Dialectical Planning

Nikos Karacapilidis and Thomas Gordon

GMD, FIT.KI
German National Research Center for Information Technology
Schloss Birlinghoven, 53757 Sankt Augustin, Germany
e-mail: {nikos.karacapilidis, thomas.gordon}@gmd.de

Abstract
Planning of real world instances has to be performed through a lot of debates, negotiations and arguments between groups of planning agents. Conflicts of interest are inevitable and support for achieving consensus and compromise is required. The identification and selection among alternative courses of action have to be supported with rational, fair and effective decision-making models, especially in the prevailing cases of limited and uncertain information. Dialectical Planning is among the most promising application areas for what we call Computational Dialectics, that is computational models of norms of rational discourse. Work presented in this paper focuses on the design and implementation of such a mediating system for supporting group planning founded on a normative model of limited rationality.

1 Introduction
The role of planners in the classical AI planning paradigm is to construct plans revealing states that satisfy particular conditions, that is goals. In spite of the success of the predominant form of hierarchical non-linear planning in search reduction, such systems lack a sufficient basis for choice of action [Wellman and Doyle, 1991] and an adequate representation for consistency and replanning [Petrie, 1992], [Wilkins and Myers, 1994]. The rough distinction of states into those satisfying and those not satisfying some goals poses problems in real world instances, where uncertainty and incomplete knowledge of the world are inherent. In these cases, planning agents may assert uncertain objectives that can be partially satisfied. Decision making formalisms could provide the desired platform, yet lack some computational advantages of planners.

More concretely, decision making in real planning environments has to confront conditions such as (see also [Gordon, 1994a]): (i) Reasoning is defeasible, that is further
information can trigger another alternative to appear preferable than what seems best at the moment; (ii) Planning has to be performed through a lot of debates, negotiations and arguments (supporting and against) between groups of agents. Conflicts of interest are inevitable and support for achieving consensus and compromise is required. Each agent may adopt and, consequently, suggest his own strategy that fulfils some goals at a specific level; (iii) The expected value of the known alternative decisions is not high enough to make it cost effective to invest substantial resources in implementing the knowledge base for a planner; (iv) The existence of both not enough and too much information; (v) However much information is available, opinions differ about its truth, relevance or value for deciding an issue; (vi) Factual knowledge is not always sufficient for making a decision. Value judgements are the critical points requiring attention.

Planning systems require the specification of a language with which to express beliefs and goals. Such a language provides the basis upon which to build representations of actions, plans and operators. Operators usually represent the actions, at different levels of abstraction, that the system may perform in the given domain (see for example [Petrie, 1992]). A planning hierarchy tree is constructed by breaking up goals into subgoals and assignments (Figure 1). The primary representation task of an operator is to describe how the world changes after the action it represents has been performed. Consequently, operators contain information about the objects that participate in the actions, the constraints that must be placed on them, the goals that the actions are attempting to achieve, the way actions in this operator relate to more or less abstract descriptions of the same action, and the necessary preconditions before the actions can be performed.

The rest of the paper is organized as follows: The Dialectical Planning formalism is introduced in Section 2; data types and concepts are defined, the layers of the proposed system are sketched, and aspects of the decision making procedure are discussed. Section 3 applies the proposed formalism in a planning instance, and Section 4 illustrates a World Wide Web gateway for the system. Finally, motivations for this work, as well as related and future work are discussed in Section 5.
2 The Dialectical Planning formalism

It has been made clear that the ubiquitous task in planning problems is the identification and selection among alternative courses of action. Artificial Intelligence has to provide more adequate tools for supporting and reasoning about rational and effective decision-making, when information is limited and uncertain, and conflicts of interest and opinion are common. Models of rational decision-making in groups are needed, taking these considerations into account. Dialectical Planning is among the most promising application areas for what we call Computational Dialectics, that is computational models of norms of rational discourse [Gordon, 1994a], [Gordon, 1994b]. Work being done in Zeno project aims at the development of a mediating system for supporting argumentation, negotiation and decision making in groups, where the above conditions prevail. We focus on the design and implementation of a mediating system for supporting group planning founded on a normative model of limited rationality.

Dialectical Planning embraces such services to be provided as managing the dependencies between arguments, claims, positions and issues, helping the participant agents to be aware of their rights and obligations in a planning problem, and providing access to procedures for negotiation and conflict resolution. The specified role of the system is that of an assistant and advisor, that is it recommends its solutions leaving the final enforcement of decisions and actions to the agents. The system interpreter is being implemented as a World Wide Web Common Gateway Interface (CGI) script, using the Scheme Shell. Developing a suite of intelligent tools needed for the planning problem solution, and providing a public-domain assistant application, any WWW browser, such as Mosaic or Netscape, will be sufficient for an agent to take part in mediated planning discussions.

The proposed system consists of the Logic Layer, where the notions of necessary consequence and contradiction are defined, the Argumentation Framework Layer, where the concepts of proposition, supporting argument, counterargument and issue as well as linguistic constructs for arguing about priority relationships among competing arguments are specified, the Speech Act Layer, where the possible kinds of actions an agent may have during the planning procedure are defined, and finally, the Protocol Layer where norms about the duties and the rights of the agents to perform actions defined at the previous layer are specified, dependent on their roles, or/and the evolution of the planning procedure so far. The first two layers define a kind of nonmonotonic logic, founded on argumentation principles. Work presented in this paper mainly focuses on them. An abstract model of mediating systems for group decision making is extensively discussed in [Brewka et al., 1995]. In the remainder of this section we introduce the basic concepts behind the Argumentation Framework Layer, and discuss aspects of the decision making procedure.

We consider a proposition to be the lowest level of granularity in our framework. Any kind of data an agent wants to assert during the planning procedure can be used in order to represent a proposition. A proposition may be represented by a text,
spreadsheets, graphics, parts of databases, etc. The proposition used may be true or false, important or irrelevant for the corresponding problem, and may become acceptable or non-acceptable. An issue consists of a set of alternative propositions and a set of related constraints. In other words, propositions represent the positions (or claims) asserted so far, while the issue is which alternative position to prefer, if any. Each issue, a position should be selected exclusively (see also [Brewka and Gordon, 1994]). Each issue also includes a “dummy” position, called nil. This dummy position can be interpreted as the “selection of none of the current positions” and provides a means of handling our aims, i.e., the system indication that none of the alternative positions of the issue is recommended. Arguments are assertions about the positions regarding their properties or attributes, which speak for or against them. We distinguish them in supporting arguments (pro) and counterarguments (con). An argument links together two propositions of different issues. We should also mention here that, in the current implementation, we don’t allow for cycles, that is the structure of the system is tree-like. Arguments can be defeated by stronger counterarguments, or combinations of counterarguments. Two competing arguments can be accepted simultaneously (see also [Brewka, 1994a]).

The following (briefly sketched here) data types can fully represent classical operator schemes (i.e., the concepts of goals, operators and assignments illustrated in Figure 1), and provide an efficient way to map links between the asserted positions and traverse the corresponding planning tree (however, the complete modelling of operator schemes with our formalism is beyond the scope of this paper):

```
<proposition> (data issue parent_argum supporting_argum counterargum)
where,
  data is the position asserted by an agent;
  issue stands for the issue to which the proposition belongs;
  parent_argum states the parent argument of the proposition, and
  supporting_argum and counterargum are lists of supporting arguments
  and counterarguments for this proposition, respectively.

<issue> (level positions constraints)
where,
  level is an index of the position of the issue in the planning tree;
  positions and constraints are lists consisting of the alternative
  propositions and the imposed constraints of the issue, respectively.

<supporting_argument> (ant_prop consq_prop)
where,
  ant_prop is the antecedent proposition of the supporting argument, and
  consq_prop is the “consequent” proposition of it.

<counterargument> (ant_prop consq_prop)
  (similar to <supporting_argument>).
```

Rather than hardwiring into the logic we use a single principle, such as specificity, for resolving conflicts among arguments, knowledge about preferences is encoded as part of the domain theory. We assume that any position related to an attribute has a qualitative value out of a predetermined domain, keeping the value identical for each alternative it refers to. Arguments pro and con each choice are weighed against each other. Combined weak arguments may defeat a strong argument, and preferences may be expressed qualitatively. Constraints provide a qualitative way to argue about
preferences and value judgements in order to weigh reasons for and against a certain option. In other words, they give to the users the ability of ranking the quality of alternative propositions. Constraints comprise positions (i.e., attributes) that are linked to the appropriate node via supporting and counter arguments. Constraints are interpreted as *meta-issues*, also allowing the attachment of a nil proposition, as well as of constraints on them. Asserted constraints on attributes provide the means of weighing among them. Representing real world cases, we allow for not totally ordered attributes. For instance, consider the case in which one has to conclude an issue with two alternative propositions $P_1$ and $P_2$, knowing that “$P_1$ has the attributes $a_1$, $a_2$, and $a_3$”, while “$P_2$ has the attributes $a_1$, $a_4$, and $a_5$”, while no information regarding the ordering of $a_2$, $a_3$, $a_4$, and $a_5$ has been given.

The initial set of conclusions computed by our system comprise the instances "Recommend Accept", "Recommend Reject", and "No Recommendation". Since we intend the final decision to be made by the planning agents, a similar set of decisions (User’s "Accept", "Reject" and "Undecided") has been specified. Decisions will automatically be reviewed at each stage of a plan’s evolution to enable an agent to check that a decision taken towards a certain planning direction is still recommended. To review these selections we have to specify the appropriate algorithm for updating those previously holding, according to the attachment of more detailed information when the planning plot is expanded. A first sketch of the algorithm is illustrated in [Brewka et al., 1995].

3 An example

We apply our formalism to the following instance: the goal of a part of our planning problem is to find a constructor for a part of a car engine. There are three (asserted so far) alternative choices, that is to construct it in the same factory where the fitting of the various parts is being performed (DIY), or to order it from two candidate subcontractor companies, Constructor-1 and Constructor-2. The asserted attributes concern the quality, service, delivery time and cost that each of the alternatives provide. There is incomplete information regarding the ordering of attributes. In addition, there may be no complete linking between each alternative of an issue and every asserted attribute. For example, there may be no argumentation about service provided for the DIY and Constructor-1 alternatives. Instances of constraints handled here are the following:

$c$-1.1: fair cost $>$ good quality;
$c$-1.2: meet due date $<$ fair cost;
$c$-1.3: good quality $+$ meet due date $>$ fair cost.

The data types introduced above provide a simple way in order to map links between the asserted propositions, and traverse the corresponding planning tree. The following examples illustrate instances of these data types:

- proposition
  ( Choose Constructor 1
1.0
argument -> find constructor for BOO-1
(provides free 1 year service,
offers fair cost,
engages to meet due date)
(no guarantee of good quality)
}

• issue
  { 1.0
      (Do It Yourself,
       Choose Constructor 1
       Choose Constructor 2
       nil-1.0)
      { ( 1.1
           ( fair cost > good quality,
             nil-1.1 )
           ( ) ),
         ( 1.2
           ( fair cost > meet due date,
             nil-1.2 )
           ( ) ),
         ( 1.3
           ( good quality + meet due date > fair cost,
             nil-1.3 )
           ( ) )
      )
  }

• supporting argument
  { Choose Constructor 1
    find constructor for BOO-1 }

• counterargument
  { no engagement of meeting due date
    Do It Yourself ).

Figure 2: An instance of the planning tree.
An instance of the planning tree for the example above is sketched in Figure 2. Positions are denoted with ellipses, issues with rectangles, and arguments with arrows. Constraints may appear in the second part of each issue. Due to space limitations, they are simply shown here with shadowed rectangles, although they retain the structure of issues.

4 The World Wide Web gateway

Figure 3 illustrates a mock-up of a World Wide Web gateway through which each agent can assert its own positions and constraints in a planning paradigm. The File menu includes the usual commands such as New, Open, Close, Send, Save, Print or Quit a plan. Each paradigm contains all corresponding positions and constraints asserted so far via our mediated system. Specification of rights and duties among agents at the Speech Act layer would affect their potential access to the list of available commands. Several agents can open and modify the same plan simultaneously. An agent can modify the dialectical graph by asserting new positions, and consider alternative decisions in spite of the system’s recommendations. “What-if” scenarios might be tested before an agent decides about what he finally wishes to assert. The Edit menu includes the usual Undo, Cut, Copy, Paste, Clear, Select, Find and Replace commands. Similarly, the View, Navigate, Options and Help menus include well-tried commands from WWW browsers adapted in our formalism.

The instance illustrated in Figure 3 is related to the example of Section 3. The corresponding file has been retrieved and its asserted issues are listed in the first scrollable pane under the main menu bar. The agent can select any of them and click either on the "Propositions in the Issue", or on the "Constraints in the Issue" button to see what has been asserted (second scrollable pane). Automatically, he would find out the system’s conclusion for the issue by observing for which proposition the "Recommend Accept" button is on. Possible weakness for solving the issue will be represented by the ‘No Recommendation’ button being on. ‘Recommend Reject’ for a proposition, indicates that the system has identified a better alternative in this issue. Preserving the mediating role we intend for the system, an agent would be able to select an alternative, and assert its own opinion by clicking on the User’s "Accept", "Reject" or "Undecided" buttons. Working in this way, agents would be able to observe the consequences their decisions cause at the higher levels of the planning tree, and evaluate alternative plans.

The bottom part serves for the commitment of new positions or constraints in a plan. The scrollable pane would include the description of the position. The linking of a new-asserted proposition with an existing one can be made by clicking on one of the "Pro" and "Con" buttons (declaring intention for a supporting or a counter argument, respectively), after the selection of the corresponding proposition. The Navigate menu provides the usual commands for the tracing of the dialectical graph. For instance, the "Top" command leads to the prime goal of the plan, and the "Up" and "Down" commands trace the issues at the various abstraction levels. "Next" and "Previous" commands cycle through the other arguments of a selected proposition. Finally, the
View menu provides suitable decision-making graphs and options for overall representations of a plan. For instance, other views of the dialectical graph, such as a temporal list of past messages will be also useful.

**Figure 3:** The World Wide Web gateway.

5 Discussion

The concept of Dialectical Planning embraces integrated hypertext and groupware technologies, smoothly applied on the amalgamation of a rhetorical model and
classical planning algorithms (Figure 4). Hypertext systems feature machine-supported links, both within and between documents, that have opened exciting new possibilities for using the computer as a communication and thinking tool [Conklin, 1987]. Groupware [Ellis et al., 1991] is known as the multi-user software supporting Computer-supported cooperative work (CSCW), that is computer-assisted coordinated activity, such as communication and problem solving carried out by a group of collaborating individuals [Greif, 1988]. The rhetorical model can enhance the quality of the dialogue process within a (conceptual) organization by providing the structure for the discussion of complex problems. The motivations behind the joint integration of a rhetorical model and planning algorithms are:

- to support and reason about commonly prevailed conditions in dynamic planning environments, such as argumentation, negotiation, and conflict resolution;
- to facilitate and rationalize the communication among the multiple agents in the planning process;
- to extend and cumulate the existing knowledge across the planning agents, and
- to adapt supported planning algorithms on defeasible and qualitative decision making environments.

Our formalism is in accordance with previous arguments stated in [Le Pape, 1994], stimulating to a decomposition of planning and scheduling activities into constraint propagation, decision-making, and intelligent control of both. Architectural issues mentioned there, regarding the integration of such a flexible software for predictive or reactive, as well as for centralized or distributed problem instances are successfully addressed through our model.

As discussed above, Dialectical Planning is a contribution to the recently introduced field of Computational Dialectics, which investigates computer models of norms for real-life discourse. The proposed system should provide adequate tools for supporting rational, effective and fair decision-making in planning, when resources, such as information, time and memory are limited, and conflicts of claims and arguments are common. Dialectical Planning is performed via a mediating system, built on a
normative model of limited rationality. Thus, the implementation of a fair, efficient and rational rhetorical model plays a key role in such a system. Among the most prominent related work, we mention here the early work of Toulmin on argumentation theory [Toulmin, 1958], Pollock’s OSCAR model of defeasible reasoning [Pollock, 1988], Rescher’s work on formal disputation theory [Rescher, 1977] and its reconstruction by Brewka [Brewka, 1994b], and the Issue-Based Information System (IBIS) rhetorical method developed at MCC [Conklin, 1992], [Yakemovic and Conklin, 1990]. The legitimate aspect of logic behind such a rhetorical model, in order to implement a set of norms for regulating this kind of discourses, has been highlighted in Toulmin’s work, and is extensively discussed in [Gordon, 1993] and [Prakken, 1993].

We intend to further explore the assertion of additional, probably by default, constraints, for example of the type “\( p > \neg p \)”, where \( p \) represents a proposition and \( \neg p \) its negation (see [Doyle et al., 1991]). Constraints of this type will provide a means of weighing positions declaring a negative attribute (such attributes are not considered in our first implementation). Jointly regarding the constraints holding and the arguments asserted to alternative positions, we also intend to introduce concepts concerning the optimistic or pessimistic, and the credulous or skeptical conclusion of an issue. The optimistic (pessimistic) conclusion of an issue is entailed regarding the maximum (minimum) possible values of the alternative choices, while the credulous (skeptical) conclusion is entailed regarding the full (common) set of asserted attributes of the alternative choices.

The conclusion of an issue implies usually the solution of a constraint optimization problem. Exploiting abilities of a constraint satisfaction programming language, the system can guarantee consistency checking for the asserted constraints. Initial experiments have been done with the ECLiPSe language. Future work aims at the efficient solving of this problem, jointly exploring constraint satisfaction techniques and a logic for handling qualitative values.

6 Conclusion

The concept of Dialectical Planning introduced in this paper, as a specific application of Computational Dialectics, conceived as a research field focusing on the implementation of appropriate tools for supporting and reasoning about fair, rational and effective group decision-making in planning. Data types and formalisms were specified, and aspects of the decision-making procedure, especially in cases where information is limited and uncertain, and conflicts of interest and opinion are common, were discussed. A WWW gateway for such systems suggested, aiming (at least) at providing the broadest access among agents (i.e., via Netscape), and assuring portability and platform-independence. Finally, related work, focusing on the rhetorical model, has been presented and some issues regarding future work have been raised. Key issues of Dialectical Planning are the awareness of the planning agents, multi-user interfaces, communication and coordination within the group of planning agents, shared information space and the support of a heterogeneous, open environment which integrates existing single-user applications.
As a final note, we argue here that Dialectical Planning could be a paradigm shift for the planning research area, in that it emphasizes on a human-human coordination, communication and problem solving, rather than on a human-machine one.

References


