Course. Introduction to Machine Learning
Lecture 7. Lazy learning

Dr. Maria Salamó Llorente

Dept. Mathematics and Informatics,

Faculty of Mathematics and Informatics

University of Barcelona (UB)

Introduction to Machine Learning

- Unsupervised Learning
  - Cluster Analysis
    - K-Means, Fuzzy C-means, EM
  - Factor Analysis
    - PCA, ICA
  - Visualization
    - Self Organized Maps (SOM) , Multi-Dimensional Scaling
- Supervised Learning
  - Non Linear Decision
    - Lazy Learning (K-NN, IBL, CBR)
    - Overfitting, model selection and feature selection
    - Kernel Learning
    - Ensemble Learning (Trees, Adaboost)
  - Linear Decision
    - Perceptron, SVM
- Decision Learning Theory
  - Basic concepts of Decision Learning Theory
    - Bias/Variance, VC dimension, Practical advice of how to use learning algorithms

Contents

Introduction to Lazy Learning

Introduction to lazy learning

- Lazy learning delays generalization until a query is made to the system
- Advantages:
  - The target function is approximated locally
  - It can simultaneously solve multiple problems
  - It deals successfully with changes in the problem domain
  - Suitable for complex and incomplete problem domains
- Disadvantages:
  - It requires a large space to store the entire training
  - It may be slow for solving a problem but it has a fast training

Lazy learning approaches

In this lesson we will learn about:

- Nearest Neighbour
- Instance-based learning
- Case-based reasoning

{case, exemplar, instance, memory}
-based-
{learning, reasoning}

Nearest Neighbor

(NN)

Nearest Neighbour classifier

Basic idea:

1. Get some example set of cases with known outputs
   – E.g. Diagnoses of infectuous diseases by experts
2. When you see a new case, assign its output to be the same as the most similar known case
   – Your symptoms most resemble Mr. X
   – Mr. X had a flu
   – Ergo you have the flu

General learning task

- There is a set of possible examples, $X = \{x_i\}$
- Each example is an n-tuple of attribute values, $\vec{x}_1 = \langle a_1, \ldots, a_k \rangle$
- There is a target function that maps X onto set Y, $f: X \rightarrow Y$
- The data is a set of duples <example, target function values>

$$D = \{< \vec{x}_1, f(\vec{x}_1) >, \ldots < \vec{x}_m, f(\vec{x}_m) >\}$$

- Find a hypothesis h such that

$$\forall \vec{x}, h(\vec{x}) \approx f(\vec{x})$$

Nearest neighbour

- Task: Given some set of training data

$$D = \{< \vec{x}_1, f(\vec{x}_1) >, \ldots < \vec{x}_m, f(\vec{x}_m) >\}$$

and a query point $\vec{x}_q$, predict $f(\vec{x}_q)$

distance function

1. Find the nearest member of the data set to the query

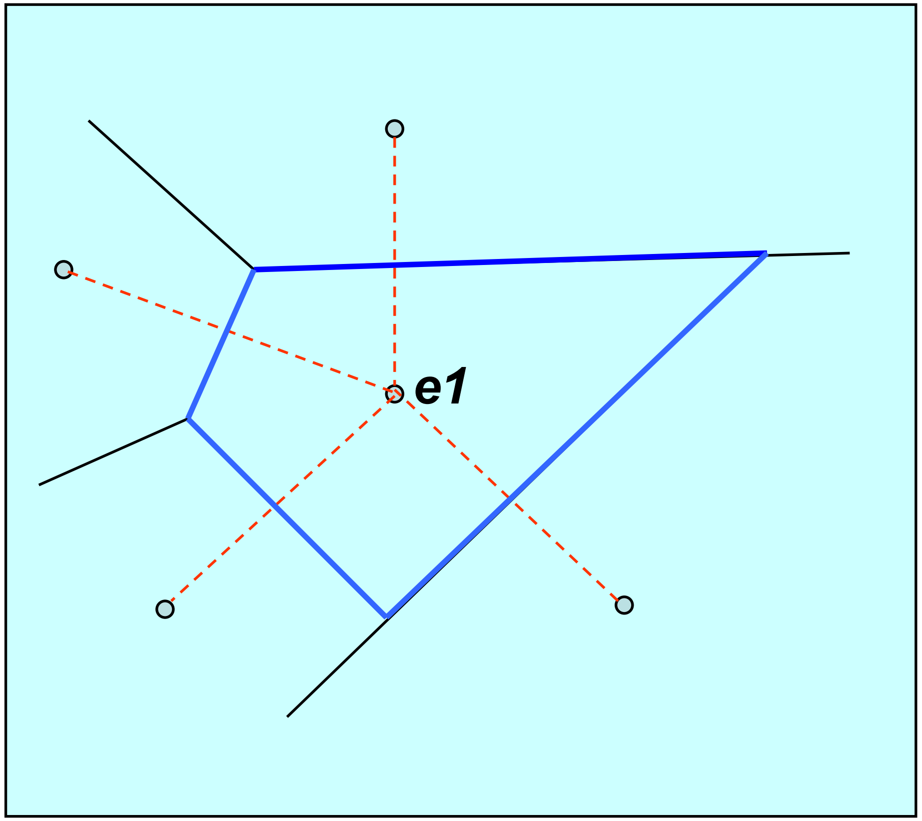
$$\vec{x}_{nn} = \arg\min_{x \in D}(d(\vec{x}, \vec{x}_q))$$

2. Assign the nearest neighbour output to the query

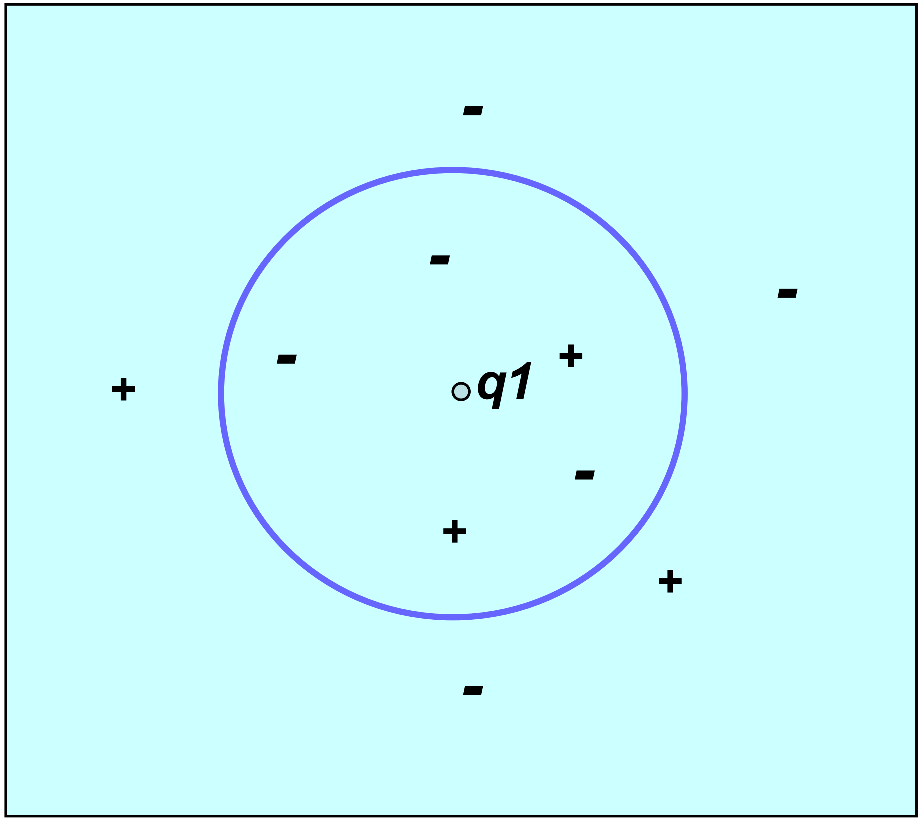
$$h(\vec{x}_q) = f(\vec{x}_{nn})$$

Our hypothesis

Voronoi diagram

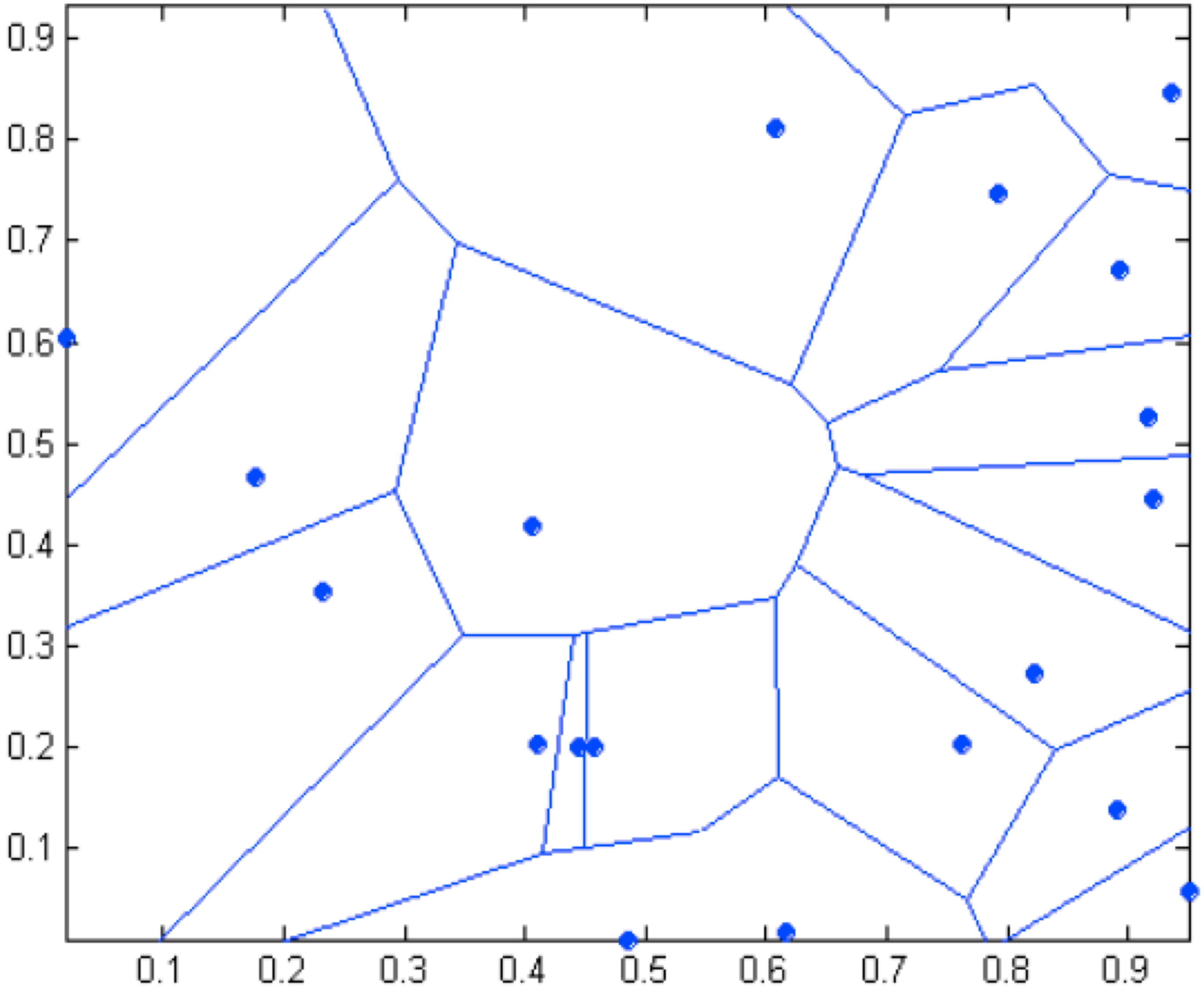


1-nearest neighbor:
the concept represented by e1



5-nearest neighbors:
q1 is classified as negative

Voronoi diagram



Instance-based Learning

(IBL)

Instance-based learning

- A lazy learning algorithm
- One way of solving tasks of approximating **discrete or real valued** target functions
- It produces local approximations to the target function
- Key idea:
  - just store the training examples, D
  - when a test example is given then **find the closest** matches

What makes an instance-based learner?

- A distance measure
  - Nearest neighbour: typically Euclidean
- Number of neighbours to consider
  - Nearest neighbour: one
- A weighting function (optional)
  - Nearest neighbour: unused (equal weights)
- How to fit with neighbours
  - Nearest neighbour: same output as nearest neighbour

K-Nearest Neighbor

(KNN)

K-Nearest Neighbour

- A distance measure
  - typically Euclidean
- Number of neighbours to consider
  - k
- A weighting function (optional)
  - Unused (equal weights)
- How to fit with neighbours
  - Vote using K nearest neighbours (or take average, for regression)

UNIVERSITAT DE BARCELONA

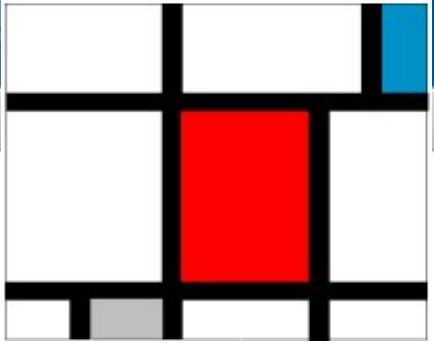
When to consider NN algorithms

- Instances map to points in $\Re^n$
- Not more then say 20 attributes per instance
- Lots of training data

- Advantages:
  - Training is very fast
  - Can learn complex target functions
  - Don't lose information

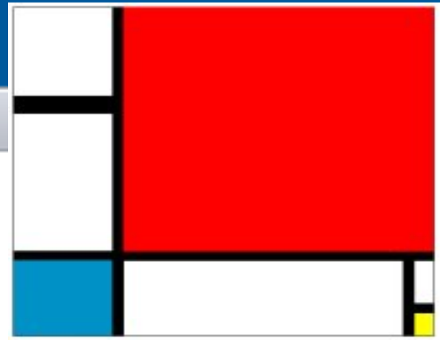
- Disadvantages:
  - ??? (We will see them shortly…)

Mondrian Example



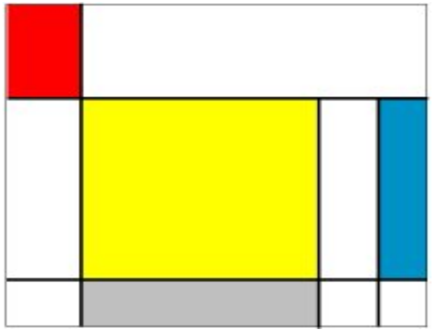
one

two



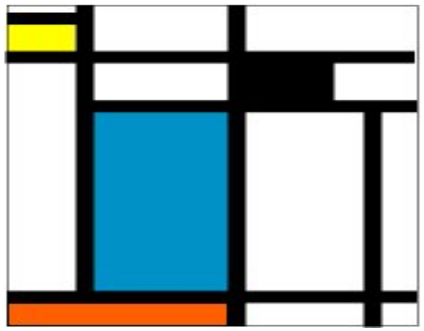
three

four

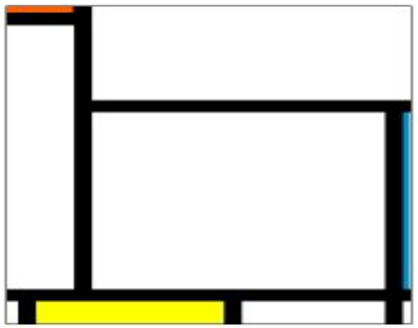


five

six



seven



Eight ?

Training data

Number	Lines	Line types	Rectangles	Colours	Mondrian?
1	6	1	10	4	*No*
2	4	2	8	5	*No*
3	5	2	7	4	Yes
4	5	1	8	4	Yes
5	5	1	10	5	*No*
6	6	1	8	6	Yes
7	7	1	14	5	*No*

Test instance x_q

Number	Lines	Line types	Rectangles	Colours	Mondrian?
8	7	2	9	4	

Keep data in normalised form

One way to normalise the data $a_r(x)$ *to* $a'_r(x)$ *is*

$$x_t' \equiv \frac{x_t - \bar{x}_t}{\sigma_t}$$

$$\bar{x}_r \equiv mean\ \ of\ \ t^{th}\ attributes$$

$$\sigma_t \equiv standard\ \ deviation\ of\ \ t^{th}\ attributes$$

Keep data in normalised form

Another way to normalise the data is rescaling the range of features to the range [0,1] or [-1 ,1]

$$x_t' \equiv \frac{x_t - \min}{\max - \min}$$

$$\min \equiv \min \textit{ of } \ t^{th} \textit{ attributes}$$

$$\max \equiv \max \textit{ of } \ t^{th} \textit{ attributes}$$

UNIVERSITAT DE BARCELONA

Normalised training data

Number	Lines	Line types	Rectangles	Colours	Mondrian?
1	0.632	-0.632	0.327	-1.021	*No*
2	-1.581	1.581	-0.588	0.408	*No*
3	-0.474	1.581	-1.046	-1.021	Yes
4	-0.474	-0.632	-0.588	-1.021	Yes
5	-0.474	-0.632	0.327	0.408	*No*
6	0.632	-0.632	-0.588	1.837	Yes
7	1.739	-0.632	2.157	0.408	*No*

Test instance (x_q)

Number	Lines	Line types	Rectangles	Colours	Mondrian?
8	1.739	1.581	-0.131	-1.021	

Computing distances

Example	Distance of test instance to the example	Mondrian?
1	2.517	*No*
2	3.644	*No*
3	2.395	Yes
4	3.164	Yes
5	3.472	*No*
6	3.808	Yes
7	3.490	*No*

Classification

1-NN	*Yes*
3-NN	*Yes*
5-NN	*No*
7-NN	*No*

Distance-Weighted kNN

- We might want to weight nearer neighbours more heavily

$$f(\mathbf{x}_q) := \frac{\sum_{i=1}^{k} w_i f(\mathbf{x}_i)}{\sum_{i=1}^{k} w_i} \quad \text{where } w_i = \frac{1}{d(\mathbf{x}_q, \mathbf{x}_i)^2}$$

- Then it makes sense to use *all training examples* instead of just k (Stepard's method)

Pros and Cons

- Advantages
  - Fast training (it's a lazy algorithm)
  - Learn complex functions easily
  - Do not lose information
- Disadvantages
  - Slow at query time
  - Needs lots of storage
  - Easily fooled by irrelevant attributes
    - Use statistical methods to remove irrelevant ones (e.g., PCA or cross-validation)

Irrelevant attributes

- Nearest neighbour easily misled when X high-dim
- Low-dimensional intuitions do not extend to high dim
- **The curse of dimensionality**
  - When dimensionality increases, the volume of the space increases so fast that the available data becomes sparse.
  - The number of features is too large relative to the number of training samples

This makes a poor generalization ability of the classifier.

Case-Based Reasoning

(CBR)

Introduction to CBR

- *In this course we focus on the basic principle rather than on specific applications or tools*

- Briefly
  - CBR is an advanced instance-based learning applied to more complex instance objects
  - Objects may include complex structural descriptions of cases and adaptation rules
  - The power comes from the organisation and content of the cases themselves

Faced this situation before?

- Oops the car stopped.
  - What could have gone wrong?
- Aah.. Last time it happened, there was no petrol.
  - Is there petrol?
    - Yes.
  - Oh but wait I remember the tyre was punctured (ban bocor)

This is the normal thought process of a human when faced with a problem which is similar to a problem he/she had faced before.

How do we solve problems?

- By knowing the steps to apply
  – from symptoms to a plausible diagnosis
- But not always applying causal knowledge
  – diseases cause symptoms
  – symptoms do not cause diseases!
- How does an expert solve problems?
  – uses same “book learning” as a novice
  – but quickly selects the right knowledge to apply
- Heuristic knowledge (“rules of thumb”)
  – “I don’t know why this works but it does and so I’ll use it again!”
  – difficult to elicit

Another way we solve problems?

- By remembering how we solved a similar problem in the past
- This is Case Based Reasoning (CBR)!
  - memory-based problem-solving
  - re-using past experiences
- Experts often find it easier to relate stories about past cases than to formulate rules

UNIVERSITAT DE BARCELONA

Problems we solve this way

- Medicine
  - doctor remembers previous patients especially for rare combinations of symptoms
- Law
  - English/US law depends on precedence
  - case histories are consulted
- Management
  - decisions are often based on past rulings
- Financial
  - performance is predicted by past results

What is Case-based Reasoning

- Case-based reasoning is […] reasoning by remembering – *[Leake, 1996]*
- A case-based reasoner solves new problems by adapting solutions that were used to solve old problems - *[Riesbeck & Schank, 1989]*
- Case-based reasoning is both […] the ways people use cases to solve problems and the ways we can make machines use them – *[Kolodner, 1993]*
- Case-based reasoning is a recent approach to problem solving and learning […] - *[Aamodt & Plaza, 1994]*

Case-based reasoning is ...

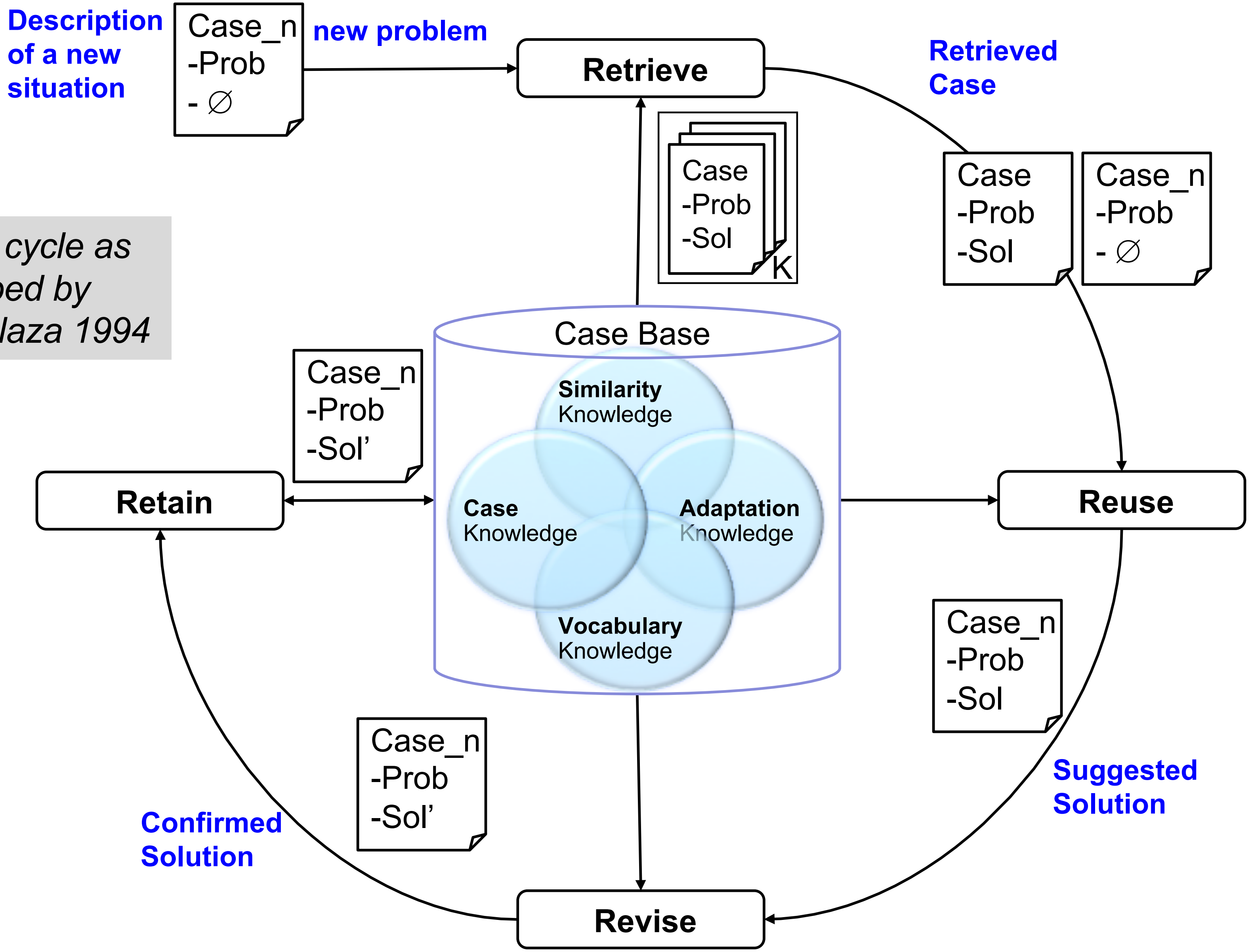
- A methodology to model human reasoning and thinking
- A methodology for building intelligent computer systems
- CBR in a nutshell:
  - Store previous experience (cases) in memory
  - To solve new problems:
    - Retrieve similar experience about similar situations from the memory
    - Reuse the experience in the context of the new situation : complete or partial reuse, or adapt according to differences
    - Store new experience in memory (**learning**)

A bit of history

- Roots of CBR is found in the works of Roger Shank on dynamic memory in 1983
- Other trails into the CBR field has come from
  - Analogical reasoning in 1990
  - Problem solving and experimental learning within philosophy and psychology
- The first CBR system, CYRUS developed by Janet Kolodner at Yale university (1983)

CBR cycle

The CBR cycle as described by Aamodt&Plaza 1994

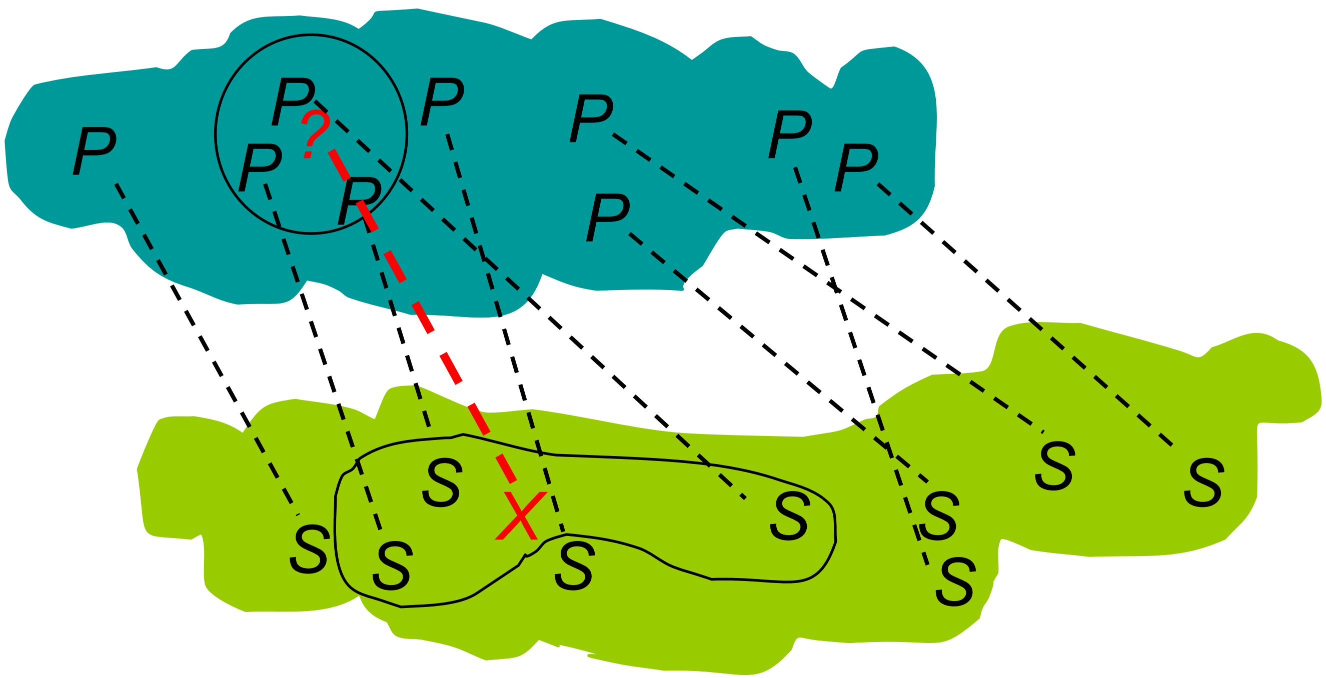


CBR System Components

- Case-base
  - database of previous cases (experience)
- Retrieval of relevant cases
  - index for cases in library
  - matching most similar case(s)
  - retrieving the solution(s) from these case(s)
- Adaptation of solution
  - alter the retrieved solution(s) to reflect differences between new case and retrieved case(s)

Main CBR Assumption

- New problem can be solved by
  - retrieving similar problems
  - adapting retrieved solutions
- Similar problems have similar solutions



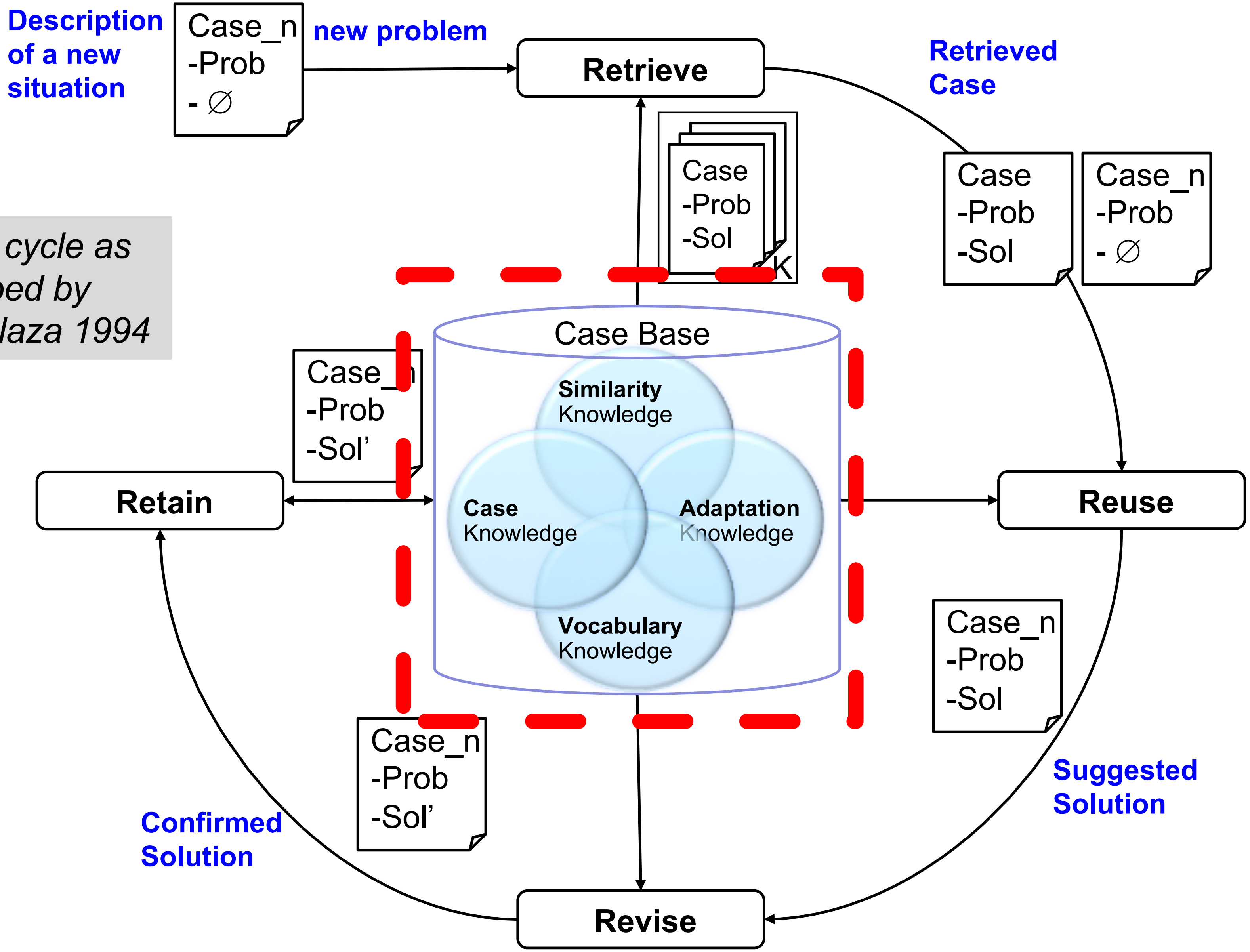
CBR Assumption(s)

- The **main assumption** is that:
  - *Similar problems have similar solutions:*
    - e.g., an aspirin can be taken for any mild pain
- More assumptions:
  - The world is a regular place: what holds true today will probably hold true tomorrow
    - (e.g., if you have a headache, you take aspirin, because it has always helped)
  - Situations repeat: if they do not, there is no point in remembering them
    - (e.g., it helps to remember how you found a parking space near that restaurant)

Two big tasks of CBR

- Classification tasks (good for CBR)
  - Diagnosis - what type of fault is this?
  - Prediction / estimation - what happened when we saw this pattern before?
- Synthesis tasks (harder for CBR)
  - Engineering Design
  - Planning
  - Scheduling

CBR cycle

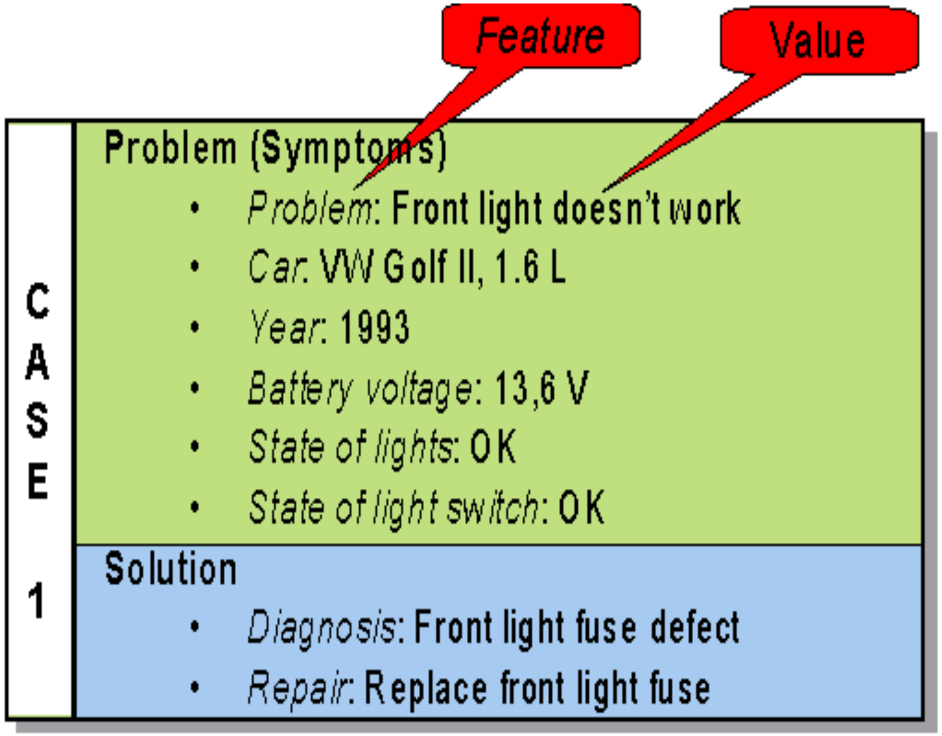


The CBR cycle as described by Aamodt&Plaza 1994

A simple example: What's a case?

- A case describes one particular diagnostic situation
- A case records several features and their specific values occurred in that situation
- A case is not a rule !!!

Technical Diagnosis of Car Faults



Figures of this example come from:

R. Bergmann, University of Kaiserslautern

Feature

Value

C A S E 1

Problem (Symptoms)

- *Problem*: Front light doesn't work
- *Car*: VW Golf II, 1.6 L
- *Year*: 1993
- *Battery voltage*: 13,6 V
- *State of lights*: OK
- *State of light switch*: OK

Solution

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

C A S E 2

Problem (Symptoms)

- Problem: Front light doesn't work
- Car: Audi A6
- Year: 1995
- Battery voltage : 12,9 V
- State of lights: surface damaged
- State of light switch: OK

Solution

- Diagnosis: Bulb defect
- Repair: Replace front light

A case base with two cases:

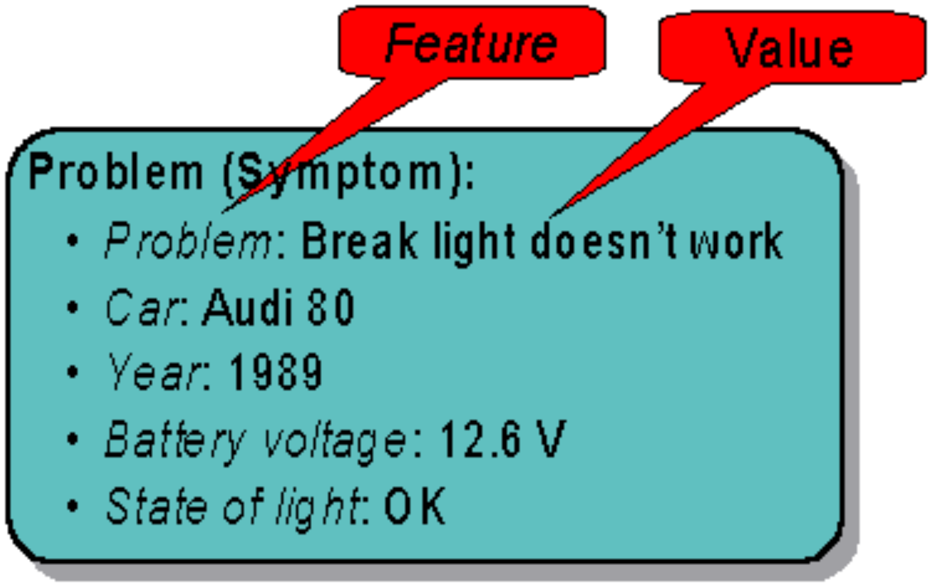
- Each case describes a particular situation
- All cases are independent from each other

Case Representation

- Flat feature-value list
  - Simple case structure is sometimes sufficient for problem solving
  - Easy to store and retrieve in a CBR system
- Object Oriented representation
  - Case: collection of objects (instances of classes) in the sense of OO
  - Required for complex and structured objects
- For special tasks:
  - Graph representations: case = set of nodes and arcs
  - Plans: case = (partially) ordered set of actions
  - Predicate logic: case = set of atomic formulas
- The choice of representation is
  - Dependent on requirements of domain and task
  - Structure of already available case data

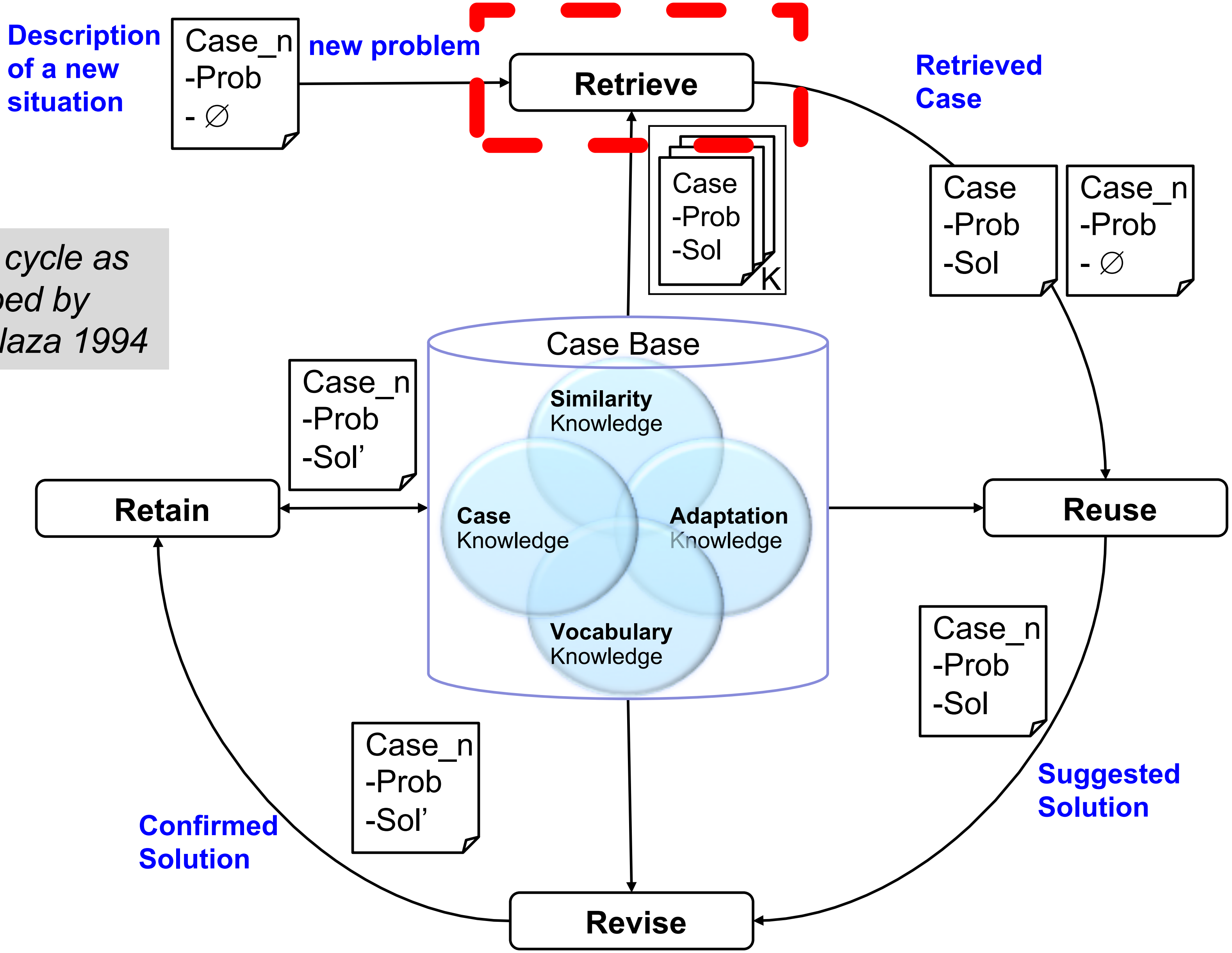
Solving a new diagnostic problem

- A new problem must be solved
- We make several observations in the current situation
- Observations define a new problem
- Not all feature values must be known
- Note: The new problem is a case without solution part



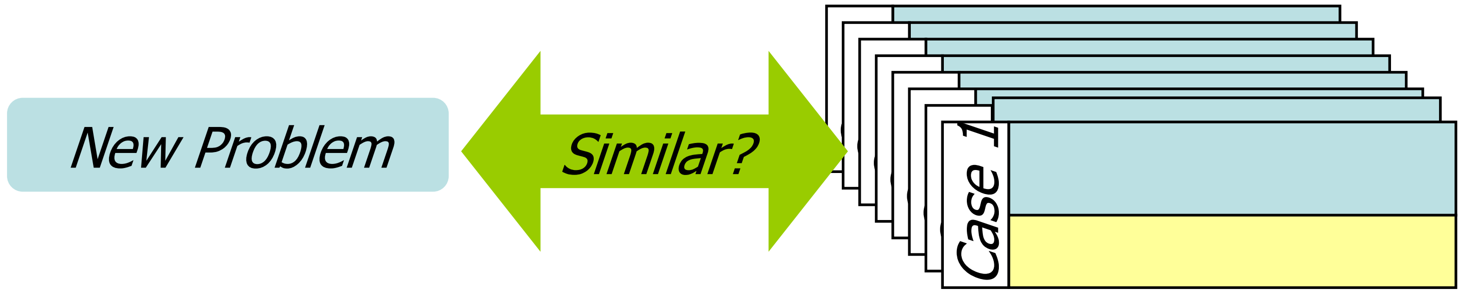
CBR cycle

The CBR cycle as described by Aamodt&Plaza 1994



Retrieving A Car Diagnosis Case

- Compare new problem to each case
- Select most similar



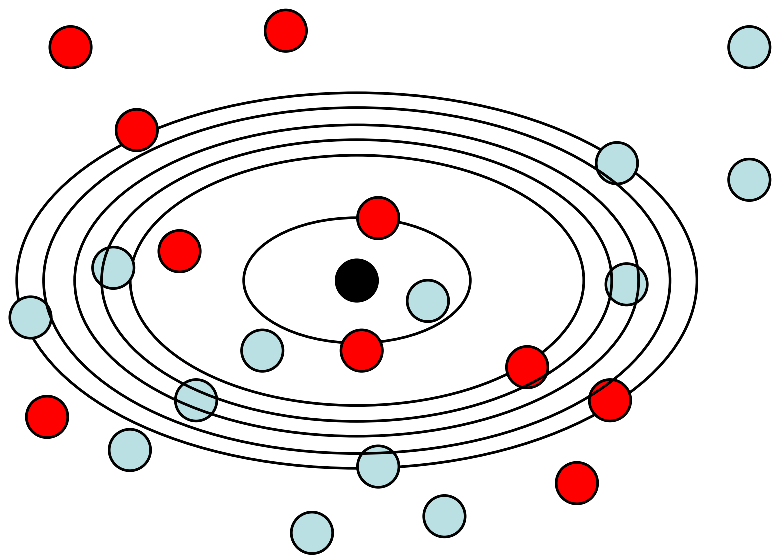
- Similarity is most important concept in CBR
  – When are two cases similar?
  – How are cases ranked according to similarity?
- Similarity of cases
  – Similarity for each feature
    - Depends on feature values
    - BUT: Importance of different feature values may be different

Retrieve: What is similarity?

- Purpose of similarity:
  - Select cases that can be adapted easily to the current problem
  - Select cases that have (nearly) the same solution than the current problem
- Basic assumption: similar problems have similar solutions
- Degree of similarity = utility /reusability solution
- Goal of similarity modeling: provide a good approximation
  - Close to real reusability
  - Easy to compute

Nearest Neighbour Retrieval

- Retrieve most similar
- k-nearest neighbour
  - k-NN
  - like scoring in bowls or curling
- Example
  - *1-NN*
  - *5-NN*



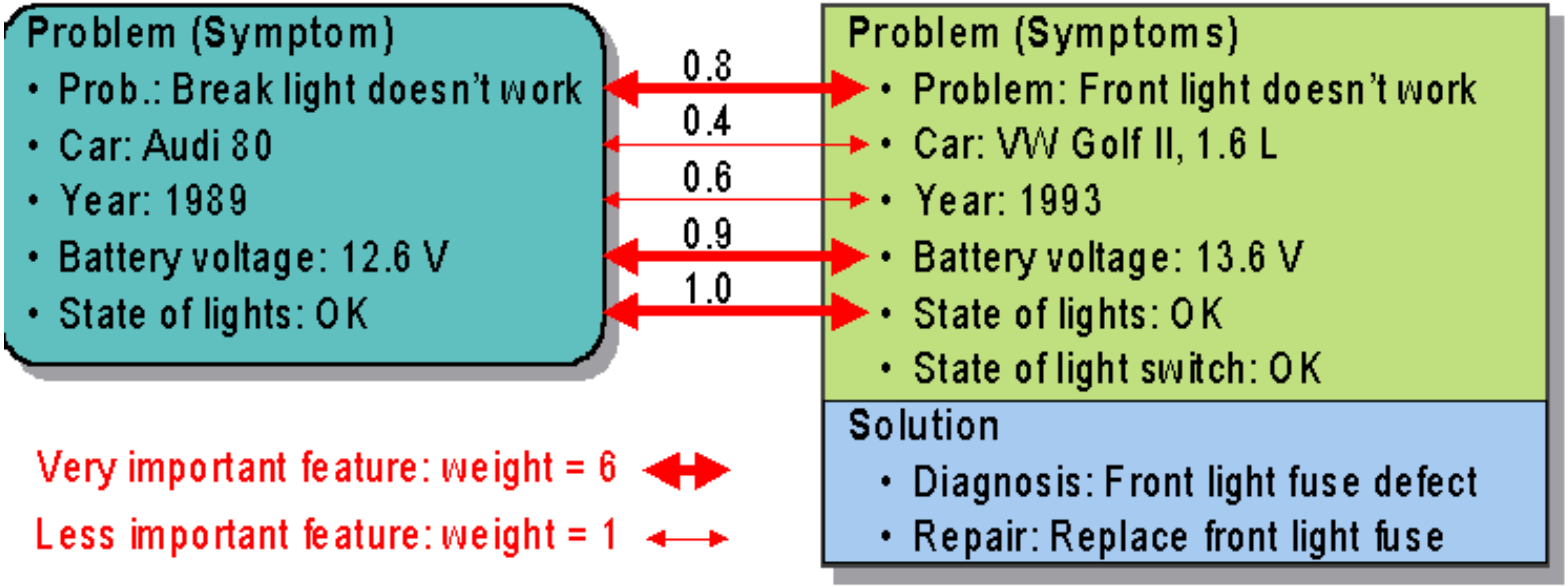
Retrieve: Modeling similarity

- Different approaches depending on case representation
- Similarity measures:
  - Function to compare two cases

    sim: Case x Case ➔[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)
  - (Sub-)Graph isomorphism for graph representations
  - Logical inferences

UNIVERSITAT DE BARCELONA

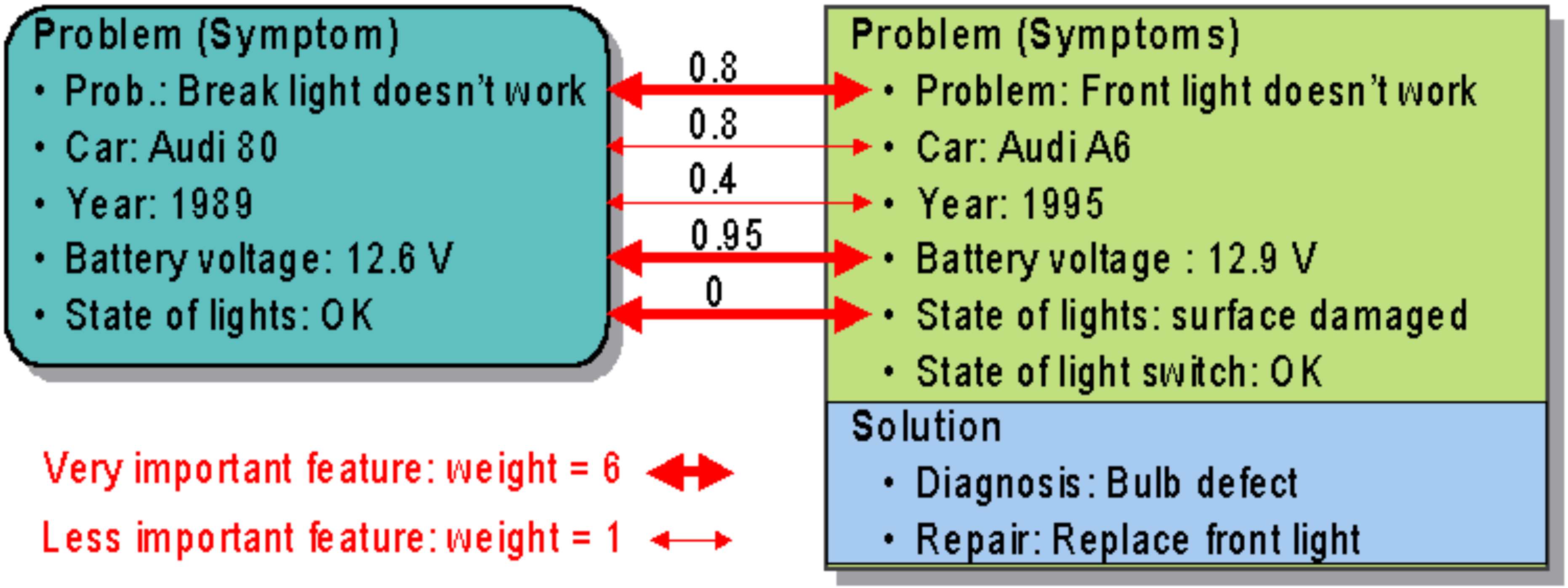
Similarity Computation for case 1



Similarity Computation by Weighted Average

similarity(new,case 1) = 1/20 * [6*0.8 + 1*0.4 + 1*0.6 + 6*0.9 + 6* 1.0] = **0.86**

Similarity Computation for case 2



Similarity Computation by Weighted Average

similarity(new,case 2) = 1/20 * [6*0.8 + 1*0.8 + 1*0.4 + 6*0.95 + 6*0] = **0.585**

Retrieve, but efficiently ...

- Efficient case retrieval is essential for large case bases
- Different approaches depending
  - On the case representation
  - Size of the case base
- Organisation of the case base:
  - Linear lists, only for small case bases
  - Index structures for large case bases
    - Kd-trees, Retrieval nets, discrimination nets, ...
- How to store cases:
  - Databases: for large case bases or if shared with other applications
  - Main memory: for small case bases, not shared

Reuse Solution from Case 1

New Problem

- Symptom: brakelight does not work
- Car: Ford Fiesta
- Year: 1997
- Battery: 9.2v
- Headlights: undamaged
- HeadlightSwitch: ?

Case 1

Problem

- Symptom: headlight does not work
- …

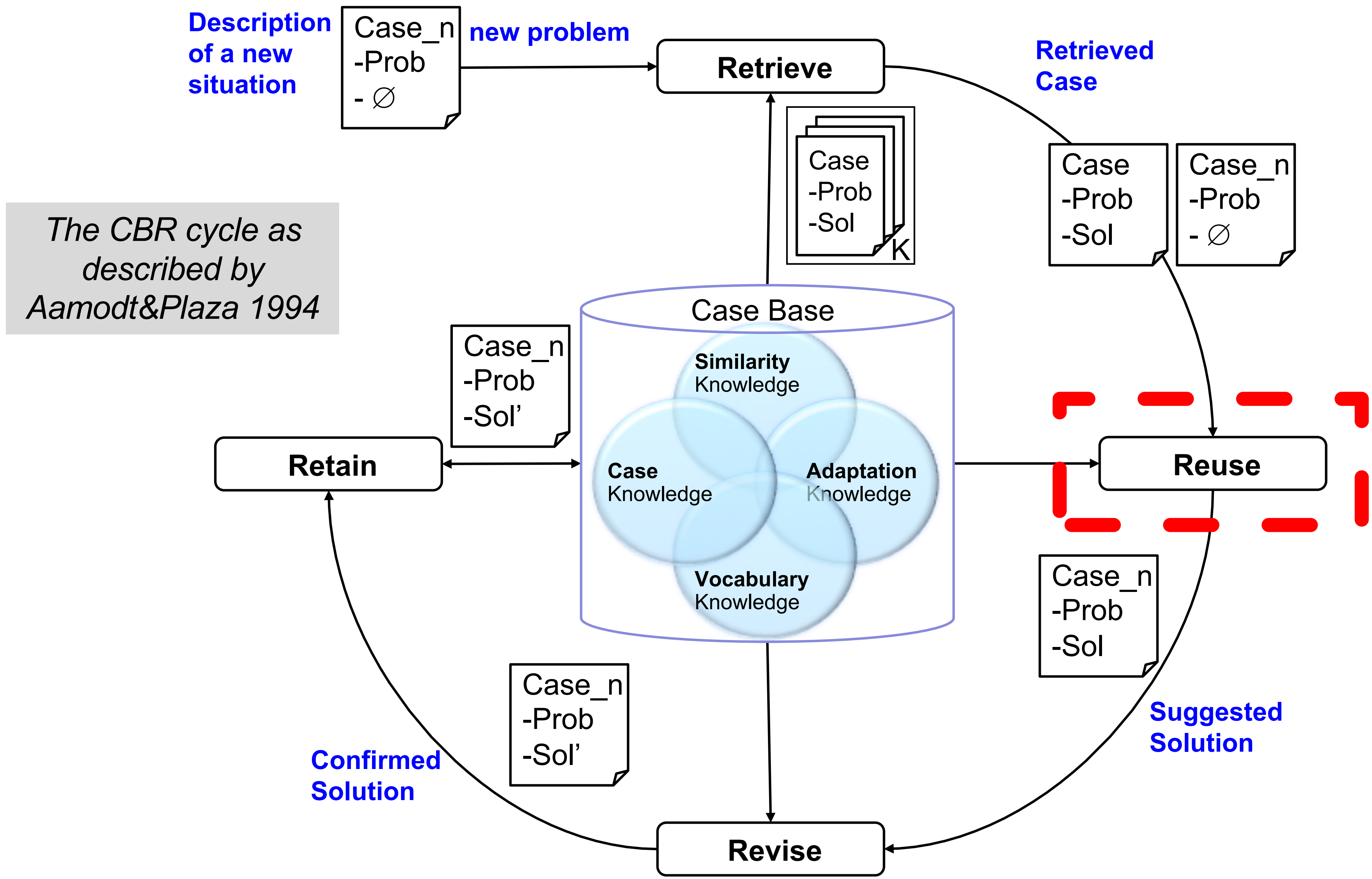
Solution

- Diagnosis: headlight fuse blown
- Repair: replace headlight fuse

– Solution to New Problem
  - Diagnosis: headlight fuse blown
  - Repair: replace headlight fuse

– After Adaptation
  - Diagnosis: brakelight fuse blown
  - Repair: replace brakelight fuse

CBR cycle


Description of a new situation
Case_n
-Prob
- ∅
new problem
Retrieve
Retrieved Case
Case
-Prob
-Sol
K
Case
-Prob
-Sol
Case_n
-Prob
- ∅
The CBR cycle as described by Aamodt&Plaza 1994
Case Base
Similarity Knowledge
Case Knowledge
Adaptation Knowledge
Vocabulary Knowledge
Case_n
-Prob
-Sol'
Retain
Reuse
Case_n
-Prob
-Sol
Suggested Solution
Case_n
-Prob
-Sol'
Confirmed Solution
Revise

Re-Using Retrieved Solutions

- Single retrieved solution
  - Re-use this solution
- Multiple retrieved solutions
  - Vote/average of retrieved solutions
    - Weighted according to
      - Ranking
      - Similarity
- Iterative retrieval
  - Solve components of the solution one at a time

How to Adapt the Solution

- Adaptation alters proposed solution:
- Null adaptation - copy retrieved solution
  - Used by CBR-Lite systems
- Manual or interactive adaptation
  - User adapts the retrieved solution (Adapting is easier than solving?)
- Automated adaptation
  - CBR system is able to adapt the retrieved solution
  - Adaptation knowledge required

Automated Adaptation Methods

- *Substitution*
  - change some part(s) of the retrieved solution
  - simplest and most common form of adaptation
- *Transformation*
  - alters the structure of the solution
- *Generative*
  - replays the method of deriving the retrieved solution on the new problem
  - most complex form of adaptation

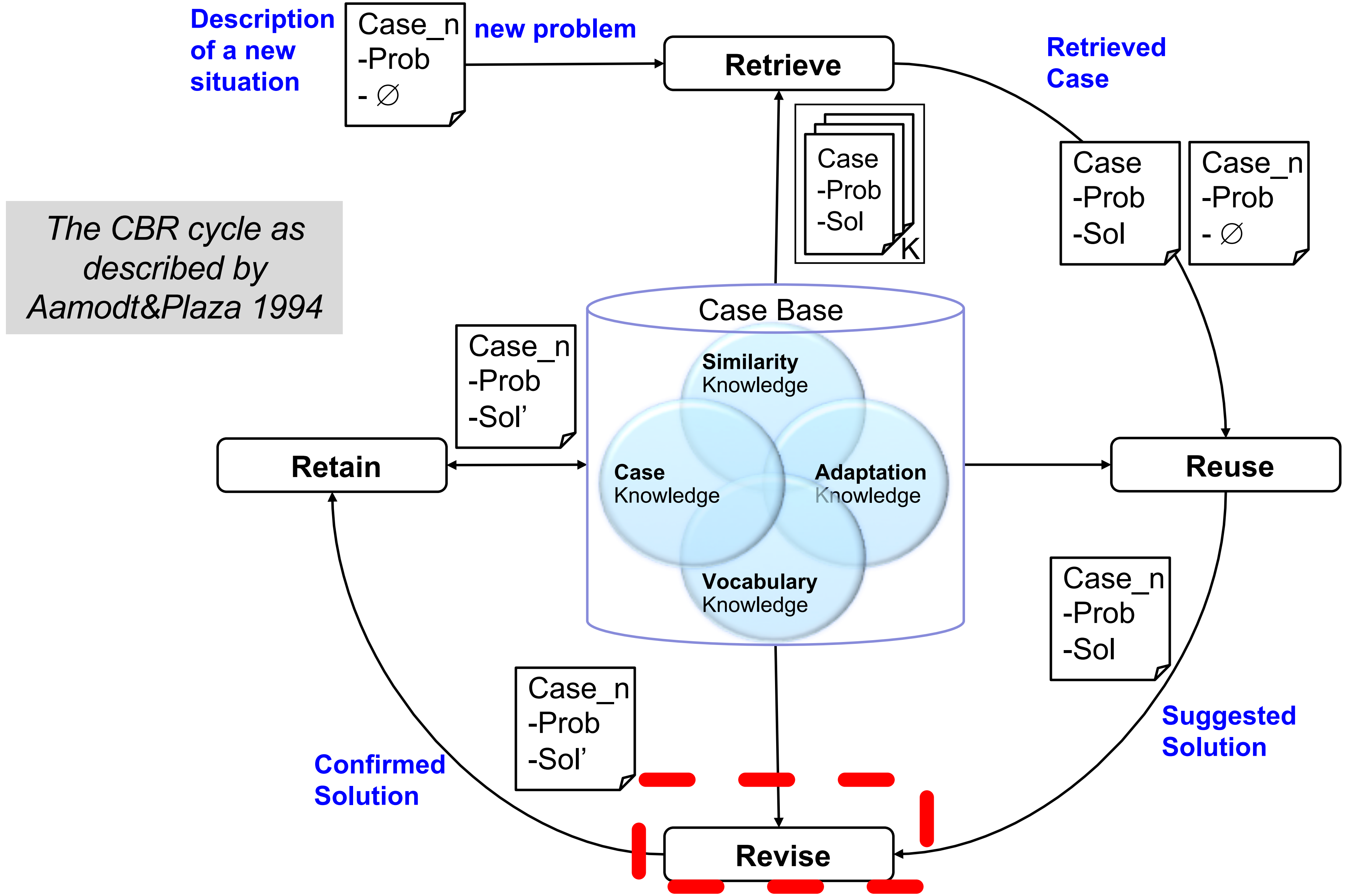
Examples of Adaptation

- CHEF
  - CBR system to plan Szechuan recipes
    - Hammond (1990)
- *Substitution* adaptation
  - substitute ingredients in the retrieved recipe to match the menu
    - Retrieved recipe contains beef and broccoli
    - New menu requires chicken and snowpeas
    - Replace chicken for beef, snowpeas for broccoli
- *Transformation* adaptation
  - Add, change or remove steps in the recipe
    - Skinning step added for chicken, not done for beef

Examples of Adaptation

- Car diagnosis example
  - Symptoms, faults and repairs for brake lights are analogous to those for headlight
  - *Substitution*: brake light fuse
- Planning example
  - Train journeys and flights are analogous
  - *Transformation*: flights need check-in step added

CBR cycle



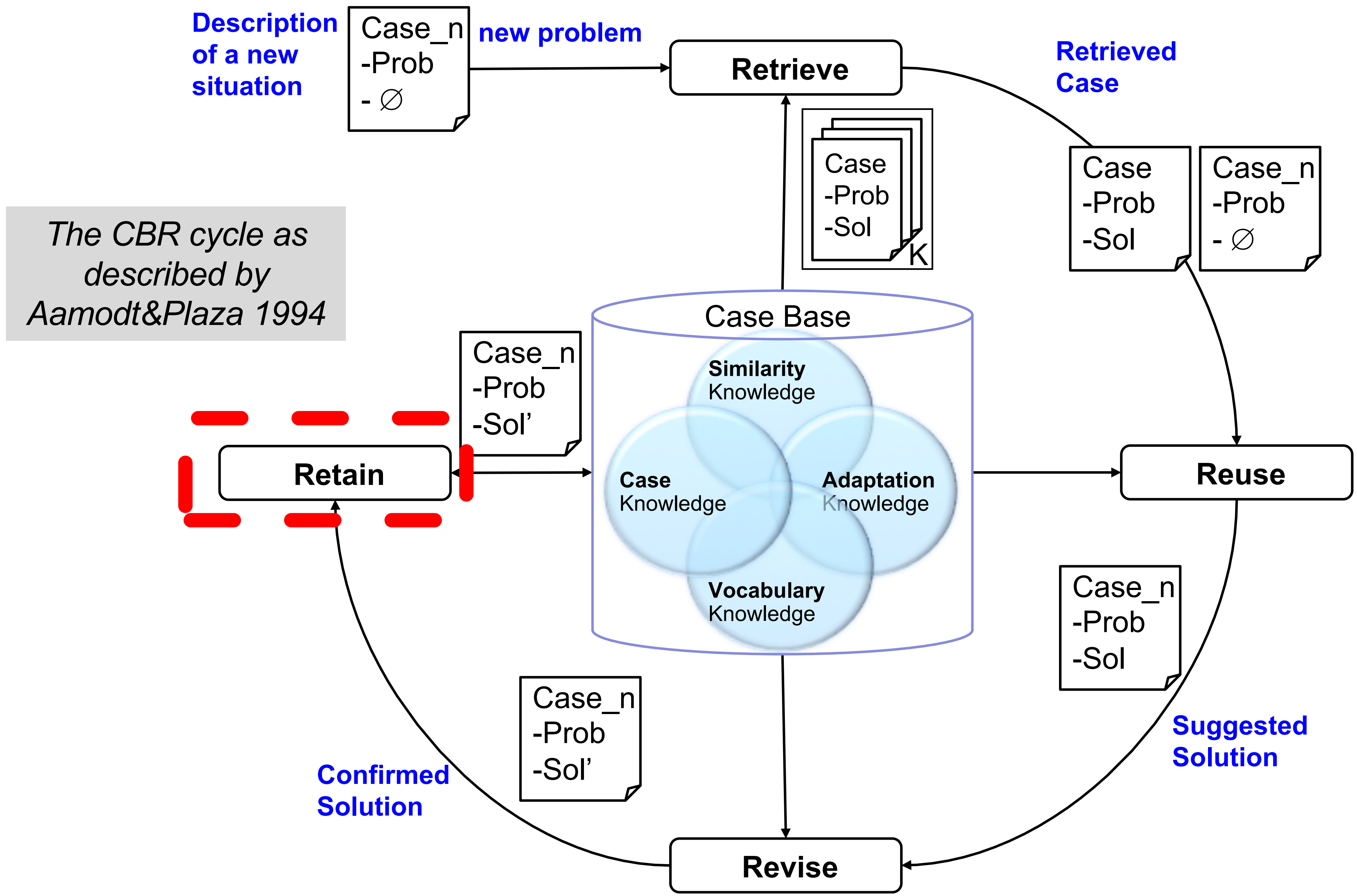
Description of a new situation
Case_n
-Prob
- ∅
new problem
Retrieve
Retrieved Case
Case
-Prob
-Sol
K
Case
-Prob
-Sol
Case_n
-Prob
- ∅
The CBR cycle as described by Aamodt&Plaza 1994
Case_n
-Prob
-Sol'
Case Base
Similarity Knowledge
Case Knowledge
Adaptation Knowledge
Vocabulary Knowledge
Retain
Reuse
Case_n
-Prob
-Sol
Suggested Solution
Case_n
-Prob
-Sol'
Confirmed Solution
Revise

Revise: verify and correct solution

- Revise phase:
  - No revise phase
  - Verification of the solution by computer simulation
  - Verification / evaluation of the solution in the real world
- Criteria for revision
  - Correctness of the solution
  - Quality of the solution
  - Other, e.g., user preferences

CBR cycle

UNIVERSITAT DE BARCELONA



Retain: Learning from problem solving

- What can be learned:
  - New experience to be retained as new case
  - Improved similarity assessment, importance of features
  - Organization / Indexing of the case base to improve efficiency
  - Knowledge for solution adaptation
  - Forgetting cases (learn to forget)
    - For efficiency or because out of date
    - Deleting an old case
      - Old is not necessarily bad
      - Does it leave a gap?

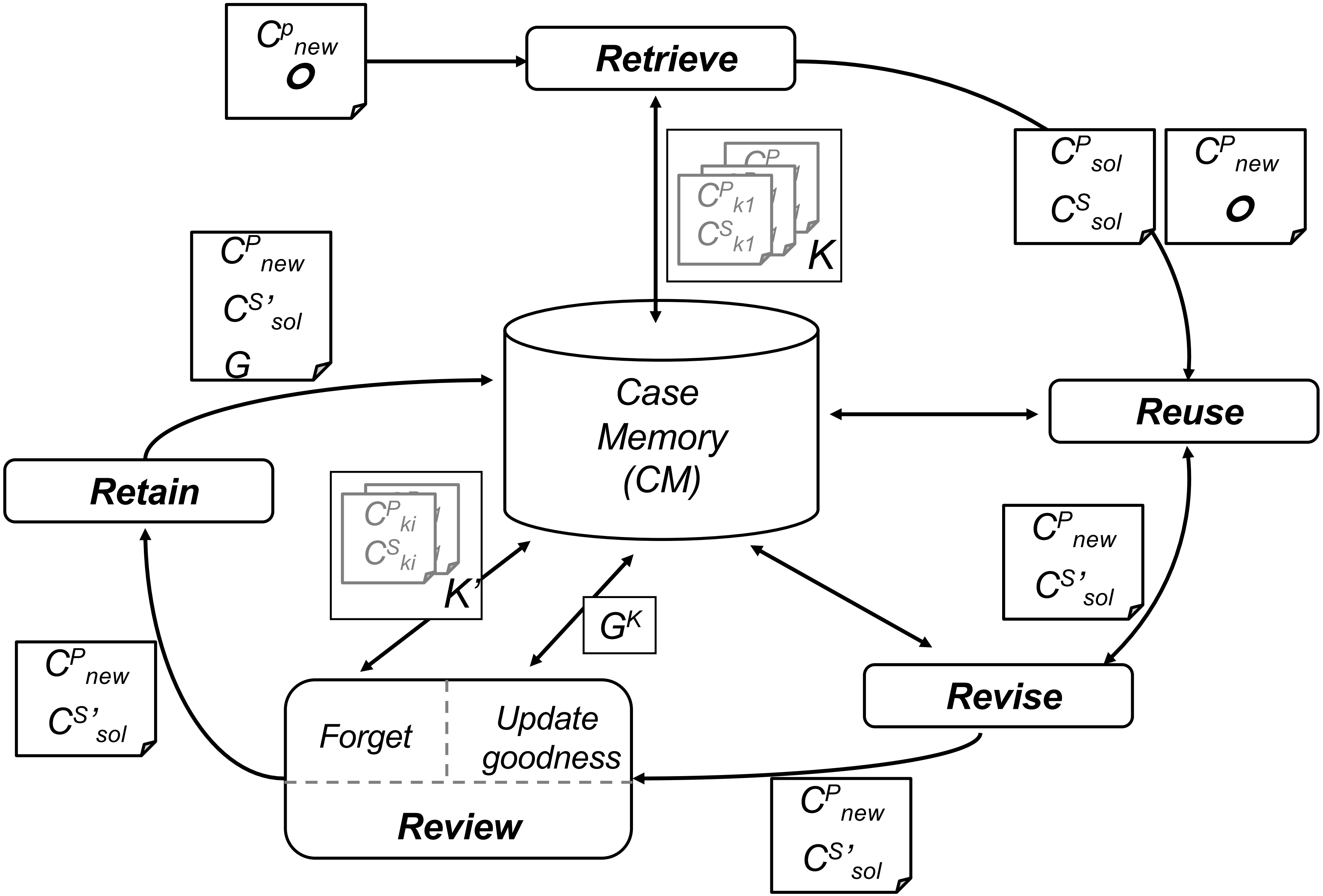
Pros & Cons of CBR

- Advantages
  - solutions are quickly proposed
    - derivation from scratch is avoided
  - domains do not need to be completely understood
  - cases useful for open-ended/ill-defined concepts
  - highlights important features
- Disadvantages
  - old cases may be poor
  - library may be biased
  - most appropriate cases may not be retrieved
  - retrieval/adaptation knowledge still needed

Advantages of CBR over other techniques

- Reduces the knowledge acquisition effort
- Requires less maintenance effort
- Improves problem solving performance through reuse
- Makes use of existing data, e.g., in databases
- Improve over time and adapt to changes in the environment
- High user acceptance

A CBR approach: ACBR Cycle



Summary

- CBR is a technique for solving problems based on experience
- CBR problem solving involves four phases
  - Retrieval, Reuse, Revise, and Retain
- Different techniques for:
  - Representing the knowledge, in particular, the cases
  - Realizing the four phases
- CBR has several advantages over tradition KBS
- Several applications
  - Classification, diagnosis, decission support, planning, configuration, design, ...

Course. Introduction to Machine Learning
Lecture 7. Lazy learning

THE END