Case-Based Reasoning

Similarity / Diagnosis
Prof. Ian Watson

The CBR-cycle

1. RETRIEVE
   - similarity
   - retrieved knowledge

2. REVIEW
   - selected knowledge

3. REVISE
   - new knowledge

4. REFINE
   - selected knowledge

5. RETAIN
   - new knowledge

6. MEMORY
   - knowledge requirement

Similarity ???

- Purpose of similarity, either:
  - Select cases that can be adapted easily to solve the current problem
  - Select cases that have (nearly) the same solution to the current problem

- Basic assumption:
  - similar problems have similar solutions
Similarity ???

- Degree of similarity = utility or reusability of the solution
- Similarity is an *apriori* approximation of reusability
- Goal of similarity modelling:
  - provide a good approximation
  - close to real reusability
  - And easy to compute

Similarity ???

- Assumptions
  - 2 similar problem descriptions have similar solution descriptions
  - It is easier to adapt the solution of a similar problem than the solution of a less similar problem
Similarity ???

- Description of new problem to solve
- Description of solved problems
- Stored solutions
- New solution created by adaptation

Modeling similarity

- Different approaches depending on case representation
- Similarity measures (metrics):
  - Functions to compare two cases:
    \[ \text{sim}(\text{Case}_1,\text{Case}_2) = [0..1] \]
  - Local similarity measure: similarity on feature level
  - Global similarity measure: similarity on case or object level
    - Combines local similarity measures
    - Takes care of different importance of attributes (weights)

Hamming distance

- A measure of the difference between two messages, each consisting of a finite string of characters, expressed by the number of characters that need to be changed to obtain one from the other.
- E.g., 0101 and 0110 has a Hamming distance of two
Hamming distance

- In CBR each case feature in target case is compared to features of cases in entire case-base

<table>
<thead>
<tr>
<th>Target case</th>
<th>case-base</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apples</td>
<td>Apples</td>
</tr>
<tr>
<td>Bread</td>
<td>Bread</td>
</tr>
<tr>
<td>Eggs</td>
<td>Butter</td>
</tr>
<tr>
<td>Beer</td>
<td>Beer</td>
</tr>
<tr>
<td>Steak</td>
<td>Steak</td>
</tr>
</tbody>
</table>

Hd: $4/5 = 0.8 \quad 3/5 = 0.6$

Hamming distance

- Problem - no attempt to indicate the importance or value of individual features

<table>
<thead>
<tr>
<th>Target case</th>
<th>case-base</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apples</td>
<td>Apples</td>
</tr>
<tr>
<td>Bread</td>
<td>Bread</td>
</tr>
<tr>
<td>Eggs</td>
<td>Butter</td>
</tr>
<tr>
<td>Beer</td>
<td>Beer</td>
</tr>
<tr>
<td>Steak</td>
<td>Steak</td>
</tr>
</tbody>
</table>

Hd: $4/5 = 0.8 \quad 3/5 = 0.6$

Similarity

- We therefore need more complex local similarity metrics
  - $4 == 10? \quad \Delta = 6$
  - $10 == 20? \quad \Delta = 10$
  - $20 == 100? \quad \Delta = 80$

  Therefore the range is important & outliers are bad
Similarity

- We therefore need more complex local similarity metrics
  - Sim = R - ∆
- 4 == 10?  ∆ = 100 - 6  Sim = 94%  
- 10 == 20?  ∆ = 100 - 10 Sim = 90%  
- 20 == 100?  ∆ = 100 - 80 Sim = 20%

Similarity

- Symbolic attributes are commonly modelled using symbol hierarchies

```
  plant
    fruit
      citrus fruit
      orange
    vegetable
      apple
      potatoe
      onion

Orange & Mandarin share the same parent (1 node higher)
Orange & Onion's shared ancestor is 3 nodes higher
Orange is 3x more similar to Mandarin than Onion
Orange == Mandarin Sim = 75%
Orange == Onion Sim = 25%
```

Similarity Metrics

- We'll use CBR-Works as an exemplar
  - Symbols
  - Numbers
  - Intervals
  - Sets
  - Ordered sets
  - Strings
  - Taxonomy
**Similarity**

- **Symbols (unordered)**
  - Similarity defined by developer
  - Similarity values stored in a decision table

<table>
<thead>
<tr>
<th>Similarity Value</th>
<th>stated</th>
<th>symmetric</th>
<th>asymmetric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car</td>
<td>0.6</td>
<td>0.0</td>
<td>0.8</td>
</tr>
<tr>
<td>Coach</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>Plane</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Train</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Symmetric vs asymmetric similarity**

- In symmetric similarity the result is independent of the role of the values being compared
  - Sim(Car, Train) = 0.8
  - Sim(Train, Car) = 0.8
- In asymmetric similarity the role is important
  - Sim(Car, Train) = 0.8
  - Sim(Train, Car) = 0.3

**Numbers**

- **Step function**
  - Computes a sim of 1 if the difference is < step point otherwise a sim of 0
- **Polynomial function**
  - Computes a sim between 0 & 1 using a polynomial curve (ie the closer to the query the greater the sim)
- **Function with a smooth step point**
  - sim will be in (0.0; 1.0) in respect to the distance between query- and case-value.
Similarity

Perfect options
- *Greater is perfect*
  - More horsepower
  - More days holiday
- *Less is perfect*
  - Less money
  - Less time

Similarity

Intervals
- if the intervals in query and case do not intersect
  the similarity is higher the closer the gap
- if the intervals intersect the similarity is higher the closer the bounds
- if the case completely covers the query the
  similarity is 1
- if the query completely covers the case the similarity is higher the closer the bounds

Similarity

Sets – three similarity methods

<table>
<thead>
<tr>
<th>Similarity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intersection</td>
<td>The similarity computes to the proportion of the intersection of two sets to its union.</td>
</tr>
<tr>
<td>Case inclusion</td>
<td>The similarity equals to 1, if the case is included in the query. Entires included in case but not in query lower the similarity.</td>
</tr>
<tr>
<td>Query inclusion</td>
<td>The similarity equals to 1, if the query is included in the case. Entires included in query but not in case lower the similarity.</td>
</tr>
</tbody>
</table>
Similarity

- Ordered Symbols
  - Similar to numbers
  - The symbols are mapped to a numeric range

Similarity

- Strings
  - exact match - two strings are similar if they are spelled the same way
  - spelling check - compares the number of letters in two strings which are the same (Useful for strings consisting of one word only)
  - word-count - counts the number of matching words of two cases. (Useful for strings consisting of several words).

Similarity

- Taxonomy
  - A classification hierarchy defines similarity for concepts
  - Inner nodes of the tree are assigned similarity values
  - Leaves under a node will share the nodes similarity

Taxonomy tree: [Diagram of a taxonomy tree with nodes and branches labeled with similarity values]
Retrieve, but efficiently ...

- Efficient case retrieval is essential for large case bases
- Different approaches depending on the:
  - case representation
  - size of the case base
- Organization of the case base:
  - Linear lists, only for small case bases
  - Index structures for large case bases
    - Kd-trees: index structure for large case bases (Wess)
    - Retrieval nets: index structure for textual (conversational) CBR (Lenz)

Technical diagnosis

- Simple example: Car Faults
  - Symptoms are observed (e.g. engine doesn't start) and values are measured (e.g. battery voltage = 6.3V)
  - Goal: Find the cause for the failure (e.g. battery empty) and repair strategy (e.g. charge battery)
- Case-Based Diagnosis:
  - A case describes a diagnostic situation and contains:
    - description of the symptoms
    - description of the failure and the cause
    - description of a repair strategy

Technical diagnosis

- Technical diagnosis
- Technical diagnosis
Each case describes one situation
Cases are independent of each other
Case are not rules

### Solving a diagnostic problem

- Make several observations about new problem
- Not all features must be known
- The new problem is a case without the solution part

### Similarity

<table>
<thead>
<tr>
<th>Problem (Symptoms)</th>
<th>Solution</th>
</tr>
</thead>
</table>
| - Problem: Front light doesn't work  
- Car: VW Golf 1.6 L  
- Year: 1993  
- Battery voltage: 13.6 V  
- State of lights: OK  
- State of light switch: OK | - Diagnosis: Front light fuse defect  
- Repair: Replace front light fuse |

<table>
<thead>
<tr>
<th>Problem (Symptoms)</th>
<th>Solution</th>
</tr>
</thead>
</table>
| - Problem: Front light doesn't work  
- Car: Audi A8  
- Year: 1995  
- Battery voltage: 12.0 V  
- State of lights: surface damaged  
- State of light switch: OK | - Diagnosis: Bulb defect  
- Repair: Replace front light |
Similarity

- Similarity is assessed for each feature
- Depends on the feature value
- Features can have different weights (importance)
  - Feature: Problem
    - Front light doesn't work: 0.8
    - Break light doesn't work: 0.4
    - Engine doesn't start: 0.6
  - Feature: Battery voltage (similarity depends on the difference)
    - 12.6 V: 0.9
    - 13.6 V: 1.0
    - 12.6 V: 0.5
    - 6.7 V: 0.3

Different features have different importance

- High importance:
  - Problem: Battery_voltage: State_of_light:
- Low importance:
  - Make: Model: Year: Colour:

Compare new problem with case 1

Problem(Symptoms)
- Prob. Front light doesn't work: 2.0
- Car: Audi S8
- Year: 1996
- Battery voltage: 12.8 V
- State of lights: OK

Problem(Symptoms)
- Prob. Front light doesn't work: 1.0
- Car: VW Golf III: 1.8 L
- Year: 1995
- Battery voltage: 12.8 V
- State of lights: OK
- State of light switch: OK

Solution
- Diagnosis: Front light fuse defect
- Repair: Replace front light fuse

Similarity Computation by Weighted Average
\[
similarity(weight, case 1) = \frac{1}{2 \times 3} \left( 0.8 \times 1.0 + 0.4 \times 1.0 + 0.6 \times 0.9 + 0.9 \times 6.0 + 0.1 \times 1.3 \right) = 0.86
\]
Compare new problem with case 2

Problem (Symptoms):
- Problem: Front light doesn't work
- Car: Audi 80
- Year: 1989
- Battery voltage: 12.8 V
- State of lights: OK

Problem (Symptoms):
- Problem: Front light doesn't work
- Car: Audi A6
- Year: 1995
- Battery voltage: 13.9 V
- State of lights: Surface damaged
- State of light switch: OK

Solution:
- Diagnosis: Bulb defect
- Repair: Replace front light

Similarity Computation by Weighted Average
\[ \text{similarity(new, case 2)} = 1.00 \times (0.8 + 1.0 + 1.8 + 1.9 + 0.95 + 0.0) = 0.585 \]

Reuse the Solution of Case 1

CASE 1

Problem (Symptoms):
- Problem: Front light doesn't work

Solution:
- Diagnosis: Frontlight fuse defect
- Repair: Replace front light fuse

Adapt Solution:

How do differences in the problem affect the solution?

- New Solution:
  - Diagnosis: Break light fuse defect
  - Repair: Replace break light fuse

Store the New Experience

If diagnosis is correct:
store new case in the memory.
(retain)