Understanding Dubious Future Problems

Oğuz Mülâyim and Josep Lluís Arcos

IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Spain
{oguz,arcos}@iiia.csic.es

Abstract. Being able to predict the performance of a Case-Based Reasoning(CBR) system against a set of future problems would provide invaluable information for design and maintenance of the system. Thus, we could carry out the needed design changes and maintenance tasks to improve future performance in a proactive fashion. This paper proposes a novel method for identifying regions in a case base where the system gives low confidence solutions to possible future problems. Experimentation is provided for RoboSoccer domain and we argue how encountered regions of dubiosity help us to analyse our case base and the reasoning mechanisms of a CBR system.

1 Introduction

When we use Case-Based Reasoning (CBR) [1] for solving problems, we count on the main assumption underlying this methodology[2], viz. *similar problems have similar solutions*. Wouldn't it be nice if we could anticipate to what extent the CBR assumption holds for future problems? A positive feedback in this direction would increase the reliability of the system. Contrariwise, we would be aware of the need for carrying out the required design and maintenance tasks throughout our system to improve its future performance in a proactive fashion [3].

Indeed, this preanalysis would give us important clues about the future. For instance, we could discover deserted regions in our case base (CB) where we do not have available cases to reason with, or we could encounter overcrowded zones in which we would have difficulty to classify our problem among cases of diverse classes.

Furthermore, this analysis would not only yield predictions about the case base but it could also give us valuable insight about the reasoner itself helping us to verify the functioning of CBR mechanisms like retrieval and reuse in advance.

The question, of course, is how this preanalysis could be performed. Case-Base Maintenance (CBM) techniques proved useful for improving the performance of CBR systems. Most of the existing CBM techniques focus on removing redundant or erroneous cases while preserving the system's competence [4–6]. More recent research introduces a complexity measure for highlighting areas of uncertainty within the problem space [7]. The common assumption of these techniques is that analysis of the cases provided in the case base is a good approach for estimating the performance of the system for future cases. This assumption is known as the representativeness assumption. Nevertheless, new problems are expected to be slightly different from the existing cases.

Thus, the possibility of systematically assessing the performance of a system in a set of problems different from the existing cases becomes an interesting issue. One way for such an assessment is confronting the CBR system with possible future problems to detect deficiences beforehand. Though the idea sounds intuitive, the task of finding possible future problems that lead to system deficiencies is far from being trivial for most of the domains where the problem space is too vast or even infinite depending on the features that characterise a domain.

A common approach to attacking such a vast space is to use heuristics that guide the search. We believe that confidence measures can be used as effective heuristics to find system deficiences as they state how sure the system is about the solution it proposes for a given problem (where a solution with a low confidence indicates an inaccurate solution). The importance of the availability of such a measure is emphasised in recent research introducing possible confidence indicators and calculations for a CBR system [8–10].

In preliminary work [11] we have proposed a method inspired on evolutionary techniques to detect problematic future problems in terms of confidence. We call these future problems with low confidence solutions *Dubious Future Problems* (DFPs). In this paper we extend the previous work for detecting and characterising dubious regions in the problem space.

To effectively scan the problem space for finding dubious regions, we propose a method based on four steps: First, we explore the problem space to find dubious future problems. Then, we carry out an exploitation phase to better identify these problems by focusing the search on additional future problems in their neighborhoods. Next, to help the understanding of the regions where dubious future problems are located, we associate each DFP with a neighborhood pattern (e.g. hole, border). Finally, to focus on regions in the case base that suffer from the same deficiency rather than dealing with individual problems, we group DFPs according to these patterns.

In Section 2 we summarize our evolutionary approach for scanning the problem space to find dubious future problems. The definitions of dubiosity patterns and the grouping algorithm are described in Section 3. In Section 4 we give an example of how to explore dubious future problems and how to group them by patterns that they exhibit in a Robosoccer system. We interpret the results showing how they helped us to analyse our CBR system. We finally conclude discussing the outcomes of the methodology introduced in this paper and giving directions for future work in Section 5.

2 Exploring Dubious Future Problems

Given a domain ontology associated with a CBR system, we are interested in identifying possible future problems that: 1) are similar enough to the current cases and, 2) that the confidence on their solutions provided by the CBR system is low. Thus, the exploration of the problem space to find DFPs requires only three knowledge components in a CBR system: a domain ontology (specifying at least the features and their data types used for defining cases); a similarity metric; and a confidence measure that attaches a confidence value to each solution proposed by the CBR system.

In the search for dubious problems, the search space is the space of all problems that can be generated according to the domain ontology. As indicated above, this space can be too vast or even infinite depending on the features that characterise the domain. To find DFPs we use Genetic Algorithms (GA) as they have demonstrated their capabilities for exploring such vast search spaces. They have the advantage of scanning the search space in a parallel manner using a fitness function as heuristics and their implementations can be domain independent.

With a diverse initial population of possible future problems and an appropriate fitness function, DFPs will evolve as the GA runs, where the less confident the CBR system is about a problem's solution the more it will prefer to regard that problem as a DFP. However, as commonly seen in practice, GAs might have a tendency to converge towards local optima [12]. In our case, this would result as getting stuck to a low confidence zone and generating problems only within that locality instead of scanning a wider region in the problem space. In many GAs, mutation is the trusted genetic operator to avoid this problem as it introduces diversity to the population, nevertheless it is usually not a guarantee.

Our approach to effectively search the problem space and to avoid local minima has been to divide the search into two steps, namely *Exploration* and *Exploitation* of dubious future problems. In the Exploration step, the aim is to find DFPs which are similar enough to existing cases and which are as dissimilar as they could be to each other. The similarity to existing cases argument is to avoid dealing with irrelevant(although possibly not unlikely) problems which have no neighbour cases in the CB. The confidence for a solution to a generated problem which has no similar neighbours would probably be very low, but since this would already be an expected result, it would not be of much interest to bring these problems to the expert's inspection. Additionally, the dissimilarity between DFPs is for the sake of obtaining diversity in the results of Exploration to achieve a richer set of future problems and their neighbours after the Exploitation step.

Successively, in the Exploitation step our objective is to find future neighbours of the DFPs encountered in the Exploration step for providing a more precise analysis of the low confidence local regions.

Both, Exploration and Exploitation steps, incorporate two proximity limits in terms of similarity to an existing case or a future problem. These limits define the preferred region in the problem space during the search for DFPs and their neighbours. We will explain both limits in detail for each step in the next subsections. We also added a *Diversity Preservation* feature to our GAs for both steps to keep the population's diversity at a desired level.

The following sub-sections describe the details of the Exploration and Exploitation steps.

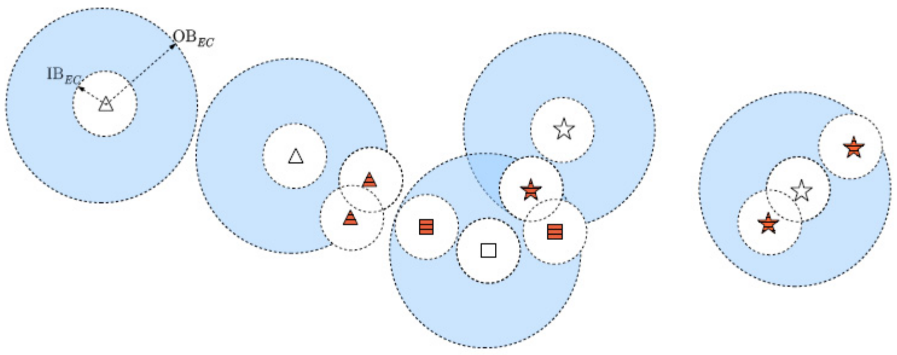

Fig. 1. Graphical representation of the Exploration step. Hollow shapes are existing cases (where different shapes refer to different classes); filled shapes are the encountered Dubious Future Problems; IB_{EC} and OB_{EC} are, respectively, inner and outer bounds.

2.1 Exploration

The goal of the Exploration step is to identify an initial set of dubious problems similar enough to the cases defined in a case base. A problem is considered dubious when its confidence is lower than a given threshold. Since the minimum value for considering a solution as confident may vary in each CBR application, the decision about the confidence threshold is domain dependent.

For the Exploration step, the proximity limits mentioned above define the preferred region of the search for dubious problems. The outer limit OB_{EC} defines the border for the less similar problems, while the inner limit IB_{EC} defines the border for the most similar ones to an existing case in the CB. We also use the inner limit to draw a border around the found DFPs since we are looking for DFPs that are as diverse as possible in this step. A graphical representation of the Exploration step is provided in Figure 1.

The decision of the proximity limits depends on the answer of how similar a problem can be to a case to be regarded as a relevant problem for the domain and application. The similarity among existing cases may give an idea of the range of possible values for these limits. For example; if these two limits are chosen so that their sum is closer to the similarity value between two nearest cases of different classes, then preffered proximities will overlap thus giving us the possibility to discover borders for the classes in the CB.

Throughout the execution of the GA for Exploration, we maintain a list of encountered future problems with low confidence solutions $\mathcal{LCFP}$. During the evaluation of a population, each time we come across a chromosome representing a dubious problem we add it to the $\mathcal{LCFP}$ list.

The concepts used in the GA for the Exploration step are explained below:

Chromosomes: Each chromosome in our population represents a future problem where each gene is a feature of the problem. The value of a gene is thus one of the possible values for the associated feature.

Initial Population: The initial population is formed by chromosomes generated by the *Random-Problem-Generator* function (RPG). RPG is a function able to generate a new problem by assigning random values for each problem feature. Values for problem features can be easily generated using the definitions of features in the domain ontology (feature definitions explicitly state the data type and the set of possible values for a feature). It should also be considered that in the existence of domain constraints, the Random-Problem-Generator function generates valid problems that conform to those constraints. Otherwise, generated future problems might be non-valid or irrelevant in the domain. The size of the population directly depends on the vastness of the problem space of the CB that is being worked on.

Fitness Function: The fitness of a chromosome is determined by two parameters: the confidence value of the solution to the problem represented by the chromosome and the similarity of the problem to the nearest problem in the CB. The fitness function has to be adapted in each different domain or CBR system. However, the following guidelines should be used in Exploration regardless of the domain or the application:

– The lower the confidence value is for a chromosome, the better candidate is that chromosome.
– A chromosome in the preferred proximity of an existing case is a better candidate than a chromosome which is not in this proximity.
– The confidence factor of the fitness is more significant than the similarity factor. This is not surprising since we are searching for dubious problems.

Our proposal for the fitness function definition is the following:

$$Fitness(c) = Confidence(c)^2 \times SimilarityFactor(c)$$

where c is the chromosome to be evaluated; $Confidence$ returns the confidence value supplied by the CBR application after solving c; and $SimilarityFactor$ takes into account the similarity to both cases and DFPs. $SimilarityFactor$ is calculated as follows:

$$SimilarityFactor(c) = partSimEC(c) + partSimDFP(c)$$

where $partSimEC$ refers to the similarity of c to existing cases and $partSimDFP$ refers to the similarity of c to DFPs in $\mathcal{LCFP}$. $partSimEC$ is defined as:

$$partSimEC(c) = \begin{cases} 1 - (OB_{EC} + IB_{EC} - Sim(c, CB)) & \text{if } Sim(c, CB) \geq IB_{EC} \\ 1 - Sim(c, CB) & \text{otherwise} \end{cases}$$

where $Sim(c, CB)$ is the similarity value of c to the most similar case in the CB (i.e. the highest similarity); IB_{EC} and OB_{EC} are, respectively, the inner and outer bounds of similarity to the existing cases. $partSimDFP(c)$ is defined as:

$$partSimDFP(c) = \sum_{p \in FP} (similarity(c,p) - IB_{EC})$$

where $FP \subset \mathcal{LCFP}$ is the set of future problems to which c is more similar than the allowed value IB_{EC} and $similarity(c,p)$ is the similarity value of c to the problem p.

Following the previously defined guidelines, $SimilarityFactor$ penalizes the chromosomes that are too close to either cases or future problems discovered in previous iterations (i.e. inside the radius defined by the inner threshold).

It should also be noted that for a desired chromosome (i.e. representing a dubious future problem which is in the preferred proximity of an existing case) our proposed function produces a fitness value which is lower than a non-desired one.

Selection: We defined a fitness-proportionate selection method. Fitness-proportionate selection is a commonly used and well studied selection mechanism where each chromosome has a chance proportional to its fitness value to be selected as a survivor and/or parent for the next generations. However, since we are interested in chromosomes with lower fitness values as explained above, to comply with our fitness function, selection of a chromosome was inversely proportional to its fitness value.

Crossover: We use single-point crossover as it is simple enough and widely used. Depending on the observed convergence of the GA, this method could easily be replaced by Two-Point or n-Point crossover methods.

Mutation: Generally, one random gene value is altered for a number of offspring chromosomes in the population. If a local minima problem is observed, more genes and/or more chromosomes can be mutated.

Diversity Preservation: We decided to use a diversity threshold that can be tuned for each application. Specifically, at each generation when the number of twins exceeds the diversity threshold, they are removed probabilistically using as probability their fitness value (i.e. twins with higher fitness have a higher probability to be deleted).

In our approach, the validity of a problem is another important issue. Due to the application of genetic operators in the evolution cycle, they are likely to reproduce offspring chromosomes which are non-valid. We may deal with these chromosomes basically in two ways: we may replace them with new valid chromosomes or we may let some of them survive hoping them to produce nice offspring in the following generations. In the former option, the replacement can be done in the Diversity Preservation. In the latter option, either a validity check can be incorporated into the fitness function reducing the fitness of non-valid chromosomes or simply non-valid chromosomes can be excluded from the $\mathcal{LCFP}$ after the termination of the Exploration step. In the current implementation we adopted this last solution.

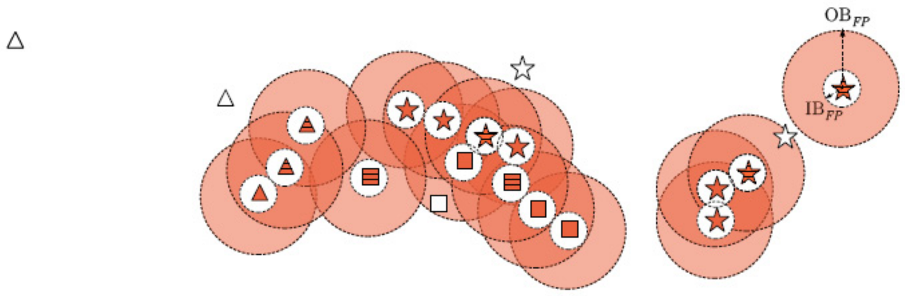

Fig. 2. Graphical representation of the Exploitation step. Hollow shapes are existing cases; filled shapes are the encountered and exploited Dubious Future Problems. IB_{FP} and OB_{FP} are, respectively, inner and outer bounds.

Termination: The termination criterion for the GA can be reaching a number of generations or a number of dubious future problems. We let the population evolve for a certain number of generations.

Result: As the result of the GA we obtain the list of future problems with low confidence solutions $\mathcal{LCFP}$.

2.2 Exploitation

The goal of the Exploitation step is to explore the neighbourhood of the low-confidence problems discovered in the Exploration step. Similarly to the Exploration step, during the execution of the GA for the Exploitation step we maintain a list of Low Confidence Problem Neighbours $\mathcal{LCPN}$. We initialise this list with the members of the $\mathcal{LCFP}$. In other words, the members of this list are the dubious future problems that we want to exploit.

For the Exploitation phase, the proximity limits define the preferred region of the search for neighbour problems. The outer limit OB_{FP} defines the border for the less similar problems, while the inner limit IB_{FP} defines the border for the most similar ones to any member of the $\mathcal{LCPN}$. A graphical representation of the Exploitation step is provided in Figure 2. Notice that, comparing with the Exploration step, the proximity limits for Exploitation step are narrower since in this step we are looking for neighbours of the DFPs.

All DFPs satisfying the proximity limits are added to the $\mathcal{LCPN}$ list. The confidence threshold for dubiosity is the same value used in the Exploration.

The concepts used in the GA for the Exploitation step are the following:

Chromosomes, Selection, Crossover, Mutation, Diversity Preservation: These concepts have the same definitions as the corresponding ones previously given in the Exploration step.

Initial Population: We partially feed the initial population with the $\mathcal{LCFP}$ set hoping to reproduce similar problems. We use the Random-Problem-Generator to reach to the desired initial population size when needed.

Fitness Function: The fitness of a chromosome c in the Exploitation step depends only on its neighbourhood to any member of $\mathcal{LCPN}$. The fitness function is defined as follows:

$$Fitness(c) = \begin{cases} 1 - (OB_{FP} + IB_{FP} - Sim(c, \mathcal{LCPN})) & \text{if } Sim(c, \mathcal{LCPN}) \geq IB_{FP} \\ 1 - Sim(c, \mathcal{LCPN}) & \text{otherwise} \end{cases}$$

where $Sim(c, \mathcal{LCPN})$ is the similarity value of c to the most similar problem in $\mathcal{LCPN}$; IB_{FP} and OB_{FP} are, respectively, the inner and outer proximity bounds of similarity to the previously found future problems.

Termination: We let the population evolve for a certain number of generations in Exploitation as well.

Result: At the end, the Exploitation step provides the list $\mathcal{LCPN}$ which contains dubious future problems found both in the Exploration and Exploitation steps.

3 Regions of Dubiosity

Exploration and Exploitation of DFPs give us a foresight of a possible bad performance of the CBR system. To inspect the underlying reasons of such a malfunction, the encountered DFPs may be presented directly for the domain expert's attention. Experts in turn may use this future map of the case base to initiate maintenance tasks if needed. However, depending on their number, analysing DFPs manually may become a difficult task as domain experts would have to check each DFP together with its neighbours to reveal the system deficiencies.

To be able to assist the domain experts in the endeavour of analysing DFPs, we have defined six dubiosity patterns. Each DFP is tagged with a dubiosity pattern which indicates the possible reason of being classified as dubious. Furthermore, when we have a numerous list of DFPs, we propose a grouping algorithm for helping the expert to focus on regions in the case base that suffer from the same deficiency.

3.1 Dubiosity Patterns

DFPs are good pointers to possible future system weaknesses as their solutions have low confidence values. But to identify the cause of the low confidence result, and thus, the needed policies for eliminating these weaknesses, DFPs themselves alone are not much of a help. This is because confidence measures, in general, do

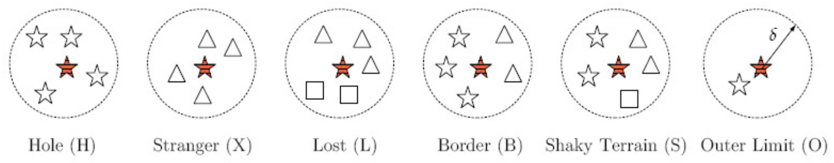

Fig. 3. Graphical representation of DFP Patterns. Hollow shapes are cases and the filled one is a Dubious Future Problem. Each shape represents a different solution class. δ is the similarity threshold delimiting the neighbourhood of a DFP.

not provide detailed explanations of the judgement they make while attaching a confidence value to a solution. Thus, for analysing why DFPs were considered as dubious the expert should inspect the indicators of confidence used by the confidence measure of the CBR system.

On the other hand, since usually the similarity measure plays an important role in confidence calculus[9, 10], looking at the neighbour cases of a DFP would give strong clues for analizing DFPs. For this aim, we have defined six dubiosity patterns according to the solution classes of a DFP and of its neighbour cases. Given a similarity threshold δ defining the neighbourhood, we say that a DFP exhibits a pattern of type (see Figure 3 for a graphical representation):

- **Hole (H)** when all of its neighbour cases are of the same class as the DFP;
- **Stranger (X)** when all of its neighbour cases are of the same class which is different from the DFP's;
- **Lost (L)** when there are at least two different groups of neighbour cases, according to their solution classes, where none of the groups is of the same class as the DFP;
- **Border (B)** when its neighbour cases can be grouped into two groups with different solution classes and one group shares the same class as the DFP;
- **Shaky Terrain (S)** when its neighbour cases can be grouped into at least three groups of different solution classes and one group shares the same class as the DFP. This pattern indicates regions where adding or removing a case might redraw borders for multiple classes.
- **Outer Limit (O)** when it has only one neighbour case sharing the class. This pattern may indicate outer limits for a particular class or it may point out isolated cases in the case base.

After the Exploitation step, for each DFP in $\mathcal{LCPN}$ we check the solution classes of its neighbour cases together with its own and we associate a pattern to each DFP according to the above pattern definitions. The similarity threshold δ value should be coherent to the OB_{EC} value in the Exploration step since we were looking for similarity between DFPs and existing cases. We propose to choose a value slightly bigger than the OB_{EC} value for the δ.

3.2 Grouping Dubious Future Problems

Preliminary experiments for exploring DFPs have shown that depending on the features that characterise a domain and on the CBR inference mechanism we may end up with a lengthy list of explored DFPs. In one sense, a high number of DFPs is attractive since the more DFPs we encounter the more possible future deficiencies we are discovering. However, using this lengthy list to carry out maintenance tasks may turn out to be a tedious task in both manual and automated maintenance of a CBR system. Although the associated dubiosity patterns help us to analyse DFPs, each DFP still requires special attention to identify the needed maintenance tasks.

To overcome this overhead when we have too many DFPs to deal with, we propose to group the DFPs according to their patterns and the similarity of the DFPs among each other. Grouping DFPs in this way makes it easier to identify regions in the problem space that suffer from the same deficiency. Thus, any maintenance task that eliminates a common deficiency in such a region will probably make the CBR system more confident of its solutions for similar future problems that will fall into that region. We call these regions *Regions of Dubiosity*.

Given the list $\mathcal{LCPN}$ and a similarity threshold δ', the grouping algorithm performs the following two steps:

1. **Identification** of the Regions of Dubiosity by transitively grouping all DFPs that are neighbours at similarity δ'. This step forms different isolated regions. Each region is a graph where the nodes are the DFPs and edges connect two nodes when their similarity is, at least, δ'.
2. **Characterization** of the Regions of Dubiosity by grouping all the DFPs that share the same pattern and are directly connected. Thus, each subregion is a subgraph highlighting a pattern.

δ' should be coherent(if not equal) with the OB_{FP} value in the Exploitation step since we are looking for similarity between DFPs for grouping them.

At the end, the regions of dubiosity that we obtain from the above algorithm help us to identify the problematic zones in the CBR system. Moreover, each subregion of shared patterns serves to detect zones that suffer from the same deficiency, thus preventing us from having to deal with individual DFPs.

A graphical representation of an example for identifying Regions of Dubiosity is given in Figure 4. In the figure two different regions have been detected. The first one on the left only has two DFPs identified as outer limits. The big region on the right has three border sub-regions, one stranger sub-region, and one central shaky terrain sub-region. Connecting lines between two DFPs show that they are neighbours according to a given similarity threshold δ'. Note that region surfaces are only painted with the purpose of highlighting the dubiosity regions in the problem space.

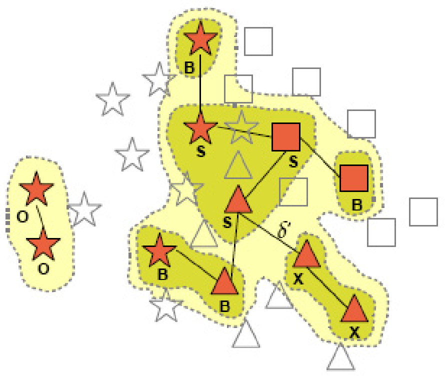

Fig. 4. Regions of Dubiosity. Hollow shapes are cases and filled shapes are DFPs. Subscripts point out the patterns associated to each DFP.

4 Experimentation

We have performed the analysis of DFPs on a CBR system developed for the Four-Legged League (RoboCup) soccer competition [13]. In RoboCup two teams of four Sony AIBO robots compete operating autonomously, i.e. without any external control. The goal of the CBR system is to determine the actions (called *gameplays*) the robots should execute given a state of the game.

The state of the game is mainly represented by the position of the ball and the positions of the players (both teammates and opponents). The positions are constrained by the field dimensions (6 m long and 4 m wide). Moreover, since robots occupy a physical space in the field, a state of the game is considered valid whenever the distances among the robots are higher than their dimensions (30 cm long and 10cm wide).

The 68 cases stored in the system can be grouped into three main behaviors: cooperative behaviors (where at least two teammates participate); individualistic behaviors (only one player is involved); and back away behaviors (where the position of the opponents forces a player to move the ball back).

The confidence measure provided by the application took into account not only the similarity of the problem to the cases but also the actual distance of the current position of the players to the ball. Therefore, although all similar cases share the same solution, when the players are away from the ball the confidence of the solution is low.

The first goal of our experiments was to foresee whether there exist states of the game where the CBR system has difficulties in determining the best behavior, i.e. the confidence on the proposed solution is low. The secondary goal was to detect if there were any bad performing mechanisms of the CBR system.

The experimentation settings were the following: 40% of the population was selected as survivors to the next generation; 60% of the chromosomes were selected as parents to reproduce offspring; mutation was applied to a randomly

Table 1. Average (Avg) and standard deviation (σ) of DFPs discovered in the Exploration ($\mathcal{LCFP}$) and Exploitation ($\mathcal{LCPN}$) steps in 63 experiments. RD shows the number of the Regions of Dubiosity created from DFPs. H, X, L, B, S, O are, respectively, the percentage of the number of the experiments in which *Hole*, *Stranger*, *Lost*, *Border*, *Shaky Terrain*, and *Outer Limit* patterns appeared.

	$\mathcal{LCFP}$	$\mathcal{LCPN}$	RD
Avg	59.26	183.21	55.95
σ	34.12	105.78	30.89

H	X	L	B	S	O
95%	100%	50%	85%	15%	85%

chosen 5% of the offspring modifying a gene's value for each chosen chromosome; the diversity threshold for the twin chromosomes was 5% (we kept this amount of twins in the new generation and replaced the rest of them with new ones created by the RPG).

Taking into account the similarities among existing cases, we chose the test range `[0.93, 0.99]` for the proximity limit values and we kept the proximity for Exploitation narrower than Exploration (see subsections 2.1 and 2.2). The similarity threshold (δ in Figure 3) for associating patterns to DFPs was always a value slightly bigger than the OB_{EC} value in the Exploration step. Analogously, the similarity threshold for grouping DFPs (δ' in Figure 4) to form Regions of Dubiosity, was the same as the OB_{FP} value in the Exploitation step (see subsections 3.1 and 3.2). Finally, the test range for confidence threshold was chosen as `[0.3, 0.7]`.

We ran different experiments for analysing the sensitivity in identifying DFP regions by changing parameters such as the size of the initial population, the number of generations, the confidence threshold, and the proximity limits. Moreover, because of the random nature of GAs, for each setting we executed the Exploration and Exploitation steps several times to get an average value for the number of identified DFPs and regions of dubiosity.

Throughout experimentation we have seen that we may encounter a higher number of DFPs when, the initial population is larger in size or GAs evolve during enough generations or the preferred proximity is wider.

The results show (see Table 1 left) that the number of the DFPs encountered in the Exploration step is closely related to the number of the Regions of Dubiosity. The difference between these two numbers is due to linking of DFPs found in Exploration with other DFPs found in the Exploitation step. This happens when we reach a DFP previously found in the Exploration step while we are exploiting another DFP problem found in the same step. Therefore, when no linking is achieved between such DFPs, the number of the Regions of Dubiosity is equal to the number of the DFPs found in the Exploration step.

Another interesting result is the analysis of the DFP patterns discovered in the experiments (Table 1 on the right summarizes the percentage of experiments where each pattern is detected). This analysis allows a better understanding of the regions of the problem space where the CBR system is not performing confidently:

– Holes in the soccer domain occured when although all the closer cases shared the same individualistic solution, the players were far from the ball. This was due to the provided confidence measure explained above. To obtain more confident solutions the neighbourhood of holes can be populated with new cases.

– Stranger DFPs were problems that proposed cooperative behavior but whose neighbours were individualistic cases. We saw that this was because of the design of the system for favouring cooperative behavior. If there is a close cooperative case to the problem, the application proposes cooperative solution even if there are more similar cases with different solutions. Stranger DFPs helped us to discover regions where the influence of the cooperative cases was excessive. To improve the confidence, we proposed to reduce that influence for similar regions.

– Lost DFPs were problems that proposed cooperative behavior in a region where their neighbours were individualistic and back away cases. This was again due to the excessive influence of a cooperative case nearby. The proposed maintenance task was the same as in the previous pattern.

– Border DFPs identified the regions where neither individualistic nor cooperative behaviors reach a significantly better confidence. The confidence in these regions could be improved by incorporating new cases into the case base to be able to mark the borders better between these two classes.

– Shaky terrain DFPs does not seem to be significant in the Robosoccer domain due to the distribution of the cases in the CB. There are only three solution classes and back away behaviors are mainly close to individualistic behaviors. Hence, in only 15% of the experiments we encountered this pattern when the proximity limits were chosen to be too wide and the proximity of the cases of all three classes were overlapping in some regions.

– Finally, the encountered Outer Limit DFPs were either in the proximity of isolated cases in deserted regions of the CB or they were on the outskirts of the proximity of cases which were themselves at the border of a class.

In Figure 5 a visualisation of an example of a dubious region in the Soccer domain is provided. Hollow and filled squares represent respectively cases and DFPs with cooperative solutions. Analogously, hollow and filled stars represent respectively cases and DFPs with indivudualistic solutions. The visualisation of cases and DFPs was built using a force-directed graph-drawing algorithm where the repulsive force between two cases is proportional to their distance. The dotted line indicates the border of the dubiosity region. We have drawn dubiosity groups at two similarity levels by using two different values for δ' (dark and light colors in figure). Neighborhood lines have been omitted for facilitating the understanding of the figures.

5 Conclusions and Future Work

In this paper we have presented a novel method for identifying future low confidence regions given an existing case base. The method was based on four steps.

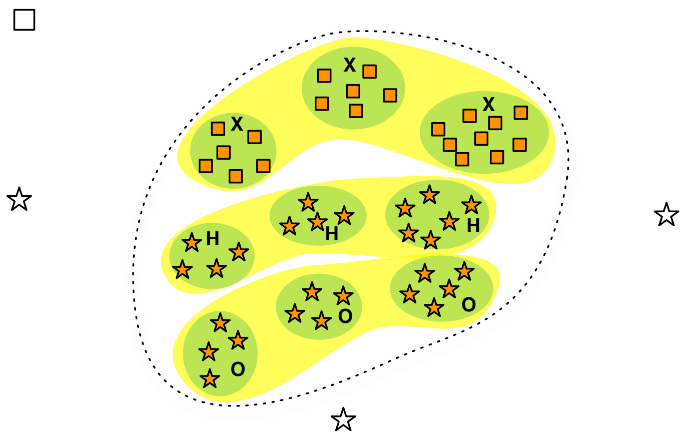

Fig. 5. Visualizing a dubious region in the Soccer problem. DFPs with individualistic solutions are represented as stars. Squares represent DFPs with cooperative solutions. H, X and O, respectively, indicate Hole, Stranger and Outer Limit patterns attached to the cases in the darker coloured regions of dubiosity.

First, we explored the problem space to find dubious future problems. Then, we exploited these problems to better locate them in the case base within their future neighbourhood. Both steps used an evolutionary approach to scan the problem space. Next, to help the understanding of the regions where dubious future problems are located, we associated each problem with one of the six dubiosity patterns defined in the paper. These patterns were based on the neighbourhood of future problems to the existing cases. Finally, we have proposed an algorithm for grouping dubious future problems according to these patterns to identify regions of dubiosity. We argued that these regions enabled us to focus on the regions in the case base that suffer from the same deficiency rather than dealing with individual problems, thus facilitating maintenance tasks.

We described the experiments performed in a Robosoccer application and have shown how DFPs associated with dubiosity patterns helped us to detect dubious regions in the case base and to analyse bad performing mechanisms of the CBR system.

We believe that the proposed method is useful for improving the performance of CBR systems in a proactive fashion. The proposed method uses only the domain ontology for generating future problems and evaluates them by using the confidence and similarity measures provided by the CBR system.

As future work we plan to relate the dubiosity patterns to possible maintenance tasks and to design a graphical tool for navigating through the problem space. We plan to join the method described in this paper with a visualisation method for case base competence based on solution qualities presented in [14].

Acknowledgements. This research has been partially supported by the Spanish Ministry of Education and Science project MID-CBR (TIN2006-15140-C03-01), EU-FEDER funds, and by the Generalitat of Catalunya (2005-SGR-00093).

References

1. de Mantaras, R.L., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M.L., Cox, M., Forbus, K., Keane, M., Watson, I.: Retrieval, reuse, revision, and retention in CBR. The Knowledge Engineering Review **20**(3) (2005) 215–240
2. Watson, I.: CBR is a methodology not a technology. Knowledge Based Systems **12**(5,6) (1999) 303–308
3. Wilson, D.C., Leake, D.B.: Maintaining case-based reasoners: Dimensions and directions. Computational Intelligence **17**(2) (2001) 196–213
4. Smyth, B., Keane, M.T.: Remembering to forget: A competence-preserving case delection policy for case-based reasoning systems. In: Proceedings of IJCAI-95. (1995) 377–382
5. Smyth, B., McKenna, E.: Building compact competent case-bases. In Althoff, K.D., Bergmann, R., Branting, K., eds.: 3rd International Conference on Case-Based Reasoning, ICCBR-99. Volume 1650 of Lecture Notes in Artificial Intelligence. Springer-Verlag (1999) 329–342
6. Smyth, B., McKenna, E.: Competence models and the maintenance problem. Computational Intelligence **17**(2) (2001) 235–249
7. Massie, S., Craw, S., Wiratunga, N.: When similar problems don't have similar solutions. In Weber, R., Richter, M., eds.: 7th International Conference on Case-Based Reasoning, ICCBR 2007. Volume 4626 of Lecture Notes in Artificial Intelligence. Springer-Verlag (2007) 92–106
8. Cheetham, W.: Case-based reasoning with confidence. In Blanzieri, E., Portinale, L., eds.: 5th European Workshop on Case-Based Reasoning, EWCBR-00. Volume 1898 of Lecture Notes in Artificial Intelligence. Springer-Verlag (2000) 15–25
9. Cheetham, W., Price, J.: Measures of solution accuracy in case-based reasoning systems. In Funk, P., Conzález-Calero, P.A., eds.: 7th European Conference on Case-Based Reasoning, ECCBR-04. Volume 3155 of Lecture Notes in Artificial Intelligence. Springer-Verlag (2004) 106–118
10. Delany, S.J., Cunningham, P., Doyle, D., Zamolotskikh, A.: Generating estimates of classification confidence for a case-based spam filter. In Muñoz-Avila, H., Ricci, F., eds.: 6th International Conference on Case-Based Reasoning, ICCBR-05. Volume 3620 of Lecture Notes in Artificial Intelligence. Springer-Verlag (2005) 177–190
11. Mulayim, O., Arcos, J.L.: Exploring dubious future problems. In Petridis, M., ed.: Twelfth UK Workshop on Case-Based Reasoning, CMS Press (2007) 52–63
12. Michalewicz, Z.: Genetic Algorithms+Data Structures=Evolution Programs. 3rd edn. Springer Verlag, New York (1996)
13. Ros, R., de Mantaras, R.L., Arcos, J.L., Veloso, M.: Team playing behavior in robot soccer: A case-based approach. In Weber, R.O., Richter, M.M., eds.: 7th International Conference on Case-Based Reasoning, ICCBR-07. Volume 4626 of Lecture Notes in Artificial Intelligence. Springer-Verlag (2007) 46–60
14. Grachten, M., Garcia-Otero, A., Arcos, J.L.: Navigating through case base competence. In Munoz-Avila, H., Ricci, F., eds.: 6th International Conference on Case-Based Reasoning, ICCBR-05. Volume 3620 of Lecture Notes in Artificial Intelligence. Springer-Verlag (2005) 282–295