What is CBR?

6

Overview

- What is CBR?
- Motivations for CBR
- Comparison with Other Methods
- Why/When CBR Works

7

7

What is Case-Based Reasoning?

- CBR is ***reasoning by remembering:*** **It** takes *specific prior examples* as the starting point for new reasoning
- CBR is a cognitive model of human reasoning
- CBR is a methodology for reasoning and learning in intelligent systems

8

8

The Types of CBR

- *Problem-solving CBR* solves new problems by retrieving and adapting records from similar prior problems.
- When do you do this in everyday life?

9

9

Examples of Problem-Solving CBR

- Diagnosing a computer problem based on a similar prior problem.
- Predicting how an opponent will act in a game based on how they acted under similar past circumstances.
- Reusing a route you've used before

10

10

Interpretive/Classification CBR

- *Interpretive/classification CBR* understands new situations by comparing and contrasting them to similar situations in the past
- When do you do this in everyday life?

11

11

Examples of Interpretive/Classification CBR

- Predicting whether you'll like a movie by whether you liked a similar one.
- Deciding whether you'll like a restaurant by how you liked a similar one.
- Assessing a vacation plan by comparing and contrasting it to the one your friend took.

12

12

Reasoning in Problem-Solving CBR

- *Case adaptation* revises the stored case, *case evaluation* reasons about the results
- Adaptation is illustrated by remembering the route taken to get to a new restaurant and taking the same route home, but
- ***Reversed, and adjusted for one-way streets.***
- Adaptation can be arbitrarily sophisticated, leading to creative reuse

13

13

Reasoning in Interpretive CBR

- Justification adds reasoning to determine both applicability and potential reuse
- Illustrated by:
  - Avoiding a movie similar to one you liked before--- *because the star would have trouble doing justice to that type of role.*
  - *Weighing the factors in a vacation package to provide the analysis needed to argue that you should get a lower price.*

14

14

Learning in CBR

- Records of problem-solving and interpretation are stored in memory
- Cases contain *ungeneralized* experiences and lessons
- Learning can be based on:
  - Successes to reuse
  - Failures to anticipate and avoid

15

15

The Nature of CBR

CBR =

Memory + Analogy + Adaptation + Learning

16

16

The CBR Cycle

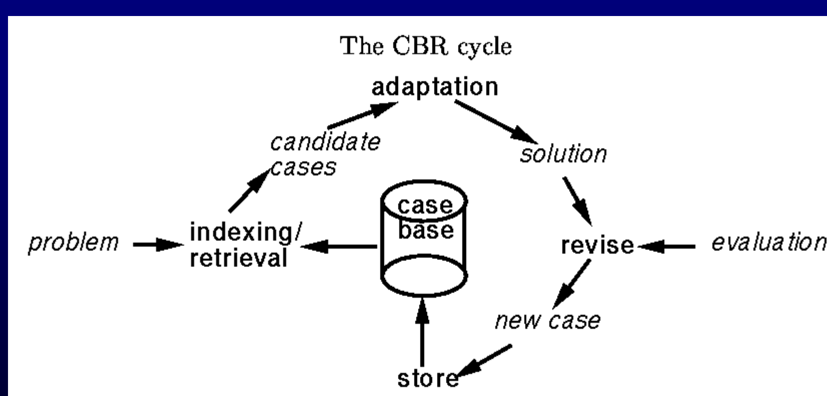


Figure by Boi Faltings, after (Aamodt & Plaza 91)

17

17

Some CBR History

- **1979:** IJCAI-79, Schank talks about dynamic memory theory – "Why computers' memories should be more like people's."
- **1983 :** AAAI, Hammond, Planning and Goal Interaction, and Cognitive Science, Kolodner: Maintaining organization in a dynamic long-term memory
- **1988 :** First DARPA CBR Workshop at Clearwater FL
- **1989**: First Fielded CBR Business Application: PRISM (Goodman)
- **1991:**
  - First conversational CBR tool, CBR-Express, by Inference
  - First European CBR company, Acknosoft
- **1992:** First French and German CBR workshops
- **1993:**
  - First European CBR Workshop
  - INRECA project begins
  - CBR review paper Aamodt & Plaza, AI Communications (the CBR-cycle originated here)
  - First UK CBR Workshop
- **1995:** ICCBR'95: First International CBR Conference in Portugal
- …
- 2004: ECCBR held in Madrid at Univ. Complutense
- 2005 ICCBR held in Chicago
- 2006: ECCBR held in Turkey at fabulous resort
- 2007: ICCBR held in Belfast
- …

18

18

CBR Is Applied to Many Applications

- Power load forecasting
- Software quality control
- Design of plastics
- Aircraft maintenance
- Recommender systems
- …

20

20

Part 2: Motivations for CBR

21

Reasons for Using CBR

- Benefits include:
  - Direct functional benefits such as efficiency, simple learning…
  - Indirect benefits from connections with human reasoning:
    - Aiding knowledge acquisition
    - Aiding explainability
    - Providing a natural vehicle for interactive systems in which human and computer each exploit their individual strengths
- Understanding these requires a look at the CBR cognitive model

22

22

The Cognitive Model

- Early study of CBR was largely motivated by interest in human reasoning
- Ashley and Rissland studied the role of cases in American legal reasoning
- Schank studied the phenomenon of *reminding* during understanding.

23

23

Two Key Cognitive Science Questions

Why are people reminded?

How are people reminded?

24

24

Why AI People Care

- Humans are effective reasoners (usually)
- Studying reminding can give insights into:
  - How knowledge is organized
  - How specific and general knowledge interact
  - How to get more flexible reasoning

25

25

Some Historical Roots from Research on Story Understanding

- Much early CBR grew out of Yale story understanding research
- Their first attempts at story understanding viewed understanding as being able to build up causal chains of inferences to establish coherence

26

26

Pros and Cons of Scripts

- Pros:
  - Efficiency
  - Identifying important inferences (shared background for summarization, etc.)
  - Cognitive validity
- Cons:
  - Lack of flexibility
  - Problems as a model of memory: Script confusions

31

31

Beyond Scripts

- When scripts fail can give important clues to human processing
- Anomalous events can prompt *remindings* during understanding
- The study of reminding is the study of how memories are organized, retrieved, and applied
- This is central to case-based reasoning

32

32

CBR View: Remindings Permeate Reasoning

Does thinking really involve thinking?

(Riesbeck & Schank, 1989)

41

41

CBR is Commonplace

- *Anticipating and avoiding failures*: Feeling congested and remembering that you should have seen a doctor sooner last time.
- *Problem-solving*: Adapting prior code to a new task.
- Understanding anomalous events and predicting consequences, even in careful analysis.

"Following the precedent of the Challenger disaster in 1986, it's unlikely that NASA will undertake any further shuttle missions … for the next two years"
Time Magazine

42

Experts Often Use Experiences

In a [medical] conference attended by a respected professor from another hospital, the chief of a service calls on that [professor], … with a request not for the latest news of research from the journals but for an anecdote: Anybody had any experience with this?
(Hunter, 86)

43

43

Experiences Can Help in Hard-to-Codify, Complex Situations

A Replay of Vietnam in Iraq?
Derrick Jackson, Boston Globe

44

44

Psychological Studies Provide Evidence for Human CBR at Many Levels

- Programming [Faries & Schlossberg, 94]
- Explanation [Read & Cesa, 91]
- Diagnosis [Lancaster & Kolodner, 87, 88]
- Decision-making [Klein & Calderwood, 89]

45

45

One Motivation for AI CBR: Cognitively-Inspired Expert Systems

- Human experts have trouble generating rules, but easily relate experiences
- Analogical reasoning can be helpful when not enough is understood to reason from scratch

46

46

Case-Based Planning

56

Why CBR for Planning?

- Knowledge Issues for planning
  - Generative planning requires full knowledge of relevant factors
  - Identifying them is hard
  - Representing them is hard

57

57

Why CBR for Planning? (continued)

- Computational Complexity of Planning
  - For many domains, planning is NP complete.
  - Even simple domains can have huge search spaces. Nau (2002) illustrates a toy domain “dock worker robots”:
    - 5 locations
    - 3 piles of items to move
    - 3 robots
    - 100 containers
    - Result:~ 10^{277} states (vs. ~ 10^{87} particles in the universe!)
  - Partial order planning helps by focusing and enabling goals to be addressed separately---but problems remain.

58

58

Why Planning is Useful for Understanding CBR

- Unlike the pure retrieval of CCBR, case-based planning must address the *reasoning* of CBR: case adaptation.
- Case-based planning illustrates *problems encountered* vs. *problems in principle*: The interesting part of the space is limited.
- Case-based planning illustrates the chasm between *knowledge for generation* and *knowledge for adaptation*.

59

59

A Case Study of Case-based planning

CHEF (Hammond, 89) builds new recipes from cases representing previous recipes: A natural CBR task.

A sample task:

- Input goals:
  - make a stir-fry dish
  - include beef
  - include broccoli

This toy task led to various real-world applications: bioprocess planning, military force deployment, color matching for plastics…

60

60

What Must we Define for This Task?

61

Similarity and retrieval

- CHEF bases similarity judgments on a hierarchy of important features
- Dish type is most important, then meat type, then vegetable types, …
- Example: Given the goal of beef and broccoli, CHEF retrieves recipe for beef and green beans.

64

64

CHEF's Adaptation Knowledge

CHEF uses two types of adaptation knowledge:

- *modification rules* for structural changes (e.g., additions and reorderings)
- *ingredient critics* to adjust for individual items

66

66

A *modification rule* for adding fruit to a souffle

```
(add:mod
  index (fruit style-souffle)
  amount (cup number (1))
  steps ((do (chop object ?new-item size (pulp)))
            (before (pour object ?object
                           into (nine-inch-baking-dish))
               do (mix object ?new-item with ?object))))
```

67

67

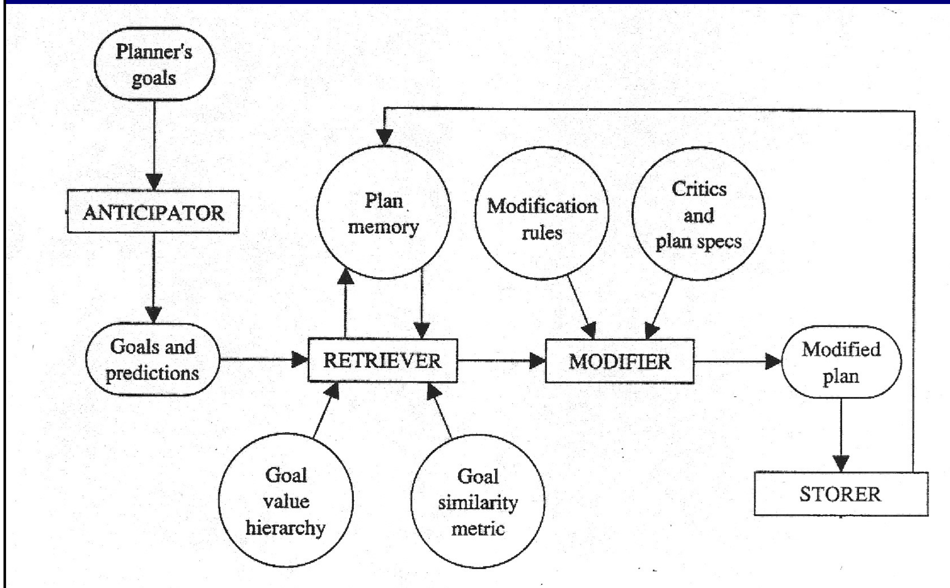
An *ingredient critic* for shrimp

```
(add:crit shrimp
    binds (shrimp *new-item*)
    steps ((before (cook-step object *new-item*)
                do (shell object *new-item*))))
```

68

68

Summary of CHEF's Architecture



69

CHEF's Plan Generation for Beef and Broccoli

- Retrieve beef and green beans
- Apply modification rule to substitute broccoli for green beans
- Apply ingredient critic to add chopping step and adjust cooking time

70

70

Voilà! Beef with broccoli

- Ingredients: 1/2 pound beef, 1 tsp sugar, 2 Tbs soy sauce, 1/2 lb. Broccoli, 1 tsp rice wine,1 tsp salt, 1/2 tsp corn starch,1 clove garlic
- Chop the garlic into pieces the size of matchheads.
- Shred the beef.
- Marinate the beef in the garlic, sugar, corn starch, rice wine and soy sauce.
- Chop the broccoli into chunks.
- Stir fry the spices, rice wine and beef for one minute.
- Add the broccoli to the spices, rice wine and beef.
- Stir fry the spices, rice wine, broccoli and beef for three minutes.
- Add the salt.

71

71

Evaluating result

- Check result against adapted recipe goals:
  - The beef is tender.
  - The dish tastes savory.
  - The broccoli is crisp.
  - The dish is salty.
  - The dish is sweet.
  - The dish tastes like garlic.

- Plan is executed (in simulator) and checked against expectations.

The broccoli is soggy!

72

72

CHEF's recovery from failure

- Explain: Beef releases water => broccoli steams.
- Identify problem pattern:
  - Water released by cooking the beef disabled ``dry wok."
  - How could this be fixed?

73

73

Designing Adaptation of Recipes

- What knowledge is needed?
- How should that knowledge be organized and accessed?

74

74

CHEF's Repair Approach

- Explain the failure
- Generate a failure characterization (how this is done is crucial to generality)
- Use the failure characterization as an index into repair strategies

75

75

Representing Plan Failures: Remember TOPs

- TOPs = Thematic Organization Points (TOPs)
- TOPs describe plans at an abstract, domain-independent level

TOP = MG;OO: "Mutual goal; outside opposition"

76

76

Forming a TOP for the Cause of Soggy Beef

- To keep description abstract, need to describe in terms of *plan steps* and *relationships*.
- Observations relevant to a response:
  - Release of juice wasn't required by the plan
  - Release of juice nullified the dry pan, which was required for cooking the broccoli
  - Juice gathered because the steps were being performed concurrently
- The resulting TOP:
  - SIDE-EFFECT:DISABLED-CONDITION:CONCURRENT
- Note its generality!

77

77

Repair Strategies for the TOP

- Repairs for SIDE-EFFECT:DISABLED-CONDITION:CONCURRENT:
- Split-and-reform.
- Alter-plan:side-effect
- Adjunct-plan

78

78

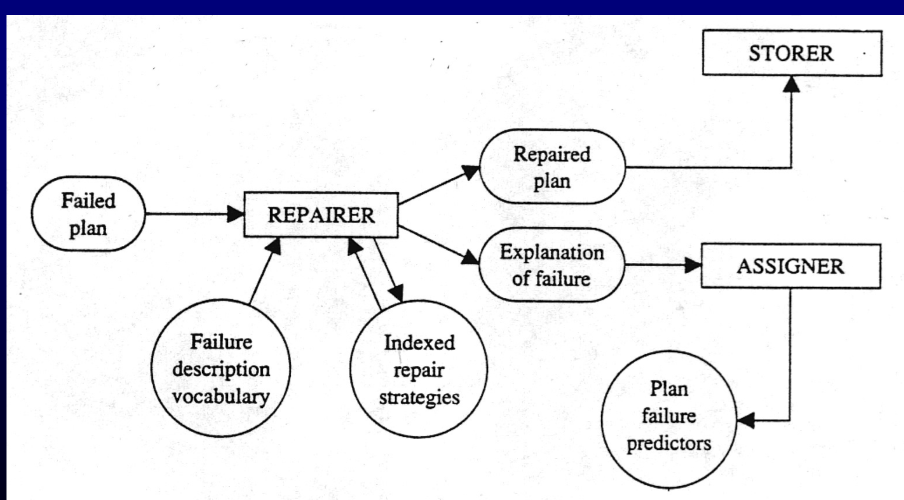
Back to the CHEF Domain

- Split-and-reform applies.
- New plan works!

79

79

Failure Recovery Architecture



80

Learning from experience

- CHEF stores plans under the combinations of goals they satisfy, including the problems they avoid.
- It learns:
  1. A new case: the recipe for beef and broccoli, indexed under
     - Goals (stir-fry, beef, broccoli)
     - Problem avoided (avoids bad meat/vegetable interaction)
  2. A rule for anticipating meat/crisp vegetable interaction problems

81

81

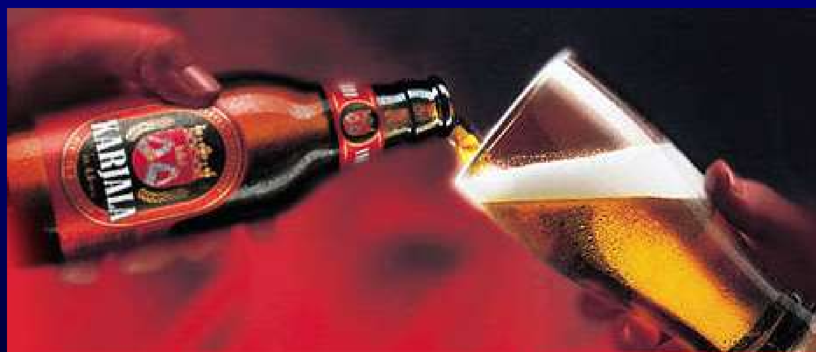
CHEF Illustrates the Range of Learning in CBR

- Learning from success
  - Save successful solutions
- Learning from failures
  - Learn to anticipate and avoid problems
- Effects of learning
  - Improving competence
  - Speedup

82

82

Better Living Through Case-Based Planning



VTT Biotechnology in Helsinki has applied case-based planning to beer production and fielded their system in Finnish breweries.

84

84

Thinking Critically

- When is case-based planning the *wrong* approach?

85

85

MicroCHEF Code is Freely Available

- Simplified "micro" versions of CBR dissertation projects developed as a starting point for pedagogical use are available from link under "Resources" on the class home page.

86

86

We've Now Examined CBR Knowledge Containers Individually, How do They Relate?

- Indexing knowledge
- Similarity assessment knowledge
- Case knowledge
- Adaptation knowledge

87

87

A Useful Property:
The Knowledge Containers Overlap!

- One container's strengths can compensate for another's weaknesses
- Consequently:
  - A system need not have rich knowledge in all containers
  - The designer can place knowledge where most convenient---i.e., can use the knowledge that's easiest to capture.
- Let's look at some examples.

88

88

Knowledge Container Overlap Example 1

- If system has a case for every problem it may encounter, what does that mean for the needed similarity assessment knowledge?
- *Similarity assessment requires only exact match*
- What does it mean for the needed adaptation knowledge?
- *No adaptation knowledge is needed*

89

89

Knowledge Container Overlap Example 2

- If system has all the adaptation knowledge that could ever be needed, what does that mean for the number of cases needed?
- *Only one case is needed; all solutions can be generated from it*

90

90

Knowledge Container Overlap Examples 3

- If system has no adaptation knowledge, but a perfect similarity metric, what does that mean for the number of cases for a classification task classifying into n categories?
- *Only n cases are needed.*

91

91

CBR as Machine Learning

92

Case Learning

95

95

What New Cases Can Contribute

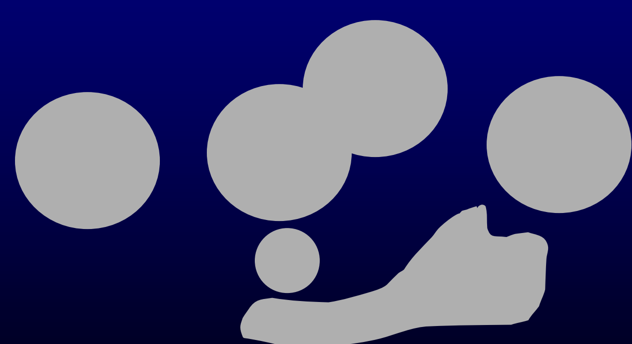
- How do people improve with experience?
  1) Competence (Range of problems they can solve correctly)
  2) Speed-up
- Case acquisition is *lazy learning*: Examples are retained and generalized as needed (contrast neural nets, d-trees, explanation-based learning,…)

96

96

Competence and Case Acquisition

- Case availability and adaptation knowledge are limiting factors on competence
- Adaptation knowledge fills in inter-case spaces



97

97

Case Acquisition and Speedup: A First Pass

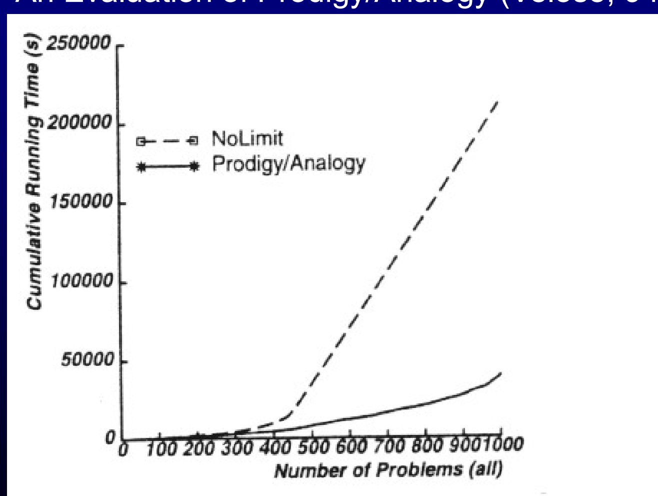
- More similar cases require less adaptation
- Reduced adaptation reduces adaptation cost
- Reduced adaptation cost increases efficiency

98

98

The Desired Picture

An Evaluation of Prodigy/Analogy (Veloso, 94)



99

99

What could go wrong?

100

A Complication: The Utility Problem for CBR Smyth & Cunningham (96)

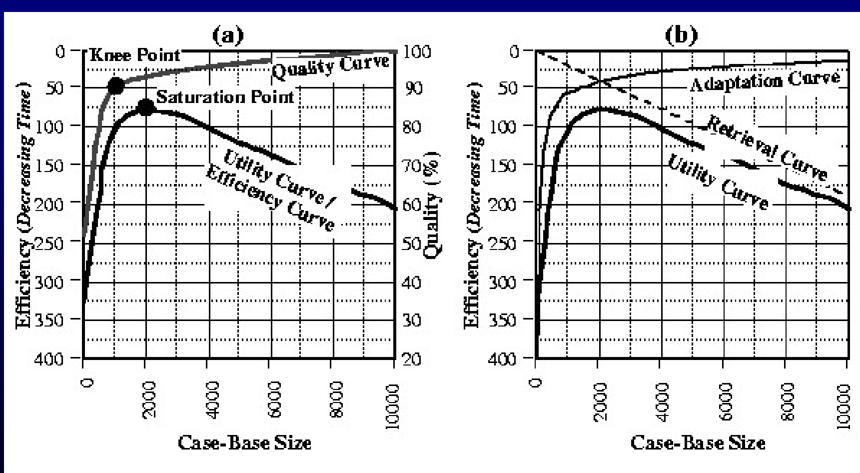


Figure 2. (a) Quality vs. Efficiency (b) The Utility Factors; Retrieval vs. Adaptation

101

Case Acquisition and Speedup: Second Pass

- More similar cases require less adaptation
- Reduced adaptation reduces adaptation cost
- Reduced adaptation cost increases efficiency,
- But…
- More cases increase retrieval/similarity assessment cost
- Consequently, case base growth must be controlled.

102

102

In-Class Exercise

- With the people in your row, come up with a scheme for addressing one or more problem(s) which arise as a case-base reasoning system saves cases over time. Delineate:
  - The problem(s) you're addressing
  - The solution(s) you propose

103

103

Case Base Maintenance

- Many strategies have been explored:
  - Competence-preserving deletion aims to decrease case base size without sacrificing coverage (Smyth & Keane 95)
  - Learning by failure with forgetting aims to save only valuable cases (Portinale & Torasso 99)
  - Performance-guided maintenance aims to maximize expected performance (Leake & Wilson, 2000)

104

104

Case-Base Maintenance

- Initial work was motivated by concerns about
  - Case storage capacity
  - The *utility problem,* in which retrieval costs outweigh benefits.
- Other motivations may include:
  - Distributed case bases with multiple contributors may have inconsistent representations or conflicting information.
  - Short-term domain changes may change desired solutions.
  - Long-term use of CBR systems may result in obsolescence.
  - Transfer costs may limit useful case-base size

105

105

A Definition of Case-Base Maintenance

Case-base maintenance implements policies for revising the contents or organization of the case-base in order to facilitate future reasoning for a particular set of performance objectives.

(Leake & Wilson, 98)

106

106

Facets of CBM

- Revision of case-base contents to change case representation, domain content, or ``accounting'' information
- Revision of organizational information such as indices and indexing structures (e.g. tree structures)

These may change case-bases at the *implementation level* or the *knowledge level* (Dietterich, 86).

107

107

Components of Maintenance Strategies

- Data collection type: None, based on current state, based on trends
- Triggering characteristics: Events or trends that prompt maintenance
- Operation types: What can be changed
- Execution: When the operations are performed

108

108

Case Update in Basic CBR Learning

CBR systems routinely maintain the case base by adding new cases as problems are solved.

- Data collection type: None
- Triggering characteristics: Every cycle
- Operation types: Add case
- Execution: Immediately

109

109

Case Update in Non-Learning CBR Systems

Expert runs tests and updates as needed.

- Data collection type: Use statistics for each case
- Triggering characteristics: Periodically check for unused cases and bad performance on verification tests
- Operation types: Add/delete/revise by hand
- Execution: When expert detects problems

110

110

Deletion Risks for CBR

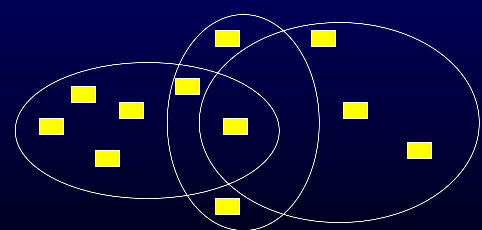
- The utility problem was first identified for explanation-based learning, by Minton (88).
- Minton’s solution: Select rules to delete (how should system choose?)
- Why is deletion safe for EBL but not CBR?
- EBL systems can re-derive rules as needed; CBR can't
- Predicting usefulness is hard:
  - Usage patterns may change: Underused cases now may be vital next month
  - A seldom used case may be crucial some day

111

111

Competence-Preserving Deletion
(Smyth and Keane, 95)

- Goal: Controlling case base growth without harming *case-base competence*.
- Strategy: Selectively delete cases by observing their coverage regions



113

113

A Competence-Preserving Deletion Strategy

- Setting up the strategy:
  - Data collection type: Snapshot of case-base size
  - Triggering characteristics: Exceeds size limit
  - Operation types: Delete unneeded cases
  - Execution: On-line or off-line
- What is an “unneeded” case?

114

114

How to Decide What to Delete (Smyth & Keane, 1995, Smyth & McKenna, 1999)

- Define basic properties of cases' contributions:
  - Coverage(c) = {c' ∈ CB | Adaptable(c, c')}
  - Reachability(c) = {c' ∈ CB | Adaptable(c', c)}
- How can we know these when only part of the problem space has been seen?
- The *representativeness assumption*: The existing CB is a good proxy for the entire problem space (why is that reasonable to assume?)
- Estimate
  - Coverage by the cases in the case-base a case can solve
  - Reachability by the cases in the case-base that can solve it

115

115

Building Compact Competent Case Base

Condensed nearest neighbor (Hart, 1968)

```
O-SET ← Original training examples
E-SET ← {}
CHANGES ← true

While CHANGES Do
          CHANGES ← false
          For each case C∈O-Set Do
            If E-SET cannot solve C Then
              CHANGES ← true
              Add C to E-SET
              Remove C from O-Set
            EndIf
          EndFor
EndWhile
```

117

117

How Should Cases be Ordered? Nearest Unlike Neighbor (NUN)

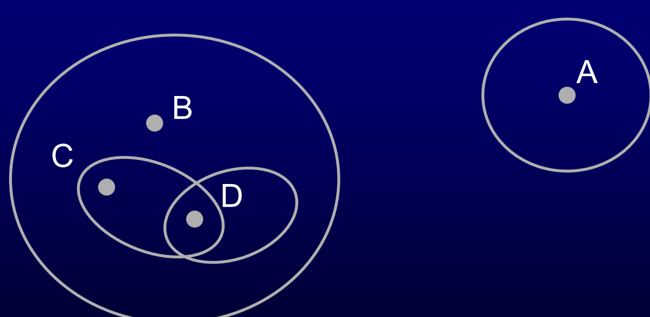
- Nearest unlike neighbor: Add cases with smallest distances to unlike neighbors first.
- What's the rationale?

118

118

Making it Smarter

- Given the representativeness assumption, which cases should be added first?



119

119

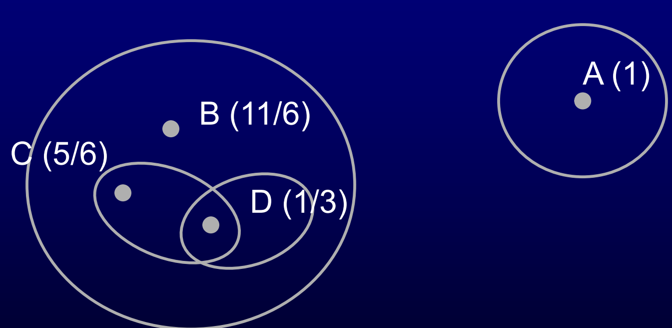
Relative Coverage

$$RC(c) = \sum_{c' \in CoverageSet(c)} \frac{1}{|ReachabilitySet(c')|}$$

120

120

Calculating Relative Coverage



121

121

Applying RC to CNN

- Add cases with greatest relative coverage first

Dataset/Editing	CNN	NUN	RC
Ionosphere	61.93 85.78	46.39 84.44	49.47 85.3
Credit	344.84 58.85	297.43 58.95	299.19 60.44
Travel	184.28 89.25	196.98 88.72	165.42 86.4
Property	55.19 95.92	57.81 95.53	45.44 94.62

122

122

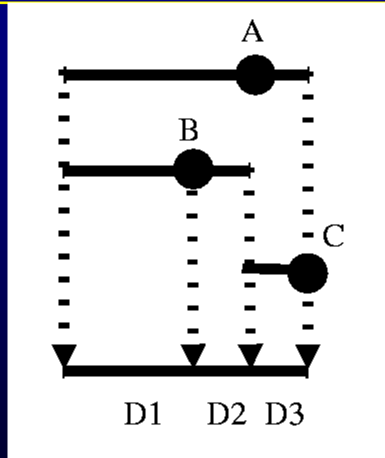
Performance-Guided Maintenance

- Compactness usually isn't an end in itself
- Overall performance depends on adaptation costs
- Competence-guided decisions may actually sacrifice performance

123

123

An Example



Right end of A's coverage interval is open; other intervals closed.

What CB does greedy addition choose? What gives best performance?

124

124

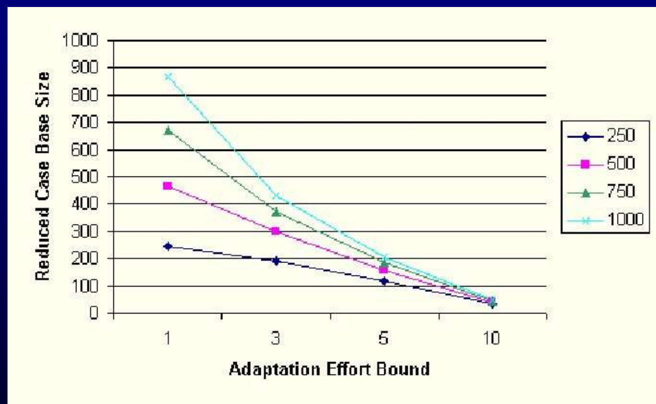
A Closer Look at Adaptability

- Smyth et al. treat adaptability as binary
- Adaptability is judged by a threshold effort limit
- This makes "competence" give a performance bound, but blurrs inter-case differences

125

125

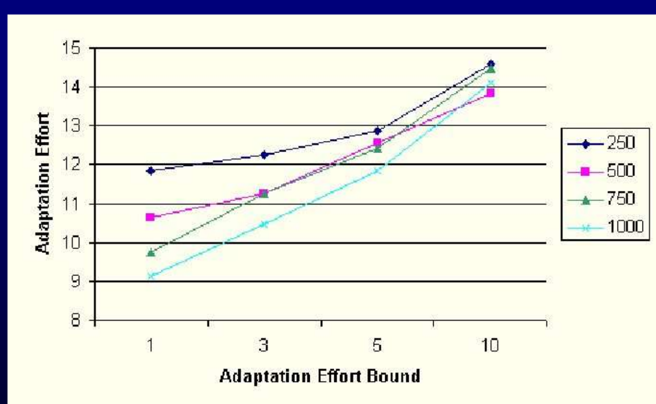
How Threshold Level Affects CB Size



126

126

How Threshold Level Affects Processing Cost



127

127

Towards Performance-Based Addition (Leake & Wilson, 2000)

- Do competence-based addition, but order based on *relative performance boost*
- In tests, improves performance 5-8% with comparable compression

128

128

Broadening the View: Other Knowledge Containers

- Knowing the relationship of the knowledge containers, it makes sense not to restrict learning to cases alone!

129

129

Index Update Based on Re-Retrieval
Fox & Leake, 95

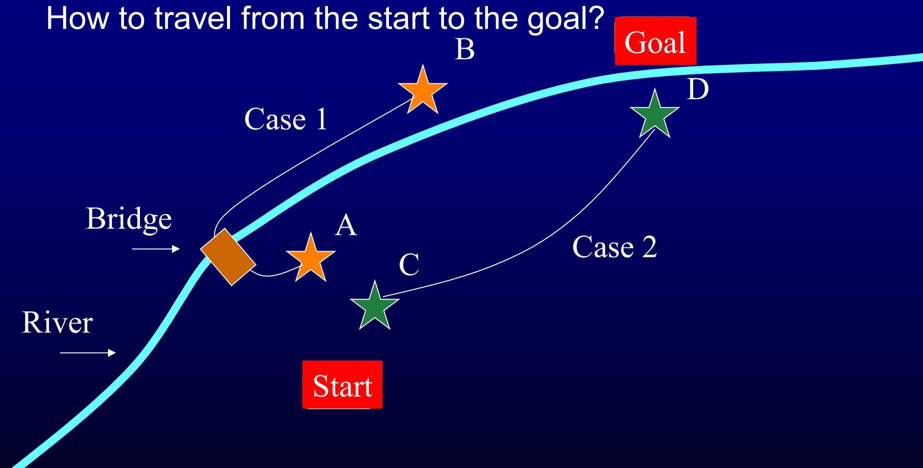
- Refining indices in a case-based planner
- How can system know that the wrong case was retrieved?
- ROBBIE'S process:
  - Perform CBR
  - After a solution is generated, retrieve based on the full solution
  - If the retrieved case is closer, indexing was flawed.
  - Adjust indices to reflect missing features

132

132

Case-Based Path Planning

How to travel from the start to the goal?



133

133

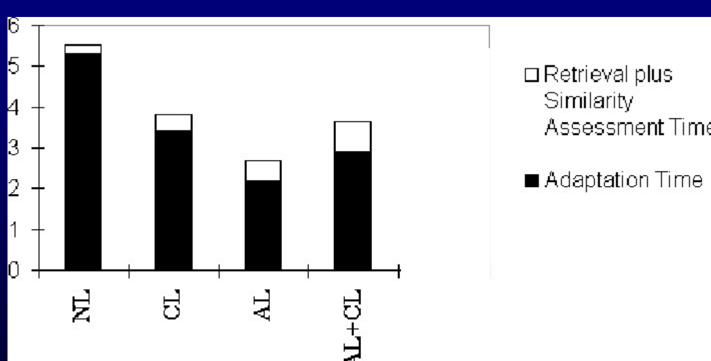
Adaptation Learning

- Explanation-based, Inductive, and Case-based methods have been used
- How could we design a case-based adaptation system?

139

139

Results from DIAL



141

141

Conclusions on Learning

- CBR provides multiple opportunities for learning
- Maintenance responds to index and case problems to improve performance, to improve quality and efficiency of processing
- Large-scale and long-term CBR use make maintenance especially crucial
- Maintenance is a major focus of current research

Relationships to Other Methods

Tasks Suited to CBR (Kolodner, 93)

- Proposing quick solutions (without rederiving them)
- Dealing with imperfectly-understood domains
- Interpreting open-ended or ill-defined concepts
- Warning of prior problems
- Evaluating solutions when no algorithmic method available

145

145

Key Properties of CBR

- Knowledge capture:
  - Cases avoid burden of distilling rules *(but burden may be shifted to indexing, similarity assessment, and adaptation knowledge)*
  - Case-bases tolerate inconsistency
  - New cases can be used as soon as stored (inertia-free)

146

146

Key Properties of CBR (continued)

- Learning
  - CBR systems learn as a byproduct of reasoning
  - Can start from a few examples, and build automatically
  - Can learn to warn of failures as well as to reuse successes
- Efficiency:
  - Effort can be reused *(but the utility problem may arise as case-base size grows)*

147

147

Key Properties of CBR (continued)

- Explainability:
  - Real examples can be used to account for system results
- Interactivity:
  - Cases provide raw material for natural human CBR process *(but people may need support to use them effectively)*

148

148

Relationship to Rule-Based Systems

- Knowledge capture:
  - Unlike CBR systems, rule-based systems
    - Often have no automatic learning
    - Require expert knowledge to be distilled into rules
    - Cannot be deployed based on partial rule sets
  - CBR systems:
    - Easily capture exceptions that may be difficult to capture in rules
    - Easily capture warnings of failures

149

149

Relationship to Rule-Based Systems (continued)

- CBR contrasts rule-based systems':
  - Startup:
    - RBS must have rules to cover every possible situation
  - Adaptation to changes:
    - RBS knowledge changed by hand
    - May be hard to anticipate how rules will interact
  - Efficiency:
    - Without learning, no benefits from experience
  - Explainability:
    - Explain by tracing chains leading to a conclusion *(but rules may not be clear to the user)*

150

150

Relationship to Information Retrieval

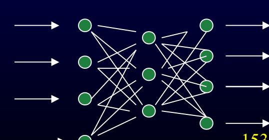
- CBR emphasizes the task: it takes a *problem* and retrieves a *solution*
- IR relates to CBR for cases with a natural textual form. In help desks, IR is a first step to select cases to consider.
- In IR, indices are generally weighted keyword vectors; in CBR, often complex structures in non-textual form. CBR matching procedures may be knowledge-based.
- IR attribute weightings are typically rooted in attribute frequencies; CBR 's on estimates from domain knowledge.
- CBR goes beyond retrieval, with case adaptation and learning

151

151

Relationship to Neural Networks

- Knowledge capture: Both neural networks and CBR stress learning
- Learning methods:
  - Networks generalize immediately and discard examples; CBR does lazy learning and retains them
  - Network learning requires expensive training, with complete retraining for new examples; CBR only requires adding cases.
- Explainability: Networks can't explain decisions; CBR can explain based on real prior examples



153

153

Is CBR a Methodology or Technology?

- Case-based reasoning is a methodology of reasoning from specific experiences, which may be applied using various technologies (Watson 98)

154

154

Why/When CBR Works

155

Why/When CBR Works:
A Picture of Similarity Assessment and Adaptation

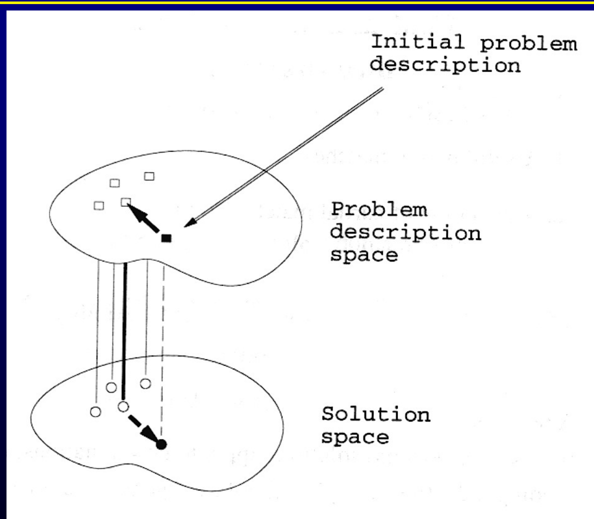


Figure from (Leake 96)

156

156

When Does it Best Apply?

- Because CBR can exploit different types of knowledge containers, it is promising whenever knowledge in some of these containers is readily available (retrieval, similarity, case, adaptation)
- CBR depends on problem-solution regularity, and problem-distribution regularity. If these don't apply, CBR won't work.
- CBR often works well in partially-understood domains, for which both experience and background knowledge come into play
- Other methods may be preferable for:
  - Very poorly-understood domains with many examples (consider, e.g., neural networks)
  - Very well-understood domains with few examples (consider, e.g, explanation-based learning)

157

157

Formalizing the Key Regularities

- CBR depends on two assumptions
  - “similar problems have similar solutions”
  - “experiences tend to be useful for new problems”
- Leake & Wilson (99) defined two terms to measure these properties:
  - *Problem-solution regularity:* Similar problems have similar solutions
  - *Problem-distribution regularity:* New problems are similar to those previously encountered

158

158

Measuring the Regularities (Leake & Wilson, 99)

Given:

- A CBR system
- A problem distance function, PDist, used by the system to measure distance between problems during retrieval
- A “real distance” function, used by an external observer to judge retrieval performance
- An initial case-base B
- A problem sequence Q = $p_i, p_{i+1}, \ldots, p_j$

159

159

1. Define Closest Cases

Closest Cases to Problem $CCP(PDist, p, B)$ =
$\{c \in B | PDist(p, c) = min_{c' \in B} PDist(p, c')\}$

Real Closest Cases $RCC(RDist, p, B, \epsilon)$ =
$\{c \in B | RDist(p, c) \leq min_{c' \in B} RDist(p, c') + \epsilon\}$

160

160

2. Define Precision of Similarity Metric

$$SimPrecision(PDist, RDist, p_k, B_k, \epsilon) = \frac{CCP(PDist, p_k, B_k) \cap RCC(RDist, p_k, B_k, \epsilon)}{CCP(PDist, p_k, B_k)}$$

161

161

3. Problem-Solution Regularity is Average SimPrecision

$$ProbSolnReg(PDist, RDist, Q, B_i, \epsilon) = \frac{\Sigma_{k=i,\ldots,j} SimPrecision(PDist, RDist, p_k, B_k, \epsilon)}{j - i + 1}$$

162

162

4. Problem-Distribution Regularity Reflects Performance on a Given Problem Stream

Problem-distribution regularity is percent of cases in the problem stream with sufficiently close prior cases.

$$ProbDistReg(Q, B_i, \epsilon) = \frac{1}{j - i + 1} \Sigma_{k=i,\ldots,j} \begin{cases} 1 & min_{c \in B_k} PDist(p_k, c) < \epsilon \\ 0 & \text{Otherwise} \end{cases}$$

These definitions enable assessing (1) quality of the similarity metric, to guide maintenance, and (2) the suitability of CBR for a given task.

163

163

Summary of Parts 1 and 2

- Case-based reasoning is a useful paradigm for problem-solving and interpretation, inspired by human reasoning and learning
- CBR exploits *problem-solution regularity* and *problem-distribution* regularity
- Learning is a natural byproduct of CBR

164

164

General References

- Riesbeck & Schank, *Inside Case-Based Reasoning*, Erlbaum, 1989.
- Kolodner, *Case-Based Reasoning*, Morgan Kaufmann, 1993.
- Leake, *Case-Based Reasoning: Experiences, Lessons, and Future Directions*, AAAI/MIT Press, 1996. Overview chapter "CBR in Context: The Present and Future" is on-line at www.cs.indiana.edu/~leake/papers/a-96-01.html
- Watson, *Applying Case-Based Reasoning: Techniques for Enterprise Systems*, Morgan Kaufmann, 1997.
- Bergmann, *Experience Management · Foundations, Development Methodology*, and Internet-Based Applications, Springer, 2002

165

165