Case-Based Goal Selection Inspired by IBM's Watson

Dustin Dannenhauer and Héctor Muñoz-Avila

Department of Computer Science and Engineering, Lehigh University, Bethlehem PA 18015, USA

Abstract. IBM's Watson uses a variety of scoring algorithms to rank candidate answers for natural language questions. These scoring algorithms played a crucial role in Watson's win against human champions in Jeopardy!. We show that this same technique can be implemented within a real-time strategy (RTS) game playing goal-driven autonomy (GDA) agent. Previous GDA agents in RTS games were forced to use very compact state representations. Watson's scoring algorithms technique removes this restriction for goal selection, allowing the use of all information available in the game state. Unfortunately, there is a high knowledge engineering effort required to create new scoring algorithms. We alleviate this burden using case-based reasoning to approximate past observations of a scoring algorithm system. Our experiments in a real-time strategy game show that goal selection by the CBR system attains comparable in-game performance to a baseline scoring algorithm system.

1 Introduction

This work presents a new solution to the problem of goal selection within a goal-driven autonomy agent. Goal-driven autonomy (GDA) is a reasoning model in which an agent selects the goals it will achieve next by examining possible discrepancies between the agent's expectations and the actual outcome of the agent's actions. GDA agents explain these discrepancies and generate new goals accordingly [10, 13–15]. The computer program Watson developed by IBM achieved fame when it defeated two previous (human) winners from the United States television show *Jeopardy!*. Watson's use [16, 17] of a variety of scoring algorithms to rank answers from evidence snippets can be applied to a game playing agent for ranking which goal to achieve next. We present two goal selection implementations: a baseline system inspired by Watson's answer scoring algorithms and a case-based reasoning system that approximates this baseline system.

In RTS games, players manage armies of units to defeat an opponent. Actions in the game are executed in real time (i.e., players do not wait for the opponent to make a move). RTS games follow a combat model where units of a certain kind are particularly effective against units of some other kind but particularly vulnerable against units of a third kind. For this reason RTS games are frequently used in case-based reasoning research [9]. A challenge of using CBR (or any

S.J. Delany and S. Ontañón (Eds.): ICCBR 2013, LNAI 7969, pp. 29–43, 2013.

other AI technique) in these domains is the large amount of state information. With such large state spaces, AI systems' state representations must exclude many details about the state (e.g., by using state abstraction techniques [20]). A primary motivation for borrowing the scoring algorithm technique from Watson is to make use of potentially all information in the game state.

A major drawback of this evidence-scoring technique is the significant knowledge engineering effort required to create new scoring algorithms. Each scoring algorithm contains heuristic-like knowledge that relates evidence to an answer by means of a numerical score. Such effort is apparent in Watson, which made use of thousands of scoring algorithms (sometimes referred to as evidence scoring strategies, in this paper we refer to them as evidence scorers). At least some (it is not clear how many) scoring algorithms used by Watson were NLP-based and were easily available due to research in the NLP community [19]. However, for other domains, including RTS games, it may not be the case that evidence scoring functions are readily available. Our work is partly motivated by the fact that creating evidence scorers may be too high of a knowledge engineering burden but past observations (e.g., made by experts) are available. Case-based reasoning can approximate evidence scorers by capturing and reusing the results from previous observations (e.g., scores given by human experts in previous episodes). We found that a CBR system with a simple state representation, a straightforward similarity function, and nearest neighbor retrieval approached the performance of the baseline evidence scoring system.

This paper is organized as follows: Section 2 discusses IBM's Watson and the core technique relevant to the work in this paper. Section 3 discusses the baseline Watson-inspired goal selection component using evidence scorers. Section 4 discusses an implementation of the CBR system using the ideas described in Section 3. We present our experiments and results in Section 5, followed by Related Work in Section 6, and finally end with Conclusions and Future Work in Section 7.

2 IBM's Watson's Evidence Scoring Algorithms

Jeopardy! is a game in which 3 competitors are given clues in natural language about some information that must be guessed and the first person to answer the information correctly wins. A wrong answer carries a penalty, so good players must be highly confident of their answers before choosing to respond. Successful play requires rapid understanding of English sentences and substantial background knowledge on a variety of topics [16, 17]. The heart of Watson is an extensible software architecture named DeepQA [16]. DeepQA is best thought of as a pipeline, where the question is given at the start and an ⟨answer,confidence-score⟩ pair is produced at the end (a full diagram of the pipeline can be found in [16]). This pipeline has many phases, we are only interested in the final stage of the pipeline, which ranks potential answers based on the evidence scores provided by the evidence scorers [19].

When the DeepQA pipeline reaches the final stage it has accumulated a list of candidate answers along with supporting pieces of evidence for each answer. An

example of a supporting piece of evidence may be a sentence or passage from an encyclopedia that contains keywords from the question and candidate answer. During the final stage, thousands of answer scorers each produce a numeric score representing the degree to which a piece of evidence supports or refutes a candidate answer. [19] The goal of the final stage is to combine all evidence scores for each candidate answer in order to determine the best candidate answer and its corresponding confidence score. To best combine evidence scores, DeepQA uses machine learning to train over a corpus of previously used questions and their correct answers [19]. DeepQA then produces a model describing how the evidence scorers should be combined (i.e., assigning different weights to different evidence scorers). Sometimes a single evidence scorer or group of evidence scorers are highly indicative of the correctness of an answer, and therefore should be given more importance when aggregating scores.

3 Goal Selection Using Case-Based Reasoning

3.1 Case Representation

We describe a system that takes an approach to goal selection that is inspired by IBM's Watson. As mentioned before, Watson ranks candidate answers according to scores produced by what are called evidence scorers. Evidence scorers are essentially functions that take the question posed to the contestant combined with a candidate answer, and a piece of evidence (i.e. a sentence or paragraph from an encyclopedia) and produce a score of how well that piece of evidence supports the given candidate answer for the given question. This can be represented as a triple: ⟨question, answer, score⟩. All of the scores from each piece of evidence for a candidate answer are aggregated into a single score for that candidate answer. This aggregated score reflects how well the pieces of evidence support the candidate answer. The candidate answers are then ranked based on their aggregated scores and the highest scoring answer is chosen.

Our baseline evidence scorer system takes the same approach, except instead of evidence scoring functions that take a candidate answer and a piece of textual evidence (such as a paragraph), our evidence scoring functions take a goal and features of the current game state in the RTS game Wargus. Analogous to the representation of ⟨question, answer, score⟩ in Watson playing *Jeapordy!*, we use ⟨gamestate features, goal, score⟩ as the representation in our system playing Wargus. In the same way as Watson, we produce aggregated scores for each goal that we may decide to pursue next. After each goal's scores are aggregated, the highest scoring goal is chosen.

Such an approach allows the goal selection component to neither restrict nor conform to models of the game state used in other GDA components. For example, perhaps the planning component of a GDA agent uses a compact state representation (as is the case in LGDA and GRL [1, 15]). The goal selection component is not forced to use that state representation, nor does it impose any restriction on the planning component's use of a compact game state. The goal selection component of the GDA agent may make use of more or all information

in the game state at the time of goal selection. This is a benefit of modularity and would allow a component like the one presented in this paper to easily fit into a current GDA system.

3.2 Information Flow

Our main motivation is for situations in which the system neither has access to the internal functioning of the evidence scorers nor to the evidence scorers themselves. For example, the evidence scorers are humans that we observed in past instances solving problems. Ontañón *et al.*, 2007 show how domain experts annotate input traces by the goals they achieve [6]; in our situation, we would ask the experts to also annotate the goals' scores. The primary objective is to create a system by reusing previous instances of these evidence scorers providing scores for specific situations. We present a case-based reasoning system that approximates an evidence scoring system by reusing past instances.

Given a sufficiently large number of past instances ⟨gamestate features, goal, score⟩ from an evidence scoring system, a case-based reasoning component can be constructed. For each instance, it is necessary to have the results from each evidence scorer and features from the game state at the time of the instance.

Figure 1 depicts a high level overview of the Watson-inspired evidence scoring component as well as the information available to the case-based reasoning system. Immediately to the right of the "Watson-inspired Component" are the evidence scorers, denoted as the functions $ES_1(G_i, S), ES_2(G_i, S), \ldots ES_N(G_i, S)$. Each evidence scorer is invoked for every goal, resulting in $N * M$ intermediate scores denoted by $G_{1,E_1} \ldots G_{M,E_1}, G_{1,E_2}, \ldots G_{M,E_2}, G_{1,E_N} \ldots G_{M,E_N}$. These intermediate scores are then aggregated to produce a single score, one for each goal, denoted by the $G_1 \ldots G_M$ scores. The goal with the highest aggregate score is chosen.The evidence scoring functions each take an additional argument, S, representing features from the current game state from Wargus. The case-base of the case-based reasoning component is shown in the lower right. The area within the double line represents all of the information available to be stored in each new case. Our case-based reasoning system records features from the game state, the highest scoring goal, and the score.

4 A CBR System for Goal Scoring

We now present a detailed walk-through of a system that implements the ideas discussed in the previous section. A goal is a task we want to achieve. Akin to [1], in our implementation we assume there is one way to achieve a goal. However, our ideas are amenable to situations in which there is more than one way to achieve a goal. Table 1 shows the goals used in our implementation. These are high level goals that require multiple actions in order to be achieved.

The baseline evidence scoring component that chooses goals within Wargus uses three specific evidence scoring functions, described in Table 2. Each of these evidence scorers produce a score based on specific features of the current game state and a goal. Designing evidence scorers can require significant knowledge

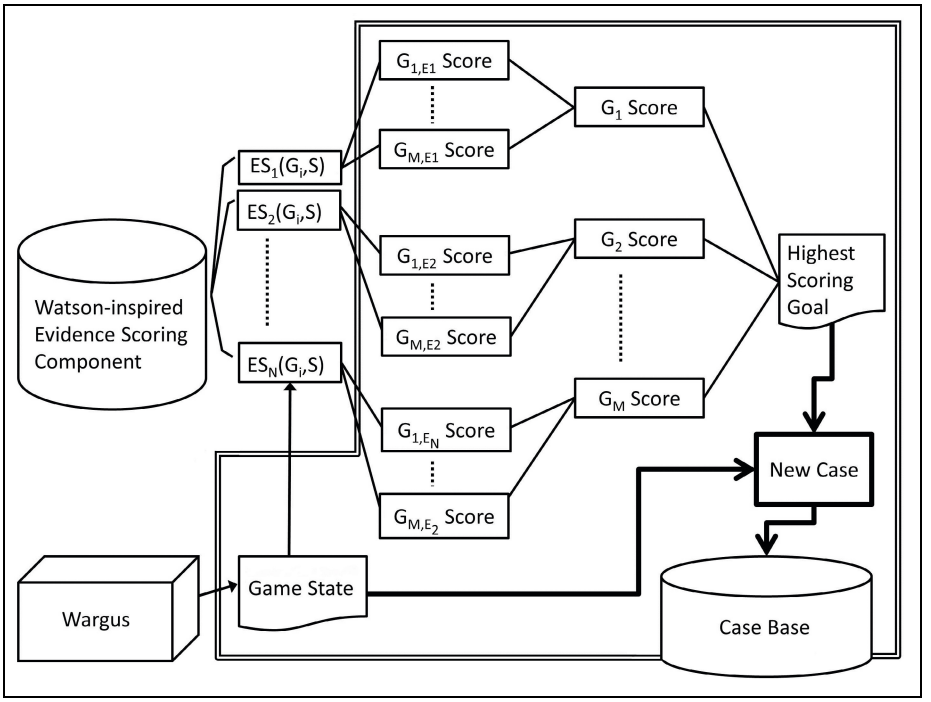

Fig. 1. Cases built from observing Watson-inspired evidence scoring component

engineering. Every instance of the evidence scoring system selecting a goal, a new case ⟨map features,goal,score⟩ is created that is then used in the case-based goal selection component.

We now walk through an example taking place using the scenario in Figure 2. In this scenario, the darker tiles are water and inaccessible by land units, and the lighter tiles are land tiles. In our experiments, goals were selected at the beginning of a scenario and their corresponding strategy was continuously executed. An agent playing full RTS games would encounter many such scenarios. In a GDA agent, components related to acting on discrepancies would determine when new goals were chosen. We take the perspective that new goals would only be chosen at the start of each micro-battle similar to those depicted in these scenarios.

4.1 Evidence Scoring System

When each game starts, the evidence scoring component will first calculate intermediate scores for each goal by invoking each evidence scorer on each goal. (Refer back to Figure 1, the intermediate scores are $G_{1,E_1} \ldots G_{M,E_1}, G_{1,E_2}, \ldots G_{M,E_2}$, $G_{1,E_N} \ldots G_{M,E_N}$). With $M = 7$ goals and $N = 3$ evidence scorers, $N * M = 21$ intermediate scores produced. It is important that the evidence scorers only score

Table 1. Summary of Goals in Wargus

Goal #	Goal Name	Strategy to Achieve Goal
1	High Range	Attack ranged enemy units first before attacking melee units
2	Passive	All units hold position and only attack if enemy units enter within attacking range. Units remain in position, even when engaged. This means melee units will not be able to attack ranged units unless ranged units are directly adjacent to the melee unit.
3	Ranged Passive	Only ranged units hold their ground, any melee units attack the closest enemy and will move to engage the enemy.
4	Half Ranged Passive	A randomly chosen group of half of the ranged units hold their ground. The remaining units attack the closest enemy and will move to engage.
5	Closest Distance	All units attack the unit closest to them.
6	Spread Out	Units attack-move* into a grid formation such that there is 1 tile of space between them and the next closest friendly unit.
7	Huddle	Calculate the center of mass of the army and order all units to attack-move* to this position. This results in a tightly packed, constantly shifting blob where all units attempt to occupy the center tile.

* attack-move is a Wargus game command that orders units to move to a specific location. Unlike the move command, units commanded to attack-move will pursue and attack any enemy that comes into their line of sight at any time.

goals that have supporting evidence. When no supporting evidence exists, the evidence scorer returns a value of zero. For the sake of space, we only show the intermediate scores that are not zero in Table 3. Observe that the goal Half Ranged Passive was scored by two evidence scorers, but because Huddle was scored so highly by a single evidence scorer, and we do not weight intermediate scores, Huddle obtains the highest aggregated score. In our implementation, evidence scorers produced scores between 0 and 7. Assigning weights to different evidence scorers is another place where knowledge engineering is required (although machine learning can be used to figure out how to best combine intermediate scores - IBM's Watson made heavy use of machine learning to combine the thousands of intermediate scores that were produced by thousands of evidence scorers over hundreds of answers. See [19] for more details.).

The Army Distance evidence scorer calculated that the distance between each army in this scenario was relatively far, and therefore gave a score of 5 to the Closest Distance goal (Table 3). The Army Distance evidence scorer produced zeros for all other goals, indicating it did not think the current game state provided any evidence to choose a goal when taking into account the distance between opposing units.

Table 2. Summary of Evidence Scorers in Wargus

#	Evidence Scorer Name	Function
1	Army Distance	Finds the single minimum distance of all distances between each friendly unit and the closest enemy unit. Goals are scored based on this distance. Distance is calculated between each units location using the formula: $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
2	Ratio Water to Land	Draws a straight line between each of friendly unit and the closest enemy unit, and counts all tiles touching this line, recording if each tile is a water tile or land tile. If the ratio of water to land is high, that indicates there is a greater chance of a choke point between the two armies. Scores goals based on this ratio of water to land tiles between the opposing forces.
3	Ratio Ranged to Melee	Calculates the ratio of ranged units to melee units. Scores goals based on the this ratio.

The Ratio Ranged to Melee evidence scorer gave a low score of 1 to goals Passive, Ranged Passive, and Half Ranged Passive, indicating that given the current number of ranged and melee units, it would be slightly advantageous for ranged units to hold position. When there are no ranged units, this evidence scorer produces a score of all zeros for every goal. This evidence scorer also produced a high score of 7 for Huddle, indicating that whenever there is a low ratio of ranged to melee units, Huddle is a strong goal. The intuition is that by huddling the army, the chances of surrounding ranged units by melee units increases and would result in ranged units having greater chances of survival and increased damage output. In our implementation, Ratio Ranged to Melee chose Huddle over Spread Out when the ratio of melee units to ranged is greater than 1.

The Ratio Water to Land evidence scorer detects a low ratio of water to land tiles in between the enemy armies, suggesting that there is a wide chokepoint, in which case it is somewhat advantageous to keep half of the ranged units holding position. Thus, this evidence scorer produces a score of 4 for the HalfRangedPassive goal. For more narrow chokepoints, it would rate Ranged Passive or Passive higher, with the intuition being that the more narrow the chokepoint, the more ranged units should hold their ground on the opening of the chokepoint.

It is easy to see that the Huddle goal is the highest ranked goal when the intermediate scores for each goal are aggregated. At this point the evidence scoring system executes the Huddle goal and the goal selection process finishes.

4.2 Case-Based Goal Selection Component Example

The case base is populated with each instance of the evidence scoring system selecting a goal. The case-based system performs on each scenario only after a

Table 3. Intermediate Scores for each goal

Evidence Scorer	Goal	Score
Army Distance	ClosestDistance	5
Ratio Ranged to Melee	Passive	1
Ratio Ranged to Melee	RangedPassive	1
Ratio Ranged to Melee	HalfRangedPassive	1
Ratio Ranged to Melee	Huddle	7
Ratio Water to Land	HalfRangedPassive	4

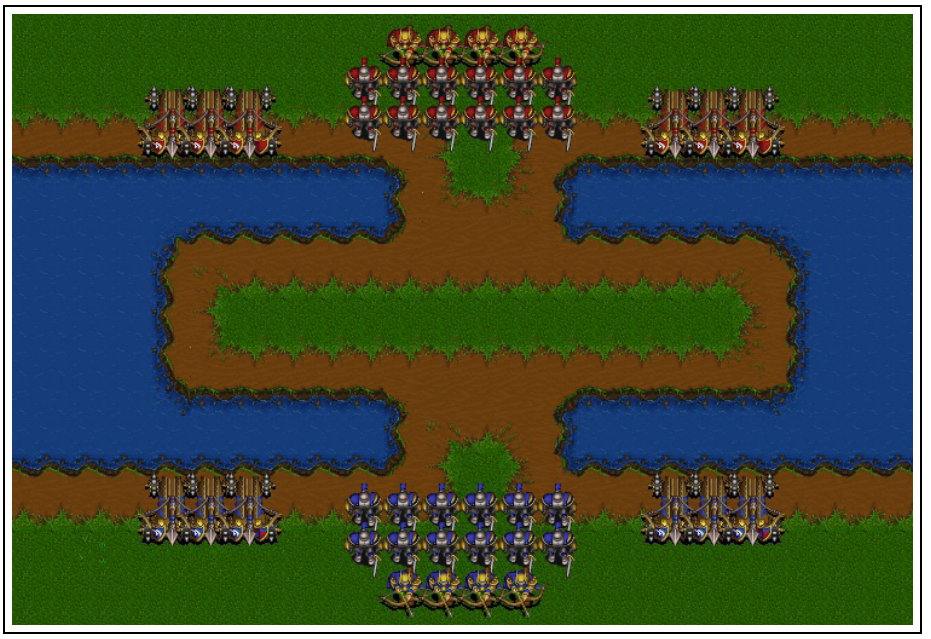

Fig. 2. Screenshot of the Example Scenario

sufficient number of cases are constructed from observing the evidence scoring system in action. At the start of the scenario, the case-based system obtains game state information, specifically the number of water and land tiles and the numbers of each type of unit (however, much more information could be used, such as the locations of each unit). In the scenario in Figure 2, it counted 6, 4, 12 for ballista's, rangers, and footmen respectively and 129 and 895 water and land tiles specifically (Figure 2 shows only the main part of the scenario; some water and land tiles are not shown). The case-based system uses a straightforward similarity function shown in Figure 3 where WT_{c_i} and LT_{c_i} represent the number

$$sim(c_1, c_2) = 1 - \left(\frac{1}{4}\right)\frac{WT_{c_1} - WT_{c_2}}{WT_{max}} - \left(\frac{1}{4}\right)\frac{LT_{c_1} - LTc_2}{LT_{max}} - \left(\frac{1}{2}\right)\sum_{i=1}^{K}\left(\frac{1}{K}\right)\frac{U_{i,c_1} - U_{i,c_2}}{U_{max}}$$

Fig. 3. CBR Similarity Function

of water tiles and land tiles for case c_i, respectively, and U_{i,c_i} represents the number of units of type i found in case c_i. WT_{max} and LT_{max} are the maximum number of water and land tiles of any scenario. K is the number of different types of units in each team.

In one of our experiments, for this scenario, the case-based reasoning system retrieved a very similar case where the number of ballistas and rangers differed by 2 and 8 respectively (the number of land and water tiles remained the same). In the retrieved case, the evidence scoring system had chosen the goal Spread Out instead of Huddle. Surprisingly, Spread Out ended up being a slightly better goal than Huddle. While the case-based reasoning system was incorrect in choosing the same goal as the evidence scoring system, it actually ended up performing better. This is the result of two important considerations. First, the evidence scoring system is not perfect, and would need machine learning techniques such as ensemble methods [21] to learn the appropriate weights to achieve a very high accuracy (as well as a proper set of evidence scorers). One reason DeepQA is attributed to the success of IBM's Watson is the ease in which different answer scorers could be experimented with. Described as a process like a running trial and error, the sets of evidence scorers that performed increasingly well were kept and continuously revised [19]. Both the set of evidence scorers and the corresponding weights play a significant role. Second, the strategies to achieve different goals in Wargus have varied performance and often pursuing multiple goals results in decent performance in some scenarios. This is much different from *Jeopardy!*, where it is rare for more than one answer to be correct.

5 Experiments

Our hypothesis is that a case-based reasoning system that is able to observe an evidence-scoring system can learn to accurately choose the same goals, given a sufficiently large and comprehensive case base. Additionally we hypothesize that the case-based reasoning system can perform close to the performance of the evidence-scoring system. While the case-based reasoning system may choose a different goal than the evidence scoring system, resulting in lower accuracy for the first hypothesis, it may still be a fairly good goal (perhaps even better), and therefore result in relatively good performance to that of the evidence scoring system.

In order to test both hypotheses, we hand crafted 48 unique scenarios for the goal-selection systems. Each scenario was one of 6 unique terrain layouts, 2 of which were pure land maps (no water tiles) and 4 of which had varying amounts of water tiles. For each of the 6 unique terrain maps, 8 different configurations of number of units and unit types were created. The configurations of units was either all melee, melee outnumbering ranged, ranged outnumbering melee, or all ranged. For each of these four relative unit configurations, two different maps were created, differing slightly. For all 8 variations for each unique terrain map, the locations of units were kept approximately the same. Every scenario was symmetrical about the terrain and units, except for 1 scenario in which one team surrounded another. Both goal systems faced the same opponent implemented by the strategy achieving the goal Closest Distance. This is the most general strategy and generally performed well in every scenario. For every match (scenario) the system played on either side and the resulting score is the average difference in scores from both runs. The score in Wargus is calculated by adding the score for every enemy unit you defeat, with different units being worth different point values. For example, rangers are worth 70 points and ballistas are worth 100. So if both teams score 1000, it means they each killed 1000 points worth of units on the opposing team, resulting in a tie.

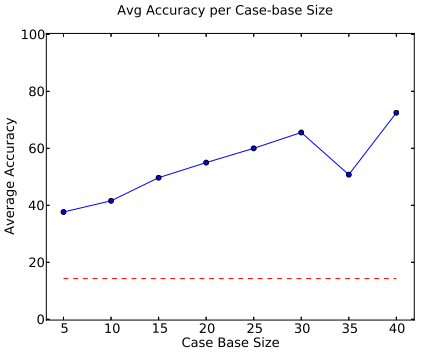

Fig. 4. Accuracy per Case-base Size

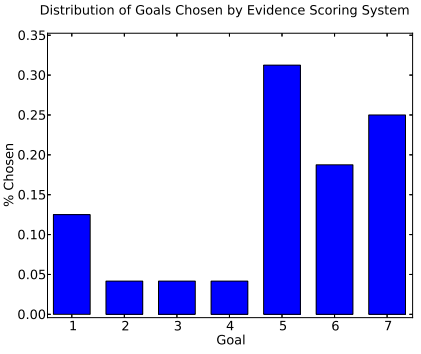

Fig. 5. Goal Selection Distribution

5.1 Results

The first graph, Figure 4, shows the accuracy of the case-based system against the evidence-based system. We ran the evidence scoring system on each of the 48 scenarios, and recorded the goals chosen. Next, we randomly picked X cases (where X varied from 5 to 40 by intervals of 5) from those 48 scenarios to use as cases in the case base. Each case only recorded the game state features and final goal chosen by the evidence scoring system. Each data point is the result of the average of 5 rounds, where each round consisted of randomly picking a case base of size X and recording the accuracy of choosing the same goal as the evidence scoring system on the remaining scenarios. For example, using a case base of size 15, the testing set was of size $48 - 15 = 33$ scenarios.

A random system would choose the correct goal once out of seven times, roughly 14% shown by the straight line in Figure 4. The first case base of size 5 exceeds random goal selection, and this is partly due to obtaining a diverse case base and to the distribution over the goals chosen by the evidence scoring system shown in Figure 5. Because goals 5, 6, and 7 were chosen much more often by the evidence scoring system, if the case base of size 5 contained cases where goals 5 and 6 were chosen, it would likely score highly in the remaining scenarios. Only if the case base consisted of mostly goals 2, 3 or 4 would the performance be near or worse than random. This distribution of goals in Figure 5 is also the reason for the dip in accuracy for case base of size 5. Depending on what scenario's were chosen in the case base, the range of the accuracy is quite large. The important result from Figure 4 is that the case base system becomes more accurate with more cases in the case base, and approaches 70% accuracy.

The following graphs, Figures 6 to 9, show the average performance of the case-based reasoning system compared to the evidence scoring system. Each bar represents the difference in score of the goal selection system against the opponent (the strategy for Closest Distance). These results are broken into 4 different figures to allow for closer inspection. Each graph depicts the difference in scores for 12 scenarios. For each scenario, the first bar is the evidence scoring system, the second is the case-based system with a case base of 5 cases, the third bar is the case based system using a case base of 10 cases, etc until the last bar is the case base containing 40 cases. The angled lines above or below each set of bars for each scenario help display the difference from the evidence scoring system and the last case based system. A angle with an end point higher than the starting point represents the case based system outperforming the evidence based system (example is scenario 31) and vice versa. Whenever the goal selection system tied with the opponent (each army defeated the other) the difference in scores is 0, and no bar is shown. Generally, we expect to see that the evidence-scoring system scores relatively highly (the first bar should be a high positive value), and the case-based systems progressively get closer and closer to the evidence scoring system score. We see this happen approximately in Figure 6 for scenarios 1, 2, 5, 11, and 12, Figure 7 for scenarios 15 and 16, Figure 8 for scenarios 26, 28, 31, 34, and 35 and in Figure 9 scenarios 38, 40, 44, 46, 47, and 48. Even when the evidence scoring system performs poorly (shown by negative bars) the case-based system approximates it. Also, notice that in Figure 7 the last 8 scenario's are almost all blank (the lone vertical bar in them is the case-based system of size 5). This is because the evidence based system and case-based system (except for case base size of 5 in some instances) chose the same goal as the opponent, Closest Distance, and resulted in tied games, and further shows that the case-based system performs approximately as well as the evidence scoring system, despite only achieving a 70% accuracy in pure goal selection. For 70% of the scenarios, when the case base had the most cases, the case-based prediction system had the same or better performance than the evidence scoring system.

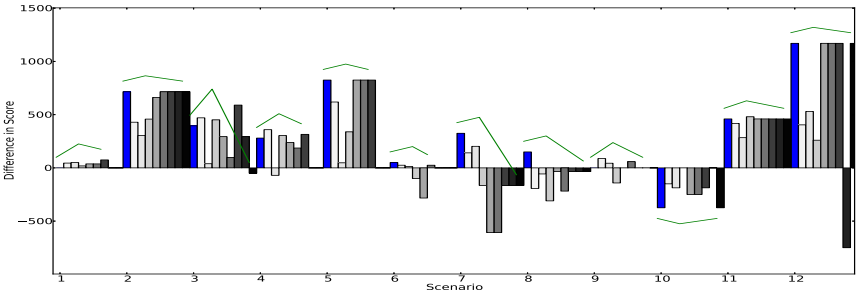

Fig. 6. Average Performance in Each Scenario (1 to 12)

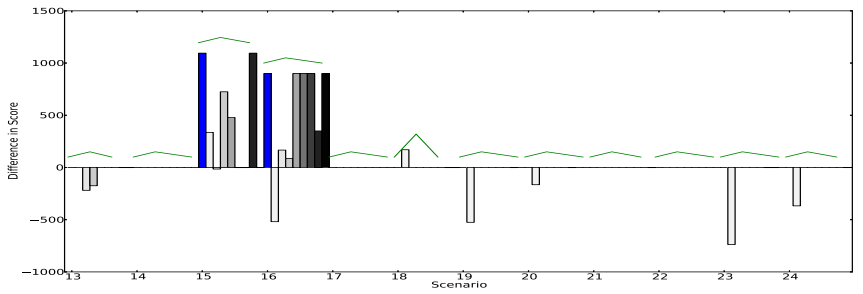

Fig. 7. Average Performance in Each Scenario (13 to 24)

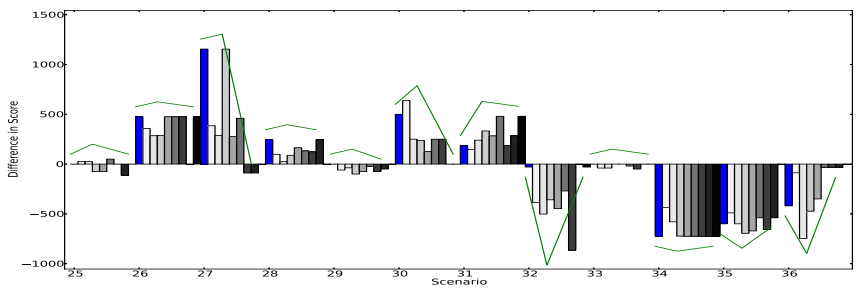

Fig. 8. Average Performance in Each Scenario (25 to 36)

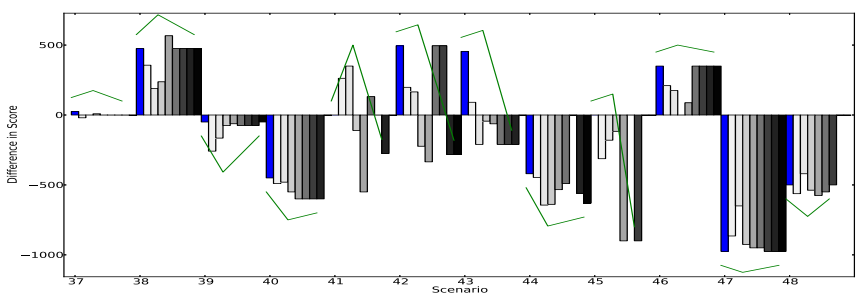

Fig. 9. Average Performance in Each Scenario (37 to 48)

When running these experiments, occasionally matches would fail for seemingly no reason, but in a deterministic manner. We ran 336 unique match ups (goal strategy and unique scenario), and only 9 of these failed before the game finished. However, 3 of these failed with only 1 unit left on one team and at least 4 units alive on the other team. For these 3, we calculated the worth of the single unit and assumed it would have been killed, and manually adjusted the winning team's score appropriately. For the remaining 6 failed matches, we re-ran the match until right before it crashed, recorded the score and the locations of each of the remaining units on each team. We then reconstructed a new match exactly as it was left off, ran it, and added the score to the match before it failed. Because there is no way to start a Wargus match so that units have less than maximum health, there was some information loss, but because this only occurred on 6 out of the 336 matches the overall results were not significantly affected.

6 Related Work

We discussed IBM Watson in Section 2. Here, we discuss other related works. As previously mentioned our objective is to embed these goal selection techniques into GDA agents. GDA agents select their goals based on the explanation of a discrepancy. A number of techniques have been suggested for goal selection in GDA research. These include using rules that map the explanation to the goal to pursue next [11, 14]. A more common technique is to rank the goals according to priority lists (i.e., the goals having higher priorities are more likely to be selected than goals that have a lower priority) (e.g., [10]). We believe that using priority lists is a natural way to integrate our work with GDA; since each goal will be assigned a score, these scores can be used to update the priorities in the list. Mechanisms will be needed to merge the priorities suggested by GDA with the scoring suggested by the CBR system.

Outside of GDA research, goal selection has been a recurrent topic in planning research [8]. Typically, the higher level goals to achieve are fixed and the problem is to select subgoals that achieve those goals. Research has been done to relax the requirement that the goals are fixed; over-subscription planning aims at finding the maximal subset of the goals that can be achieved [12]. In principle, the user could input a large set of possible goals and let the system figure out which subset of these goals can be achieved in the given situation.

Wargus has been extensively used as a testbed in case-based reasoning research. Among many others, Mehta et al. (2009) used case-based learning techniques to learn from failure patterns in a Wargus game trace [9]. It has also been used to retrieve cases aimed to counter an adversary [4] and to evaluate online case-based adaptation algorithms [3]. Outside of CBR, Wargus has been used to demonstrate concurrent reinforcement learning techniques [5] and to acquire playing strategies using evolutionary computation among many others [2]. A common motivation among these works for using Wargus is that it provides a rich environment for decision making. This is precisely the motivation for using

Wargus in our work as we want to test the goal selection mechanism and observe how it affects this environment.

7 Conclusions and Future Work

IBM's Watson demonstrated the effectiveness of using a variety of evidence scorers to rank potential answers. However, one of the biggest issues is the knowledge engineering burden to create the ranking algorithms. In this paper we explored a CBR solution to an instance of this problem in which episodic knowledge of the form ⟨gamestate features, goal, scores⟩ are retained and reused. We tested our ideas in the Wargus RTS game using a hand-crafted evidence scoring system inspired from IBM's Watson as our baseline. We use this baseline system to generate cases that are fed into a CBR system. Our experiments demonstrated that the CBR system can attain comparable performance to the baseline system after sufficient cases have been retained in the case base.

For future research directions we wish to explore a number of ideas. (1) Given the independent nature of evidence scorers, we would like to explore running more complex and potentially computationally expensive evidence scorers in parallel. (2) We will like to use ensemble methods [21] to aggregate the individual information from the scoring algorithms. The tuning of this feature can lead to significant performance improvements. (3) We will like to study the use of an adaptation algorithm, where there can be multiple alternative ways to take the k-retrieved cases and select goals (such as taking into account the intermediate scores from the evidence scorers). These alternative ways could be ranked using techniques such as the ones shown in this paper.

Acknowledgments. We would like to thank David W. Aha (Naval Research Laboratory) for suggesting the idea of using IBM Watson in the context of goal-driven autonomy research. We would also like to thank Ulit Jaidee for his code used to run automated experiments in Wargus. This research is funded in part by NSF 1217888.

References

1. Jaidee, U., Muñoz-Avila, H., Aha, D.W.: Learning and Reusing Goal-Specific Policies for Goal-Driven Autonomy. In: Agudo, B.D., Watson, I. (eds.) ICCBR 2012. LNCS, vol. 7466, pp. 182–195. Springer, Heidelberg (2012)
2. Ponsen, M., Muñoz-Avila, H., Spronck, P., Aha, D.: Automatically Acquiring Domain Knowledge For Adaptive Game AI Using Evolutionary Learning. In: Proceedings of the Seventeenth Innovative Applications of Artificial Intelligence Conference (IAAI 2005). AAAI Press (2005)
3. Sugandh, S., Ontañón, S., Ram, A.: On-Line Case-Based Plan Adaptation for Real-Time Strategy Games. In: Proceedings of the AAAI Conference (AAAI 2008). AAAI Press (2008)

4. Aha, D.W., Molineaux, M., Ponsen, M.: Learning to win: Case-based plan selection in a real-time strategy game. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 5–20. Springer, Heidelberg (2005)
5. Marthi, B., Russell, S., Latham, D., Guestrin, C.: Concurrent hierarchical reinforcement learning. In: International Joint Conference of Artificial Intelligence (IJCAI 2005). AAAI Press (2005)
6. Ontañón, S.: Case Acquisition Strategies for Case-Based Reasoning in Real-Time Strategy Games. In: FLAIRS 2012. AAAI Press (2012)
7. Ontañón, S., Mishra, K., Sugandh, N., Ram, A.: Case-Based Planning and Execution for Real-Time Strategy Games. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 164–178. Springer, Heidelberg (2007)
8. Ghallab, M., Nau, D.S., Traverso, P.: Automated Planning: Theory and Practice. Morgan Kaufmann (2004)
9. Mehta, M., Ontañón, S., Ram, A.: Using Meta-reasoning to Improve the Performance of Case-Based Planning. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 210–224. Springer, Heidelberg (2009)
10. Molineaux, M., Klenk, M., Aha, D.W.: Goal-driven autonomy in a Navy strategy simulation. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence. AAAI Press, Atlanta (2010)
11. Cox, M.T.: Perpetual self-aware cognitive agents. AI Magazine 28(1), 23–45 (2007)
12. van den Briel, M., Sanchez Nigenda, R., Do, M.B., Kambhampati, S.: Effective approaches for partial satisfaction (over-subscription) planning. In: Proceedings of the Nineteenth National Conference on Artificial Intelligence, pp. 562–569. AAAI Press, San Jose (2004)
13. Powell, J., Molineaux, M., Aha, D.W.: Active and interactive discovery of goal selection knowledge. In: To Appear in Proceedings of the Twenty-Fourth Conference of the Florida AI Research Society. AAAI Press, West Palm Beach (2011)
14. Muñoz-Avila, H., Jaidee, U., Aha, D.W., Carter, E.: Goal-Driven Autonomy with Case-Based Reasoning. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 228–241. Springer, Heidelberg (2010)
15. Jaidee, U., Muñoz-Avila, H., Aha, D.W.: Integrated learning for goal-driven autonomy. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, vol. 3. AAAI Press (2011)
16. Ferrucci, D.A.: Introduction to This is Watson. IBM Journal of Research and Development 56(3.4), 1 (2012)
17. Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J., Schlaefer, N., Welty, C.: BBuilding Watson: An Overview of the DeepQA Project. AI Mag. 31(3), 59–79 (2010)
18. Lally, A., Fodor, P.: Natural Language Processing With Prolog in the IBM Watson System (retrieved June 15, 2011)
19. Gondek, D.C., et al.: A framework for merging and ranking of answers in DeepQA. IBM Journal of Research and Development 56(3-4), 14:1–14:12 (2012)
20. Giunchiglia, F., Walsh, T.: A theory of abstraction. Artificial Intelligence 57(2-3), 323–390 (1992)
21. Dietterich, T.G.: Ensemble Methods in Machine Learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
22. Hunter, J.D.: Matplotlib: a 2D graphics environment. Computing in Science & Engineering, 90–95 (2007)