Incremental Generation of Abductive Explanations for Tactical Behavior

Thomas Wagner, Tjorben Bogon, and Carsten Elfers

Center for Computing Technologies (TZI), Universität Bremen, D-28359 Bremen
{twagner, tbogon, celfers}@tzi.de

Abstract. According to the expert literature on (human) soccer, e.g., the tactical behavior of a soccer team should differ significantly with respect to the tactics and strategy of the opponent team. In the offensive phase the attacking team is usually able to *actively select* an appropriate tactic with limited regard to the opponent strategy. In contrast, in the defensive phase the more passive *recognition* of tactical patterns of the behavior of the opponent team is crucial for success. In this paper we present a qualitative, formal, abductive approach, based on a uniform representation of soccer tactics that allows to recognize/explain the tactical and strategical behavior of opponent teams based on past (usually incomplete) observations.

1 Introduction

Abductive inference, i.e., the inference to reason from *observation* to *cause*, has already been proposed by Charles S. Pierce [10] in the early 1920 as the third fundamental inference next to induction and deduction which accounts to the generation of new knowledge. Nevertheless, in contrast to the latter inferences abduction[1] has gained only limited attention. Recent advances in a wide range of scientific research fields ranging from robotics to ubiquitous and intelligent environments have imposed strong requirements in the generation of environmental context, especially in the face of (inter)acting agents. The environmental context is to a large extent characterized by the spatial representation of the objects and by the actions of the agents who (inter)act within the environment. While the generation of the spatial context has gained strong interest the recognition of action (from complex to simple) and intentions (underlying these actions) is especially in the area of ubiquitous and intelligent environments an open and demanding task. Nevertheless the recognition of intentions and actions of collaborating and competing agents is an essentiell task for (semi)active assistance. In order to provide appropriate active assistance the intelligent environment needs to how either *what* an agents intends to do or *how* an agents intents to realize his intentions. The requirements differ to the intention recognition process differ significantly from many other areas of application like diagnosis (see following section 2). In this paper we present a new approach to plan recognition based on the works of Eiter and Makino [4] that is specially suited to the demands of (active) assistance systems.

[1] The use of abductive inference is not necessarily limited to logical representations

The rest of the paper is organized as follows. In section 2 we describe in more detail the role of tactics in human soccer and try to motivate the relevance of incrementality and configurable context sensitivity. Furthermore we motivate how the recognition process is related to abduction. In section 3 we shortly describe the previous work on plan recognition with a focus on abductive methods. The succeeding section 4 shows on the one hand how a declarative representation can be transformed into horn clause (in a uniform fashion) and how qualitative action description can be generated based on 2D- and 3D simulation data. In section 4.4 we introduce our optimization and extensions of the *Eiter et al.*-approach. After the results in section 5, we finally conclude the paper with a discussion and an outlook on future work in section 6.

2 Motivation

One of the most important tasks to be accomplished in an intelligent environment is the generation of a consistent representation of the static (e.g., the spatial representation of the static environment) and dynamic actions and entities (e.g., most importantly the agents acting within). The generation of the *situational context*[2] is one of the most fundamental and demanding tasks, since it provides the basis for every kind of assistance that e.g., an intelligent environment will be able to provide. At least two assistance scenarios can be distinguished: (1) *passive* assistance, where the acting agent actively asks for support/information for a specific task and (2) *active* assistance, where the environment actively provides support without explicit user request. The latter task is of special interest, since it allows for a fundamentally new range of application: e.g., support of elderly and/or handicapped people or support of children (avoiding dangerous situation), ... Active support is not limited to these applications but is also interesting for everyday scenarios, like pre-heating the oven, defrost foot (both, if required). Active support is not only one of the most interesting scenarios for assistance but also one of the most demanding. It requires the intelligent environment to make to fundamental decisions: (a) *when* and (b) *how* support should be provided. Both questions can only be answered with respect to the specific environmental conditions and the intentions of the users. Especially the latter one imposes new requirements to the generation of the *sc* in terms of plan-/and intention recognition. In contrast to many other approaches to plan recognition like diagnosis we are not interested in a single hypotheses which tries to predict future behavior as precisely as possible (which appears to be least extremely difficult due to the indeterministic nature). Instead we require a plan/intention recognition process that accounts for,

Different levels of granularity: A precise (but therefore possibly incorrect) prediction is usually not necessary. Assume an intelligent environment wants to provide support by pre-heat the oven. In this case we do not want to know whether he is making (marinated beef or meat loaf). The level of granularity strongly depends on the assistance services available and may differ significantly with respect to the specific situations[3].

[2] *sc* for short

[3] Can be handled at the representational level.

Focused Recognition: Additionally, we are not interested in any prediction. We need a goal directed process that allows on the recognition of actions and intentions we are interested in, e.g., identifying certain risks (e.g. in child care scenarios) or identifying scenarios that allow for active support.

Explanatory coherence: The intelligent environment should be able to explain not only what it attempts to do but also *why*. This is especially important in supporting elderly or handicapped people. An explanation is additionally useful to find out whether the provided support is accepted in future situations, e.g., in order to improve user profile.

Time Efficiency: Efficiency may play a crucial role depending on the specific area of application. Especially in risk avoidance scenarios efficiency is an essential property.

Incrementality: Generating explanations should be designed as an incremental process.

Based on the RoboCup-domain which provides us an interesting scenario for a highly dynamic scenario with competitive as well as cooperating intentional agents we will show how plan recognition can applied incrementally and efficiently.

3 Related Work

The generation of explanations of observed events/behavior has gained much interest within the research community and resulted in various applications like diagnosis [9], natural language understanding [5] and plan recognition [2]. The methods under consideration vary from probabilistic methods like bayes networks [1], classification-based approaches [3] to the already mentioned logic-based abductive methods [2]. One aspect most of these approaches have in common are the constraints of their application: most approaches focus on static scenarios with an precise model of behavior (e.g., *closed world assumption*). An popular exception is the work of [1] which applys probabilistic reasoning to an online-dungeon game but also grounded on a complete (predefined and also limited) number of actions and valid combinations. An additional approach that overcomes these limitations at least to some extent is the work of Intille and Bobick [6] who apply their classification-based approach to the multi-agent scenario of *American football*-domain. Although the *American football*-domain appears to be highly related to the *RoboCup*-scenario the differences (within the Intille-approach) are significant. The *American football*-domain provides a complete, predefined taxonomic playbook which specifies all possible (allowed) patterns of behavior and therefore allows for classification-based approaches. Additionally, they are able to use manually generated data without noise. In contrast, in the soccer domain tactics and strategies describe behavior on much higher level that allows a wide range of variations that cannot be specified in all detail. Furthermore, the given observations have to be assumed to be uncomplete due to sensor limitation.

An alternative, logic-based approach that has been proposed for a wide range of applications is abduction (see above). Abduction has been introduced by C.S. Pierce [10] as a third kind of logic inference next to induction and deduction and has gained much

interest in the late 80'th and early 90'th. Abduction does not rely on complete observations but instead supports to infer missing knowledge i.e., premisses. The abductive inference process can be generally be decomposed in two steps:

1. generation of all abductive explanation
2. selection of the most appropriate explanation

Several proposals have been made for the time consuming generation process depending on the underlying representation[4]. A serious problem for the use of abduction especially in time critical applications like RoboCup is that the generation of abductive explanations has been proofed to be NP-hard (in the general case - some time ago) [11]. Nevertheless, more recently Eiter et al. [4] developed an efficient algorithm that allows to generate all explanations of positive queries based on a logic horn-clause representation.

4 Abduction-based Generation of Explanations of Tactical Behavior

Observations are essential elements for plan recognition. In this approach the necessary information are recognized actions from a team of soccer players. The next section briefly describes the used process to gather these observations. Thereafter the usage of them for explaining tactical behavior is described.

4.1 Qualitative Action Recognition

In the following monotonicity based and threshold based qualitative propositions are distinguished (cf. [7]). Both *types of propositions* share several common properties, as the moment they have been satisfied for the first time, further called *StartTime* and the moment they stopped being satisfied, further called *FinishTime*. Additionally a copy of the world model is stored at *StartTime* and at *FinishTime*, further called *StartTimeWorldModel* and *FinishTimeWorldModel*. Thereby a proposition is defined as a tupel *Proposition[StartTime, FinishTime, StartTimeWorldModel, FinishTimeWorldModel, Type]*. *Monotonicity based Propositions* are satisfied if either a sequence of given values are monotonically increasing or monotonically decreasing, depending on their specified type through the proposition. E.g. the *ApproachingBall* proposition is satisfied as long as the distance from an agent to the ball is continues decreasing. *Threshold based Propositions* are satisfied as long as a given sequence of values under-run a specified threshold, e.g. if the distance from an agent to the ball does not overrun a threshold of 4 meters, the BallDribbleRange proposition is fulfilled. Figure 4.1 illustrates used propositions with their type and describtion.

The given propsitions are a qualitative description of the world which allow all considered actions to be defined as in expression 1 - 4. Each action inherits the common properties of the previously specified propositions, *StartTime*, *FinishTime* and so on.

[4] Abduction does not necessarily have to rely on an logic representation.

qual. proposition	type	satisfied when
ApproachingBall	Monotonically Decreasing	decreasing distance from agent to ball
DisapproachingBall	Monotonically Increasing	increasing distance from agent to ball
BallKickable	Threshold	ball is kickable for agent
BallDribbleRange	Threshold	ball is near agent
KeepingDirection	Threshold	minor direction changes of agent
Stopped	Threshold	agent not moving

Fig. 1. Used qualitative propositions for action recognition

According to the actions' name the GetBall condition (1) recognizes if an agent becomes the ball carrier:

$$ApproachingBall \wedge BallKickable \wedge \neg BallOwnedSoon_{LastValue} \tag{1}$$

The Dribble condition (2) recognizes if an agent dribbles the ball:

$$\begin{aligned} &BallDribbleRange \wedge (ApproachingBall_{StartTime} \\ &\geq BallDribbleRange_{StartTime}) \wedge (DisapproachingBall_{StartTime} \\ &\geq BallDribbleRange_{StartTime}) \end{aligned} \tag{2}$$

The pass condition (3) recognises if possibly a pass has been performed:

$$\neg BallDribbleRange \wedge DisapproachingBall \wedge BallOwnedSoon \tag{3}$$

The Move condition (4) recognises movements:

$$KeepingDirection \wedge \neg BallDribbleRange \wedge \neg Stopped \tag{4}$$

The common properties of the action conditions are assigned by checking the action conditions' satisfaction. *StartTime* and *StartTimeWorldModel* are assigned at the moment the action condition becomes satisfied, accordingly *FinishTime* and *FinishTimeWorldModel* are assigned at the moment the action condition stops beeing satisfied. At this time an action has been identified and temporal segmented. E.g. the *StartTime* of a move action is set to the first time, the regarded agent is keeping his direction, no ball is in dribble range and it is still moving (cf. 4). Analogical the *FinishTime* is set to the time the agent stops keeping its direction, a ball is getting near the agent (in dribble range) or the agent stops walking. For further application the identified actions are stored to a list. With the propositions' copys of the world model at *StartTime* and at *FinishTime*, it is ensured that all needed information is available to determine necessary parameters of the propositions.
In order to complete recognitions and delete unnecessary information, two *Modificators* have been implemented. *Modificators* are modules working on the previously created list of recognized actions. The first modificator is the *Pass Modificator*, searching in the list of actions for pass actions and get ball actions in a temporal relationship to complete the recognized pass action with information about the pass receiver, e.g. if it has been recognized that agent 4 performed a pass and agent 5 a get ball back-to-back, the pass action will be updated by the pass destination, in this case agent 5 and its position while

performing the get ball. Another modificator is the *Cut Modificator* that cuts off all gathered recognized actions when the ball is possessed by the team that is not regarded. This avoids the list of recognized actions expanding too much by deleting unnecessary information. After accomplishing this procedure all necessary information have been produced for following processing.

4.2 Generation of Tactical Knowledgebase

Our knowledge base is based on the book of soccer tactics from Lucchesi [8]. The key assumption which is essential for the use of abductive reasoning is that the (logical) implication can be interpreted not only as an inference from *cause to effect*, but also as an inference from *effect to cause*[5]. Following this pattern each single action within a complex tactical pattern can be interpreted as the cause for a possible sequence of successional actions and may itself be the effect of a previous action and therefore sequences of actions are modeled strictly as sequences of implications. The situation depicted in figure 2 is in the first step described as a sequence of implications (see figure 4) which is in the second step transfered into a *horn* representation[6]. Two limitations had to be considered (1) no cyclic horn theories and (2) no expressions like $x \rightarrow 0$ or $1 \rightarrow x$ are allowed[7]. The resulting horn-clause knowledge base is described in figure 4.

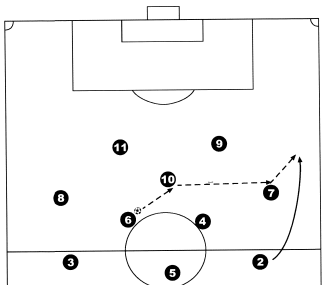

Fig. 2. A left-right Sidechange of a Human Soccer Tactic

(1) {¬PassPlayer6ToPlayer10, ¬MovePlayer2ToRightOppMid, RecievePassPlayer10}
(2) {¬RecievePassPlayer10, PassPlayer10ToPlayer7}
(3) {¬PassPlayer10ToPlayer7, RecievePassPlayer7}
(4) {¬RecievePassPlayer7, PassPlayer7ToPlayer2}
(5) {¬PassPlayer7ToPlayer2, RecievePassPlayer2}
(6) {¬HaveBallPlayer6, ¬IsFreePlayer10, PassPlayer6ToPlayer10}
(7) {¬IsFreeRightOppMid, MovePlayer2ToRightOppMid}
(8) {¬HaveBallPlayer10, ¬IsFreePlayer7, PassPlayer10ToPlayer7}
(9) {¬HaveBallPlayer7, ¬IsFreePlayer2, PassPlayer7ToPlayer2}

Fig. 3. Extracted Clauses from the Tactic.

[5] It should be mentioned that reasoning from *effect* to *cause* is *non-monotonic*.
[6] Due to space limitation we leave out the corresponding horn representation (quite straightforward)
[7] The first condition is a prerequisite of the algorithm of [4], the latter helps to ensures a consistent knowledge base

(1) PassPlayer6ToPlayer10 ∧ MovePlayer2ToRightOppMid → RecievePassPlayer10
(2) RecievePassPlayer10 → PassPlayer10ToPlayer7
(3) PassPlayer10ToPlayer7 → RecievePassPlayer7
(4) RecievePassPlayer7 → PassPlayer7ToPlayer2
(5) PassPlayer7ToPlayer2 → RecievePassPlayer2
(6) HaveBallPlayer6 ∧ IsFreePlayer10 → PassPlayer6ToPlayer10
(7) IsFreeRightOppMid → MovePlayer2ToRightOppMid
(8) HaveBallPlayer10 ∧ IsFreePlayer7 → PassPlayer10ToPlayer7
(9) HaveBallPlayer7 ∧ IsFreePlayer2 → PassPlayer7ToPlayer2

Fig. 4. Extracted Implication from the Tactic

4.3 The basic Algorithm

The basic algorithm can be decomposed into two main steps: (1) calculation of *prime implicants* and (2) calculation of abductive explanations[8] The first step results in a knowledge base of *prime implicants*.

In our example all clauses are also *prime implicants*. The following generations of abductive explanations will be done on this representation. The general idea is quite simple: in the first step the positive request clause σ is used to look up in the *consequences* of the set of all *prime implicants* (in the following *pi*). If σ is found in some prime clause ρ the corresponding *antecedents* of ρ as annotated as the first solution. Based on the first (simple) solution the algorithm tries systematically to find a more fundamental explanation by trying to find a new resolvent between ρ and some different clause in the set of *pi*. Given a new resolvent ρ is found *pi* is expanded by ρ and the same procedure is applied to *pi'* until no changes occur and therefore all solutions have been calculated.

4.4 Optimizing the Algorithm of Eiter and Makino

Although the described algorithm has not only been proved to be complete and correct it is also the only abductive algorithm known to be efficient (non NP-hard). In contrast to various different approaches to the generation of explanations an abductive algorithm is very robust with respect to redundant actions which is a serious problem in soccer in general[9]. Furthermore, an abductive algorithm can easily be applied to different knowledge bases at different levels of granularity. Nevertheless, efficiency is still a very serious problem for the application in a *RoboCup*-domain[10]. In order to improve efficiency various optimizations have been applied. It should be noted that the following

[8] Due to the space limitations we will only describe the more essential second part shortly. For more details please refer to [4].

[9] Even if a team is strictly using declarative tactical patterns of behavior, redundant actions result due to necessary adaptations as a result of unexpected opponent behavior

[10] On a more complex knowledge base the basic algorithm took 21 sec.! As we will see in section 5, with all optimization and with some restrictions the performance can be improved to 22ms.

Algorithm OPT-EXPLANATIONS;
Input: A Horn CNF φ , a positive letter q and Observation *obs*
Output: All nontrivial explanations of q from φ
Step 1. φ^* := $\emptyset$, S := $\emptyset$, and O := $\emptyset$;
OPT1.
if (*JustificationStruct* $\in \varphi$) $\wedge$ ($solutions\,(q) \in Justification Struct$) **then**
    S = $solutions\,(q)$;
    return;
end
Step 2.
OPT2.
if *usePreBuildPrimes = true;*
then φ^* = PreBuildPrimes;
else foreach c $\in \varphi$ **do**
    add any prime implicate *c'* $\subseteq$ *c* of φ to φ^*;
end
OPT3.
Mapping (q, φ^*) ;
foreach c' $\in \varphi^*$ *with* P(c') = $\{q\}$ *and* N(c') $\notin$ S **do**
    output *N(c')*;
    OPT4.
    S := S $\cup$ $\{N(c')\} \setminus$ *obs* ;
    O := O $\cup$ {(c,c') | c $\in \varphi^*$ };
end
Step 3. while *some* (c_1,c_2) $\in$ O *exists* **do**
    O:=*O* $\setminus$ $\{(c_1,c_2)\}$;
    if *(1)* q $\notin$N(c_1);
    (2) P(c_1) = $\{r\} \subseteq$ N(c_2);
    OPT5.
    if *useSatisfyTest = true* **then**
        (3) φ^* $\cup$ N(c_1) $\cup$ N(c_2) $\setminus$ P(c_1) *is satisfiable*
    end
    then
        c := resolvent of c_1 and c_2;
        compute any prime implicate c' $\subseteq$ c of φ;
        if N(*c'*) $\notin$ S **then**
            output *N*(c');
            OPT4.
            S:= S $\cup$ $\{N(c')\} \setminus$ *obs* ;
            O := O $\cup$ {(c,c') | c $\in \varphi^*$}
        end
    end
end

Algorithm 1: (Optimized) Eiter et al. algorithm for generating all explanations

optimizations may not be reasonable in all domains. I.e., although the optimized algorithm is (of course) domain-independent the optimizations can only be applied with some restrictions. Therefore they are optional with respect to different domains[11].

The complexity of the algorithm is given by *O(e * m * n * $\|\varphi\|$)*, whereas e denotes the number of solutions, m the number of clauses in φ and n the number of literals. The first significant improvement can be achieved by separating the complete knowledge base into different separat knowledge bases. As a matter of consequence, we get a special knowledge base for *counter attack on the right wing*, *counter attack on in the center*, ... This modularisation has an interesting advantage: as assumed in the motivation (see section2) a player is usually only interested in explanations that leads to an improved/adopted behavior: E.g., a right wing defender is specially interested in counter attack on the right wing and not on the left one. The modularisation allows him to focus on context sensitive, role specific tactic explanations.

Additionally, four different optimizations have been applied:

1. Pre-calculation of prime implicants: Independently of the specific request the basic algorithm calculates all prime implicants. This process is done before runtime in our realization and just has to be loaded together with the knowledge-base. The level of improvement is strongly dependent on the complexity of the causal relations and will lead at least in complex models to significant improvements.

2. Use of additional observations: In most of the cases a player has made different observations that account to a specific tactic. These observations can be used to improve the abductive reasoning process by skipping proofs. Since a player has already observed an action it is not necessary to find out under which conditions the observation is true.

3. Skipping satisfiability-test: The above handling of observations has an additional interesting side effect: In the classic abductive approach new observations are expanded in the knowledge base which in case of false observations *may* lead to an inconsistency. In order to avoid to an inconsistent knowledge base a satisfiability-test would be needed. But since observation are never expanded in the knowledge base the satisfiability-test can be skipped.

4. Pre-calculation of possible solutions: A static knowledge base offers additional advantages: it allows the pre-calculation of all possible solutions! A possible disadvantage can be the increase in memory usage which is minimal in this domain due to the modularisation of the knowledge base.

In addition to the improvements in efficiency the basic algorithm had to be adopted in order to increase robustness. Although tactical pattern described in Lucchesi [8] are strictly associated with specific tactical roles these assumption does rarely hold in the *RoboCup*-domain. Therefore we provided the abductive algorithm with a flexible role association method. The use of this method allows to detect tactical behavior independently from specific player numbers or roles and increases the robustness significantly. The obvious drawback is a decrease in efficiency since all player-number configurations have to be considered in the role assignment. The preliminary results of our extentions are described in the following section. The detailed modified algorithm is depicted in algorithm 1.

[11] And they are also optional in our implementation.

5 Experiments and Preliminary Results

Before we tried to integrate the modified algorithm in our 3D-team we evaluated the efficiency in different scenarios. In the first test scenario described in table 1 we wanted to evaluate the effect of pre-calculated solutions under (1) the varying condition of request complexity: simple vs. complex and (2) under varying goal: whether the agent wants to know if a plan is possible at all or whether he wants to get a list of all possible (opponent-) plans. The request complexity is simply changed by the action selection. In the case we observe an action that appends very late in a possible plan, the algorithm will find significant more solutions than in the inverse case. In the following tests we used 26 different plans in each case. The table 1 presents some interesting results.

Test 1	**with Pre-Calculation**		**without Pre-Calculation**	
	simp. Query	**komp. Query**	**simpl. Query**	**komp. Query**
all Time	62,8 ms	55,7 ms	69,3 ms	74,2 ms
∅ **Time for one Plan**	4,7 ms	4,5 ms	6,3 ms	10,5 ms
processed Plans	4 P.	3 P.	4 P.	3 P.
∅ **compute Possible Plans**	44,2 ms	42 ms	43,8 ms	42,2 ms

Table 1. Finds all Solution and search all Plans with a possible Solutions.

First, the calculation whether a single plan is possible is highly efficient and can be done in 4,7 ms to 10,5 ms with respect to the specific conditions. Interestingly, the pre-calculation of results is significantly more efficient than without but the difference is surprisingly small. Two main reasons can be found: (1) the complexity of our tactical model is quite low (in terms the capability of the algorithm). The efficiency decreases in the case of no pre-calculations only for complex requests. (2) The modularisation of the knowledge base appears to be highly efficient. In the case that all possible plans should be calculated (which represents the case of non-modularisation) the run-time requirements are significant higher. - But still efficient enough to be used e.g., in the 3D-simulation league, as it can be seen in table 2. In table 2 we used a 3D-trainer agent

	iterativ Test									
	all Queries					**Queries with Solutions**				
	min	**max**	∅	∅ **over**	**Tests**	**min**	**max**	∅	∅ **over**	**Tests**
P28	2	7	4,66	0,6	22,6	2	7	4,66	0,6	22,6
P53	0	4	1,06	6	182,2	1	4	2,62	6	55,8
P99	0	7	2,24	1,2	49,2	1	7	3,96	1	24,8
P130	0	6	2,6	0,6	53,4	2	6	4,6	0,2	16,6

Table 2. Used Cycles in 2D-League, watch a game with a Traineragent in 3d-League

who has been restricted to use at maximum 80ms in order to simulate cycles. In the test

the coach was required to detect whether a single plan is possible and to find all possible solutions. These conditions have been tested on four different varying plans (P28, P53, P99, P130). The test mainly showed two important results: (1) The modularisation of the knowledge base provides a basis for incremental abductive reasoning. The algorithm in our implementation can interrupt the calculation process at (relative) fixed time steps in order to allow other tasks within a single cycle (instead using complete cycles). (2) Depending on the specific condition the algorithm requires at most between 4 to 7 cycles for all solutions. These results can also be approved under different conditions e.g., used by a 3D- or a 2D-player. The detailed description of all generated results is out of the scope of this paper (space limitations). - If the reviewers consider one of the other mentioned results more interesting then we will integrate them.

6 Summary and Discussion

The role of strategy and tactic is becoming more and more important especially in the simulation- and the small-size league. The use of more complex tactics of an offensive team will require that defensive teams are at least to some extent able to detect the set of possible opponent tactical patterns in order to coordinate defensive behavior. Nevertheless, the use strategical and tactical knowledge is not limited to a specific domain but is instead relevant in any domain that is related to the cooperative and/competitive dynamic interaction between natural and artificial agents.

In this paper we presented an approach to symbolic plan recognition (more precisely generation of explanation for opponent behavior) at all relevant stages: from the generation of qualitative action- and world descriptions based on [7] to the generation of abductive explanations of these observed behavior. The algorithm of Eiter and Makino [4] has been adopted to the specific requirements of highly dynamic domains. We showed that the adopted algorithm can efficiently be used for explanation generation. Furthermore, the algorithm can be used in an incremental fashion which is especially useful for monitoring scenarios. An additional characteristic is the robustness with respect to redundant/false observations/actions which have to be handled in any physically grounded scenario. The modularisation of the knowledge base allows for role- and context sensitive requests but does not prohibit the generation of complete solutions, i.e., without respect to role and context e.g., for the trainer.

Besides the application of the modified algorithm in different domains the hypotheses generation is still an open task. Although we claim that it will be sufficient in many situations to identity possible tactical behavior (with respect to role and context) it is clear that there also exits situations where we would like a single prediction, i.e., the selection of a single hypothesis out of the set of possible explanations. Various solutions may be considered, varying from probabilistic to symbolic approaches proposed in the abduction community [9].

References

1. David W. Albrecht, Ingrid Zukerman, and An E. Nicholson. Bayesian models for keyhole plan recognition in an adventure game. *User Modeling and User-Adapted Interaction*, 8(1-2):5–47, 1998.

2. Douglas E. Appelt and Martha E. Pollack. Weighted abduction for plan ascription. *User Modeling and User-Adapted Interaction*, 2(1-2):1–25, 1991.
3. Dorit Avrahami-Zilberbrand and Gal A. Kaminka. Fast and complete symbolic plan recognition. In *IJCAI*, pages 653–658, 2005.
4. Thomas Eiter and Kazuhisa Makino. On computing all abductive explanations. In *Eighteenth national conference on Artificial intelligence*, pages 62–67, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.
5. Jerry R. Hobbs, Mark Stickel, Paul Martin, and Douglas D. Edwards. Interpretation as abduction. In *26th Annual Meeting of the Association for Computational Linguistics: Proceedings of the Conference*, pages 95–103, Buffalo, New York, 1988.
6. S. Intille and A. Bobick. Recognizing planned, multi-person action. In *Computer Vision and Image Understanding*, volume 81, pages 414–445, 2001.
7. Andreas D. Lattner, Andrea Miene, Ubbo Visser, and Otthein Herzog. Sequential pattern mining for situation and behavior prediction in simulated robotic soccer. In *RoboCup*, pages 118–129, 2005.
8. Massimo Lucchesi. *Choaching the 3-4-1-2 and 4-2-3-1*. Reedswain Publishing, edizioni nuova prhomos edition, 2001.
9. Hwee Tou Ng and Raymond J. Mooney. On the role of coherence in abductive explanation. In *National Conference on Artificial Intelligence*, pages 337–342, 1990.
10. Charles Sanders Peirce. *Collected Papers of Charles Sanders Peirce*. Harvard University Press, 1931.
11. Bart Selman and Hector J. Levesque. Support set selection for abductive and default reasoning. *Artif. Intell.*, 82(1-2):259–272, 1996.