Chapter 9

Using Past History Explicitly as Knowledge: Case-Based Reasoning Systems

Chapter Objectives

- Introduce the student to the concept of using explicit historical occurrences to solve current problems.
  - Explained in the context of rule-based systems that also use past experience to solve current problems
- Introduce case-based reasoning.
- Introduce how case-based systems can learn from their own experience

Weaknesses of rule-based systems

- Weaknesses of rule-based systems that inspired the rise of case-based reasoning:
  - Experts may not be able to externalize their experience into clean bits of knowledge that can be encoded into rules
    - Their knowledge is an accumulation and a combination of years of being exposed to many instances of similar problems (and their subsequent solutions)

Weaknesses of rule-based systems

- To manage the knowledge of experts, we must:
  - Elicit it from the expert
  - Represent or formalize it in a form suitable for computing
  - Validate and verify the knowledge
- All these contain pitfalls for the rule-based systems approach

Weaknesses of rule-based systems

- Q. Why may rule-based systems have difficulty with eliciting, representing, and validating knowledge from the expert?
- A1. Results vary depending on which expert
  - It is the experts' personal interpretation of the domain
  - Some experts are very knowledgeable, others only minimally so
  - Different experts may see the same domain from different perspectives – in fact, they may all be correct in some way

Weaknesses of rule-based systems

- Q. Why may rule-based systems have difficulty with eliciting, representing, and validating knowledge from the expert?
- A2. Transferring the codified knowledge can be difficult and error-prone
  - Experts can provide erroneous knowledge if the KE's question is ill-posed
  - The KE can misinterpret an expert's correct answer
  - Developers can misrepresent correct knowledge in the system code (or rules)

Weaknesses of rule-based systems

- Q. Why may rule-based systems have difficulty with eliciting, representing, and validating knowledge from the expert?
- A3. Too many rules may be needed to properly represent one domain.
  - Eg: GenAID had ~10,000 rules when initially deployed
  - Disadvantage 1: The rules have to be coded, verified, validated, and maintained
    - Complexity of validation and maintenance can grow exponentially with the number of rules, due to rule interaction!
  - Disadvantage 2: They have to be executed by the inference engine
    - Computational cost can become infeasible

Case-Based Reasoning (CBR)

CBR is an alternative to rule-based systems…

- Keep all the concrete **cases** that might have led to the learning by the experts
- Stick to recording the concrete details of each case, without generalizing experience into rules
- Avoid the personal influence of individual experts
- Bypass the expert and look directly at the information that allowed them to learn and acquire their expertise
- We no longer need the experts' interpretation, and thus avoid the associated drawbacks

Case-Based Reasoning (CBR)

- One approach to avoid the problem of knowledge acquisition and maintenance
- The CBR technique originates from Schank's [1982] concept of **remindings**:
  - When people are thinking (eg, solving problems), they are merely recalling past experiences that somehow remind them of the current situation
  - If the current and historical situations are sufficiently similar, then it can be inferred that the solutions to both situations are the same
  - I.e., people apply solutions of past problems to current problems that are similar in nature

Case-Based Reasoning (CBR)

- Example 1 [Klein 1985]
  - Fire ground commander coordinating his crew, while fighting a fire at a low-rise apartment building
  - Notices billboards on the building's roof
  - Recalls earlier incident where flames burned through the wooden billboard supports, causing them to crash to the street below
  - Orders that spectators be moved farther back to prevent injury from falling billboards
- Note: Highly unlikely that such a rule would have already been included in a KBS
  - Unless spectators had already previously been injured by falling billboards – but that would be too late!

Case-Based Reasoning (CBR)

- Example 2 [Klein 1985]
  - Fire ground commander notices some peculiar properties in a cloud of smoke at a fire
  - Recalls an incident in which toxic smoke had been given off showing the same features of density, color, and heaviness
  - Orders his crew to use breathing support systems
- Note: Highly unlikely that such a rule would have already been included in a KBS
  - Unless fire crew had already previously been injured by this specific kind of toxic smoke – but that would be too late!

Case-Based Reasoning (CBR)

- CBR uses explicit historical experiences to solve new problems
- Assumes that problems recur, and that “similar problems have similar solutions”
- Intuitive and simple framework that developers and users find natural to understand and work with
- Provides a natural (though simplistic) form of learning by merely adding any newly solved problem to its database of past cases

Case-Based Reasoning (CBR)

- Simplest, most basic form of CBR:
  - A repository of historical cases called the **case library**
  - A means to find and retrieve a similar case from the case library, and use its solution to solve the current problem
  - A means to add the newly solved problem and solution to the case library as a new case

Case-Based Reasoning (CBR): Adaptation

- But: What happens when the most similar case is not judged similar enough to the current problem?
- This will happen much of the time – if we just give up, CBR won't be very useful!
- In such circumstances, the solution(s) of the most similar case(s) should be **adapted** to the current problem.

Case-Based Reasoning (CBR): Adaptation

- Automatic adaptation is a very difficult problem technically.
- Except in highly limited formalisms, there is little solid mathematical theory to support practical adaptation methods.
- In many systems, adaptation has been abandoned altogether.
  - Focus instead on improving searching and learning.
- Combining CBR with other approaches is the usual way of tackling automatic adaptation, eg:
  - Use rules to make adaptations
  - Use machine learning and pattern recognition methods

Case-Based Reasoning (CBR): Successful vs failed cases

- Important to categorize cases according to whether it succeeded or failed
- Failed cases can provide as much (or more!) useful information than successful ones

Case-Based Reasoning (CBR)

- More realistic CBR systems:
  - Search the case library
    - Requires efficient indexing of the cases
  - Retrieve the most similar case(s)
    - Requires quantitative similarity metrics
  - Adapt the most similar case(s) if not suitably similar
    - Technically difficult, and optional
    - Sometimes impossible, eg, pre-filmed video clips
  - Apply the solution to the current problem
    - Capture whether it succeeded or failed, as feedback
  - Add the last case to the case library
    - Requires criteria to decide whether it is worth adding
    - Requires efficient updating of the indexing

Indexing the case library

- Indexing = labeling of data items in such a way that they are easily retrieved
- Examples:
  - Library card catalogs (usually only by title or author)
  - Library numbering system / physical organization (usually by subject)
    - Dewey Decimal System
    - Library of Congress System
  - Library electronic catalogs (more powerful searching over a wider range of attributes)
  - Library full text search engines
  - Internet full text search engines

Indexing the case library

- Issue: What attributes of cases are indexed?
- For real-world cases, there are typically a huge range of attributes you could imagine to index each case by.
- If attributes are defined too broadly, an unnecessarily large number of cases may be retrieved for examination.
- If attributes are defined too narrowly, there may be some truly similar cases that are overlooked during retrieval.
- Explanation-Based Indexing (EBI) is one AI technique for deciding which indexes, out of a predefined possible space of indexes, to use for each case. [Barletta 1988]

Indexing the case library

- Issue: What is the best way to organize a library of cases?
- The effect of retrieving an improperly matched case is often more computationally expensive than selecting the wrong rule to execute in a rule-based system.
- Inefficient searches due to poor organization can result in unacceptable performance.
- Main means of increasing search efficiency through indexing:
  - Flat library
  - Shared feature networks
  - Redundant shared feature networks

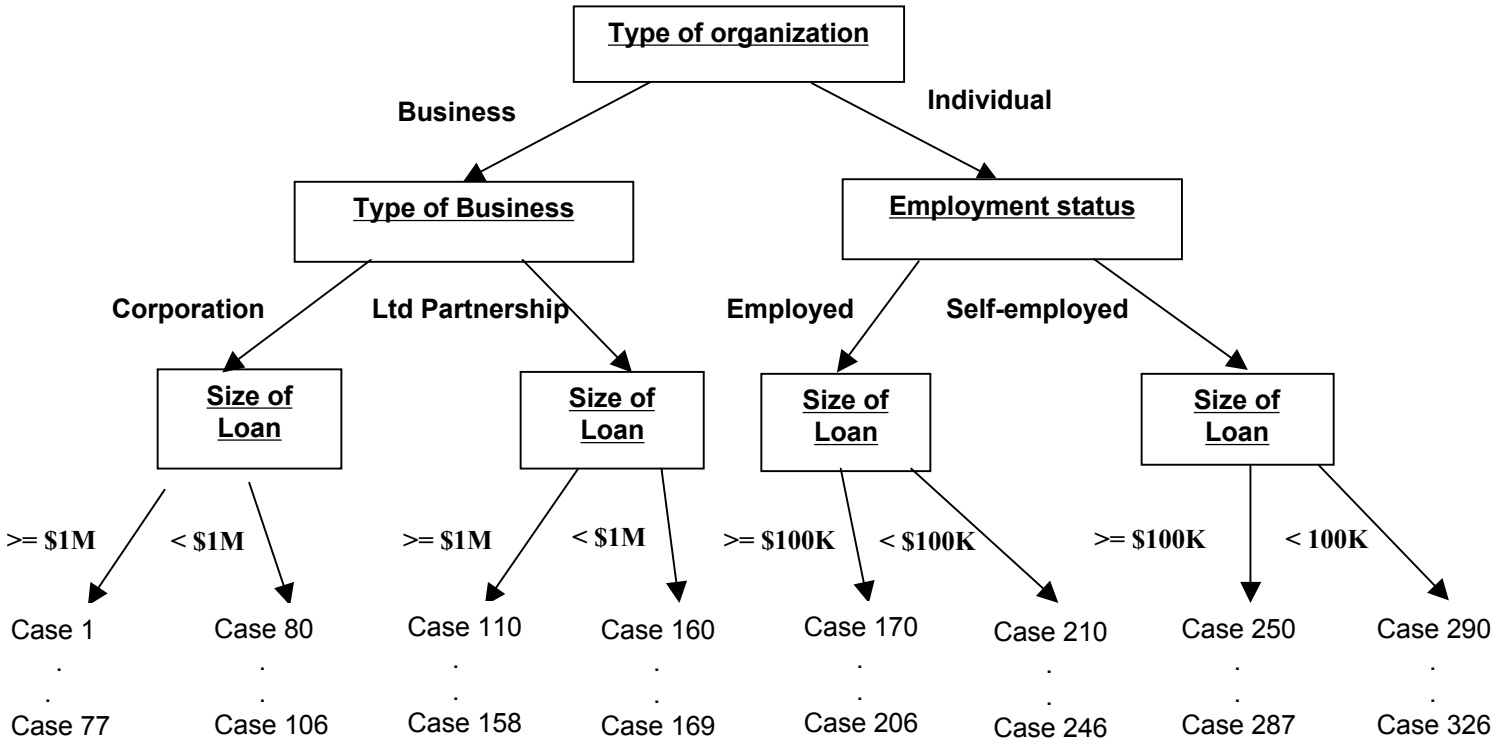
Indexing the case library: Flat library

- **Flat case libraries** are the simplest
- Cases can be placed in a list, array, or file
- Some ordering may be imposed
- May be too inefficient for larger case libraries with complex cases; additional strategies:
  - Partitioning the library (eg, individual vs corporate)
  - Relational database techniques

Indexing the case library: Shared feature networks

- Decision tree clustering
- Hierarchical organizations that segregate cases by what features they have in common
- Cluster cases as much as possible – given any node, segregate its cases by choosing the feature most universally shared
- Search process simply follows path through tree, matching features of the current problem

Fig. 9.1: Shared feature network for a loan application example



Indexing the case library: Redundant shared feature networks

- Problem: Often there are some unknown attributes in any new case. Can't find path through tree!
- **Redundant shared feature networks** attempt to overcome this by maintaining a number of different trees, each of which prioritizes different attributes.
- Choose the tree that gets you the longest path, ie, closest to the leaf level, so that the roadblock comes in at the latest possible stage in the search.
  - Tries to give the narrowest subset of similar cases.
- Still a rather "sledgehammer" approach.
- Thus, best to use hierarchical organizations in applications where there is little reason for new cases to have incomplete information.
  - Example: Property listing databases
  - Counterexample: Diagnostic cases

Matching and retrieval of cases from the case library

- A **distance metric** is used to compute the distance between historical cases and current problem
- A distance metric is a function that aggregates variation over a large number of attributes
- Attributes may be discrete/boolean or continuous
- There are always many possible aggregation functions
  - Weighted sums (weighted averages)
  - Higher-order polynomials (regression based measures)
  - Hamming distances, and more sophisticated edit distances
  - Cosine based measures
  - Information-theoretic and probabilistic measures
  - Ad hoc measures
  - etc.
- No universal truth – which distance metric is best depends heavily on the domain
  - Often must be empirically determined

Evaluation

- How to determine whether the most similar case is similar enough?
- May involve implementing the solution
  - within a simulator
  - in real life under test conditions
- Not always possible!
- May also involve looking for *negative* cases – those whose solution did *not* solve the current problem when applied

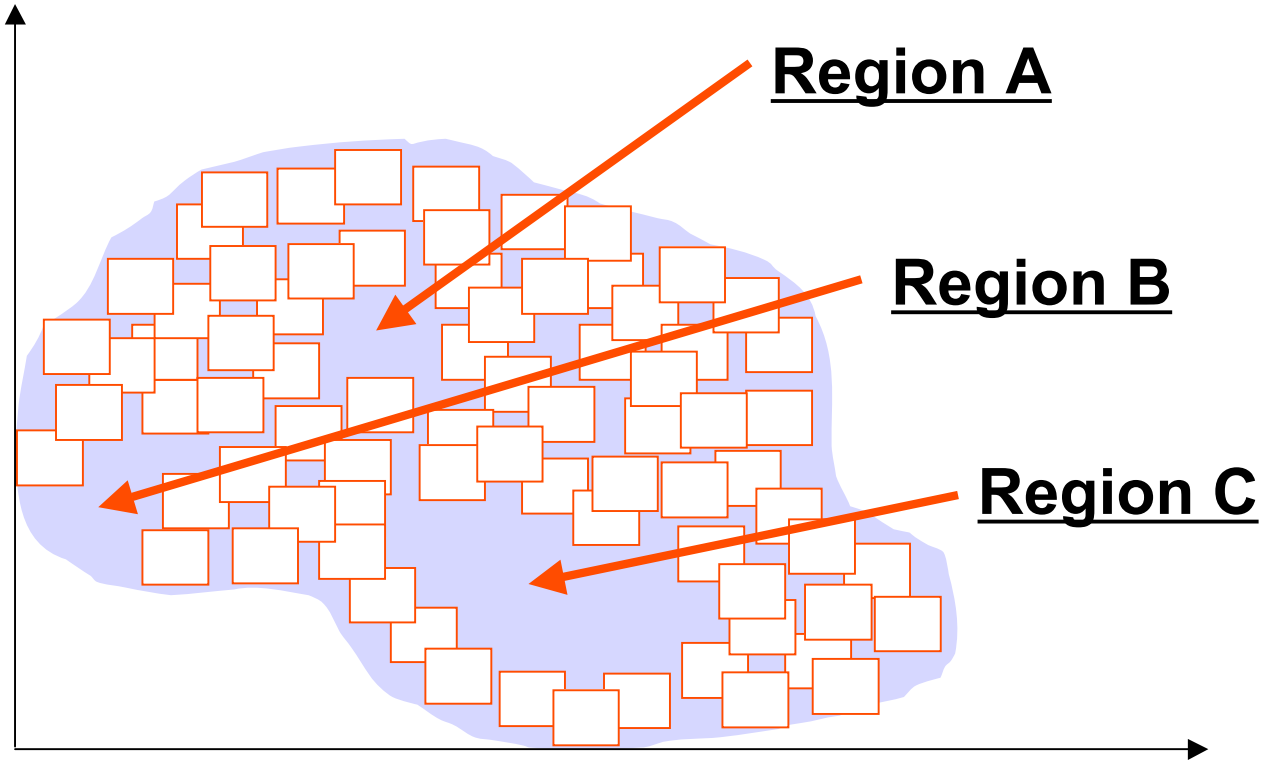
Adaptation

- How to modify the most similar case when it is not sufficiently similar to the current problem?
  - **Reinstantiation** - eg, replace beef with chicken
  - **Parameter adjustment** – eg, scaling income/credit
  - **Search** – eg, find exact location of hose leak
  - **Case-based substitution** – recursively use CBR to find a substitute for a mismatched sub-step in the case
  - **Transformation** – use some non-CBR (eg, heuristic rule-based) method to find a substitute for a mismatched sub-step in the case
  - **Model-guided repair** – use a causal model to determine the approapriate transformation

Learning

- Learning in the context of case-based reasoning can often compare very favorably with the rather painful and expensive knowledge acquisition and maintenance process for rule-based systems.
- Learning is done simply by adding new cases to the knowledge base.

Fig. 9.2: Pictorial representation of a simple 2-D problem space



Learning

- Adding cases progressively covers more and more of the problem space.
  - The more cases, the better the coverage of the problem domain.
- Many cases overlap.
  - Not a problem, as long as their solutions are consistent.

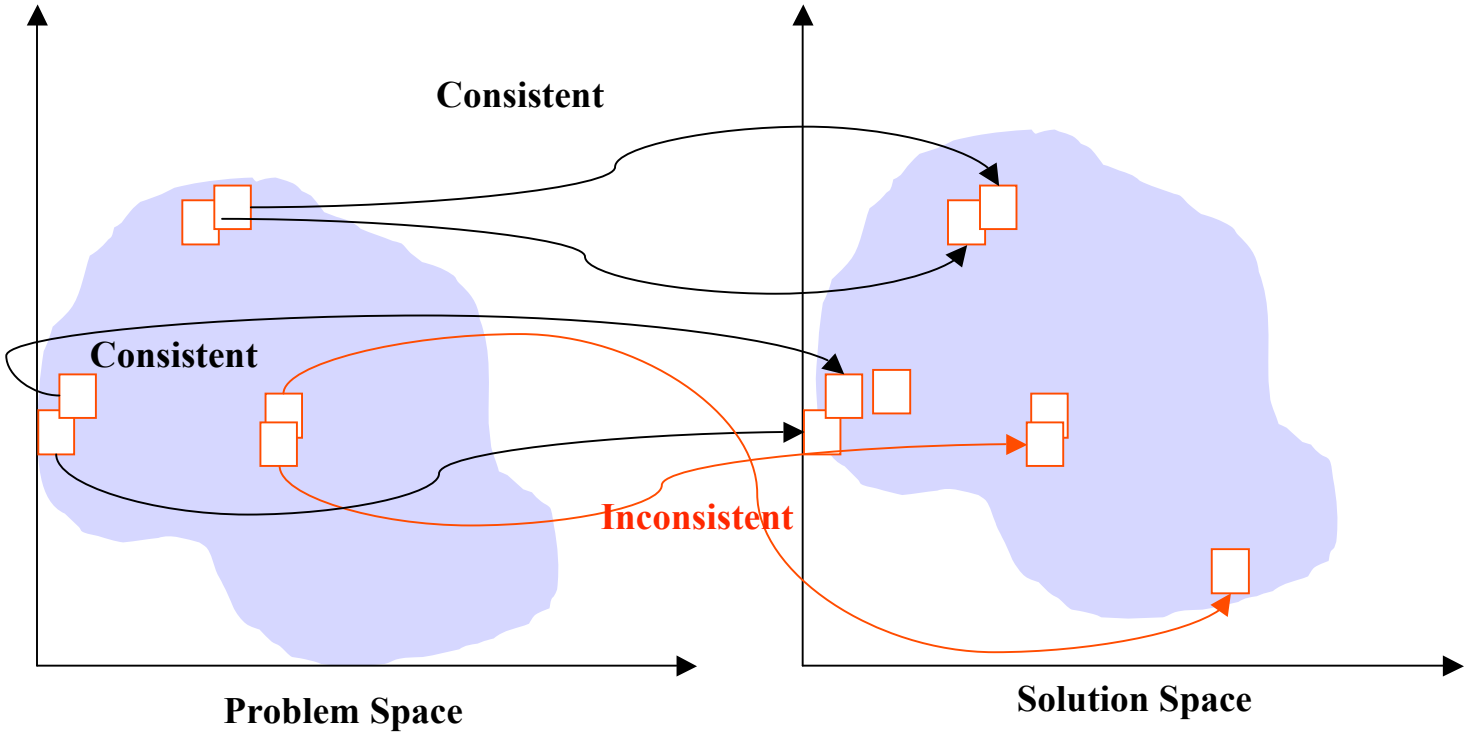
Learning

- Issues:
  - Too many cases:
    - May clog the search and retrieval process unnecessarily
  - Too little diversity among the cases (even if there are many cases):
    - May leave significant gaps in coverage
    - Eg, regions A, B, C in Figure 9-2

Learning

- Issues:
  - Inconsistency between problem and solution spaces
  - Neighboring cases in the problem space are likely to map to neighboring cases in the solution space if the problem domain
    - is based on a natural process,
    - is well understood, and
    - all cases are completely defined.
  - Solutions may conflict with each other, ie, solutions to nearly identical cases may be radically different, if the problem domain
    - is poorly understood,
    - is based on irrational human behaviors (eg, stock or property markets), or
    - is incompletely defined.

Fig. 9.3: Mapping between problem and solution spaces



Learning

- Learning in the context of case-based reasoning can often compare very favorably with the rather painful and expensive knowledge acquisition and maintenance process for rule-based systems.
- Introduce the concept of when new cases are consistent with the rest of the case library and when they are not
  - This is important when deciding whether to add new cases or not

Where CBR excels

- CBR is excellent when many well-documented histories of past problems and their solutions exist.
- Examples:
  - Law: legal cases
  - Property: appraisal

Example: CBR applied to property appraisal [Gonzalez 1992]

- Use the market data method of property appraisal
- Features (attributes):
  - Living area in square feet
  - Number of bedrooms
  - Number of bathrooms
  - Architectural style of the house
  - Age of the house
  - Location (neighborhood)
  - Date of sale
  - Type of cooling equipment
  - Type of heating equipment
  - Type of garage
  - Site or lot size
  - Availability of swimming pool

Example: CBR applied to property appraisal [Gonzalez 1992]

- Case retrieval:
  - Retrieves 10 best cases
  - Ranks in order of decreasing similarity
- Case adaptation
  - Uses **critics**: heuristic rules that increase or decrease the actual sold price of a retrieved property based on differences between it and the property being appraised
    - eg, a swimming pool critic
  - Adaptation is cumulative: done for all features of the comparison
- Case evaluation
  - Too many adaptations can result in inaccuracy
  - A **comfort factor** indicates which of the 10 cases was least extensively adapted
  - Top three are selected (traditional in appraisal business)
- Returns the average of the top three

Some issues in case-based systems

- It can take a much larger number of cases than rules to cover a domain to the same extent.
- Whether the assumption that similar problems have similar solutions really holds depends on the domain.
- The similarity metric depends on the domain and can greatly affect CBR systems (and sometimes there may not even be any good similarity metric).
- Adaptation is highly domain dependent (and in extreme cases may be equally or more complex than a rule-based system).

Advantages of case-based systems

- The knowledge acquisition process is considerably simplified in many applications, especially where the case library may already exist as corporate documentation, possibly even in an electronic database.
- The knowledge maintenance process is greatly facilitated by the learning ability of CBR systems.
- CBR is modeled after human reasoning. There is significant evidence to believe that CBR is a cognitive problem-solving model [Kolodner 1993].
- CBR performs better than rule-based systems in so-called weak-theory domains. That is, these are domains where experts may not exist, or if they exist, they do not fully understand the intricacies of the domain.
- The base of experience used can be that of an entire organization, instead of that of a few interviewed individuals. This can multiply the breadth of a knowledge base in CBR.

Disadvantages of case-based systems

- Just as efficiency is seen as an advantage, it can also be a disadvantage for large systems with poorly organized or indexed case libraries. Moreover, the matching process, if complex, can add computational cost to the CBR system, regardless of how well designed the case library may be.
- In many cases, distance calculations between the desired and actual solution can be difficult to make. This is from a conceptual as well as a computational standpoint.
- Adaptation may be quite difficult or impossible in many domains. In others, it is done with rules.
- Learning, although natural and intuitive, demands some careful considerations as to which cases are added to the case library, and how.
- Building a case library may not be easy in some situations, and may approach the difficulty of building a rule base. This may be the case where cases do not already exist or are poorly documented. In such cases, experts can be asked to create the cases from their experiences. This may be as difficult as implementing the knowledge in a conventional rule-based system. We argue that this neutralizes one otherwise major advantage of CBR.

Section 9.10 - Objectives

- Briefly introduces some variations of case-based reasoning:
  - Exemplar-based reasoning – focuses on the classification phase of CBR
  - Instance-based reasoning – emphasizes a very large number of cases (instances) represented in simple regular forms (eg, attribute vectors or feature vectors), thus facilitating fully automatic machine learning
  - Memory-based reasoning – identical to CBR
  - Analogy-based reasoning – focuses on adaptation via the mapping problem (how to map the solution of the analogue case to the current problem)

Conclusions

- The student should be familiar with:
  - The difference between how rule-based systems and case-based systems use historical knowledge.
  - The main processes of case-based reasoning:
    - Search
    - Select
    - Adapt
    - Apply
    - Learn
  - The advantages and disadvantages of case-based systems

A7: Individual Assignment
(Due at beginning of class Jul 23)

1. For one week, keep track of what you eat for breakfast, lunch, and dinner. Also record any special attribute that contributed to you ordering or preparing each specific meal. Were you particularly hungry? Did you have a special desire for something? Were you eating out? Were you celebrating some good news? By looking at these cases, decide how you will index the case library to achieve the most efficiency in searching.
2. For the problem of deciding what to eat for the following week, design a distance metric to determine similarity in cases.

Chapter 9

Using Past History Explicitly as Knowledge: Case-Based Reasoning Systems