LOGIC PROGRAMMING FOR MACHINE TOOLS

Reggie Davidrajuh
Department of Electrical & Computer Engineering, University of Stavanger, Norway.
Email:reggie.davidrajuh@uis.no

Abstract: Logic programming is a very important issue in programming for machine tools, especially for decision-making for machine operations. First, this paper presents a survey on some of the mathematical approaches for logic programming. Second, it presents a case study on logic programming a flexible manufacturing cell.

Key words: Logic programming, mathematical approach, machine tools

1. INTRODUCTION

Logic functions are very important parts of programs for machine tools as the logic functions take decisions during the execution of the program. Making wrong or bad decisions can be very costly.

The aim of logic programming for machine tools is to develop a logic model of the system so that decisions can be made out of the model. Generally, the logic models and the decision making operations run on them are based on one of the following mathematical approaches:

1) Algebraic operations on Cartesian axes,
2) Searching,
3) Mapping,
4) Geometry.

Structure of this paper: Section-2 presents the mathematical approaches for logic programming. Section-3 presents a case study on logic programming a flexible manufacturing cell.

Please use the following format when citing this chapter:
Davidrajuh, Reggie, 2006, in International Federation for Information Processing (IFIP), Volume 207, Knowledge Enterprise: Intelligent Strategies In Product Design, Manufacturing, and Management, eds. K. Wang, Kovacs G., Wozny M., Fang M., (Boston: Springer), pp. 717-722.

2. MATHEMATICAL REASONING APPROACHES

2.1 Algebraic approaches using Cartesian Axes

Propositional logic and predicate logic falls under this category. Prepositional logic uses simple Boolean connectives (like 'not', 'and', etc.), and lacks quantifiers like 'all', 'among', 'only', 'at least one', etc. Predicate logic has these quantifiers. ***Mathematical reasoning approach***: First, the fundamental logic variables are identified and each logic variable is assigned an axis, creating the whole universe of discourse. Then the logic variables are connected into premises, creating subspaces. Finally, the premises are combined to form the logic system, connecting subspaces spanned by the premises. By connection, spaces that do not satisfy the constraints are removed, leaving a smaller space that represents the feasible solution. ***Advantages and disadvantages of this approach***: This approach is useful in providing formal proofs as it offers clarity. Since a Cartesian axis is assigned to each logic variable, generating subspaces spanning all possible states of all the variables, a *complete model* is obtained. The serious shortcoming of propositional logic and predicate logic is the *exponential growth* (also known as 'combinatorial explosion') of the logic model with increasing number of variables. For example, for M Boolean logic variables, the resulting space of M axes will contain M^2 subspaces. Decision-making is slower due to this exponential growth of the subspaces with increasing number of variables.

2.2 Searching

Production rules use searching as the basic inference mechanism. A production rule has the following form: IF (condition) THEN (conclusion). ***Mathematical reasoning approach***: The reasoning approach starts with a set of facts and look for those rules in which the IF clause matches the facts; if such rules are found ('hit'), then it proceeds to the THEN clause. This reasoning is known as 'forward reasoning'. In 'backward-reasoning', searching starts with a set of desired goals and to look for those rules in which the THEN clause (conclusion) matches the goals. ***Advantages and disadvantages***: The main problem associated with the searching approach is that the decision-making is slow. In addition to this shortcoming, production rules also have a weakness: a logic system is evaluated with a couple of 'if-then' statements; thus, for M multi-valued logic variables with N values M^N 'if-then' statements are needed to span all combinations of the variables. For a system of many logic variables, it will be impossible to write so many if-then statements to take care of all possible combinations of variables; thus, creating a complete model is not easy and prone to errors. The main

advantage of production rules is that the rules are simple, easy to understand, modify, and extend.

2.3 Mapping

Fuzzy logic is a popular technology that uses mapping with mathematical approximation for decision-making. ***Mathematical reasoning approach***: Decision-making in fuzzy logic is based on fuzzy rules that connect input and output parameters (fuzzy rule base), and the membership functions for input and output parameters. Inference mechanism in Fuzzy logic is implemented in three phases: 1) Fuzzification, which converts crisp input value into fuzzy value, 2) Inference, which computes fuzzy output value using fuzzy rules base, and 3) Defuzzification, which converts fuzzy output value into crisp value. ***Advantages and disadvantages:*** The first limitation is that fuzzy logic does not guarantee completeness; it is up to the designer to include all the fuzzy rules connecting all possible combinations between the input and output parameters. The second limitation is the difficulty in generating fuzzy rule base. The fuzzy rules generated for an application must properly adhere to the process dynamics with no contradictions between the rules; this demands deep understanding of the process dynamics.

2.4 Geometry

Array-based logic uses geometry (topology of connections between the fundamental components of a system) for creating a system model and for running operations on the model. A system consists of three fundamental components, namely *elements*, *connections*, and *sources*. Elements carry all the physical properties of the system; thus, elements are the fundamental building blocks of a system. Connections reflect how elements in a system influence each other; thus, connections represent the structure of a system. Finally, sources are the environment's influence on a system. ***Mathematical reasoning approach***: The inference mechanism consists of three phases (Davidrajuh, 2000): 1) Identifying the primitive system, 2) Making the connected system, and 3) Applying the sources, and solving the connected system. ***Advantages and disadvantages***: The previous subsections state that a complete model of M multi-valued logic variables with a domain of N values contains M^N subspaces. Array-Based Logic avoids this exponential problem by compressing M^N subspaces into $M \times N$ linear representation (Møller, 1995). Array-based logic also provides mechanisms for operations to operate on the compressed representation in linear time.

2.5 Summary of the approaches

Given below are the important characteristics of the different types of mathematical approaches and the sample logic types that fall under the approaches: ***Inference cycle time***: Due to the combinatorial explosion, algebraic methods are slower. By nature, searching approaches are also slow, unless the implementations use some special data structures like 'hash coding'. Mapping by fuzzy logic is fast. Finally, array-based logic is fast due to the compact linear model and the linear-time operation it provides. ***Complete system***: Both the algebraic approaches by propositional logic and predicate logic, and the geometrical approach by array-based logic provides complete logic models. ***Implementation languages***: Most general purpose programming languages can be used for implementing propositional logic statements. Prolog can be used to implement predicate logic (Emden and Kowalski, 1979). Prolog, C-LIPS and Jess programming languages can be used for implementing production rules (Wagner, 2002). There are also numerous commercial packages for fuzzy logic, e.g. fuzzy logic toolbox for MATLAB platform. SABL is a toolbox of functions for implementing array-based logic (Davidrajuh, 2000).

3. CASE STUDY

The case study is about logic programming a flexible manufacturing cell (FMC) consisting of a vertical machining center (VMC), a horizontal machining center (HMC), a robot, and a conveyor belt. Here are the simplified operational specifications of the FMC: Step-1: To start a cycle, a raw part must be available on the incoming conveyor belt, and the robot is also available. Step-2: Robot moves a raw part from conveyor and loads it at HMC. Step-3: Milling operations are performed at HMC while robot backs off (returns). Step-4: Robot unloads semi-finished part from HMC, loads it to VMC and returns. Step-5: Drilling operations are performed at VMC; simultaneously robot perform step 2. Step-6: Robot unloads the finished part from VMC, deposits it on the outgoing conveyor belt and returns. In steady state steps 2-6 are repeated.

3.1 Vertical and Horizontal machining centers

Components of the VMC are shown in figure-1. Three sensors (micro-switches) *B*, *M*, and *E* control the operation of the VMC: Sensor *B* indicates that the boring spindle is at the rear position; Sensor *M* indicates that the boring spindle has reached the feeding position; Sensor *E*

indicates that the boring spindle has reached the final destination, and ready for backward motion.

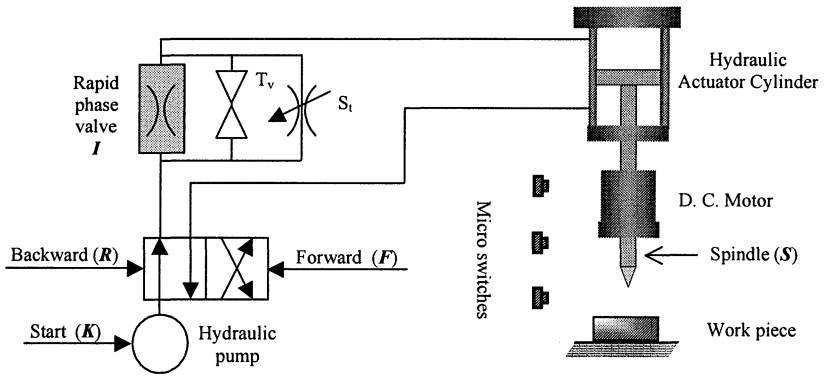

Figure 1. The Vertical Machining Center (VMC)

The spindle is at the rear position initially, and the operator switches on the system by a very short signal *K*. VMC then will go through three modes of operation: Mode-1: Starting from the initial state, hydraulic circuit will open rapid phase valve *I*, and the spindle will go forward by the opening the valve *F*. Mode-2: At position *M*, the rapid phase valve *I* will be switched off to start a controlled feed forward motion; This motion is regulated manually by the servo valve S_t. At position *M*, the spindle motor will also be switched on by a signal *S*. Mode-3: At position *E*, the backward motion *R* will begin. Simultaneously, the rapid phase valve *I* will be switched on.

Components and operations of the HMC are very similar to that of the VMC. The only difference is the orientation of the machining operations.

3.2 Logic programming the FMC

Logic programming is divided into two tasks: 1) Programming the machine centers VMC and HMC, and 2) Programming the robot.

Logic programming the machine centers VMC and HMC: Descriptions about the VMC/HMC reveal that both the VMC and the HMC have 4 inputs (K, B, M, and E) and 4 outputs (F, S, I, and R). Thus, the logic model has 8 Boolean variables, making 256 possible combinations. This means, 1) a *faster inference mechanism* is needed due to the large size of the model. In addition, 2) the logic model must be *complete model* as well, as one would minimize costly mistakes due to operating on a incomplete model. Only array-based logic based on geometry fulfills these two criteria. If speed is not that important (assuming that the VMC/HMC machines are slower) then propositional logic can also be used as they satisfy the complete modeling requirement; Though predicate logic can also be used, but it is not necessary,

as the operational specifications seemingly do not demand for logic quantifiers.

Logic programming the robot: Programming an industrial robot is done at three different levels (Nilsson, 1996): 1) The motion control level, 2) The application-specific level, and 3) The end-user level. Only the end-user programming is done by customers, using specialized robot programming tools (before delivering robots to customers, robot manufacturers do the programming at the other two levels as it requires detailed knowledge of the robotic system dynamics). Usage of specialized robot programming tools suffers from a number of drawbacks (Gustavo et al, 2005): high learning curve of the specialized languages, non-reusable code, and limited portability across different robots. Most importantly, unlike the higher-level languages, these specialized languages are *opaque* meaning they reveal no details about how the logic models are created by the compilers and how the logic variables are manipulated. ***Thus, thinking about the mathematical approaches for logic modeling is not applicable in for robot programming.***

4. CONCLUSION

This paper presents 4 fundamental mathematical approaches for logic programming for machine tools. Choosing an appropriate mathematical approach for logic programming is very important as different approaches provide different characteristics such as complete models, fast decision-making, choice of implementation languages, etc. However, as the case study proves, taking a mathematical view is not always applicable for logic programming for machine tools.

5. REFERENCES

Davidrajuh, R. (2000). Automating Supplier Selection Procedures. PhD dissertation, Norwegian University of Science and Technology (NTNU).

Emden, M. and Kowalski, R. (1979). The semantics of Predicate Logic as a Programming Language. *Journal of the Association for Computing Machinery*, Vol. 23, No. 4, October 1976, pp. 733-742

Møller, G. (1995) On the Technology of Array-based Logic. Ph.D. dissertation, Technical University of Denmark

Nilsson, K (1996). Industrial Robot Programming. PhD thesis, Department of Automatic Control, Lund Institute of Technology

Wagner, G. (2002) How to Design a General Rule Markup Language. The workshop on XML Technologies for the Semantic Web (XSW2002). Humbolt University, Berlin, Germany, June 2002