Diet Menu planning Using CBR

Rajendra Akerkar

Vestlandsforsking,
6851 Sogndal, Norway
rak@vestforsk.no

Abstract. An application of case based reasoning (CBR) in the diet menu planning is discussed. The paper begins with an outline of knowledge intensive learning methodology used in the system. An overview of computer assisted diet menu planner is provided. Finally, the problem solving in the system and some evaluation efforts are discussed.

Keywords: Case based reasoning, diet menu planner, learning, knowledge acquisition.

1 Introduction

Case-based reasoning [1, 8] simulates a type of human problem-solving behavior. It is useful whenever it is difficult to formulate domain rules and when cases are available. CBR is a rich and knowledge intensive method for capturing past experiences, enhancing existing problems solving methods, and improving the overall learning capabilities of machine [9]. Many successful applications, such as CHEF [6] and PROTOS [3], have proven the utility of this paradigm.

Planning systems are characterized by their intention to help in solving a process involving a number of steps. Therapy support is an often seen example of planning in medical systems. Three recent planning systems based on CBR approach:

- Davis *et al.* [5] are using a planning system based on the ReCall CBR shell. The system decides what kind of SMARTHOUSE devices disabled and elderly people need in their homes for independent living.
- MNAOMIA operates in the domain of psychiatric eating disorders [4]. It assists clinicians with the cognitive tasks of diagnosis, treatment planning, patient follow-up and clinical research.
- The *Auguste* project [10], is an effort to provide decision support for planning the ongoing care of Alzheimer's Disease (AD) patients. The first reported system prototype supports the decision to prescribe neuroleptic drugs for behavioural problems. The prototype is a hybrid system where a CBR part decides if a neuroleptic drug is to be given, and a Rule-Based Reasoning (RBR) part decides which neuroleptic to use.

Unlike most problem solving methodologies in artificial intelligence, CBR is memory based, thus reflecting human use of remembered problems and solutions as a starting point for new problem solving. An observation on which problem solving is

based in CBR, namely that similar problems have similar solutions, has been shown to hold in expectation for simple scenarios, and is empirically validated in many real-world domains.

This paper presents the diet menu planning system called *DietMaster* [7]. The framework discussed here represents a perspective on knowledge, reasoning, and learning that aims to satisfy fundamental system requirements. Section 2 describes the system, problem solving in DietMaster is discussed in Section 3. Overall planning model is given in Section 4, and Section 5 presents some evaluation results.

2 The System—DietMaster

The DietMaster follows a *knowledge-intensive* approach to problem solving and learning. It is an approach that is based on an intensive use of domain knowledge and relevant parts of the surrounding world in its problem-solving and learning methods.

2.1 Diet Menu Planning Process

The process of menu planning is derived using data collected from experts (dietitians, physicians, medical practitioners, and home science departments of various colleges running undergraduate and postgraduate courses) and literature of the Indian Medical Council [11, 12, 13, 14, 15, 16]. The menu-planning process is divided into five major steps:

- Determine the total calories required for an individual according to his or her physical state and activities he or she is doing.
- Determine the nutrient requirements with regard to carbohydrates, proteins, and fat to suit the person's health state and overcome the energy requirement.
- Determine the proportions of eleven food groups to fulfill total carbohydrate, protein, and fat requirements as well as restrictions that should be followed to improve the health state.
- Divide these proportions into multiple intakes within a day (say, 12 hours) period.
- Plan a sample menu by assigning proper food items to each intake, which will match the food group requirement.

2.2 Functional Components of DietMaster

DietMaster integrates both problem solving and learning architectures [2]. The flow of control and information between the knowledge base and the processes of problem solving and learning are shown in Figure 1. At a high level of abstraction, DietMaster is an integrated architecture containing three building blocks: an object-level knowledge base, a problem solver, and a learner.

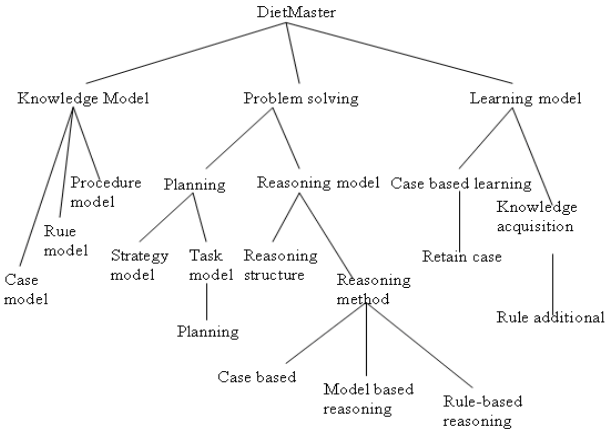

Fig. 1. Functional Architecture of *DietMaster*

The object knowledge base contains a conceptual knowledge model, a collection of past cases, and a set of heuristic rules. DietMaster contains multiple methods for reasoning and learning, where each method communicates with a part of the knowledge base.

The planning task controller and the combined reasoning controller use knowledge in the planning and reasoning modules, respectively. For example, the planning module initializes the problem-solving process through the start-up task "Understand problem," which activates a set of relevant features. This is followed by the task "Retrieve candidate solutions," which triggers the reasoning model, whose initial task is to decide whether the case base or heuristic rules should be used in the initial attempt to solve the problem. When presented with a new problem, the reasoning controller receives a problem frame containing the input features and the current goal.

In a system for planning a diet menu, for example, the initial goal could be "Find total calories required." If a feature is known to the system, it will have a frame that describes its current properties. If a feature is not available, the system uses rules to generate it and creates the corresponding slots. The system checks whether this description violates any constraints or current facts. The input features (observed features, also called *findings*) trigger a chain of activations along a particular subset of relationships in the knowledge model. This process establishes a context for further inferencing, which is guided by the current goal and subgoals and the current state of the planning model. The planning model continually updates the subgoals as the problem-solving process proceeds.

The process of selecting the initial reasoning paradigm starts when a set of relevant features of a problem has been identified. This feature set typically contains input features as well as inferred features—that is, features that the system derives from the input features by using its knowledge. If the set of relevant features gives a reminder of a previous case that is above a particular strength—called the *reminding threshold*—case-based problem solving is tried; otherwise, activation of the rule set is attempted. Relevant features may be input features or features inferred from the object domain model.

For example, if an application system has a well-developed collection of rules but has not seen many problems yet, the user may want to optimize performance by setting the threshold level to a high value—close to 1.0. The rule base will then be used in the initial attempt to solve the problem, unless there is a strong reminder to a case. As more cases are added, the ratio of case-based to rule-based successes is likely to increase, and the system will gradually lower the threshold value. The exact flow - for this process will depend on the actual application system.

The sustained learning control module is the operational part of the learning model. The learning controller guides the learning process by providing a strategy for when to perform particular learning tasks and how to combine them. DietMaster learns from every problem-solving experience. If a successful solution was copied from a previous case, the reminder to that case from each relevant feature is strengthened. In this situation, no new case is stored, but an attempt is made to merge the two cases by generalizing their feature values. If a solution was derived by revising a previous solution, a new case is stored. A new case is also created after a problem has been solved, whether from rules or the deeper knowledge model. Thus, the main target for the learning process in DietMaster is the case base. But DietMaster may also learn general knowledge through interactions with the user during problem solving.

Heuristic rules are integrated within the conceptual model and are available for the same tasks as the conceptual domain model in general. A rule may be used to support learning, for example. Even if the explanatory strength of a shallow relationship like a rule in general is low, it may add to other explanations for the same hypothesis and, thus, contribute to a justification of a hypothesis.

2.3 Representation of Knowledge

DietMaster uses different types of knowledge at the object and control levels as shown in Figure 2. Domain knowledge at the object level consists of a conceptual domain model in which specific experiences (past cases) and general heuristics (premise-conclusion rules) are integrated. Explicit knowledge about how to use domain knowledge for problem solving and learning is described partly as control level concepts, partly as control rules. The general definitions of concepts and relations at this level is part of the unified conceptual fundamental knowledge. This enables an explicit definition of each type of concept that will be reasoned about. A function, for example, may be explicitly defined by describing its input arguments, output, side effects, contextual constraints, dependencies of other functions, etc. In this way, the system is able to understand the effect and operation of a function, and use this understanding in its reasoning at a higher level (control level).

There are two types of heuristic rules within the DietMaster architecture, the premise-conclusion rule - called *conclusive rule* - and the premise-action rule - called *conditioned procedure*. While a conclusive rule expresses a `premise-> conclusion` statement, a conditioned procedure represents a `premise -> action` sequence structure.

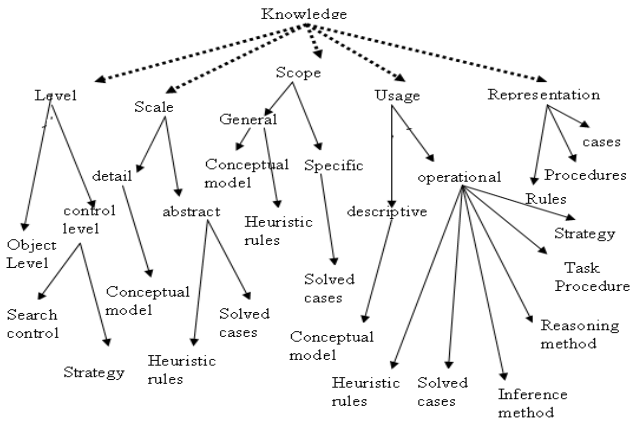

Fig. 2. Dimensions of knowledge

The conclusion part of a conclusive rule contains an assignment of a value to some variable (a frame-slot pair), which is a different type of knowledge than a specification of a sequence of steps to follow in order to reach a (sub) goal.

Thus, although they both are regarded as heuristic rules, conclusive rules and conditioned procedures are different types of knowledge, used for different purposes. Both types of heuristic rules are used at the control level, while - as noted above - the type of rule at the object level is the conclusive rule.

2.4 Case Structure

The structure of case is considered in three different modes; viz. input case, case in process, output case.

Table 1. A Case structure

Input Case	Case in process	Learned Case
Input findings	Input findings	Relevant findings
Goal	Derived findings	Case reused for generating a plan
	Solution constraints	Explanation of successful plan
	Planning states	Successful diet plan
	Possible planning	Different cases
	Explanation of planning	Book-keeping information
	Possible diet plan	
	Explanation of plan	
	Similar cases	

New client enters in DietMaster and asks for diet recommendation; he has to feed his personal information to the system. This collects some features that describe the case. This makes *input case*. When diet planning process starts, it generates some intermediate results describing the client in more detail. Those are called as *derived*

descriptors of the case, those along with input descriptors forms *case in process*. After completion of the planning process case will be stored in the case base for future reuse. Here only required information for reuse gets stored in the case base this becomes *case to store.*

The case structure is shown below. Input to the system represents the input case. It includes physical state of the client, symptoms, test details, other health complaints, exercise details and meal pattern.

Age (in years)
Weight (kg)
Height (cm)
Gender
Male / Female
Female Status
General
1st Trimester Pregnancy
2nd Trimester Pregnancy
3rd Trimester Pregnancy
0 – 6 Mon Lactation
7 – 12 Mon Lactation
Body Frame
Small / Medium / Large
Eating Habit
Vegetarian / Non Vegetarian
BFI (Waist to Hip ratio)
Work type / activity
Lose of wt / Bed patient / Sedentary / Moderate / Heavy
Test details
BP – Blood Pressure in mm
Systolic, Diastolic
BGL Test Method
Plasma / Whole Blood / HbA1c / Urine / Glycometer
BGL - Blood Glucose Level in mg/dl
Fasting BGL , Postprandial BGL
Cholesterol mg/dl
Insulin
Type
Dosage for a day in cc
Exercise
(Type, duration in minutes)n
Diabetic Symptoms
Polyuria
Polydypsia
Polyphagia
Dehydration
Loss of weight / Weakness
Less Immunity
Less Healing capacity
Degenerative Changes
Ketosis/Acidosis
Other health complaints

(Disease)^{m}
Current meal Pattern
(Intake, Food item, Unit Quantity)^{m}

In addition to input features, case has some descriptors while case is in process. These descriptors are derived features, constraints on output, actual output and book keeping information. Solved cases are retained for solving future similar problems. This case consists of *Input, Output, Outcome* and *book keeping information* descriptors. Input descriptors are useful for searching a matching case to be used to solve new problem, Output descriptors are used to apply as a solution. The outcome descriptors reflect strength of a case, which helps to find most desirable case while choosing a better match or a nearest neighbor for solving new problem. The book keeping information helps to check performance of the systems.

2.5 Rules

General knowledge of DietMaster is stored in terms of rules and models. These rules are mostly fixed. Some rules apply on input descriptors of the client (new case) and generate result or intermediate result. Intermediate results form case descriptors for solving further problem. These are referred in the model as derived descriptors, which will be further used to generate output. Some of these rules are conclusive rules whereas some are conditioned procedures. Rules are associations of Knowledge units. For instance, balanced weight for given height, age and gender has been stored in terms of associated rules. There are two sets of rules. One set maintains standard weight for children. These are maintained in the form:

(Age, Height, Gender)-> Standard Weight.

General knowledge is also stored in terms of procedures that solves problem partially whenever past cases are not available. These procedures are also used in adaptation process. Major five operations in diet menu planning along with supportive processes are generalized and kept available. These can also act as fundamental knowledge and some times with conditioned rules. Algorithms for five major procedures are being prepared. We will present one such algorithm below:

(This procedure performs initial step of menu planning to calculate Total calories.)

Input: Age, Height, Weight, Gender, Female status,
Activity and Waist to hip ratio.
Output: Total Calories required per day for the client.

Calculate Total Calories (Age, Height, Weight, Gender,
Female Status, Activity,Waist to Hip ratio)

1. Start.
2. Calculate BMI.
   BMI = (Weight in kg) / (Height in meters)
3. If Age <18 then
   Apply Rule 1.1 to find standard weight
   Otherwise

Apply Rule 1.2 to find standard weight

4. Decide weight status
   a. Actual Weight is less than 20% of Standard Weight and Age < 18 then
      Weight Status <- Underwt
   b. If Age > 18 and Actual Weight exceeds Standard Weight by 20% and BMI > 29 then
      If Gender = Male and Waist to Hip ratio > 1 then
      Weight Status <-Overwt
   Otherwise
      If Gender = Female and Waist to Hip ratio > 0.9 and Female status = Normal then
      Weight Status <-Overwt
      end if
      end if
   c. According to Weight Status decide calories change
      If Weight Status = Underwt then Caloriesded = -500 Otherwise Caloriesded = Ø
      If Weight Status = Overwt then Caloriesded = 500
5. Apply Rule 16 to check activity and decide caloric factor
6. Calculate Total Calories
   Total calories = (Standard Weight in Kg) * Caloric Factor
7. If person does exercise, Apply rule 4 and calculate sum of calories consumed for all exercises say it Extra calories
8. Total calories required = Total calories - Caloriesded + Extra calories
   from steps 4,6 and 7
9. Decide additional calories required for female client [NIN- a]
   If Female status = 1st Trimester Pregnancy then E= 10 cal.
   If Female status = 2nd Trimester Pregnancy then E= 90 cal.
   If Female status = 3rd Trimester Pregnancy then E= 200 cal.
   If Female status = 0 - 6 Mon Lactation then E = 550 cal.
   If Female status = 7 - 12 Mon Lactation then E = 400 cal.
   Total Calories = Total Calories + E
10. Stop

3 Problem Solving in *DietMaster*

Problem solving in DietMaster is controlled and guided by the planning model. This generic model specifies a goal and task structure for the planning process. The planning model considers planning a major task that is further decomposed into subtasks. These steps should be executed in sequence, as they are dependent on each other. DietMaster has to go through five major steps when planning a diet. It applies a combined approach to solve each individual task based on the retrieve–reuse/generate–revise cycle. This is a combination of case-based and rule-based approaches. The CBR cycle generally consists of four steps, called the REs: retrieve, reuse, revise, and retain. Out of those four, three REs—retrieve, reuse, and revise—are adopted here. Generate is the phase where calculations are done using general knowledge to solve the problem and generate a result. DietMaster selects appropriate features and sets an index parameter and then retrieves the matching case. It proceeds with the next step after solving the previous step; at the same time, it passes matching cases of the previous step to be used as a search space for the next step. This way, it reduces the search space and increments the search criteria until it reaches the last step. If DietMaster could not find a matching case to solve a subproblem, it solves the subproblem with the generate phase satisfying that subgoal. When a step is solved with the generate phase, DietMaster uses the complete case base as a search area for solving further steps.

While solving the problem, DietMaster maintains the explanation regarding phases undergone through the problem-solving process. The output generated is displayed and an expert user is asked to revise it to properly suit the current client. After solving the problem, the output is evaluated for relevant information, which is retained, if necessary.

4 Learning in *DietMaster*

The reuse phase of the CBR cycle works at each step of planning. In the adaptation phase, we propose a strategy of using experiences in multiple levels. The old cases get reused in the same way as in the diet planning process. DietMaster decides up to which step the current matching case (old case) can be reused. This is done by comparing corresponding features. There are two types of matching processes implemented in DietMaster: direct matching and relative matching. The direct matching method is used to find candidate-matching cases. The relative matching method selects the most desirable case. In this method, in the selection of the most successful case is made from the candidate case set. Suppose old plans are not available to solve a step, DietMaster uses general knowledge.

The primary machine learning method in DietMaster is the learning of problem specific knowledge by retaining experience in the form of cases. However, a DietMaster system is also able to take part in the refinement of its general knowledge model. The structure of learning process is shown in the Figure 3. The DietMaster system learns by retaining cases, whether representing a single problem-solving experience or merged into slightly generalized cases.

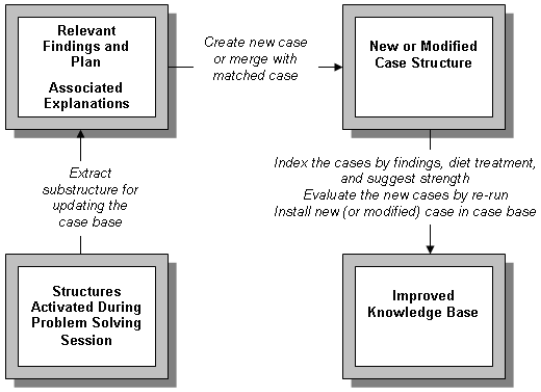

Fig. 3. Structure of the learning process in DietMaster

The criteria for when to store a new case, when to modify an existing case, and when to update indexes only are as follows: A new case is created if no similar past case was found, a past solution needed modification, or the new case has a set of relevant findings significantly different from those of the matching case. In this context, " significantly different" means that the two sets have different values and the corresponding findings cannot be generalized. If both cases have the same findings and their values are close enough to be generalized, the two cases are merged. If their values are close enough to be regarded equal, the only update made is adding "book-keeping" information to the case and adjusting the relevance factors. If a past case leads to a successful solution to the new problem, this leads to a strengthening of the relevance factor for each finding. If a past case leads to an unsuccessful solution, this leads to a weakening of the relevance factors. As the system gains experience, the frequency of problem cases that lead to the storing of a new case will gradually decrease, implying a flattening of the knowledge base growth curve over time.

5 Evaluation

Our model is evaluated by feeding data collected from Kolhapur district. The result shows that our model is efficiently working due to incremental matching process, proper organization of cases according the goal driven classification. Figure 4(a) shows results obtained with a real case base containing 1000 cases. Figure 4(b) depicts the performance of system in terms of availability of number of cases and usage of cases to solve the problem. The result shows that reuse of cases is 75%. We found that as size of case-base increases the performance is also increased.

Another evaluation we followed is effect of diet treatment on diabetes patients. Majority patients receive good result in terms of improvement in Blood Glucose Level. At the same time insulin dependent patients also experienced improvement in sugar level due to diet suggestion through the DietMaster.

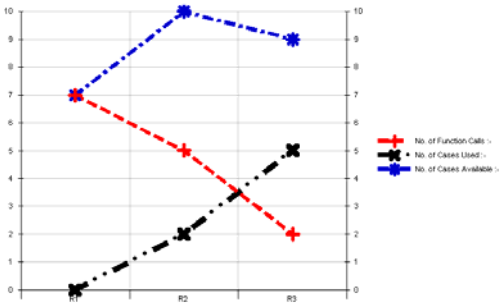

Fig. 4(a). Performance chart

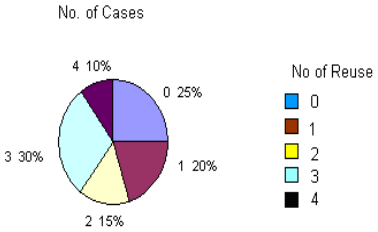

Fig. 4(b). Usage of Case base

Dietitians in Kolhapur district tested the DietMaster and found that the guess a dietitian made resembled with the result of DietMaster. They found it very handy tool and useful.

5 Conclusion

CBR can be useful in assisting diet planning and treatment. In this paper an application in diet planning is described. Important limitation of the system is that it does not consider vitamin and mineral requirement exhaustively. It selects food items rich in required vitamin and minerals but may not fulfill complete requirement of the same. With some modifications this shortfall may be overcome. The current system works for almost all types of adult diabetic patients. As this system restricts to select carbohydrate rich food items, which are prime requirement for growth of children, so output of system is not desirable to feed children. The system is under revision so that it can give proper diet for children diabetic patients also.

CBR presents an essential technology of building intelligent systems for medical diagnoses that can aid significantly in improving the decision making of the physicians. The DietMaster can serve as a starting point in developing advanced applications in diet planning. The first system prototype serves as proof of concept that CBR can help provide decision support for planning the diet.

References

1. Aamodt, A., Plaza, E. Case Based Reasoning: Foundational Issues, Methodological Variations and system Approaches*; AI Communications*, IOS Press, Vol. 7: pp. 39-59, 1994.
2. Akerkar, R. and Sajja, P. *Knowledge Based Systems*, Jones and Bartlett, 2009.

3. Bareiss, R. PROTOS : A unified approach to concept representation, classification and learning. Ph.D Dissertation, University of Texas at Austin, Dep. of Comp. Sci. 1988. Technical Report AI88-83, 1988.
4. Bichindaritz, I. MNAOMIA, Improving case-based reasoning for an application in psychiatry In Artificial Intelligence in Medicine, Applications of Current Technologies, Stanford, CA, Working Notes of the AAAI 96 Spring Symposium, 1996.
5. Davis, G.; Wiratunga, N.; Taylor, B.; and Craw, S. Matching smarthouse technology to needs of the elderly and disabled. In *Workshop on CBR in the Health Sciences*, 29–36. ICCBR'03, 2003.
6. Hammond, K. CHEF: A model of case-based planning; Proceedings of AAAI-86. Morgan Kaufmann, pp 267-271, 1986.
7. Jamsandekar, P., Akerkar, R. Dietary Planner, A Case Based Reasoning System; *In proceedings of Recent Trends in IT, A National Conference*, Amaravati, India, 2000.
8. Kolodner, J. *Case based Reasoning*: Morgan Kauffmann Publisher; San Mateo, CA, 1993.
9. Leake, D.: Combining Rules and Cases to Learn Case Adaptation; In proceedings of the 17th annual conference of cognitive science society Hillsdale (NJ): Lawrence Erlbaum, 1995.
10. Marling, C., and Whitehouse, P. Case-based reasoning in the care of alzheimer's disease patients. In *Case-Based Research and Development*, 702–715. ICCBR'01, 2001.
11. Dietary Guidelines for Indians – A Manual, National Institute of Nutrition, ICMR, Hydrabad.
12. Dietary tips for the elderly, National Institute of Nutrition, ICMR, Hydrabad
13. Nutritive Values of Indian Foods, National Institute of Nutrition, ICMR, Hydrabad
14. Nutri-calc : Web page http://www.nutri-calc.com
15. Nutribase : Web page http://www.nutribase.com
16. Nconcepts : Web page http://www.nconcepts.com