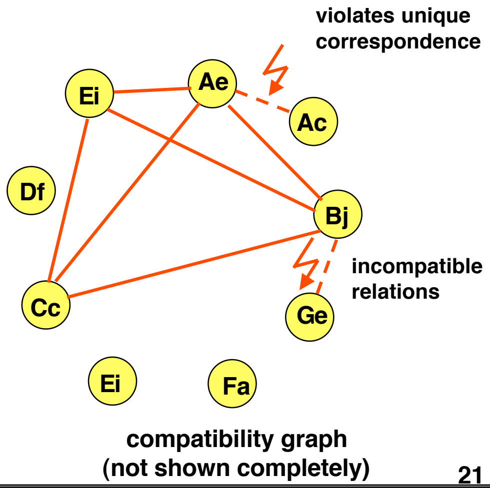
A relational image description B is compatible with a relational model M, if there exists a bijective mapping of nodes of a partial structure B´of B onto nodes of a partial structure M´of M such that

- corresponding nodes and edges are compatible
- M is described by M´ with sufficient completeness

Relational Matching with a Compatibility Graph

nodes of compatibility graph = pairs with compatible properties
edges of compatibility graph = compatible pairs
cliques in compatibility graph = compatible partial structures





compatibility graph
(not shown completely)

Finding Maximal Cliques

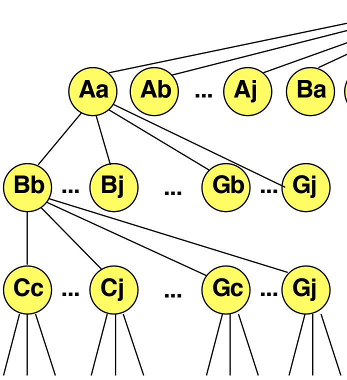
Algorithms are available in the literature, e.g.

Bron & Kerbusch, Finding all Cliques of an Undirected Graph, Communications of the ACM, Vol. 16, Nr. 9, S. 575 - 577, 1973.

- Complexity is exponential relative to number of nodes of compatibility graph
- Efficient (suboptimal) solutions based on heuristic search

Relational Matching with Heuristic Search

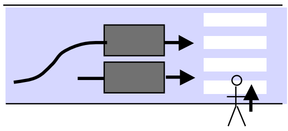
Stepwise correspondence search between model nodes {A ... G} and image nodes {a ... j}



- quality function evaluates partial matches
- accept a partial match if quality > acceptance threshold
- refute a partial match, if quality < refutation threshold

Example for Relational Event Recognition (1)

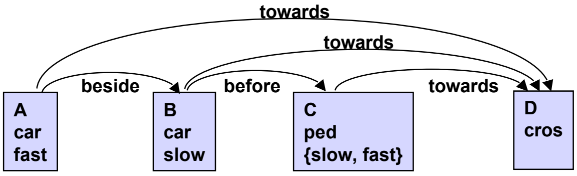
Suppose we want to recognise dangerous overtake events at pedestrian crossings:



<u>Nodes:</u> type: {car, pedestrian, plain_road, crossing}
speed: {zero, slow, fast}

<u>Relations:</u> {beside, behind, before, towards, away}

Model for dangerous overtake events:



Example for Relational Event Recognition (2)

Is there a dangerous overtake event in the following exciting traffic scene?

a car fast
b car fast
c car slow
d car slow
e ped fast
f ped slow
g ped slow
h ped slow
i ped slow
k cros
l plain_road

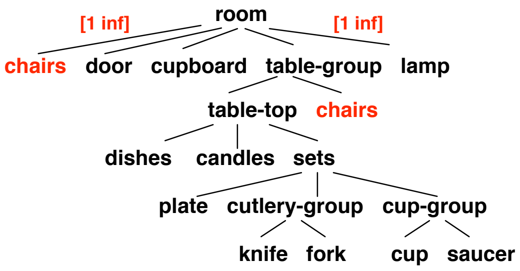
a behind b
b before a
b towards k
c away k
d towards k
b beside d
f towards l
g towards k
e beside h
i away k
h towards l

- Apply heuristic search!
- What heuristic may be useful?
- How can the approach be improved?

Shortcomings of Relational Matching for Scene Interpretation (1)

Natural hierarchical structures and groupings are not well represented by flat relational structures.

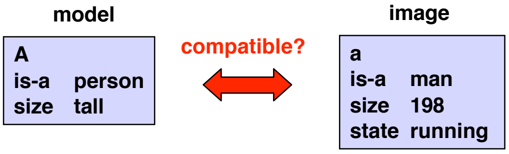
<u>Example</u>: Modelling dining room views



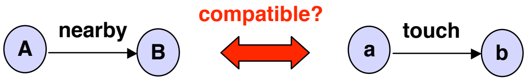
Symbolic hierarchical models allow unique representations for repeated substructures, cardinality information for part-of relations and other features not available in flat relational models.

Shortcomings of Relational Matching for Scene Interpretation (2)

Node compatibility is not clearly defined



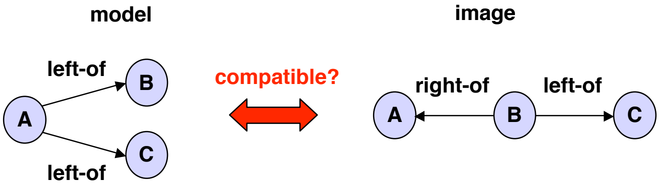
Edge compatibility is not clearly defined



Logical relations between different node descriptions and different edge labels must be represented.

Shortcomings of Relational Matching for Scene Interpretation (3)

Implicit information about the semantics of relations is not considered



Reasoning may be required to determine compatibility

How Useful is Relational Matching?

- **Relational structure captures basic high-level notions**
- **Graceful degradation w.r.t. completeness and degree of match**
- **Well-understood computional procedures**
  - finding maximal cliques in compatibility graphs
  - heuristic search
  - constraint satisfaction
  - neural network implementations
- **Improvement by hierarchical matching**

- **Multi-level aggregate structure required**
- **Differentiated compatibility measure required**
  - fuzziness
  - compatibilty vs. consistency
  - probabilities
- **Reasoning about temporal, spatial, physical relations**
- **Uncertainty management required**