CASE-BASED REASONING

- A cognitive approach for modelling human behaviour
- An engineering approach for developing and implementing intelligent systems for problem solving

"Similar problems have similar solutions"

- Using situation specific experience knowledge

Case base ← Already solved problems in

Description of past problem
↓
Description of found solution

Case

1. Retrieve past experience
    - Object oriented structure
    - Retrieval algorithms
    - Similarity measures
2. Reusing experience adapted to current problem
    - Adaption - rules, operators
    - Generalized cases
3. Revise solution
    - Mostly done manually
    - Corectness/quality
4. Retain new experience
    - Organize case base
    - Hill climbing
    - Genetic algorithms

⟶ If correct, store in case base

Homogeneous cases

→ All cases have the same structure

Heterogenous cases

→ Structure varies

Episodic cases

→ Records of real situations

Prototypical cases

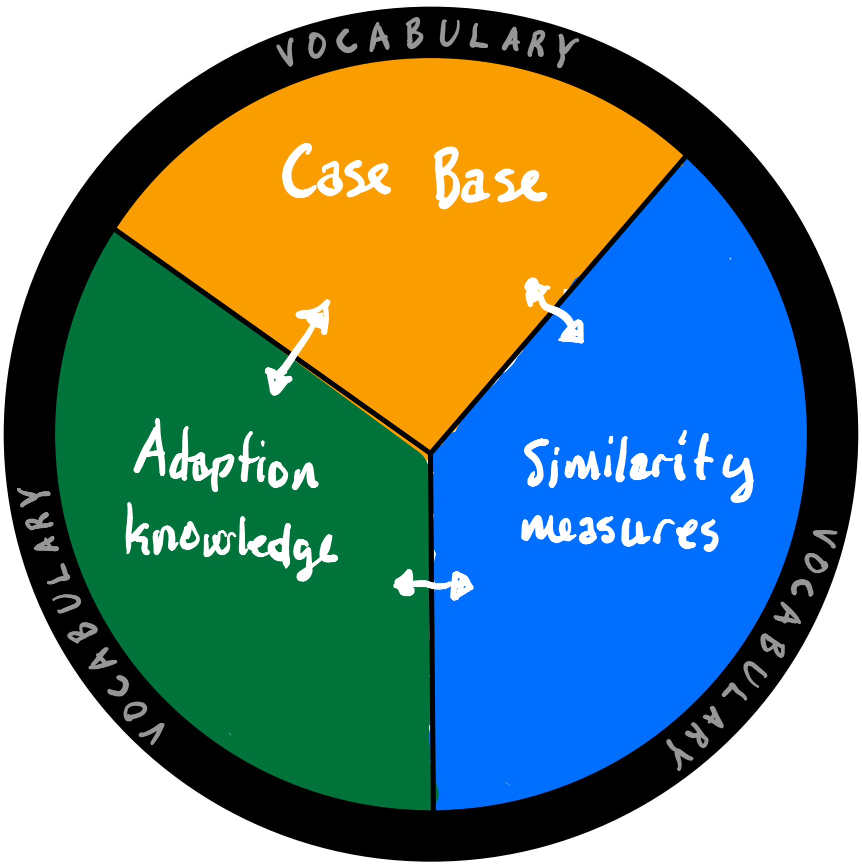
→ Examples of typical events

Solution

Components

- Solution itself (class/plan/description)
- Solution procedure (sequence)
- Information
- Justifications
- Alternative solution steps
- Failed solution steps

General knowledge



SIMILARITY

- Similarity is used as a heuristic for estimating an unknown utility function

Hamming distance:

positions at which corresponding symbols are different

$$H(x,y) = n - \sum_{i=1}^{n} x_i \cdot y_i - \sum_{i=1}^{n} (1-x_i)(1-y_i)$$

$$= |\{ i \mid x_i \neq y_i \}|$$

$$H(x,x) = 0$$

$$H(x,y) = H(y,x)$$ Symmetry

$$H(x,y) = \frac{1}{n} |\{ i \mid x_i \neq y_i \}|$$ Normalized

$$sim_H(x,y) = 1 - \frac{H(x,y)}{n} = \frac{|\{ i \mid x_i \neq y_i \}|}{n}$$

Numeric attributes

Manhattan distance:

$$dist_{1,1}(x,y) = \sum_{i=1}^{n} |x_i - y_i|$$

Euclidian distance:

$$dist_e(x,y) = \sqrt{\sum_{i=1}^{n} x_i^2 + y_i^2}$$

Maximum norm:

$$dist_{norm}(x,y) = \max_{i=1}^{n} |x_i - y_i|$$

Amalgamation function

$$F : [0,1]^n \rightarrow [0,1]$$

$$F(0, \ldots, 0) = 0$$

$$F(1, \ldots, 1) = 1$$

Weighted average

Uniform distribution

$$F(s_1, \dots, s_n) = \sum_{i=1}^{n} w_i \cdot s_i \,, \quad \sum_{i=1}^{n} w_i = 1 \,, \quad s_i = sim_i(q_i, d_i)$$

$\longrightarrow$ Generalization

$$F(s_1, \dots, s_n) = \sqrt[\alpha]{\sum_{i=1}^{n} w_i \cdot (s_i)^{\alpha}} \,, \quad \alpha \in \mathbb{R}^{+}, \ \sum_{i=1}^{n} w_i = 1$$

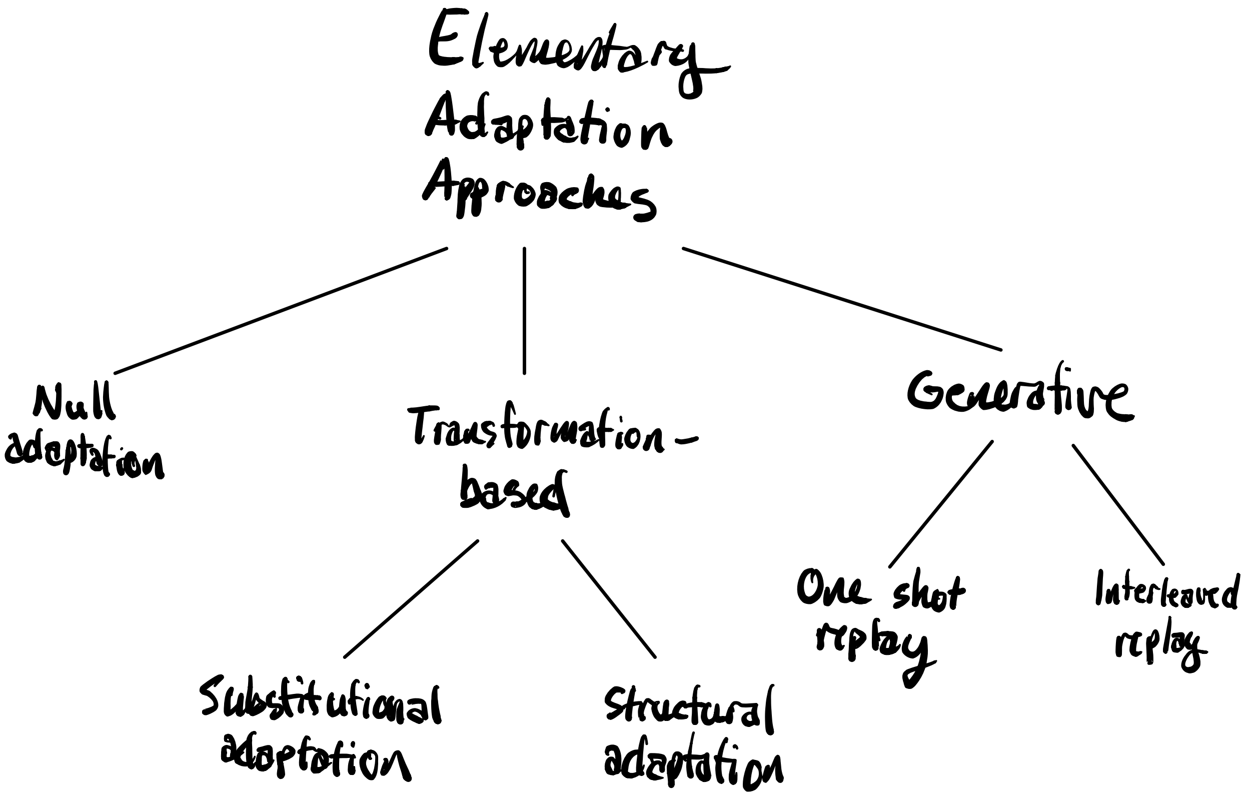
Maximum $F(s_1, \dots, s_n) = \max\{s_1, \dots, s_n\}$

Minimum $F(s_1, \dots, s_n) = \min\{s_1, \dots, s_n\}$

k-Maximum $F(s_1, \dots, s_n) = s_{i_k}, \ s_{i_1} \geq s_{i_2} \geq \dots \geq s_{i_n}$

k-Minimum $F(s_1, \dots, s_n) = s_{i_k}, \ s_{i_1} \leq s_{i_2} \leq \dots \leq s_{i_n}$

ADAPTATION



- Compositional adaptation
- Hierarchical adaptation

Derivational Analogy

→ Case = problem + solution + inference

→ Transfer inference to new problem

→ Fill in gaps with generative solver

Goal

- Increase efficiency of generative solver
- Increase solution quality