The Belief-Desire-Intention Model of Agency

Michael Georgeff[*] Barney Pell[†] Martha Pollack[‡]
Milind Tambe[#] Michael Wooldridge[×]

[*] Australian AI Institute, Level 6, 171 La Trobe St
Melbourne, Australia 3000
georgeff@aaii.oz.au

[†] RIACS, NASA Ames Research Center
Moffett Field, CA 94035-1000, USA
pell@ptolemy.arc.nasa.gov

[‡] Department of Computer Science/Intelligent Systems Program
University of Pittsburgh, Pittsburgh, PA 15260, USA
pollack@cs.pitt.edu

[#] Computer Science Department/ISI, University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292, USA
tambe@isi.edu

[×] Department of Electronic Engineering, Queen Mary and Westfield College
University of London, London E1 4NS, United Kingdom
M.J.Wooldridge@qmw.ac.uk

1 Introduction

Within the ATAL community, the belief-desire-intention (BDI) model has come to be possibly the best known and best studied model of practical reasoning agents. There are several reasons for its success, but perhaps the most compelling are that the BDI model combines a respectable philosophical model of human practical reasoning, (originally developed by Michael Bratman [1]), a number of implementations (in the IRMA architecture [2] and the various PRS-like systems currently available [7]), several successful applications (including the now-famous fault diagnosis system for the space shuttle, as well as factory process control systems and business process management [8]), and finally, an elegant abstract logical semantics, which have been taken up and elaborated upon widely within the agent research community [14, 16].

However, it could be argued that the BDI model is now becoming somewhat dated: the principles of the architecture were established in the mid-1980s, and have remained essentially unchanged since then. With the explosion of interest in intelligent agents and multi-agent systems that has occurred since then, a great many other architectures have been developed, which, it could be argued, address some issues that the BDI model fundamentally fails to. Furthermore, the focus of agent research (and AI in general) has shifted significantly since the BDI model was originally developed. New advances in understanding (such as Russell and Subramanian's model of "bounded-optimal agents" [15]) have led to radical changes in how the agents community (and more generally, the artificial intelligence community) views its enterprise.

The purpose of this panel is therefore to establish how the BDI model stands in relation to other contemporary models of agency, and where it should go next.

2 Questions for the Panelists

The panelists (Georgeff, Pell, Pollack, and Tambe) were asked to respond to the following questions:

1. *BDI and other models of practical reasoning agents.*
   Several other models of practical reasoning agents have been successfully developed within the agent research and development community and AI in general. Examples include (of course!) the Soar model of human cognition, and models in which agents are viewed as utility-maximizers in the economic sense. The latter model has been particularly successful in understanding multi-agent interactions. So, how does BDI stand in relation to these alternate models? Can these models be reconciled, and if so how?
2. *Limitations of the BDI model.*
   One criticism of the BDI model has been that it is not well-suited to certain types of behaviour. In particular, the basic BDI model appears to be inappropriate for building systems that must learn and adapt their behaviour – and such systems are becoming increasingly important. Moreover, the basic BDI model gives no architectural consideration to explicitly multi-agent aspects of behaviour. More recent architectures, (such as InteRRaP [13] and TouringMachines [5]) do explicitly provide for such behaviours at the architectural level. So, is it necessary for an agent model in general (and the BDI model in particular) to provide for such types of behaviour (in particular, learning and social ability)? If so, how can the BDI model be extended to incorporate them? What other types of behaviour are missing at an architectural level from the BDI model?
3. *Next steps.*
   What issues should feature at the top of the BDI research agenda? How can the relationship between the theory and practice of the BDI model be better understood and elaborated? Programming paradigms such as logic programming have well-defined and well-understood computational models that underpin them (e.g., SLD resolution); BDI currently does not. So what sort of computational model might serve in this role? What are the key requirements to take the BDI model from the research lab to the desktop of the mainstream software engineer?

3 Response by Georgeff

The point I wanted to make in this panel was that the notions of complexity and change will have a major impact on the way we build computational systems, and that software agents — in particular, BDI agents — provide the essential components necessary to cope with the real world. We need to bring agents into mainstream computer science, and the only way we can do that is to clearly show how certain agent architectures can cope with problems that are intractable using conventional approaches.

Most applications of computer systems are algorithmic, working with perfect information. But in a highly competitive world, businesses need systems that are much more complex than this — systems that are embedded in a changing world, with access to

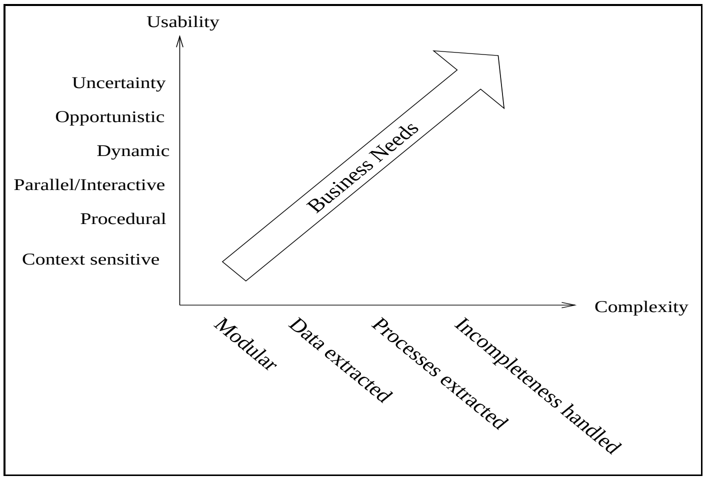

Fig. 1. Business Drivers

only partial information, and where uncertainty prevails. Moreover, the frequency with which the behaviour of these systems needs to be changed (as new information comes to light, or new competitive pressures emerge), is increasing dramatically, requiring computer architectures and languages that substantially reduce the complexity and time for specification and modification. In terms of Figure 1, business needs are driving to the top right hand corner, and it is my contention that only software agents can really deliver solutions in that quadrant.

As we all know, but seem not to have fully understood (at least in the way physicists have) the world is complex and dynamic, a place where chaos is the norm, not the exception. We also know that computational systems have practical limitations, which limit the information they can access and the computations they can perform. Conventional software systems are designed for static worlds with perfect knowledge — we are instead interested in environments that are dynamic and uncertain (or chaotic), and where the computational system only has a local view of the world (i.e., has limited access to information) and is resource bounded (i.e., has finite computational resources). These constraints have certain fundamental implications for the design of the underlying computational architecture. In what follows, I will attempt to show that Beliefs, Desires, and Intentions, and Plans are an essential part of the state of such systems.

Let us first consider so-called Beliefs. In AI terms, Beliefs represent knowledge of the world. However, in computational terms, Beliefs are just some way of representing the state of the world, be it as the value of a variable, a relational database, or symbolic expressions in predicate calculus. Beliefs are essential because the world is dynamic (past events need therefore to be remembered), and the system only has a local view of the world (events outside its sphere of perception need to be remembered). Moreover, as the system is resource bounded, it is desirable to cache important information rather than recompute it from base perceptual data. As Beliefs represent (possibly) imperfect information about the world, the underlying semantics of the Belief component should conform to belief logics, even though the computational representation need not be symbolic or logical at all.

Desires (or, more commonly though somewhat loosely, Goals) form another essential component of system state. Again, in computational terms, a Goal may simply be the value of a variable, a record structure, or a symbolic expression in some logic. The important point is that a Goal represents some desired end state. Conventional computer software is ”task oriented” rather than ”goal oriented”; that is, each task (or subroutine) is executed without any memory of why it is being executed. This means that the system cannot automatically recover from failures (unless this is explicitly coded by the programmer) and cannot discover and make use of opportunities as they unexpectedly present themselves.

For example, the reason we can recover from a missed train or unexpected flat tyre is that we know where we are (through our Beliefs) and we remember to where we want to get (through our Goals). The underlying semantics for Goals, irrespective of how they are represented computationally, should reflect some logic of desire.

Now that we know the system state must include components for Beliefs and Goals, is that enough? More specifically, if we have decided upon a course of action (let’s call it a plan), and the world changes in some (perhaps small) way, what should we do — carry on regardless, or replan? Interestingly, classical decision theory says we should always replan, whereas conventional software, being task-oriented, carries on regardless. Which is the right approach?

Figure 2 demonstrates the results of an experiment with a simulated robot trying to move around a grid collecting points [11]. As the world (grid) is dynamic, the points change value and come and go as the robot moves and plans — thus a plan is never good for long. The *y* axis of the graph shows robot efficiency in collecting points, the *x* axis the speed of change (i.e., the rate at which the points in the grid are changing). The “cautious” graph represents the case in which the system replans at every change (i.e., as prescribed by classical decision theory), and the “bold” graph in which the system commits to its plans and only replans at ”crucial” times. (The case of conventional software, which commits to its plans forever, is not shown, but yields higher efficiency than classical decision theory when the world changes slowly, but rapidly becomes worse when the world changes quickly). In short, neither classical decision theory nor conventional task-oriented approaches are appropriate — the system needs to commit to the plans and subgoals it adopts but must also be capable of reconsidering these at appropriate (crucial) moments. These committed plans or procedures are called, in the AI literature, *Intentions*, and represent the third necessary component of system state. Computationally, Intentions may simply be a set of executing threads in a process that can be appropriately interrupted upon receiving feedback from the possibly changing world.

Finally, for the same reasons the system needs to store its current Intentions (that is, because it is resource bound), it should also cache generic, parameterized Plans for use in future situations (rather than try to recreate every new plan from first principles). These plans, semantically, can be viewed as a special kind of Belief, but because of their computational importance, are sensibly separated out as another component of system

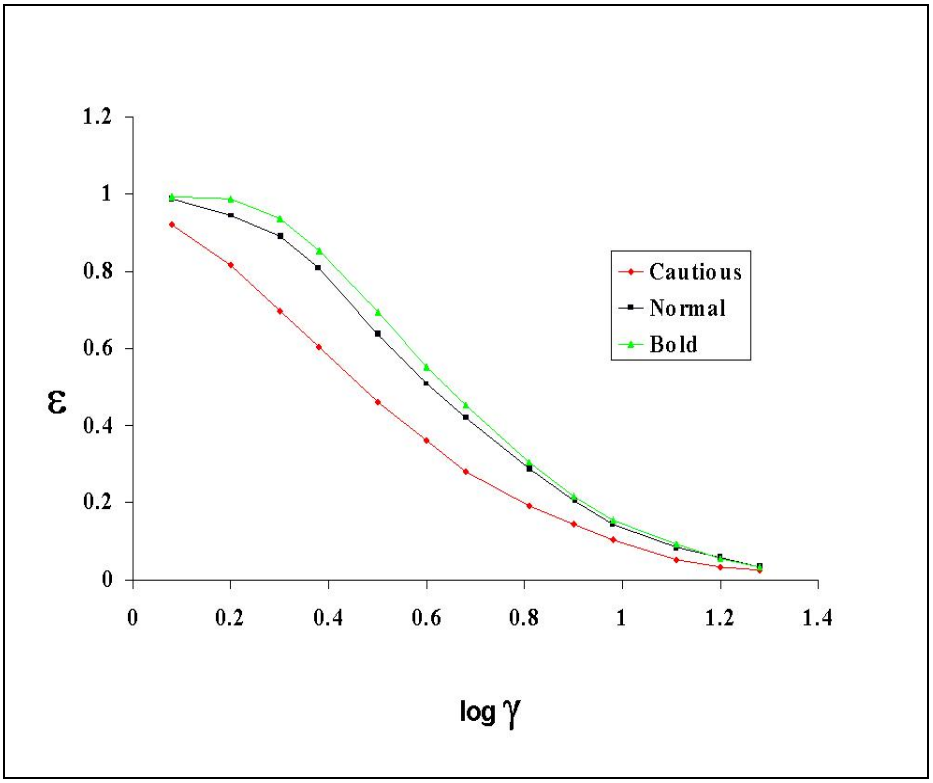

Fig. 2. Rational Commitment

state.

In summary, the basic components of a system designed for a dynamic, uncertain world should include some representation of Beliefs, Desires, Intentions and Plans, or what has come to be called a BDI agent. I have said here nothing about the way in which these components are controlled and managed, which is of course crucial to the way in which BDI agents cope with uncertainty and change in a way that is not possible with conventional systems. There is much in the literature about this, and many different and interesting approaches.

Finally, because of the logical or physical distribution of information and processing, it is important that agent systems be distributed, giving rise to so-called multi-agent systems. Apart from the usual benefits provided by distributed systems, multi-agent systems also have the substantial benefit of containing the spread of uncertainty, with each agent locally dealing with the problems created by an uncertain and changing world.

4 Response by Pollack

I want to begin by clarifying the distinction between three things:

- Models of practical reasoning that employ the folk-psychology concepts of belief, desire, and intention, perhaps among others. Let's call these Belief-Desire-Intention (BDI) models.
- Particular BDI models that center on claims originally propounded by Bratman [1] about the role of intentions in focusing practical reasoning. Specifically, Bratman argued that rational agents will tend to focus their practical reasoning on the intentions they have already adopted, and will tend to bypass full consideration of options that conflict with those intentions. Let's call this Bratman's Claim, and let's call computational models that embody this claim IRMA models (for the "Intelligent Resource-Bounded Machine Architecture" described in [2]).
- The Procedural Reasoning System (PRS) [7, 6], a programming environment for developing complex applications that execute in dynamic environments and can best be specified using BDI concepts.

One can reject Bratman's Claim, but still subscribe to the view that BDI models are useful; the converse, of course, is not true.[1] And while it is possible to build a PRS application that respects Bratman's Claim — indeed, as mentioned in the Introduction, several successful applications have done just this — it is also possible to build PRS applications that embody alternative BDI models. It is up to the designer of a PRS application to specify how beliefs, desires, and intentions affect and are influenced by the application's reasoning processes; there is no requirement that these specifications conform to Bratman's Claim.

The questions set out in the Introduction might in principle be asked of each of the three classes of entity under consideration: BDI models, IRMA models, or PRS-based applications. However, I think it makes the most sense here to interpret them as being addressed at IRMA models, in part because these are the most specific of the three classes (it would be difficult to address all BDI models within a few pages), and in part because IRMA models have received significant attention within the AI community, both in their realization in several successful applications, and in a number of detailed formal models.

Bratman's Claim addresses at a particular, albeit central, question in practical reasoning: how can an agent avoid getting lost in the morass of options for action available to it?[2] The formation of intentions and the commitments thereby entailed are seen as a mechanism — possibly one amongst many — for constraining the set of options about which an agent must reason. Practical reasoning tasks such as means-end reasoning

[1] However, one could reject all BDI models, including IRMA ones, arguing that they have no explanatory value. The debate over this question has raged in the philosophical literature; see, e.g., Carrier and Machamer [3, Chap. 1-3].

[2] Bratman actually came at things the other way round. He wondered why humans formed intentions and plans, and concluded that doing so provides them with a way of focusing their practical reasoning.

and the weighing of alternatives remain important for IRMA agents. But IRMA agents' intentions help focus these reasoning tasks.

In response to the first question posed, then, it seems clear that both Soar and the utility maximization models include important ideas that can potentially be integrated in an IRMA agent. As just noted, IRMA agents still need to perform means-end reasoning (in a focused way), and Soar, with its chunking strategies, can make the means-end reasoning process more efficient. Again, IRMA agents still need to weigh alternatives (in a focused way), and to do this they may use the techniques studied in the literature on economic agents. It has been generally accepted for many years that agents cannot possibly perform optimizations over the space of all possible courses of action [17]. Bratman's Claim is aimed precisely at helping reduce that space to make the required reasoning feasible.

The second question concerns the development of techniques to enable IRMA agents to learn and to interact socially. Certainly, if Bratman's Claim is a viable one, then it must be possible to design IRMA agents who can learn and can interact with one another. However, all that is required is that Bratman's Claim be compatible with (some) theories of learning and social interaction: Bratman's Claim *itself* does not have to tell us anything about these capabilities.[3] To date, I see no evidence that there is anything in either Bratman's Claim or its interpretation in IRMA models that would make an IRMA agent inherently poorly suited to learning or social interaction.

The third question asks about an appropriate research agenda for those interested in IRMA models. What seems most crucial to me is the development of computationally sound accounts of the various practical reasoning tasks that must be performed by IRMA agents. There has been a great deal of attention paid to questions of commitment and intention revision, and this is not surprising, given that these questions are central to Bratman's Claim. But there are other reasoning tasks that all IRMA agents must perform as well. For example, they must deliberate about alternatives that are either compatible with their existing plans or have "triggered an override" [2]); recently, John Horty and I have been developing mechanisms for weighing alternatives in the context of existing plans [10]. Another example is hinted at in my earlier comments: all IRMA agents need to be able to perform means-end reasoning. But unlike standard means-end reasoning in AI (plan generation), an IRMA agent must do this reasoning taking account its existing plans. Work on plan merging, notably that of Yang [18], may be relevant here. One final example: IRMA agents must be capable of committing to partial plans. If they were required always to form complete plans, they would over-commit, and filter out too many subsequent options as incompatible. But this then entails that IRMA agents must have a way of deciding when to add detail to their existing plans—when to commit to particular expansions of their partial plans. To my knowledge, this question has not been investigated yet.

In addressing questions like these, we need to focus, at least for now, on the development of computationally sound mechanisms: algorithms and heuristics that we can employ in building IRMA agents (perhaps using PRS). Formal underpinnings can, and if at all possible, should accompany these mechanisms, but unless they underpin

[3] However, it might contribute to them; see, e.g., Ephrati *et al.* [4] for some preliminary work on using the intention-commitment strategy in multi-agent settings to increase cooperation.

specific algorithms and heuristics they seem unlikely to have much impact.

5 Response by Tambe

I was invited on this panel as a representative of the Soar group with particular interests in multi-agent systems. Thus, in this short response, I will mainly focus on the relationship between Soar and BDI models. For the sake of simplicity, one key assumption in my response is considering PRS, dMARS, and IRMA to be the paradigmatic BDI architectures. Of course, it also should be understood that despite my twelve years of research using Soar, I alone cannot possibly capture all of the diverse set of views of Soar researchers.

I will begin here by first pointing out the commonality in Soar and BDI models. Indeed, the Soar model seems fully compatible with the BDI architectures mentioned above. To see this, let us consider a very abstract definition of the Soar model. Soar is based on operators, which are similar to reactive plans, and states (which include its highest-level goals and beliefs about its environment). Operators are qualified by preconditions which help select operators for execution based on an agent's current state. Selecting high-level operators for execution leads to subgoals and thus a hierarchical expansion of operators ensues. Selected operators are reconsidered if their termination conditions match the state. While this abstract description ignores significant aspects of the Soar architecture, such as (i) its meta-level reasoning layer, and (ii) its highly optimized rule-based implementation layer, it will sufficient for the sake of defining an abstract mapping between BDI architectures and Soar as follows:

1. *intentions* are selected operators in Soar;
2. *beliefs* are included in the current state in Soar;
3. *desires* are goals (including those generated from subgoaled operators); and
4. *commitment strategies* are strategies for defining operator termination conditions. For instance, operators may be terminated only if they are achieved, unachievable or irrelevant.

Bratman's insights about the use of commitments in plans are applicable in Soar as well. For instance, in Soar, a selected operator (commitment) constrains the new operators (options) that the agent is willing to consider. In particular, the operator constrains the problem-space that is selected in its subgoal. This problem-space in turn constrains the choice of new operators that are considered in the subgoal (unless a new situation causes the higher-level operator itself to be reconsidered). Interestingly, such insights have parallels in Soar. For instance, Newell has discussed at length the role of problem spaces in Soar.

Both Soar and BDI architectures have by now been applied to several large-scale applications. Thus, they share concerns of efficiency, real-time, and scalability to large-scale applications. Interestingly, even the application domains have also overlapped. For instance, PRS and dMARS have been applied in air-combat simulation, which is also one of the large-scale applications for Soar.

Despite such commonality, there are some key differences in Soar and BDI models. Interestingly, in these differences, the two models appear to complement each other's strengths. For instance, Soar research has typically appealed to cognitive psychology and practical applications for rationalizing design decisions. In contrast, BDI architectures have appealed to logic and philosophy. Furthermore, Soar has often taken an empirical approach to architecture design, where systems are first constructed and some of the underlying principles are understood via such constructed systems. Thus, Soar includes modules such as chunking, a form of explanation-based learning, and a truth maintenance system for maintaining state consistency, which as yet appear to be absent from BDI systems. In contrast, the approach in BDI systems appears to be to first clearly understand the logical or philosophical underpinnings and then build systems.

Based on the above discussion, it would appear that there is tremendous scope for interaction in the Soar and BDI communities, with significant opportunities for cross-fertilization of ideas. BDI theories could potentially inform and enrich the Soar model, while BDI theorists and system builders may gain some new insights from Soar's experiments with chunking and truth maintenance systems. Yet, there is an unfortunate lack of awareness exhibited in both communities about each others' research. The danger here is that both could end up reinventing each others' work in different disguises.

In my own work, I have attempted to bridge this gap, roughly based on the mapping defined above. For instance, Cohen and Levesque's research on joint intentions [12], and Grosz and Kraus's work on SHAREDPLANS [9] has significantly influenced the STEAM system for teamwork, which I have developed in Soar. However, this is just one such attempt. This panel discussion was an excellent step to attempt to bridge this gap in general.

Acknowledgements

Thanks to David Kinny, who helped to generate the questions and issues raised on the panel.

References

1. M. E. Bratman. *Intentions, Plans, and Practical Reason.* Harvard University Press: Cambridge, MA, 1987.
2. M. E. Bratman, D. J. Israel, and M. E. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4:349–355, 1988.
3. M. Carrier and P. K. Machamer, editors. *Mindscapes: Philosophy, Science, and the Mind.* University of Pittsburgh Press, Pittsburgh, PA, 1997.
4. E. Ephrati, M. E. Pollack, and S. Ur. Deriving multi-agent coordination through filtering strategies. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 679–687, Montréal, Québec, Canada, August 1995.
5. I. A. Ferguson. Integrated control and coordinated behaviour: A case for agent models. In M. Wooldridge and N. R. Jennings, editors, *Intelligent Agents: Theories, Architectures, and Languages (LNAI Volume 890)*, pages 203–218. Springer-Verlag: Berlin, Germany, January 1995.
6. M. P. Georgeff and F. F. Ingrand. Decision-making in an embedded reasoning system. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, pages 972–978, Detroit, MI, 1989.

7. M. P. Georgeff and A. L. Lansky. Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pages 677–682, Seattle, WA, 1987.
8. M. P. Georgeff and A. S. Rao. A profile of the Australian AI Institute. *IEEE Expert*, 11(6):89–92, December 1996.
9. B. Grosz and S. Kraus. Collaborative plans for group activities. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, pages 367–373, Chambéry, France, 1993.
10. J. F. Horty and M. E Pollack. Option evaluation in context. In *Proceedings of the Seventh Conference on Theoretical Aspects of Rationality and Knowledge (TARK-98)*, 1998.
11. D. Kinny and M. Georgeff. Commitment and effectiveness of situated agents. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 82–88, Sydney, Australia, 1991.
12. H. J. Levesque, P. R. Cohen, and J. H. T. Nunes. On acting together. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90)*, pages 94–99, Boston, MA, 1990.
13. J. P. Müller. *The Design of Intelligent Agents (LNAI Volume 1177)*. Springer-Verlag: Berlin, Germany, 1997.
14. A. S. Rao and M. Georgeff. Decision procedures of BDI logics. *Journal of Logic and Computation*, 8(3):293–344, 1998.
15. S. Russell and D. Subramanian. Provably bounded-optimal agents. *Journal of AI Research*, 2:575–609, 1995.
16. K. Schild. On the relationship between BDI logics and standard logics of concurrency. In J. P. Müller, M. P. Singh, and A. S. Rao, editors, *Intelligent Agents V — Proceedings of the Fifth International Workshop on Agent Theories, Architectures, and Languages (ATAL-98)*, Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg, 1999. In this volume.
17. H. A. Simon. A behavioral model of rational choice. *Quarterly Journal of Economics*, 69:99–118, 1995.
18. Q. Yang. *Intelligent Planning: A Decomposition and Abstraction Based Approach*. Springer-Verlag: New York, 1997.

This article was processed using the LaTeX macro package with LLNCS style