Introduction to Case-Based Reasoning

Dr. Ralph Bergmann

bergmann@informatik.uni-kl.de

Centre for Learning Systems and Applications

Research Group - Prof. Michael. M. Richter -

Department of Computer Science

University of Kaiserslautern

Zentrum für Lernende Systeme & Anwendungen

Case-Based Reasoning in 45 Minutes ...

- More an introduction than an overview ...
- Focus on the basic principle rather than on specific applications or tools
- Goal of the talk:
  - Brief history of CBR
  - A simple example
  - Introduction to the common vocabulary
  - Spectrum of the techniques applied
  - Advantages of CBR over alternative methods
  - Application fields

Zentrum für Lernende Systeme & Anwendungen

What is Case-Based Reasoning (CBR)

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 a recent approach to problem solving and learning [...] *Aamodt & Plaza, 1994*

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 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)

From an engineering perspective:
We are mainly interested in building intelligent systems

History of CBR in U.S.A.

Roger Schank, Yale University: Cognitive Science

1977: Scripts for knowledge representation (Schank, Abelson)

1983: Dynamic Memory Theory, Memory Organization Packets
CYRUS: First implemented CBR-System (Kolodner)

1983-1988: Other Systems, e.g. : JUDGE, SWALE, CHEF

Bruce Porter, Austin Texas: Concept Learning

1986-89: System PROTOS (Exemplar-based concept representation)

Edwina Rissland, U. of Massachusetts: Cases in Law (since 1983)

1990-92: Systems HYPO (Ashley) and CABARET (Skalak)

Jaime Carbonell & Manuela Veloso, Carnegie Mellon U.: Analogy

since 1990 Prodigy/Analogy: Case-based Planning using analogy

Interest in CBR is increasing in USA (new research groups)

since 1988 several DARPA and AAAI Workshops

History of CBR in Europe

Michael M. Richter, U. Kaiserslautern, Germany: CBR for Expert Systems

1988-1991 Systems MOLTKE and PATDEX (technical diagnosis)

since 1991 Case-Based Planning: Systems Caplan/CbC, PARIS

since 1992 European Projects INRECA, INRECA-II

Ramon Mantaras, Enric Plaza, IIIA Blanes, Spain: CBR and ML

1990 Case-Based Learning for medical diagnosis

Agnar Aamodt, U. Trondheim, Norway: CBR and Knowledge Acquisition

1991 System CREEK: Integration of Cases and general knowledge

Mark Keane, Trinity College, Dublin: Cognitive Science

since 1988 Theory of analogical reasoning

Since 1991 Increasing interest in Europe (Several new research groups)

1991 First German CBR Workshop (AKCBR, GWCBR)

1993 First European CBR Workshop (EWCBR)

1995 First International CBR Conference (ICCBR)

Case-Based Reasoning Today

- Research on CBR in more than 35 universities and institutes all over the world.
- 15 commercial tools involving CBR
- Many applications already in daily use
- Several regular scientific and application-oriented events: from national workshops to the international conference
- Recent information on the World Wide Web: http://www.cbr-web.org/
- Upcoming Events:
  - 4th European CBR Workshop in Dublin, September 1998
  - 3rd International CBR Conference in Munich, July 1999

Zentrum für Lernende Systeme & Anwendungen

A Simple Example (Overview)

Technical Diagnosis of 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 a 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
- Store a collection of cases in a case base
- Find case similar to current problem and reuse repair strategy

Zentrum für Lernende Systeme & Anwendungen

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 !!

Feature

Value

CASE 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

Zentrum für Lernende Systeme & Anwendungen

A Case Base with Two Cases

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

CASE 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

CASE 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

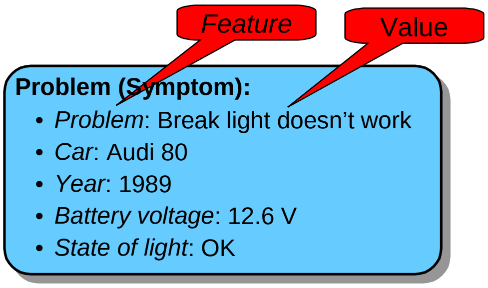
Solution

- Diagnosis: Bulb defect
- Repair: Replace front light

Zentrum für Lernende Systeme & Anwendungen

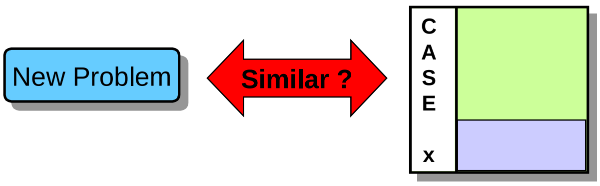
Solving a New Diagnostic Problem

- 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



Zentrum für Lernende Systeme & Anwendungen

Compare the New Problem with Each Case and Select the Most Similar Case



- When are two cases similar?
- How to rank the cases according to their similarity?
- **Similarity is the most important concept in CBR !!**
- We can assess similarity based on the similarity of each feature
- Similarity of each feature depends on the feature **value**.
- BUT: Importance of different features may be different

Zentrum für Lernende Systeme & Anwendungen

Similarity Computation

- Assignment of similarities for features values. **Not similar** **Very similar**
- Express degree of similarity by a real number between 0 and 1

  Examples:
  - Feature: *Problem*

    Front light doesn't work ←0.8→ Break light doesn't work

    Front light doesn't work ←0.4→ Engine doesn't start
  - Feature: *Battery voltage* (similarity depends on the difference)

    12.6 V ←0.9→ 13.6 V

    12.6 V ←0.1→ 6.7 V

- Different features have different importance (weights) !
  - High importance: Problem, Battery voltage, State of light, ...
  - Low importance: Car, Year, ...

Compare New Problem and Case 1

Problem (Symptom)
- Prob.: Break light doesn't work
- Car: Audi 80
- Year: 1989
- Battery voltage: 12.6 V
- State of lights: OK

Similarities: 0.8, 0.4, 0.6, 0.9, 1.0

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

Very important feature: weight = 6 ⟷

Less important feature: weight = 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**

Compare New Problem and Case 2

Problem (Symptom)
- Prob.: Break light doesn't work
- Car: Audi 80
- Year: 1989
- Battery voltage: 12.6 V
- State of lights: OK

Similarity
0.8
0.8
0.4
0.95
0

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

Very important feature: weight = 6

Less important feature: weight = 1

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**

Case 1 is more similar: due to feature "State of lights"

Zentrum für Lernende Systeme & Anwendungen

Reuse the Solution of Case 1

CASE

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

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

Problem (Symptom):1
- Prob.: Break light doesn't work
- Car: Audi 80
- Year: 1989
- Battery voltage: 12,6 V
- state of break light: OK

Adapt Solution:

How do differences in the problem affect the solution?

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

Zentrum für Lernende Systeme & Anwendungen

Store the New Experience

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

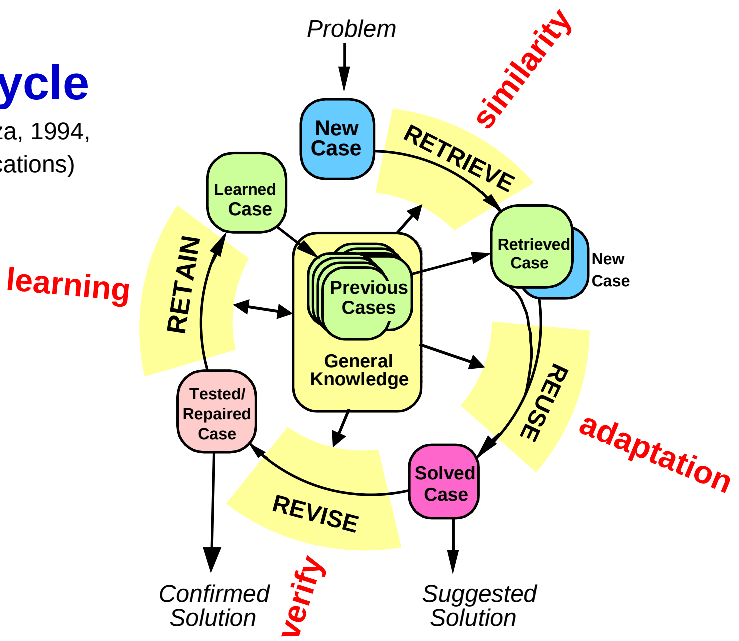
CASE 3

Problem (Symptoms):
- Problem: Break light doesn't work
- Car: Audi 80
- Year: 1989
- Battery voltage: 12.6 V
- State of break lights: OK
- light switch clicking: OK

Solution:
- Diagnosis: break light fuse defect
- Repair: replace break light fuse

CBR Cycle

(Aamodt & Plaza, 1994,
AI Communications)

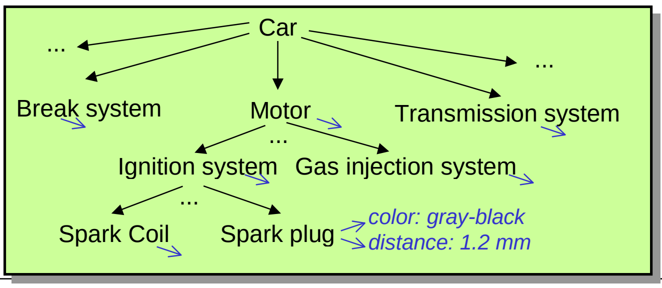


Representing Cases

- **Many different case representations are used:**
  - Depend on requirements of domain and task
  - Structure of already available case data
- **Flat feature-value list**
  - Simple case structure is sometimes sufficient for problem solving
  - Easy to store and retrieve in a CBR system
- **Object-oriented representations**
  - 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*

Object-Oriented Case Representations

- A case consists of a set of *objects*
- An object represents a closed part of the situation
- Objects described by a *set of features*
- *Relations* between objects (e.g. part-of)
- Each object belongs to an *object-class*.
- Object-classes are organized in a *inheritance hierarchy*.
- Case representation language CASUEL (developed in INRECA)



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 of solution
- Similarity is an *a-priori approximation* of utility / reusability
- Goal of similarity modeling: provide a *good* approximation
  - close to real reusability
  - easy to compute

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**

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 CBR *(Lenz)*
    - Discrimination nets: used with representations in logic
    - ...
- **How to store cases:**
  - Databases: for large case bases or if shared with other applications
  - Main memory: for small case bases, not shared

Zentrum für Lernende Systeme & Anwendungen

Reuse: How to Adapt the Solution - Different Options -

- **No modification** of the solution: just copy
- Manual/interactive solution adaptation **by the user**
- **Automatic** solution adaptation
  - **Transformational Analogy:** transformation of the solution
    - Rules or operators to adjust solution w.r.t. differences in the problems
    - Knowledge required about the impact of differences
  - **Derivational Analogy:** replay of the problem solving trace
    - Complete generative problem solver
    - Knowledge required about how to solve the problem in principle
  - **Compositional adaptation:** combine several cases to a single solution

Zentrum für Lernende Systeme & Anwendungen

Revise: Verify and Correct Solution

- **Revise phase:** *little attention in CBR research today*
  - 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

Retain: Learning from Problem Solving

- **What can be learned:**
  - New experience (new case)
  - Improved similarity assessment, importance of features
  - Organization/indexing of the case base to improve efficiency
  - Knowledge for solution adaptation
  - Forgetting cases, e.g., for efficiency or because out-of-date
- **Methods**
  - Storing cases in the case base
  - Deleting cases from the case base
  - Explanation-based learning
  - Induction, e.g. of decision trees
  - Neural net style learning

Where does a CRR System Store Knowledge ?

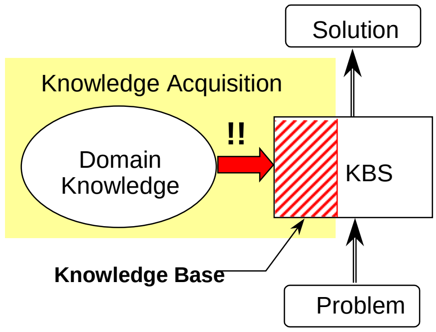
- **CBR systems store knowledge in four different knowledge containers (*Richter, 1995*):**
  - Vocabulary (used features)
  - Case base
  - Similarity assessment
  - Solution adaptation
- **Advantage of CBR**
  - High flexibility:
    - knowledge can be distributed between the four containers according to application needs
    - in principle, every container can hold the whole knowledge
  - Focus on knowledge in the case base
  - Knowledge in the case base can be updated and maintained very easily

Advantages of CBR over other Techniques

- Reduces the knowledge acquisition effort
- Requires less maintenance effort
- Improve 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

Avoid (Partially) Knowledge Acquisition Effort

Traditional Knowledge-Based Systems



Acquisition of general knowledge is very difficult !!

CBR Systems

- Require less general knowledge
- Most knowledge in case base
- Case knowledge is easier to acquire (sometimes already available)

Less Effort Required for Maintenance

What is the impact of changes of the environment ?

- **Rule bases or models are difficult to maintain**
  - Many dependencies between rules
  - Rules of KBS often difficult to understand for non AI experts
  - Effects of changes of the rule base are hard to predict
  - Maintenance by the domain expert impossible !!
- **Case bases are easier to maintain**
  - Cases are independent from each other
  - Domain experts and novices understand cases quite easy
  - Maintenance of the CBR system (partially) by adding/deleting cases
  - However, changes in the vocabulary container require (little) more effort

CBR for Analytic Tasks
- Classification, Diagnosis, Decision Support -

- **Analytic tasks: Focus on analyzing a situation**
  - Classification of the situation always involved
  - Often: fixed number of classes
  - Additional steps may be required: e.g., test selection strategy
- **Characteristics of CBR systems for analytical tasks**
  - Typical case structure: case = < problem, class >
  - Focus on case retrieval
  - Solution adaptation usually not required
- **Examples**
  - Classification of biological objects (e.g. marine sponges)
  - HOMER (**HO**tline **M**it **ER**fahrung)
  - Electronic Commerce applications involving product selection

CBR for Synthetic Tasks
- Planning, Configuration, Design -

- **Synthetic Tasks: Synthesizing a new solution**
  - Compose a solution from different components
  - Problem = description of requirements
  - Usually infinite (or at least very large) solution space
- **Characteristics of CBR systems for synthetic tasks**
  - Typical case structure: < problem,solution > or < problem,solution-trace>
  - Typically, solution adaptation is mandatory
  - Much general knowledge required in addition to the cases
  - Cases often used to improve the performance
- **Examples**
  - Manufacturing planning, transportation planning, etc.
  - Electronic Commerce applications involving product configuration
  - Architectural design (FABEL)

Summary

- CBR is a technique for solving problems based on experience
- CBR problem solving involves four phases:

  Retrieve, Reuse, Revise, Retain
- CBR systems store knowledge in four containers:

  Vocabulary , Case Base , Similarity Assessment, Solution Adaptation
- Different techniques for:
  - representing the knowledge, in particular, the cases
  - realizing the four phases
- CBR has several advantages over traditional KBS
- Applications of CBR for analytic and synthetic tasks