STUDIA INFORMATICA
Nr 1/2(7) Systemy i technologie informacyjne 2006

Innovative Realization of Quantitative Goals in BDI Agents via Partial Utility Functions

Michał Przybylski[1], **Paweł Cichocki**[2]
[1] Faculty of Mathematics, Informatics and Mechanics, Warsaw University, Banacha 2, 02-097 Warsaw, Poland
[2] Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland

Abstract. Agents play an important role in high level artificial intelligence in such areas as distributed decision support, robot control, computer games, etc. Currently, the most popular high-level agent architectures are based on the belief-desire-intention (BDI) model. BDI agents are usually specified in modal logic. This is efficient for defining event goals. However, defining quantitative goals can be very difficult in many popular formalisms. In this paper we propose a method for expressing quantitative goals by associating partial utility functions with agent's goals. We propose a modified BDI agent architecture which is loosely based on fuzzy logic. In this architecture, approximation of partial derivatives of those functions enables us to use gradient based optimization algorithms in the intention reconsideration step to weight some action specializations. Using the proposed approach allows us to easily combine quantitative and event goals, and consider them all while planning. This paper also describes a simple language which can be used to elegantly describe generic action libraries in accordance to the proposed model.

Keywords. AI, Agents, BDI, Planning, Intention reconsideration, Goals, Optimization

1 Introduction

Agents are often used to control single entities in a dynamic environment. This is especially true in games and robotics [9], [30]. Probably the most popular high-level agent architecture is the *belief-desire-intention* (BDI) model. The BDI model is a philosophical model introduced by Michael Bratman [6], [7]. Strong formal logic models were later developed by Cohen and Levesque [4], [5]. Another approach was presented by Rao and Georgeff [1], [2].

Since Bratman first described the BDI model, many publications with slightly different interpretations of beliefs, desires, and intentions [4], [6], [10], [19] have been published. Let us briefly explain how we understand, utilize, and extend those concepts in this paper.

An agent has a knowledge base (KB) containing a set of facts that is not always complete or fully correct, called *beliefs*.

The agent's KB also contains a set of *goals* or *desires* it tries to fulfil. Some researchers distinguish between goals and desires, stating that goals can contradict one another, while desires need to be consistent [15]. Our method was designed to handle contradictive goals with no additional cost. In this article, we assume that an agent already has a set of goals and we do not deliberate here on their origin, which is a broad topic, beyond the scope of this paper. Discussion on this topic can be found, inter alia, in [3], [21].

To achieve its goals an agent has a set of available *actions*. Each action can be *specialized* to be realizable in different ways or states of the environment.
Another important concept in BDI systems is a *plan*. It describes a few alternative action sequences that should be executed depending on the circumstances. By a *specialized plan* we will understand a fixed chain of specialized actions.

Next, by an *intention* we understand the fulfillment of a goal by executing a specialized plan. The process of choosing which goal to realize next and creating a specialized plan is called *intention reconsideration*.

A graphic interpretation with dataflow marked by arrows can be found in Figure 1.

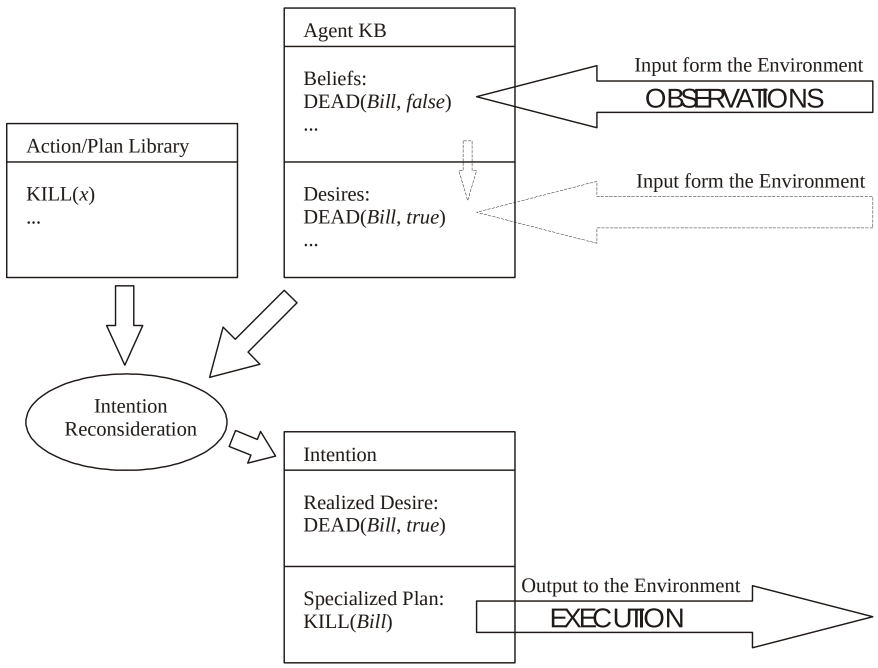

Figure 1. BDI interpretation in our system

Much work is devoted to improve intention reconsideration [17], [20], [21], which is one of the major problems in agent systems. Many sophisticated methods for creating plans to achieve binary event goals, have been created. By *event goals* we mean goals that can be achieved by obtaining a specific state of the environment for example obtaining an object. However, very little attention has been devoted to describing *quantitative goals* such as maximizing money, minimizing cost, or keeping a moderate speed while minimizing fuel consumption. Optimization is often pushed to a lower implementation level (for example the keeping of a moderate speed and minimizing fuel consumption would usually be put into the code of a high-level travel action). This makes the logic of the agent simpler but has several serious disadvantages:

- quantitative goals are described on a different level than event goals,
- no planning can be done for quantitative goals,
- weighting the priorities of goals cannot be done in a consistent manner.

Another problem with the orthodox modal logic approach is that it is not flexible. Everything can either be true or false. However, the uncertainty of various events can be approximated and used. It seems to be inappropriate to treat two plans, one based on very reliable beliefs and the other based on very vague beliefs, in the same manner.

2 Problem Definition

Real world examples often require us to define quantitative goals. Many problems can be viewed as optimization problems where the value of a function of some quantitative variables needs to be optimized. Such goals are not easily defined in a framework based on modal logic. While expressing a desire for a variable to attain a certain value is still relatively easy, simple maximization requires using more complex concepts such as persistent goals or achievement goals defined by Cohen and Levesque [4]. For example, consider a situation where an agent wants to maximize its savings. Such a goal can be expressed as in [33]: "I always want more money than I have"

$$\begin{aligned}
&\text{LATER } p = \neg p \wedge \Diamond p \\
&\text{A-GOAL } x\ p = \text{GOAL } x\ (\text{LATER } p) \wedge \text{BEL } x\ \neg p \\
&\text{GOAL } I\ \exists\, x, y\ (\text{HAVE } I\ x) \wedge (y > x) \wedge (\text{LATER } (\text{HAVE } I\ y)) \\
&\forall x (\text{KNOW } I\ (\text{HAVE } I\ x)) \Rightarrow \text{A-GOAL } I\ \exists y\ ((y > x) \wedge (\text{HAVE } I\ y))
\end{aligned}$$

Another approach could be to add a goal “Have more money than *b*” and once this goal is achieved a similar goal with a greater value *b* must be added, and so on.

However, there still remain the following questions:

- how important should such a goal be in comparison to other coexisting event goals,
- should the importance of such a goal change over time and how,
- how to optimize a function of more than one variable,
- how to decide which variable to optimize with the next action,

- how to perform or define more complex optimization than minimization/maximization.

Generally the problem of choosing between two plans that both lead to the fulfilment of the same desire can be well addressed by using fuzzy logic [29], [31], [32]. Using weights in BDI systems has been already described in many publications, for example [3], [20], [28]. Other examples of using fuzzy logic in conjunction with BDI include [13], [30].

3 Our Proposal

3.1 The Agent Model

Our implementation relies heavily on a simplified fuzzy formula structure which we describe using the following grammar:

```
fuzzyformula ::= formula ","
                 probability
                 "and"
                 fuzyformula
               | formula ","
                 probability

formula ::= predicate
          | formula "or"
            formula
          | formula "and"
            formula
          | "not" formula

predicate ::= name(varlist)

varlist ::= var "," varlist
          | var

var ::= STRING
```

In our current implementation there can be no alternatives within a fuzzy formula. This allows for a much simpler satisfaction probability calculation which we describe in this paper.

Our implementation has an action library described in the following manner:

```
action ::= name
           "need:" need
           "evaluation:"
           evaluation
```

```
            "effect:" effect
            "expected time:"
            CONSTANT
            "execution: "
            SCRIPT_CODE

name ::= STRING

need ::= formula need
       | ε
evaluation ::= evaluation
               EXPRESSION
             | ε

effect ::= fuzzyformula
           effect
         | fuzzyformula
```

We also describe an agent's KB:

```
agent_description ::=
            AGENT "believes: "
            beliefs
            AGENT "goals: "
            desires

beliefs ::= fuzzyformula
            beliefs
          | ε

goals ::= goal goals
        | ε

goal ::= formula ","
         importance

importance ::= CONSTANT
             | FUNCTION
```

We bind a function we call the *partial utility function* with quantitative goals. This function should describe how an agents utility changes with the change of parameters that are directly affected by an action. The utility function can be expressed as a sum of partial utility functions [34], [35].

In our model each agent can have its own set of actions or they can share a common action/plan library.

3.2 Intention Reconsideration

Let us assume that we have a correct derivation system. Now we can define derivation of a fuzzy formula set Δ by a set of fuzzy formulas Γ, $D(\Gamma, \Delta, v_0)$ as an operation returning a tuple $\langle v, p, s\rangle$, where v is the most general substitution needed to derive Δ from Γ in this system, p the probability that Δ is consistent with Γ, s is a boolean stating whether Δ can be derived from Γ at all, and v_0 is some initial substitution. Of course $s=0 \Rightarrow p=0$ and v_0 is a more general substitution than v.

Let us consider one step of planning, that is choosing the best action specialization for a single action plan only. Firstly, for each action the agent calculates the probability stating the chances of successful completion of that action, which is the probability that the preconditions or *needs* of the action are satisfied by the agent beliefs obtaining

$$\langle v_{a_n},\ p_{a_n},\ s_{a_n}\rangle = D(bel,\ a_n,\ v_p),(1)$$

where *bel* is the set of the agents beliefs and v_p is either Ø or some substitution from an earlier planning step. Then for each action where p_{a_n} is greater than 0 by a small threshold value we try to satisfy each goal (the weight of the formula is assumed one if its importance is a function) by the action *effects* using the partial substitution v_{a_n}. Thus we obtain

$$\langle v_{a_e d},\ p_{a_e d},\ s_{a_e d}\rangle = D(a_e,\ d,\ v_{a_n}).(2)$$

Let us introduce a weight for each action specialization we obtain from each action template - goal pair

$$w(a, d) = p_{a_n}\ p_{a_e d}\ \left.\frac{\partial u_d}{\partial t}\right|_{t=T}. \quad (3)$$

If there is a possibility that the specialization of an action A consisting of action a_A and specialization v_A will be generated more than once with nonzero weights we should sum up the corresponding weights

$$w(A) = \sum_{d \in \{d : v_{a_e d} = v_A\}} w(a_A, d)\,. \quad (4)$$

If the partial utility functions are not orthogonal and the execution of an action specialization does affect the values of partial utility functions associated with other goals the weight an action specialization A receives can be calculated as

$$w(A) = \sum_{d \in des} p_{a_n} p_{a_e d} \left. \frac{\partial u_d}{\partial t} \right|_{t=T}, \quad (5)$$

where *des* is the set of all goals. Assuming that the change of parameters is linear during action execution $\partial u_d / \partial t \big|_{t=T}$ can be approximated simply as the difference between the values of the partial utility function before and after action execution. For event goals we simply ignore $\partial u_d / \partial t \big|_{t=T}$.

4 Results

4.1 Sample Problem

An agent *Salomon* has to manage a kingdom consisting of cities *A* and *B*. His goal is to maximize the profit in taxes. The cities mine silver and gold from which they pay taxes. The amount of mined silver and gold depends on the level of respective mines. However, the deeper the mine is dug the smaller the increase in its yield.

Let us define the yields of silver and gold mines respectively as

$$\begin{aligned} y_s(l) &= 2\sqrt[3]{l} \\ y_g(l) &= \sqrt{l} \end{aligned}, \quad (6)$$

where *l* is the level of the mine. Further, assume a city can be loyal to the king or not. If the city is loyal it pays taxes. If not, it hides its income from the king and little or no tax income is generated for the kingdom, thus the partial utility functions are

$$\begin{aligned} u_s(c,l) &= \mathrm{LOYAL}(c, true)\, y_s(l) \\ u_g(c,l) &= \mathrm{LOYAL}(c, true)\, y_g(l) \end{aligned}, \quad (7)$$

where *c* is the city, *l* is the level of the respective mine and LOYAL is *true* only when the second term is consistent with the loyalty of the city. Note that in our system the evaluation of the predicate would rather return a probability than binary 0 or 1.

For the sake of simplicity, let us assume the king can perform only two actions: tell his craftsmen to extend a gold mine in a city, or tell his craftsmen to extend a silver mine in a city. Thus, the action library would be:

```
EXTEND_SILVER_MINE
need:
 SILVER(city, level)
evaluation:
 newlevel←level+1;
effect:
 SILVER(city, newlevel), 0.9
```

```
expected time:
 1
execution:
 world.buildSilverMine(city);
EXTEND_

GOLD_MINE
need:
 GOLD(city, level)
evaluation:
 newlevel←level+1;
effect:
 GOLD(city, newlevel), 0.9
expected time:
 1
execution:
 world.buildGoldMine(city);
```

SILVER(c, l) and GOLD(c, l) are true only if the level of the respective mine in city c is equal to l.

4.2 Sample Solution

Assume the agent *Salomon* starts with the following KB:

```
Salomon beliefs:
 SILVER(A, 0), 1
 GOLD(A, 0), 1
 LOYAL(A, true), 1
 SILVER(B, 0), 1
 GOLD(B, 0), 1
 LOYAL(B, false), 1

Salomon goals:
 SILVER(A, l),us(A, l)
 GOLD(A, l), ug(A, l)
 SILVER(B, l), us(A, l)
 GOLD(B, l), ug(B, l)
```

Let us now see how *Salomon* would choose an action specialization.

Table 1. All possible action specializations in the first step

Action Desire	Build Silver Mine	Build Gold Mine
SILVER (*A*, *l*), u_S	v_{11}={A/city, 0/level, 1/newlevel}, w_{11}=1.8	v_{12}={A/city, 0/level, 1/newlevel}, w_{12}=0
GOLD (*A*, *l*), u_G	v_{21}={A/city, 0/level, 1/newlevel}, w_{21}=0	v_{22}={A/city, 0/level, 1/newlevel}, w_{22}=0.9
SILVER (*B*, *l*), u_S	v_{31}={B/city, 0/level, 1/newlevel}, w_{31}=0	v_{32}={B/city, 0/level, 1/newlevel}, w_{32}=0
GOLD (*B*, *l*), u_G	v_{41}={B/city, 0/level, 1/newlevel}, w_{41}=0	v_{42}={B/city, 0/level, 1/newlevel}, w_{42}=0

Assuming that the loyalty of cities will not change, the agent will extend mines only in city *A*. From Tab. 1 we clearly see that the specialization EXTEND_SILVER_MINE with the substitution set containing *A/city* should be chosen. The results of further planning lead the agent to chose a plan as shown graphically in Figure 2.

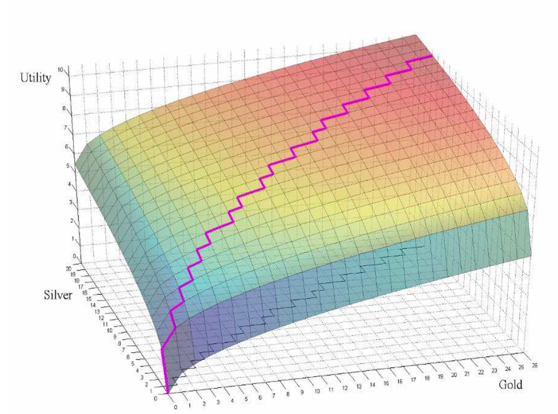

Figure 2. The actions taken by the agent mapped over the utility function

Let us investigate the level of silver and gold mines in city A and the total utility obtained. As we can see in Figure 3, the utility growth obtained by our algorithm is much better than what would be obtained building just silver or gold mines all the time. This scales to much more complex examples.

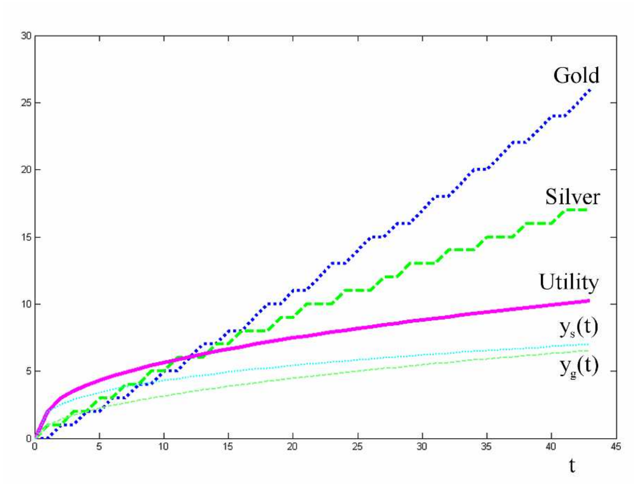

Figure 3. Obtained utility, gold and silver mine amounts as functions of time and partial utility function (yield) plots

5 Conclusion and Future Work

We have presented a novel, elegant method for expressing quantitative goals in fuzzy BDI-like agents. We have shown how our approach can be used to make a BDI agent effectively optimize a few values using an approach corresponding to gradient steepest descendent.

The simplicity and effectiveness of the proposed method encourages us to investigate it further. We are currently experimenting with extending the approach to perform optimization analogous to the Levenberg-Marquardt method.

6 Acknowledgments

The authors are grateful to Professors Barbara Dunin-Kęplicz from Warsaw University and Jan Zabrodzki from Warsaw University of Technology for their continuous advice and support.

References

1. A.S. Rao and M.P. Georgeff, "BDI Agents: From Theory to Practice," *in Proceedings of the First International Conference on Multi-Agent Systems*, 1995, pp. 312-319.
2. A.S. Rao and M.P. Georgeff, "Modelling Rational Agents within a BDI-Architecture", *in Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, 1991, pp. 473-484.
3. R. Evans, *Varieties of Learning, AI Game Programming Wisdom*, Charles River Media, 2002, pp. 567-578.

4. P.R. Cohen and H.J. Levesque, "Intention Is Choice with commitment", *in Artificial Intelligence, Vol. 42*, 1990, pp. 213-231.
5. P.R. Cohen and H.J. Levesque, "Teamwork", *in Nous, Vol. 25, No. 4, Special Issue on Cognitive Science and Artificial Intelligence*, 1991.
6. M. Bratman, "Two faces of intention", *in The Philosophical Review, Vol. 93, No. 3*, 1984, pp. 375-405.
7. M. Bratman, D.J. Israel and M.E. Pollack, "Plans And Resource-Bounded Practical Reasoning", *in Computational Intelligence, Vol. 4, No. 4*, 1988, pp. 349-355.
8. E. Norling and L. Sonenberg, "Creating Interactive Characters with BDI Agents", *in Proceedings of the Australian Workshop on Interactive Entertainment (IE2004)*, 2004.
9. E. Norling, "Capturing the Quake Player: Using a BDI Agent to Model Human Behaviour", *in Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2003, pp. 1080-1081.
10. E. Norling, "Folk Psychology for Human Modelling: Extending the BDI Paradigm", *in Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2004, pp. 202-209.
11. C. Sioutis, N. Ichalkaranje and L. Jain, "A framework for interfacing BDI agents to a real-time simulated environment", *in Design and Application of Hybrid Intelligent Systems*, 2003, pp. 743-748.
12. R. Witte, "Fuzzy Belief Revision," *in Proceedings of the 9th International Workshop on Non-Monotonic Reasoning (NMR'02)*, 2002, pp. 311-320.
13. X. Luo, C. Zhang and N.R. Jennings "A Hybrid Model For Sharing Information Between Fuzzy, Uncertain And Default Reasoning Model In Multi-Agent Systems", *in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2002, pp. 401-450.
14. H.J. Levesque, P.R. Cohen and J.H.T. Nunes, "On Acting Together", *in Proceedings of the National Conference on Artificial Intelligence*, 1990.
15. B. Dunin-Kęplicz and R. Verbrugge, "Collective Intentions," *Fundamenta Informaticae, Vol. 51, No.3*, 2002, pp. 271-295
16. B. Dunin-Kęplicz and A. Radzikowska, "Nondeterministic Actions with Typical Effects: Reasoning about Scenarios", *in Formal Models of Agents*, 1999, pp. 143-156.
17. B. Dunin-Kęplicz and R. Verbrugge, "A Reconfiguration Algorithm for Distributed Problem Solving", *in Engineering Simulation, Vol. 18*, 2001, pp. 227-246.
18. B. Dunin-Kęplicz and R. Verbrugge, "Evolution of collective commitments during teamwork", *in Fundamenta Informaticae*, 2003.
19. M. Xinjun, W. Huaiming, W. Gang and Z. Jianming, "Formal Model of Joint Maintenance Intention," *in Journal of software, Vol. 10, No. 1*, 1999, pp. 43-48.
20. M. Schut and M. Wooldridge, "Principles of Intention Reconsideration", *in Proceedings of the 5th international conference on Autonomous agents*, 2001, pp. 340-347.

21. M. Schut, M. Wooldridge and S. Parsons, "The theory and practice of intention reconsideration", *in Journal of Experimental and Theoretical Artificial Intelligence, Vol. 16, No. 4*, 2004, pp. 261-293.
22. S. Parsons, O. Pettersson, A. Saffiotti and M. Wooldridge, "Intention reconsideration in theory and practice", *in Proceedings of the 14th European Conference on Artificial Intelligence (ECAI-2000)*, 2000, pp. 378-382.
23. M. Dastani and L. van der Torre, "Games for Cognitive Agents," *in Proceedings of the 9th European Conference on Logics in Artificial Intelligence (JELIA'04),* 2004, pp. 5-17.
24. F.R. Meneguzzi, A.F. Zorzo and M. da Costa Móra, "Propositional Planning in BDI Agents", *in Proceedings of the ACM Symposium on Applied Computing*, 2004, pp. 58-63.
25. M. P. Georgeff and F. F. Ingrand, "Decision-Making in an Embedded Reasoning System", *in Proceedings of the International Joint Conference of Artificial Intelligence*, 1989, pp. 972-978.
26. H.S. Nwana, "Software Agents: An Overview," *in Knowledge Engineering Review, Vol. 11, No. 3*, 1996, pp. 205-244.
27. M. Dastani and L. van der Torre, "Programming BOID-Plan Agents deliberating about conflicts among defeasible mental attitudes and plans", *in Third International Joint Conference on Autonomous Agents and Multi-Agent Systems, Vol. 2*, 2004, pp. 706-713.
28. M. Dastani and L. van der Torre "Decisions, Deliberation, and Agent Types CDT – QDT – BDI – 3APL – BOID", *in Focus in Computer Science*, Nova Science, 2005.
29. L.A. Zadeh, "Fuzzy sets", *in Information and Control, Vol. 8*, 1965, pp. 338–353.
30. Y. Li, P. Musilek and L. Wyard-Scott, "Fuzzy Logic in Agent-Based Game Design", *in Proceedings of the Annual Meeting of the North American Fuzzy Information Processing Society*, 2004, pp. 734-739.
31. P. Hajek, *Fuzzy predicate calculus and fuzzy rules, Fuzzy IF-THEN rules in computational intelligence*, Kluwer, 2000, pp. 27-36.
32. V. Novak, I. Perfilieva and J. Mockor, *Mathematical principles of fuzzy logic*, Kluwer 2000.
33. W. van der Hoek, "Logical Foundations of Agents", *in Advanced Course on Artificial Intelligence (ACAI-01)*, 2001.
34. Slodzian and S. Aknine, "Intelligent agents for tracking racist documents on the Internet", *at the Workshop on Intelligent Techniques for Web Personalization, 18th international joint conference on artificial intelligence (IJCAI 03)*, 2003.
35. M. Monticino, M. Acevedo, B. Callicott, T. Cogdill, M. Ji and C. Lindquist, "Coupled Human and Natural Systems: A Multi-Agent Based Approach", *in the Proceedings of International Environmental Modelling and Software Society (iEMSs 2004)*, 2004.