Department of Administrative Sciences

REASONING WITH ASSUMPTIONS, DEFEASIBLY, IN MODEL FORMULATION

Hemant K. Bhargava
and
Ramayya Krishnan

October 4, 1991

Working Paper No. 91-04

Naval Postgraduate School
Monterey, California

Working papers of the Naval Postgraduate School Department of Administrative Sciences are preliminary materials circulated to stimulate discussion and critical comment. The views stated herein are the author's and not necessarily those of the Department of the Navy or the Naval Postgraduate School.

List of working papers on inside backcover.
For additional copies, write to:

Department of Administrative Sciences
Working Paper Series
Code AS
Naval Postgraduate School
Monterey, California 93943-5026
(408) 646-2471

Reasoning with Assumptions, Defeasibly, in Model Formulation

Hemant K. Bhargava *
Naval Postgraduate School
Monterey CA 93943
5186p@navpgs.bitnet
(408) 646-2264

Ramayya Krishnan
Carnegie-Mellon University
Pittsburgh PA 15213
rk2x+@andrew.cmu.edu
(412) 268-2174

October 4, 1991

Abstract

This paper examines the relevance of reasoning with assumptions in two processes that are desired to be supported in model management systems, namely model formulation and model version management. We submit, and illustrate with an example, that the ability to represent and reason with assumptions in modeling languages could lead to significant improvement in the functionality of model management systems. We also argue that the process of reasoning with assumptions is non-monotonic and propose that defeasible reasoning is a useful candidate for modeling this process.

1 Introduction

This paper examines the relevance of reasoning with assumptions in two processes that are desired to be supported in model management systems, namely model formulation and model version management. A model is often defined as a collection of assumptions. In this sense developing, and reasoning with, assumptions is a fundamental process in modeling. Yet, few languages and systems for model management provide useful ways to represent, and to reason with, assumptions. It then becomes relevant to pose the following two questions. One, would the ability to represent and reason with assumptions lead to significant improvement in the functionality of model management systems? Two, how should assumptions be represented in a language for model management, and what inference mechanisms would yield the desired functionality? In this paper we mainly attempt to provide an affirmation of the first question by motivating the need for an explicit representation of assumptions, and mechanisms for reasoning with them, in modeling languages. It is our secondary purpose to provide partial answers to the second question.

The model construction process usually involves the development of mathematical abstractions corresponding to selective aspects of a problem situation [6, 8, 15]. The specific mathematical formulation depends largely on what assumptions are made by the modeler, and its usefulness depends partly on how reasonable these assumptions are. Yet, in studies of modeling practice, Gass [8] found that "analysts do not document, cannot or will not write well, will not state modeling assumptions, ..." While recently developed algebraic modeling languages (e.g., [3, 7]) support the modelers' algebraic notation directly, and even provide means for the representation of additional qualitative information (e.g., [2, 1, 4, 9]), they have few features for the representation and use of assumptions.

*This author's work on this paper was performed in conjunction with research funded by the Naval Postgraduate School.

Research in computer-aided model formulation is concerned with the analysis, design and development of computer systems to assist human modelers in the formulation of models. We argue that the process of reasoning with assumptions during model development is non-monotonic, in that a change in (or addition of) an assumption might cause the modeler to delete previously developed components of the model. We will illustrate with an example that defeasible reasoning is a suitable method for (non-monotonic) reasoning with assumptions in model management systems. That is the subject of §3. First, we give a quick introduction to defeasible reasoning in the next section.

2 Defeasible Reasoning

Predicate logic and sentence logic are systematic methods of reasoning, which, for most practical purposes, can be viewed as reasoning with a set of rules that can be stated in the form: IF conditions $\phi_1, \ldots, \phi_n$ are true, THEN conclusion ψ holds (or, $\phi_1, \ldots, \phi_n \rightarrow \psi$). Such logics have the property that they are *monotonic*, i.e., the addition of new premises may lead to new conclusions but cannot override earlier conclusions. Defeasible reasoning is a form of *non-monotonic* reasoning, in that it allows tentative conclusions to be *defeated* in the face of new, relevant information.

Defeasible reasoning works with three kinds of rules, called *absolute* rules, *defeasible* rules, and *defeaters* [12].[1] In this sense, the rules of first-order logic are all absolute, in that a conclusion of a rule must hold if all its conditions are true. An example is the rule

$\forall$ x (penguin(x) $\rightarrow$ bird(x)) (Rule A)

A defeasible rule is a rule whose conclusion is *normally* true when its antecedents are, but which conclusion may be defeated in the face of new information. An example is the rule

$\forall$ x (bird(x) $\Rightarrow$ flies(x)) (Rule B)

which represents the observation that, typically, birds fly. Of course, penguins and ostriches and sick birds do not fly. Defeasible reasoning allows us to conclude, in the absence of other information, that a bird flies. And it prevents such a conclusion when appropriate information is available. Defeasible rules can be defeated by other (conflicting) defeasible rules, or by defeaters. In the first case, a defeasible rule simply prevents the firing of the defeasible rule whose conclusion it contradicts. A defeater's role in defeasible reasoning is to prevent a defeasible inference from taking place. An example is

$\forall$ x (bird(x), sick(x) $\mapsto$ $\neg$ flies(x)) (Rule C)

Given a sick bird, this rule alone will not allow us to conclude that it does not fly (indeed, there are sick birds that do fly), but it will prevent the earlier defeasible rule (B) from being used alone to conclude that it does fly. The final conclusion will depend on the other rules available and on the specificity of different rules that apply. The calculus of defeasible reasoning (really calculi, since there are several versions of it) specifies how conclusions are reached in the presence of possibly conflicting absolute rules, defeasible rules, and defeaters, some of whose antecedents we may have no information about. Nute's version of defeasible reasoning [12] uses a defeasible reasoning meta-interpreter to place the calculus of defeasible logic within a first-order logic framework, and is supported by an implementation called d-Prolog [13]. Causey [5] describes a shell for defeasible reasoning, EVID, which differs from Nute's d-Prolog in its treatment of defeasible rules and negation by failure. One of the interesting charactersitics about EVID is the built-in meta-predicates (such as **why**, **howdefeatit**) that explain why the system did or did not reach a certain conclusion, or what would be required to defeat a certain conclusion.

[1] We will use the operator $\rightarrow$ for absolute rules, $\Rightarrow$ for defeasible rules, and $\mapsto$ for defeaters.

3 Reasoning with Assumptions: Model Development

There is general agreement among researchers in computer-aided model construction that the cognitive process employed in model creation involves the application of a series of general model formulation rules constituting a modeler's knowledge about models, model classes, and modeling paradigms, to the information the modeler obtains about the specific problem situation [10, 11]. Consequently, considerable research on model construction has focused on building systems that combine a set of such general purpose inference rules with a domain-specific knowledge base. We believe, however, that there is a significant difference in the way modelers use such rules and in the way model formulation systems have attempted to do so.

Most of the earlier research efforts directed at developing rule-based systems to support the construction of mathematical programming models have either ignored, or have made implicit, the rôle of assumptions in the modeling process. For example, a system for linear programming formulation [10] automatically and implicitly assumes, on detecting a problem with "sources" and "destinations," that the demand at the destinations must be fulfilled. Thus the rules in such systems implicitly rely on certain assumptions that may not be made clear to the user and that may often not be verified. Further, we find that the process of reasoning with such assumptions is defeasible. In this section we show that the theory of defeasible reasoning can be used to represent, and accurately model the process of reasoning with, assumptions.

Our application of defeasible reasoning to model construction is based on the premise that modelers first develop initial versions of a model based on certain core and obvious information about the problem, and on some default assumptions. They then retract or modify some of the earlier conclusions after deeper examination of the problem situation and of the assumptions that underlie these earlier versions. Thus the process of reasoning while applying modeling knowledge during model construction is non-monotonic. If a rule-based system is to support this process, it must also be able to make tentative conclusions and revise them in the face of additional information. In what follows, we illustrate that a system using defeasible reasoning in model construction has the following kinds of advantages: a) for a given problem, the system can support the development of multiple alternative mathematical formulations which contain differences in their assumptions, b) the system can support model revision and maintenance as the problem situation or beliefs about it change over time, and c) the rule base of the system can be methodically and easily revised over time to incorporate new knowledge just as modelers change their rules over time as they learn. We do so with the following example.

Example 1 *Power Transmission*

> Electric power needs to be transmitted from a set M of power plants to a set N of electric companies. Company j requires d_j units of power. We have a_i units of power available at plant i. It will cost c_{ij} to transmit one unit from plant i to company j, and we would like to minimize the transmission costs. What we need to determine is the number of units x_{ij} that go from plant i to company j.

This description suggests that the problem can be formulated mathematically as a transportation model. Of course, this formulation assumes that the transmission cost is directly proportional to the number of units transmitted.

Model 1a

$$\text{Minimize} \sum_{i \in M} \sum_{j \in N} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{j \in N} x_{ij} \leq a_i \quad \forall i \in M$$

$$\sum_{i \in M} x_{ij} \geq d_j \quad \forall j \in N$$

$$x_{ij} \geq 0 \quad \forall i \in M \ \forall j \in N$$

This is an appropriate formulation given the available information. Notice that this formulation assumes that there are no losses during shipment (transmission). Indeed that is a reasonable assumption to make in general, and one that most rule-based systems would make. In a rule-based system the second set of constraints would be derived using rules of the following type.

Amount x_{ij} is shipped from source i to destination j

$\rightarrow$ total shipment to destination $j = \sum_{i \in M} x_{ij}$ (1)

Demand at destination $j = d_j$

$\rightarrow$ constraint(total shipment

to destination $j \geq d_j$) (2)

Similarly, the objective function might be derived using the following rule.

Amount x_{ij} is shipped from source i to destination j

AND unit cost of shipment from source i

to destination j is c_{ij}

$\rightarrow$ objective(Minimize $\sum_{i \in M} \sum_{j \in N} c_{ij} x_{ij}$) (3)

However, now suppose we wish to take account of losses in shipment, which in our example are transmission losses. Then the conclusion derived using rule 1 appears to be incorrect, though it might still be appropriate if the losses are negligible or if we choose to ignore them in our optimization. In any case, it is reasonable to prevent, without further investigation, the firing of this rule. We can accomplish that by making rule 1 defeasible (to obtain rule 4), and by adding a defeater (rule 5) which negates ($\neg$) that rule's conclusion.

Amount x_{ij} is shipped from source i to destination j

$\Rightarrow$ total shipment to destination $j = \sum_{i \in M} x_{ij}$ (4)

There are shipment losses

$\mapsto \neg$(total shipment to destination $j = \sum_{i \in M} x_{ij}$) (5)

Notice here that there could be several other defeaters for each conclusion, some of which will not be relevant to our example. For instance, rule 1 could also be defeated if the customer (destination) could reject certain shipments or could return them at a later time. Since the preconditions of these defeaters are not satisfied, they do not enter the reasoning process at all. Returning to the defeater of rule 5, however, the system would now be forced to search for an alternate rule that had a similar consequent (total shipment) and whose antecedents were true. The following rule from our defeasible rule base would now apply.

Amount x_{ij} is shipped from source i to destination j

AND a fraction l_{ij} is lost in shipment

from source i to destination j

$\Rightarrow$ total shipment to destination j

$$= \sum_{i \in N} (1 - l_{ij}) x_{ij} \quad (6)$$

Further, rules 2 and 3 assume respectively that demand must be met (or exceeded), and that the selling price is the same for each (i, j) pair. Now suppose that we want the system to model the fact that the selling price can vary, that it may not even be profitable to supply certain customers, and that

a revenue would be earned for only as many units as each customer requires. In our example, assume that an electric company j is willing to pay p_{ij} for 1 unit of power received from plant i. Any supply over a company's total requirement would be wasted and would earn no revenue, invalidating the previous demand constraint and objective function. The following defeaters would prevent what earlier seemed to be obvious conclusions.

Marginal revenue for supply exceeding demand is zero

$\mapsto \neg$(constraint(total shipment
to destination $j \geq d_j$)) (7)

Selling price can vary over (i, j) pairs

$\mapsto \neg$(objective(Minimize $\sum_{i \in M} \sum_{j \in N} c_{ij} x_{ij}$)) (8)

The following rules from our defeasible rule base would be used to reach new conclusions.

Demand at destination $j = d_j$

AND marginal revenue for exceeding demand is zero

$\Rightarrow$ constraint(total shipment
to destination $j \leq d_j$) (9)

Amount x_{ij} is shipped from source i to destination j

AND unit cost of shipment from source i
to destination j is c_{ij}

AND selling price of a unit shipped
from source i to destination j is p_{ij}

$\Rightarrow$ objective(Maximize $\sum_{i \in M} \sum_{j \in N} (p_{ij} - c_{ij}) x_{ij}$) (10)

Our problem would now have the following mathematical formulation.

Model 1b

$$\text{Maximize} \sum_{i \in M} \sum_{j \in N} (p_{ij} - c_{ij}) x_{ij}$$

$$s.t. \quad \begin{array}{ll} \sum_{j \in N} x_{ij} \leq a_i & \forall i \in M \\ \sum_{i \in M} (1 - l_{ij}) x_{ij} \leq d_j & \forall j \in N \\ x_{ij} \geq 0 & \forall i \in M \ \forall j \in N \end{array}$$

What we have illustrated thus far is how a defeasible knowledge base could be used to develop simple initial versions of a model and then to systematically revise pieces of it in the face of additional information to make the model a more accurate reflection of reality. It might appear that rather than go through a process of defeasible reasoning, we could have developed and directly used "exhaustive" absolute rules that took into account all such additional information. That would be missing the point since a) the model construction rules and process would become far more complicated if we had to reason about all possibilities at the start, and b) there would still be defeaters to these new rules that reflected other exceptional conditions.

How robust and generalizable is this technique? In other words is the example contrived to fit the defeasible rule base (or perhaps vice versa) or can such rule bases be created to handle other kinds of situations? One might argue that even with a defeasible knowledge base we might have overlooked certain conditions, and that we might learn of some such conditions at a later time. Is there a systematic way to revise the rule base that will not affect the validity of existing rules? We extend this example to illustrate how this is done

Consider the supply constraint in our previous two formulations. This would have been derived using the following rules.

Stock available at source i is a_i

$\Rightarrow$ constraint(total shipment from source i
$\leq a_i$) (11)

Amount x_{ij} is shipped from source i to destination j

$\Rightarrow$ total shipment from source $i = \sum_{j \in N} x_{ij}$ (12)

However, now suppose that it were possible to procure additional units at source i at a procurement price p_i per unit and that these additional units had an overhead transmission cost of d_i per unit. Then if there were unsatisfied demand, and it were profitable to meet it, we would want to defeat our earlier conclusions about the supply constraint and the objective function. Denote the procurement at each plant by y_i, and suppose that the total budget for this procurement is B. What we need to do is to update our knowledge base in order that the system can reason appropriately in a situation of this sort.

First, we note that rule 11 is no longer valid in case additional units can be procured. Second, the right-hand side of the constraint needs to be modified to account for the additional units. We can encode this knowledge into the following rules.

Additional units can be procured
for shipment from source i
$\mapsto \neg$(constraint(total shipment from source i
$\leq a_i$)) (13)

Stock available at source i is a_i
AND y_i additional units can be procured
for shipment from source i
$\Rightarrow$ constraint(total shipment from source i
$\leq a_i + y_i$) (14)

Note that these rules are independent of the existing rules in the knowledge base, in the sense that the earlier rules are still valid and will indeed be used when there is no information on procurement or when additional units can not be procured.

Next we wish to encode the knowledge that the total amount spent on procuring additional units must not be greater than that available. This is done by the following defeasible rules.

y_i additional units are procured
for shipment from source i
AND unit procurement price at source i is p_i
$\Rightarrow$ total procurement cost $= \sum_{i \in M} p_i y_i$ (15)

Procurement budget is B
AND total procurement cost is C
$\Rightarrow$ constraint($C \leq B$) (16)

Finally, we wish to modify the objective function to account for the overhead transmission cost for these additional units. We add a defeater to defeat the previous rule (10) and add a new defeasible rule to the knowledge base.

Overhead transmission costs exist for certain units
$\mapsto \neg$(objective(Maximize $\sum_{i \in M} \sum_{j \in N} (p_{ij} - c_{ij}) x_{ij}$)) (17)

Amount x_{ij} is shipped from source i to destination j
AND unit cost of shipment from source i
to destination j is c_{ij}
AND selling price of a unit shipped
from source i to destination j is p_{ij}
AND y_i additional units are procured
for shipment from source i
AND overhead transmission cost for
additional units from source i is d_i
$\Rightarrow$ objective(Maximize
$\sum_{i \in M} \sum_{j \in N} (p_{ij} - c_{ij}) x_{ij} - \sum_{i \in M} d_i y_i$) (18)

Now these rules would apply to the revised information about the problem situation to create the following mathematical formulation, which is quite different from the original formulation.

Model 1c

$$\text{Maximize} \sum_{i \in M} \sum_{j \in N} (p_{ij} - c_{ij}) x_{ij} - \sum_{i \in M} d_i y_i)$$

$$\text{s.t.} \quad \sum_{i \in M} p_i y_i \leq B$$

$$\sum_{j \in N} x_{ij} \leq a_i + y_i \qquad \forall i \in M$$

$$\sum_{i \in M} (1 - l_{ij}) x_{ij} \leq d_j \quad \forall j \in N$$

$$x_{ij}, y_i \geq 0 \qquad \forall i \in M \; \forall j \in N$$

We have illustrated how a knowledge-based modeler based on defeasible reasoning would represent and reason with assumptions. Specifically, we showed that defeasible rules and defeaters could be employed to make tentative conclusions based on the available information, and to revise them suitably when further information about the problem becomes available. It is easily seen that a defeasible reasoning system's built-in metapredicates (see §2 could provide useful information to a modeler in the model formulation process. For example, the predicate **howdefeatit** [5] could be used to examine under what circumstances a certain constraint or objective function would be an invalid (or valid) representation of the problem information. Our examples show that a rule-based system for model formulation could be made more useful by the inclusion of defeasible reasoning calculus to enable the system to reason with assumptions. Now we turn to a brief discussion of other ways in which the representation of assumptions could provide useful functionality for model formulation in a model management system.

The process of model development often results in several model versions, where each version corresponds to a certain set of assumptions. A change in an assumption affects not only the rules that get defeated as a consequence, but also other rules that have antecedents that are now no longer valid. For instance, in Example 1, a change in the assumption about shipment losses altered the model for total shipment at a destination. In addition, this change also invalidated the old formulation of the demand constraint and created a new one. A change in an assumption immediately raises the question "Which components of a model are affected by this change?" A system that represents model assumptions explicitly should be able to answer this question. Such a system would have the information necessary for it to a) retrieve the assumptions underlying a given model version, b) isolate the differences or commonalities in assumptions between two different model versions, c) explain the consequences of a change in an assumption, d) examine whether a model version is consistent with a given set of assumptions, and e) retrieve all model versions that are consistent with a given set of assumptions. We submit that this would be materially useful in a model development process wherein several model versions are developed and refined before the final model is formulated. While we have not specified *how* all of this might be achieved, we hope to have made clear the need to explicitly represent and reason with assumptions in a model management system.

4 Conclusions

There is general agreement among researchers that the cognitive process employed in model creation involves the application of a series of general model formulation rules constituting a modeler's knowledge about models, model classes, and modeling paradigms, to problem-specific information and assumptions. However, most research efforts directed at developing rule-based systems to support the construction of mathematical programming models have either ignored, or have made implicit, the role of assumptions in the modeling process. In addition, they have not suitably modeled the process of reasoning with assumptions. This process involves making tentative conclusions (either because of the unavailability of certain information or to keep the formulation simple at the

start) and then revising these conclusions to reflect new information about the problem. We have argued that the theory of defeasible reasoning is effective in explicitly and systematically representing the consequences of making certain assumptions, and in modeling the process of reasoning with assumptions. Modeling knowledge can be represented using absolute rules, defeasible rules, and defeaters. Defeasible rules are employed in making conclusions under some tacit assumptions that are normally satisfied. Their use is prevented by defeaters and other defeasible rules when information to the contrary is available. The details of implementing all of this in a formal system remain to be developed—and it is not clear how this might be done—but we believe that the issues discussed in this paper raise some interesting research challenges for the logic modeling and model management communities.

References

[1] Bhargava, H.K., and S.O. Kimbrough, "Model Management: An Embedded Languages Approach," *Decision Support Systems,* forthcoming, 1992.

[2] Bhargava, H. K., "A Logic Model for Model Management: An Embedded Languages Approach," *Ph.D. Dissertation,* University of Pennsylvania, 1990.

[3] Bisschop, J., and A. Meeraus, "On the Development of a General Algebraic Modeling System in a strategic planning environment," *Mathematical Programming Study*, 20, 1982.

[4] Bradley, Gordon H. and Robert D. Clemence, Jr., "Model Integration with a Typed Executable Modeling Language," *Proceedings of the Twenty-First Annual Hawaii International Conference on System Sciences, Vol. III, Decision Support and Knowledge Based Systems Track,* B.R. Konsynski, ed., (January 1988), 403-10.

[5] Causey, R.L., "EVID: System for Interactive Defeasible Reasoning," *Decision Support Systems* Special Issue on Logic Modeling, forthcoming, 1992.

[6] Clements, R.R., *Mathematical Modeling: A Case Study Approach*, Cambridge University Press, New York, NY, 1989.

[7] Fourer, R., D. Gay, and B.W. Kernighan, "A Mathematical Programming Language," *Management Science,* 36: 5, May 1990.

[8] Gass, S., "Managing the Modeling Process: A Personal Reflection," *European Journal of Operations Research*, Vol. 3, No. 1, 1987.

[9] Geoffrion, Arthur M.,"The SML Language for Structured Modeling," Working Paper No. 378, Western Management Science Institute, UCLA, 1990. (Two-part extract forthcoming in *Operations Research.*)

[10] Krishnan, R., "PM*: A Logic Modeling Language for Model Construction," *Decision Support Systems*, Vol. 6, pp.123-152, 1990.

[11] Murphy, F., and E. Stohr, "An Intelligent System for Formulating Linear Programs," *Decision Support Systems*, Vol. 2, pp. 39-47, 1986.

[12] Nute, D., "Defeasible Logic and the Frame Problem," in *Knowledge Representation and Defeasible Reasoning*, Studies in Cognitive Systems, Kluwer Academic Publishers, Boston, 1990.

[13] Nute, D., and M. Lewis, "A User's Manual for d-Prolog,", Advanced Computational Methods Center, University of Georgia, Athens, Georgia, 1990.

[14] Raghunathan, S., "An Artificial Intelligence Approach to the Formulation and Maintenance of Models," *Ph.D. Dissertation,* University of Pittsburgh, 1990.

[15] Saaty, T., and J. Alexander, *Thinking with Models: Mathematical Models in the Physical, Biological, and Social Sciences*, Pergamon Press, New York, NY, 1981.

Distribution List

Agency	No. of copies
Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
Dudley Knox Library, Code 0142 Naval Postgraduate School Monterey, CA 93943	2
Office of Research Administration Code 08 Naval Postgraduate School Monterey, CA 93943	1
Library, Center for Naval Analyses 4401 Ford Avenue Alexandria, VA 22302-0268	1
Department of Administrative Sciences Library Code AS Naval Postgraduate School Monterey, CA 93943	10
Ramayya Krishnan SUPA Carnegie-mellon University Pittsburgh, PA 15213	1
Sumitra Mukherjee SUPA Carnegie-mellon University Pittsburgh, PA 15213	1
Hemant K. Bhargava Code AS/Bh Naval Postgraduate School Monterey, CA 93943	5

List of Recent Working Papers

1991

91-01 **Shu S. Liao**, **Thomas P. Moore**, and Andrew G. Mackel
"Modeling the Transportation and Logistic Support System for the Aviation Assets Aboard a Navy Aircraft Carrier", October 1990.

91-02 **Nancy Roberts**
"Case Development Program Naval Postgraduate School: An Overview," May 1991.

91-03 **Thomas P. Moore**
"Collection of Wartime Data-are Reserves the Solution?," August 1991.

1990

90-01 **Hemant K. Bhargava**
"A Simple and Fast Numerical Method for Dimensional Arithmetic," March 1990.

90-02 **Thomas P. Moore**
"A Multiple Repairable Equipment and Logistics-Maintenance System (REALMS) Model," April 1990.

90-03 **Daniel R. Dolk**
"Structured Modeling and Discrete Event Simulation," August 1990.

90-04 **Daniel R. Dolk**
"Model Integration and Modeling Languages," March 1990.

90-05 **William R. Gates** and **Katsuaki L. Terasawa**
"The Economics of Defense Alliances," June 1990.

90-06 **Katsuaki L. Terasawa** and **William R. Gates**
"Allies, Adversaries and Commitment in Defense Alliances," September 1990.