Case Based Session Modeling and Personalization in a Travel Advisory System

Bora Arslan and Francesco Ricci

eCommerce and Tourism Research Laboratory
ITC-irst
via Sommarive 18
38050 Povo, Italy
{arslan,ricci}@itc.it

Abstract. Knowledge intensive session modeling and mixed initiative recommendation are introduced in the CBR framework. The advantages of this approach, with respect to traditional web logs plus Data Mining, are shown. On-line, dynamic identification of in-context past similar sessions is exploited for building personalized recommendations. Session components, which are hierarchically structured, are selectively exploited in the system range of recommendation functions. The application of these ideas to an interactive, case-based travel recommender system, that guides European travelers for their travel decision making processes, is described.

1 Introduction

Recommender systems usually extract the knowledge required for a truly personalized advise, logging each single page visited and then analysing this data with Data Mining techniques ([12, 5]). This usually produces large data bases, which are hard to manage and where it is difficult to single out an expressive representation of what the user was trying to do, in which context, and what finally achieved. Especially when dealing with a complex decision process, such as travel planning, it is clear that, to model in a truly reusable fashion the user session, a knowledge-intensive analysis is required [10, 9] before logging user activity. Therefore, an accurate session model is pivotal, and we propose to attain it in a Case-Based Reasoning (CBR) framework. CBR is a cognitively appropriate approach for modeling and explaining human problem solving that has been gaining popularity in recommender systems [3, 6, 8].

The motivation of our research is a travel advisory system, called Dietorecs[1]. The objective of Dietorecs is to guide the user to build his personalized travels, i.e. aggregation (bundling) of a set of travel related products/services, here called travel items. In this content, we remark the separation of two types of objects to

[1] This work has been partially funded by CARITRO foundation (under contract "eCommerce e Turismo") and by the European Union's Fifth RTD Framework Programme (under contract DIETORECS IST-2000-29474).

be recommended in this domain: single travel items, and complete travel bags. A travel item is a basic travel asset like a hotel to stay, a destination to visit, an activity to perform, etc., and a travel bag is a bundle (aggregation) of such travel items. Recommending single travel items, and letting the user to build his own travel bag in an iterative way (e.g. first selecting a destination and adding it to the travel bag, then a hotel, then activities, and so on) is just one face of the Dietorecs system. Dietorecs also provides complete travel recommendations (travel bags), which makes it special among existing systems, none of which can support the user in building a complete itinerary. The idea of recommending complex products, that are made up of individual parts, has been investigated by [18], where they follow a hierarchical problem solving methodology: decompose the given complex problem into sub-problems recursively till the sub-problems can be solved easily. Our methodology is different from this, we rather follow a mixed initiative approach, i.e. stressing the dynamic integration of human and machine problem solving (see Section 2). Moreover the hierarchical travel bag representation is mostly exploited for implementing different similarity measures, according to the abstraction level. These measure are selectively used in particular stages of the problem solving process, e.g. identification of relevant past sessions (more abstract) vs. item scoring (less abstract).

This paper describes our approach to the problem of supporting a user in the task of composing a leisure travel, i.e. bundling elementary travel products/services. We show how a deep structured model of the recommendation session can be incrementally built and exploited to cope with user needs. We are here mostly focused on the model of the session and its usage within a case-based recommendation framework. The approach here described, is the extension of another Case-Based Recommendation system, the Intelligent Travel Recommender System (ITR), which is focused on single item recommendation and interactive query management [14].

The rest of the paper is organized as follow. Section 2 illustrates the basic CBR problem solving cycle and points out the changes and extensions we propose to tackle with the specific problems raised by the single item and complete travel recommendation functions. The model of the recommendation session is explained in Section 3. We talk about the session similarity computations in Section 4 and we list some future work in Section 5 where we conclude the discussion.

2 CBR Learning Cycle

Case-Based Recommendation systems implement a continuous learning cycle, where the aim is to improve system behaviour by using the lessons learned from past experiences. In [1], this learning cycle is structured in five steps:

- 1. *Retrieve:* Given a problem, retrieve a set of stored cases (e.g. problem-solution couples).
- 2. *Reuse:* Apply one or more solutions from these retrieved cases, perhaps by combining them with each other or with other knowledge sources.

– 3. *Revise:* Adapt the retrieved solution(s), as needed, in an attempt to solve the new problem.
– 4. *Review:* Evaluate the outcome(s) when applying the constructed solution to the current problem. If the outcome is not acceptable, then the solution will require further revision.
– 5. *Retain:* Consider adding the new learned case (a new problem + solution couple) to the case base[2]

The left part of Figure 1, e.g. (a), covers the mentioned classical CBR cycle, plus an extra step "6. Iterate" that is particular to our approach. Those that are shown in boxes are the points where we introduce some changes to the classical framework, each of which will be detailed throughout this paper. The separation between the left and right parts of the figure (e.g. (a) and (b)) underlines one of the main differences of the Dietorecs system from the basic CBR cycle. In Dietorecs, the case base contains structured cases, i.e. user driven

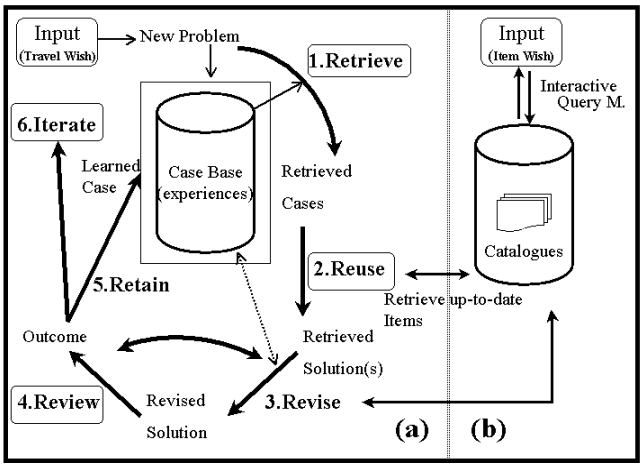

Fig. 1. CBR Learning Cycle

recommendation sessions, whose rather complex hierarchical structure will be described in the following Section. The travel items selected in a recommendation session, i.e. the items that forms the travel bag are actually pointers to items in catalogues (products data base). The case base provides information about good bundling of products and is therefore used for learning this knowledge and for ranking items selected in the catalogues. The catalogues are exploited for obtaining up-to-date information about currently available services.

[2] The author in [1] uses "<problem, solution, outcome> triples" instead of problem-solution couples, and "Case Library" instead of Case Base. But, for the sake of consistency with the rest of the paper and to avoid any ambiguity we will follow the postulated terminology above.

Dietorecs provides two main types of recommendations: *Single Travel Item* and *Complete Travel*, which are going to be described below.

2.1 Single Travel Item Selection

The single travel item recommendation function assists the user in the sequential identification of travel products/services (e.g. accommodation or attractions) that best matches to his needs (wishes).

The user first provides his general needs and constraints for the complete travel in his mind ("Travel Wish"). For example, he may be interested in a one week travel in March, with a limited budget, suitable for a young couple. The system accepts this input as the (partially defined) problem specification, and filters out from the case base those cases that do not match these wishes.

Then the system allows the user to specify his wishes about the specific item he is looking for. Let us say that the user is currently looking for a hotel, and specifies his wishes about the hotel item. For example, a hotel near to the city center, with a half-board catering service, having TV in the room, etc. This is a second level of Input, which we call Item Wish in Figure 1.

Having both types of inputs, the system retrieves from the case base a set of cases (i.e recommendation sessions) that are most similar to the current one (step "Retrieve" in Figure 1). This collection, called Reference Set, is dynamically computed, i.e. each time a recommendation function is called, a new contextual Reference Set is retrieved, matching all the user wishes accumulated at that stage of the recommendation. We will describe briefly the computation of the Reference Set in Section 4.

Single item recommendations are retrieved from the catalogue(s). To select the items that best match to the user's wishes, the system gives personalized weights to the item features. For instance, if the system understands that Air Conditioning (AC) is an important feature for this user (although he has not specified), it will recommend hotels with AC, that are selected among those matching the wishes explicitly mentioned by the user [13]. Such a knowledge is gathered from the Reference Set, and this is the main point where we exploit the experienced recommendation sessions for this type of recommendations.

At this stage, the user can assess the recommendations, modify his wishes, and ask the system for an other list of recommendations. This corresponds to the "Revise" and "Review" steps in Figure 1. In case of failure retrieving items from the catalogues, that is, if there are too many or no items matching to the user's wishes, the system suggests the user useful changes to the query conditions (relaxation or tightening) [15]). After accepting one (or more) recommendation and hence adding it to his travel bag, the user may either decide that the travel bag is completed and stop the session, or go ahead for other items. If the user terminates the interaction at this stage, the whole recommendation session is added to the case base as a learned case (i.e. the step "Retain" in Figure 1). If the user goes for another type of item, which is the "Iterate" step, then new item wishes are acquired, a new Reference Set is computed and new items are suggested.

2.2 Complete Travel Selection

The complete travel recommendation function provides the user with an already aggregated set of travel items. Here the user may specify both travel wishes and item wishes. In response, the system generates the Reference Set as above in the *Single Travel Item* recommendation, but now a selection of these cases is shown to the user as travel templates. The cases in the Reference Set are used to generate bundles of travel items appropriate to the user. Whenever the user shows a propensity to one of these templates, the system updates the items in that template. Simply, it goes to the catalogue, finds the most similar items to those in the template, gets the up-to-date items[3], adapts the template and recommends it. Therefore Dietorecs, implements the "Reuse" and "Revise" steps in Figure 1 exploiting both the case base and the catalogue. So, we do reuse previous cases but in a different fashion, mainly to extract from them typical bundling of services and to drive the search in the catalogues.

Dietorecs users are always allowed to reconfigure the recommended bundles. The user may replace, add or remove one (or more) of the items in the recommended travel. For example, the user can keep the destination item, say Trento, and ask for hotel and activity items different from those recommended, but well suit to the selected destination. This refers to the "Review" step in Figure 1. In this view, we manage the Review step in collaboration with the user, in a mixed-initiative process. When the user finally accepts the outcome and stops modifying the travel bag, this is stored in the case memory. That is, a complex (and fuzzy) problem and its solution to be used for further recommendations. We stress that in the course of a single session the CBR cycle is potentially iterated ("Iterate" step in Figure 1). This happens each time a new item selection or complete travel recommendation is initiated by the user.

In this view, the "problem" part of a case becomes a cloudy and dynamic concept[4], in the sense that, at each iteration, the item(s) already selected (that are already added to the travel bag) becomes a part of the problem. In fact, new recommendations should not only match the user needs, but also satisfy some compatibility with the already selected items. For example, if the user has already selected a hotel in Innsbruck-Austria, recommending an activity in Trento-Italy may not be rational[5].

3 Hierarchical Structure and The Case Model

The case is modeled in a hierarchical tree of components and sub-components. A case is decomposed into: travel wishes, travel bag, user, navigation history and reward. Figure 2 illustrates an example of a case.

[3] Clearly, if the items in the template are still available, the similarity will pick the same ones from the catalogue

[4] Similar ideas have been discussed in [4].

[5] Actually, it is quite likely that such a recommendation can be meaningful for some tourists coming from long distances. Learning such a knowledge is another objective for us, which may add outstanding values to personalization task.

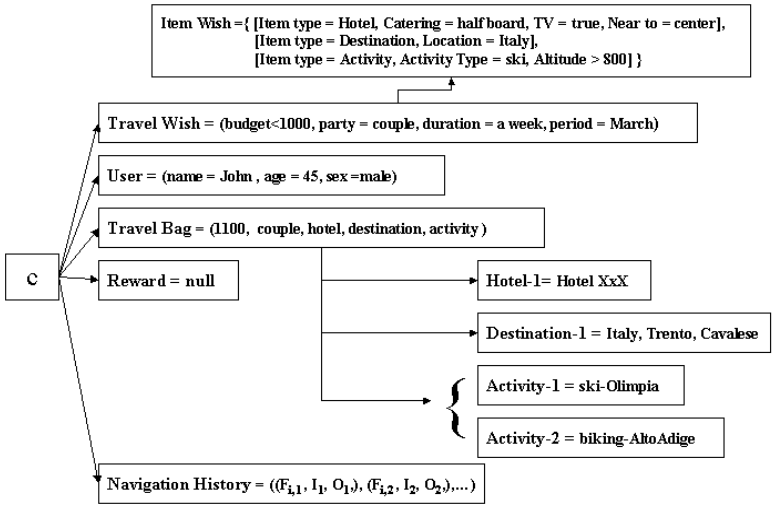

Fig. 2. An Example of a Case

- **Case** (c) A case $c = (tw, tb, u, nh, r)$ model a complete recommendation session, where tw (travel wish), tb (travel bag), u (user), nh (navigation history), and r (reward) are its sub-components.
- **User** (u) This block carries the user-specific data for registered users, and is modeled as a simple features vector. For non-registered users, all these data will remain null.
- **Travel Wish** (tw) This will carry the user's constraints (and needs) about the travel in his mind. Travel wish consists of constraints over the travel features, that are specified by the user during his interaction with the system.
  - **Item Wish** This will carry more specific data about travel items. While travel wish contains constraints about the whole travel plan, item wish consists of item-specific feature-value pairs provided by the user. For sake of simplicity, we will assume here that there are tree types of items, hotel, destination and activity), mentioned as type 1, type 2 and type 3 respectively.

  Mathematically, we represent the travel wish as $tw = (gw, iw_1, iw_2, iw_3)$, where $gw, iw_1, , iw_2, iw_3$ are all symbolic or numeric valued vectors. For example, in Figure 2, we have:

$gw =$ [budget<1000, party = couple, duration = a week, period = March]
$iw_1 =$ [type = Hotel, catering = half-board, TV = true, near to = center],
$iw_2 =$ [type = Destination, country = Italy],
$iw_3 =$ [type = Activity, ski = true, altitude $>$ 800]

This refers to a travel bag which is suitable for a couple with budget limited to 1000 Euros. More specifically, he is looking for a hotel with half-board

catering service and TV equipment, near to a city center in Italy, where he can practice some ski activity.

- **TravelBag** (tb) Travel Bag is a complex data structure that collects together the items (sub-components) selected during a case session. It is represented in a tree-like hierarchical way, denoted $tb = (gd, d_1, d_2, d_3)$. gd is a vector that describes general properties of the travel bag. For instance, as in Figure 2, gd = (1100, couple, hotel, destination, activity) is a travel bag constructed for a couple, the budget required for this travel is about 1100 Euros, and includes tree types of item. $d_i = [d_{i1}, d_{i2}, \ldots]$, $(1 \leq i \leq 3)$ denotes the items(s) of type i in the bag, $d_{i,j}$ stands for the description of the j^{th} item of type i in the travel bag.
  Hotel XxX is the pointer to a hotel item in the catalogue, possibly described like $d_{11} = (swimmingpool = true, catering = halfboard, location = Trentino - Italy, TV = true, sofa = true, nearto = center \ldots)$. In the same travel bag, the user has two items of type activity: ski-Olimpia and biking-Alto Adige, respectively having descriptions d_{31} and d_{32} (e.g. $d_{31} = (activitytype = ski, difficultylevel = high, location = Cavalese, altitude > 1000 \ldots)$)
- **Navigation History** (nh) This block stands for tracking the user's clicking behavior. Since it is out of the scope of this paper, we do not detail here our approach to users' click behavior tracking. But, shortly we need to say that nh is more than standard user logs. We model each session stage (i.e. status-quo between two function calls) as a triple: (F,I,O), where F is the function called (e.g. search, find similar item, etc.), I is a representation of the input of the function (e.g. wishes specified by the user, description of an item, etc.), and O models the output (I and O are lists of feature/value pairs). Basically, nh is a sequence of such triples.
- **Reward** (r) Reward is a sequence of rates provided by the user for each item in the travel bag. More precisely $r = (gr, [r_{11}, r_{12}, \ldots], [r_{21}, r_{22}, \ldots], \ldots)$, where r_{11} is the rate for the first item of type 1, r_{12} is the rate for the second item of type 1, r_{21} is the rate for the first item of type 2, etc. gr is the general rate for the whole travel bag computed as a composite average of rates of items.

4 Reference Set and Similarity

To retrieve the Reference Set we compute the similarity between the current case and those in the case base. This similarity-based retrieval is applied after a logical (context dependent) filter has been enforced. For instance if the destination has been provided as input, only cases in the selected destination will be included in the reference set (see [13]).

Given two cases $c = (tw, tb, u, s, r)$ and $c' = (tw', tb', u', s', r')$, the case similarity is computed as follow:

$$Sim(c, c') = \frac{1}{5}(w_w Sim(tw, tw') + w_b Sim(tb, tb')$$

$$+w_{wb}Sim(tw, tb') + w_u Sim(u, u') + w_r Rew(r')) \quad (1)$$

where $w_w, w_b, w_{wb}, w_u, w_r$ (real numbers between 0 and 1) are the weights that are used to balance the relative components importance in case similarity computation[6].

Note that the "cross" similarity between the travel wishes of c and the travel bag of c' is typically used when a new case c is going to be built. At that point the system must match the wishes of c with the bag of c', not only the user wishes that were at the base of c'. This is motivated by the observation that there is always a mismatch between the explicit user wishes and the selected items. This is due both to psycological inconsistencies between desires and real actions, and the approximation brought by the CBR problem solving approach. Moreover, the term $Rew(r')$ does not represent any comparison between cases, but takes into account the importance of the retrieved case. The motivation here is to consider first those highly rated (i.e. successful) travel bag, so having a better rate makes that case more important. In this way, the Equation 1 makes more similar a case that have been rated highly by the user that built the case.

For each of the sub-similarities in Equation 1, a different similarity measure is needed. Detailed description of similarity computations can be found in [13], but here we want to mention that the sub-similarity components in Equation 1 (e.g. $Sim(tw, tw')$) point out a basic problem in similarity computation: comparison of two non-trivial structures made of sub-components ([2, 7]). If we consider, for the sake of the simplicity, tw and tw' to be two subsets $tw = \{w_1, \ldots, w_M\}$ and $tw' = \{w'_1, \ldots, w'_N\}$ of a space W (space of wishes, i.e. constraints over feature spaces), and we assume there is a definition of similarity in W, i.e. $Sim(w, w')$ is a known positive real number between 0 and 1, for all $w, w' \in W$, then we define two auxiliary similarities:

$$Sim_l(tw, tw') = \frac{1}{M} \sum_{i=1}^{M} \max_{j=1}^{N} Sim(w_i, w'_j),$$

$$Sim_r(tw, tw') = \frac{1}{N} \sum_{j=1}^{N} \max_{i=1}^{M} Sim(w_i, w'_j)$$

where, for instance, $Sim_l(tw, tw')$ gives the average similarity of the elements in tw when they are matched with the best element in tw', i.e., for each w_i is considered the element in tw' that gives maximal similarity. The the final similarity is computed as follows:

$$Sim(tw, tw') = \min\{Sim_l(tw, tw'), Sim_r(tw, tw')\}$$

This similarity function share many advantages of other more complex metrics based on sub-graph isomorphism problem solution (see e.g. [16]) but reduces sensibly the computation cost.

[6] Equation 1 is generated with the spirit of the principle "two objects are similar if they contain similar objects" [11], which is a quite customized version of the integration function proposed in [11].

5 Future Works and Conclusions

This paper presents a research project that addresses a fundamental issues in CBR recommendation, i.e. knowledge intensive modeling of recommendation sessions, and its usage in a mixed-initiative computer supported travel planning system. The system here described (Dietorecs) extends ITR [15, 14], a recommender system based on a simpler case structure and supporting only single item selection. Dietorecs is at the system architecture development stage and the functions here described have been defined in collaboration with experts of consumers behaviour [10, 9].

We had no chance to explain here all the aspects, and details of our approach. We have simplified the description of the item selection process omitting, for instance, the interactive query management issues (inherited from [15]). If there is a failure in the user query (too many or no results), the system suggests alternative query modifications for tightening or relaxing the query using statistics computed over the catalogues (see [7] for a similar approach). In fact, we are planning to extend this approach and personalize these query refinement suggestions using the introduced session model, and deriving wishes' relative importance from statistics computed on line on the reference set.

Also, as we mentioned earlier, the user input is gathered in two abstraction levels: Travel Wish and Item wish. Sequencing of the acquisition of this user input, that is asking the most important features for the user based on previously gathered input, is another subject for us to learn.

One shortcoming of similarity based recommendations is the lack of the diversity of recommendations ([17]). If all the recommendations delivered to the user are quite alike, then if the user does not like one of the recommendations he may not like any of them. To remedy this problem, we are investigating one approach based on the partitioning of the case base with respect to some predefined features (e.g. cost or travel type). Having some predefined clusters of cases, the Reference Set can be formed by retrieving the most similar case from each of these clusters [13]). The central issues is to learn the level of this diversity: how much similar and how much apart the recommendations should be? For all these purposes we believe that the session model we described here will provide a valuable source of knowledge.

References

[1] David W. Aha. The omnipresence of case-based reasoning in science and application. *Knowledge-Based Systems*, 11(5-6):261–273, 1998.

[2] Ralph Bergmann and Armin Stahl. Similarity measures for object-oriented case representations. In B. Smyth and P. Cunningham, editors, *Advances in Case-Based Reasoning*, volume 1488 of *Lecture Notes in Computer Science*, pages 25–36. Springer, 1998.

[3] Robin Burke. *Encyclopedia of Library and Information Science*, volume 69, chapter Knowledge-based Recommender Systems. Marcel Dekker, 2000.

[4] H.-D. Burkhard and P. Pirk. Technical diagnosis: Fallexperte-d. In *System Development and Evaluation*, 4th German Workshop on CBR, Berlin, Humboldt University, 1996.

[5] R. Cooley. *Web Usage Mining: Discovery and Application of Interesting Patterns from Web Data*. PhD thesis, University of Minnesota, 2000.

[6] Pdraig Cunningham, Ralph Bergmann, Sascha Schmitt, Ralph Traphner, Sean Breen, and Barry Smyth. Websell: Intelligent sales assistants for the world wide web. In R. Weber and C.G.von Wangenheim, editors, *Procs. of the Workshop Programme at the Fourth International Conference on Case-Based Reasoning*, 2001.

[7] Michelle Doyle and Pádraig Cunningham. A dynamic approach to reducing dialog in on-line decision guides. In *Advances in case-based reasoning: 5th European workshop, EWCBR-2000, Trento, Italy, September 6–9, 2000: proceedings*. Springer, 2000.

[8] Mehmet H. Göker and Cyntia A. Thomson. Personalized conversational case-based recommendation. In *Advances in case-based reasoning: 5th European workshop, EWCBR-2000, Trento, Italy, September 6–9, 2000: proceedings*, pages 99–111. Springer, 2000.

[9] K. Grabler and A.H. Zins. Vacation trip decision styles as basis for an automated recommendation system: Lessons from observational studies. In *Proceedings of the ENTER 2002 Conference*, Innsbruck, Austria, January 22-25 2002. Springer Verlag.

[10] Y.-H. Hwang, U. Gretzel, and D. R. Fesenmaier. Behavioral foundations for human-centric travel decision-aid systems. In *Proceedings of the ENTER 2002 Conference*, Innsbruck, Austria, January 22-25 2002. Springer Verlag.

[11] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. Technical report, Stanford University, October 2001.

[12] Yesha Y. Krishnapuram R. Joshi K.P., Joshi A. Warehousing and mining web logs. In *Proceedings of the ACM CIKM Workshop on Web Information and Data Management*, pages 63–68, WIDM'99, 1999.

[13] Francesco Ricci, Bora Arslan, Nader Mirzadeh, and Adriano Venturini. Cbr methodologies in dietorecs. Technical Report D4.1, DieToRecs IST-2000-29474, 2002.

[14] Francesco Ricci, Dennis Blaas, Nader Mirzadeh, Adriano Venturini, and Hannes Werthner. Intelligent query management for travel products selection. In *ENTER 2002 Conference*, Innsbruck, Austria, January 22-25 2002.

[15] Francesco Ricci, Nader Mirzadeh, and Adriano Venturini. Intelligent query managment in a mediator architecture. In *IEEE International Symposium "Intelligent Systems'*, Sunny Day, Bulgaria, September 10-12 2002.

[16] Francesco Ricci and Luca Senter. Structured cases, trees and efficient retrieval. In B. Smyth and P. Cunningham, editors, *Advances in Case-Based Reasoning*, volume 1488 of *Lecture Notes in Computer Science*, pages 88–99. Springer Verlag, 1998.

[17] B. Smyth and P. McClave. Similarity vs diversity. In *Proceedings of the 4th International Conference on Case-Based Reasoning*, Springer-Verlag, 2001.

[18] Armin Stahl and Ralph Bergman. Applying recursive cbr for the customization of structured products in an electronic shop. In *Advances in case-based reasoning: 5th European workshop, EWCBR-2000, Trento, Italy, September 6–9, 2000: proceedings*, pages 297–308. Springer, 2000.