

To appear in Journal of Logic Programming

# Modeling Simultaneous Events with Default Reasoning and Tight Derivations

Allen Van Gelder\*

April 7, 1988

## Abstract

This report describes how an arbitrator for the game of Diplomacy was implemented in Prolog, and discusses the advantages of logic programming for this type of problem. The design of the program is greatly simplified by using a default reasoning style to express many of the game rules. The technique of default reasoning is to specify a predicate (or property or situation) by stating sufficient conditions for a general rule, and then stating exceptions to it. Logic programming supports the expression of default reasoning quite directly with negation as failure. Tight derivations were employed to analyze cycles correctly. A tight derivation is one in which no goal has an identical ancestor. Speaking informally, any provable goal is provable by a tight derivation, so restricting the proof searches to tight derivations does not lose any proofs. Only the parts of the program that are susceptible to infinite recursions check for tightness. It is known that the use of tight derivations corresponds to a certain "preferred," or "canonical," minimal model for many programs that contain negation as failure; in such programs, there is normally no *minimum* model. Care must be taken in the way defaults are expressed to achieve proper correspondence between the model and the game rules.

## 1 Introduction

This report describes how an arbitrator for the game of Diplomacy was implemented in Prolog, and discusses the advantages of logic programming for this type of problem. This problem is a member of a large class of problems that can be formulated as discrete systems whose changes of state are governed by a set of "control" operators. Automated process control systems and multiple robot systems are examples of such discrete systems. A characteristic of large "real world" systems that is often absent from "toy" systems (but

---

\* Author's address: Computer and Information Sciences, Room 225AS, University of California, Santa Cruz, CA 95064, USA (avg@saturn.ucsc.edu)

present in Diplomacy) is that many control operators can be applied simultaneously. Typically, different operators mainly affect different parts of the system, but have some degree of interaction that prevents them from being treated as independent.

### 1.1 Discrete Systems

Discrete systems for our purposes are systems that change only at discrete points in time and are described by a finite or countably infinite set of parameters. Parameter values may have a continuous range, such as real numbers, in general, but we shall limit discussion to finite systems for the most part. Frequently in artificial intelligence applications an underlying continuous system is abstractly represented by a finite discrete system as an approximation.

Many models of discrete systems are based on the concepts of a *state*, *operators*, and rules that govern *changes in state*. To be computationally tractable, the rules should be local in nature. The *state* is frequently represented (actually or conceptually) as a large collection of facts. Rules for change are easiest to understand and apply if they involve testing and changing relatively few elements of the state. Complex problems frequently arise if many operators are applied simultaneously.

Issues of simultaneous operations also frequently arise in database update situations. Databases are in a consistent state when they satisfy certain constraints. An update transaction may require changes to many records, or facts. After all changes, the database should again be in a consistent state; however, if the changes are done serially, intermediate inconsistent states might be created. The problem is how to treat the required changes properly as simultaneous operations.

Another view of Diplomacy arbitration is that the rules are a body of law to be applied to a case, the facts of which are represented by the players' orders. The idea of applying logic programming to the interpretation of law has been explored by Sergot *et al.* [SSK\*86]. As is frequently the case with laws, the rules of Diplomacy have evolved since the game was introduced by Games Research, Inc. in 1961; the original rules were stated in 8 pages, and by 1982 had expanded to 11 pages.

### 1.2 The Game of Diplomacy

Diplomacy is a board game in which seven players, representing "great powers" of Europe in the early 1900's, manage their military units by simultaneously giving orders for all of their units. Each unit (a fleet or an army) may be ordered to move, hold, or assist another unit. Players can see the current positions of all units and can negotiate with each other before deciding upon their orders, hence the game's name. Orders are written privately, then opened all at once. The rules of the game are applied to arbitrate conflicts and determine which orders succeeded. Successful moves are then carried out on the board and another round of play commences. Although there are many other elements to the game, such as retreating, disbanding, and

building new units, in this paper we shall concentrate on the problem of arbitrating the players' orders in accordance with the published rules of the game. Avalon Hill Game Co., which currently markets Diplomacy, publishes a booklet containing these rules [Ava82], and includes it with the game equipment.<sup>1</sup> In later sections we introduce enough of the rules to allow a reader who is unfamiliar with the game to follow the discussion of programming issues involved.

### 1.3 The Implementation of an Arbitrator

Diplomacy arbitration was chosen as a vehicle for investigation because it has sufficient complexity to bring issues to light, yet is well defined and compact enough to be manageable by one person. Representation as a discrete system with operators is straightforward: The positions and owners of the military units comprise the state, and the orders represent the operators.

Other concurrent efforts to implement a Diplomacy arbitrator in other languages offered an opportunity for comparison of approaches. (The other efforts are not finished at this writing.) In another research project at Stanford, Diplomacy is being used to study problems of cooperation and commitment among independent autonomous agents whose goals are partially conflicting and partially consistent [GY87].

Logic programming has also been used to implement expert systems for problems in the games of bridge [SN89], and other games.

The program discussed here is written in Cprolog in a portable style that is easily converted to DEC-20 Prolog and compatible commercial versions. It represents about 100 hours of work. An initial version was put into "operation" about after about four weeks. (By this we mean it was given to others to use.) A substantial revision of the initial version was undertaken when the author obtained a copy of the up-to-date rules. (The author is indebted to Kevin Knight of Carnegie Mellon University for coding the database statements that represent the map, for providing and reviewing many program tests, and for advising the author that his idea of the rules was out of date.) The current program comprises about 1000 lines, including comments and a reasonable user interface. It has been in bug-free use for about six months as of this writing.

A fair amount of the effort went into making the program usable by someone with virtually no knowledge of Prolog. Prolog's operator declaration feature was used to allow orders to be written in a natural style close to the way players normally write their orders. A parser that "runs both ways" converts orders between external and internal representation. The parser inserts uninstantiated variables when certain parts of an order are elided, and fills them in as more information becomes available. For example, if two of the orders are:

```
fra a bel s nth => hol.  
eng f nth => hol.
```

---

<sup>1</sup> *Avalon Hill* has requested the author to state that the company has no association with the author or this article.

then the first order, when written out after the analysis, appears as

```
fra a bel s eng f nth => hol.
```

which is read by Diplomacy players as "French army in Belgium supports English fleet in North Sea into Holland." (Players actually use this terse style for writing orders; standard abbreviations are suggested in the rule booklet.)

After loading the program into Cprolog (which has no compiler), the user types

```
ro <input file name>.
```

which causes the orders to be read in and checked for errors. If satisfied, the user then types

```
res <output file name>.
```

which causes the orders to be arbitrated, and the results written on the named output file. Other options are displayed by typing "h." The commands can be repeated to process other sets of orders, to re-read orders after they have been corrected, etc. Each set of orders is processed in a few seconds on a Digital Equipment VAX 780.

## 2 Modeling Discrete Systems

Logic programming is a natural vehicle for simulation of a discrete system in which the effects of operators are described by rules. This is especially true when the system is largely nonnumeric. In many cases the operator has an effect on only one or a few state elements. In a purely logical specification an additional formalism is needed to specify that the operator does *not* effect other state elements. One approach is to use *frame axioms* [Nil80]. However, these prove to be computationally expensive, and do not generalize naturally to simultaneous application of several operators.

*Non-monotonic reasoning* encompasses a class of alternatives to frame axioms that are the subject of much current research. Two similar techniques in this class are the *closed world assumption* [Rei78] and *circumscription* [McC80, McC84]. Informally, the closed world assumption states that, with various qualifications, if a fact cannot be proven true, then it may be assumed to be false; that is, certain facts are *false by default*. This idea is closely connected to and generalized by circumscription. The circumscription axioms generally state that if a model, represented as a set of positive facts (or points where predicates are true), has a proper subset that also is a model, then the larger model should be rejected. The intuition is that the rejected model contains "unnecessary" facts. One might say that the closed world assumption and circumscription are biased against positive facts. This bias reflects the experience that normally in the real world most things are not true, and if they are, there is a reason.

## 2.1 Axiomatization of the Problem

The first step in developing a program to simulate a discrete system is to describe the system and its behavior by axioms. Normally, these axioms fall into several distinct categories:

- Facts that are true in all states
- Constraints on states
- Rules for operators

These axioms are static. The state of the system at a particular time can usually be represented as a set of facts. The state can change for three reasons:

- Intrinsic operators, often called *equations of motion* in continuous systems
- Control operators, applied by some agent(s) for a purpose
- Random operators

The reasons for change of state can interact, and not all three kinds need be present in the system.

Perhaps the fundamental problem for a discrete system is the forward (or backward) *time projection problem*:

Given a set of simultaneously applied operators and the current state, what is the successor (or predecessor) state?

Diplomacy arbitration requires the solution of a forward time projection problem, with the players' orders filling the role of operators.

## 2.2 Elementary Axioms of Diplomacy

The static axioms are derived by translation of the rules of the game. A complete description of the rules would be inappropriate, but we give enough details to illustrate the problems involved in automating them.

The axioms that express the "eternal truths" are simply facts:

- the names of spaces on the board, and whether each is land or sea;
- which spaces are adjacent for army movement, which for fleet movement;
- which spaces may be occupied by armies (land); which by fleets (sea, and land adjacent to sea);
- the names of the seven "great powers," and other miscellany.

The constraints of "legal" states are also quite simple to state informally:

