Plan Recovery in Factory Automation Domain

Xuemei Wang
Rockwell Science Center
444 High St. #400
Palo Alto, CA 94301
mei@rpal.rockwell.com

Shivakumar Sastry
Advanced Technology
Rockwell Automation
1201 South Second Street
Milwaukee, WI 53204
ssastry@ra.rockwell.com

Abstract

Over the past decade, artificial intelligence conceptsand techniques have been applied to many aspects of manufacturing, ranging from product and process development, to production management, to process diagnosis and quality control. Planning technology will play an important role in the future of intelligent manufacturing, especially in factory automation systems.

Drawn from our experiences on automation in the manufacturing domain, we present lessons learned as well as issues arisen in order to successfully employ planning technology in this real world domain. We primarily focus on the execution environment, and discuss how plan generation and execution impact each other. A preliminary approach is described.

Introduction

Factory automation systems are commonly used in manufacturing various products as varied as cookies, cars, and paint. Such applications are typically controlled by a special computing device called a Programmable Logic Controller (PLC) that is configured to operate, along with many other electronic modules, in a larger system that controls the application. This paper presents issues that arise when applying AI Planning technology in the domain of factory automation systems. Some lessons learned from our early real-world experiences are discussed.

AI planning technology is relevant in the domain of factory automation since this technology has good potential to significantly improve the efficiency of operations, including both programming efficiency and factory run-time efficiency. Traditionally within the context of AI, "planning" usually refers to the process of "finding a sequence of operations to achieve a set of goals, given the initial state". In the context of factory automation, planning must go beyond identifying such a sequence and must aim to integrate with many diverse technologies that are collectively necessary for an effective operation of the automation application. The remainder of this paper elaborates on this central thesis.

We will start by introducing the factory automation domain as a planning domain. We then describe how process plans are manifested in factory control, and show why efficient failure recovery is crucial to planning applications. We analyze situations where recovery is difficult, and present our preliminary and proposed approaches that require effective plan monitoring and mixed-initialtive recovery. We conclude with suggestions on future research in applying planning to factory automation domains.

Factory Automation as Planning Application

Consider a typical instance of a factory automation problem. The end goal is to produce as many products as possible, in a given unit of time, to meet certain cost objectives, quality and safety constraints. Generally, three tasks are completed: (1) product design and development, (2) process design and development, and (3) production (Chang et al 1998). Product design specifies desired properties of the product, including geometry (tolerances, datums), model features (material, surface), and assembly features (components, welds/adhesives), and is usually carried out in a CAD system. Process design, a.k.a. process planning, is the function that establishes which processes and parameters are to be used, as well as those machines capable of performing these processes, to convert parts from their initial form to the final product. These machines must be arranged ergonomically to satisfy safety constraints, optimize floor space, and minimize costs (such as wiring costs and piping costs). Finally, during the production phase, a.k.a. plan execution, the product is manufactured as a sequence of operations performed by these machines with human guidance. It is important to recognize that an effective control system is necessary to facilitate the production process.

Process planning, i.e., the generation of sequence of operations, has received significant attention among researchers (Kempenaers et al 1994, Kiritsis 1995), justifiably so because of its knowledge and labor intensive nature. Planning in manufacturing domains not only involves determining the sequence of operations, but also equally important is the task of determining the tools and fixtures. However, the "production" aspect of the manufacturing process has not received its fair share of attention. Activities that commonly occur during the production phase have strong upstream requirements on the initial design and programming of the control system. In such a dynamic context, AI planning technology must be able to integrate with other diverse technologies that can provide an intelligent and efficient execution environment.

Plan Execution Environment

Factory automation systems are dynamic systems involving many (often hundreds) of machines. Numerous reasons may lead to production downtime. For example, machines may fail, components (especially when there is contact) wear down, parts may be out of sequence when multiple products are manufactured on the same plant, human operators may forget or fail to complete some steps, critical safety constraints may be violated.

The costs of disruptions that occur are high. In general, manufacturing process experience significant cost increases due to downtimes. For example, in many automobile plants, a minute of downtime costs the company a few thousands dollars, and there is on average one case of over 40 minutes downtime per shift (8 hours) per plant. Our field studies indicate that timely recovery from equipment and other types of failures plays a critical role in reducing downtime and increasing plant efficiency. Thus, factory automation systems are expected to recovery quickly after any kinds of operation failure. In addition, they must also be safe even when catastrophic events such as fire, accidents, or other natural disruptions occur.

Thus, to ensure efficiency and increase productivity, planning must be augmented with techniques for effective monitoring during plan execution, post-failure plan robustness, and speedy recovery from undesirable operating conditions. The following sections address each of these issues in detail.

From Process Plans to Executable code

Programmable Logic Controllers (PLCs) were first introduced in 1968 to replace hard-wired relay systems that were used for automatic control. Modern PLCs are highly optimized for high speed input and output. Many PLCs support advanced features to perform control specific tasks such as PID control and enhanced data transfers with a predictable response time.

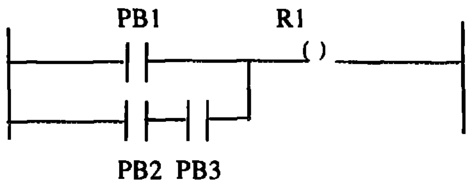

Figure 1. RLL rung

PLCs are commonly programmed using Relay Ladder Logic (RLL). A ladder program consists of a series of rungs, where each rung specifies a mapping from one or more input conditions and internal states to one or more output conditions and states. Figure 1 shows the logic represented by an equation: R1 = PB1 OR (PB2 AND PB3). Ladder programs are typically developed using a finite state machine as the model of abstraction. Other programming languages for PLCs are Structured Function Charts, Structured Text, and Flow Charts. Essentially, the finite state machine abstraction continues to be the base abstraction.

Since RLL is a dominant language used for programming PLCs, the remainder of this discussion deals with control systems programmed in RLL. The issues are equally relevant even when other programming languages are used to program factory automation systems.

Process plans are ultimately manifested in RLL code. Currently, this translation of high-level and intermediate-level process plans to executable ladder code is an expensive, manual process that is error prone. To ensure safety of equipment, operators, and the environment, ladder programs are enhanced with instructions that guarantee these safeties. Our study revealed that less than 20% of the final ladder code directly implements the process plan. The remainder or the ladder program is dedicated to diagnostics (20 to 40%), factory information systems (20 to 30%) and Human-Machine-Interface (25%). These ratios highlight the importance of these aspects for successful plan execution.

Why Plan Recovery Is Difficult

Through reverse-engineering of existing RLL code that is controlling assembly lines, we have discovered that the process plan as encoded in RLL generally only handles normal operation sequence. In other words, only the states that occur during normal operation sequences are encoded in the controller. If human operators try to start automated operations when the factory is in any other state, the controller would not be able to recognize the state and thus unable to actuate any correct outputs. For example, when equipment (such as a robot) breaks, factory personnels must first repair the failure equipment. In the process of repairing, they often manually change the state of devices (for example, by pushing the valves of clamps to open/close the clamps). The factory is thus often left in a state not recognized by the PLC controller. When the human operator relinquishes control to the controller after the individual device is repaired, normal workcell operations still could not be resumed because this state is not encoded in the controller.

An even trickier situation is when effects of operations are time-sensitive. Time-sensitive effects are outcomes of activities that have durations, that is, the outcome signals are only present for duration of time and no longer hold after the duration. This situation is very common including both device states and process states, and a form of visualization of the model and consequences of additions/changes.

in complex devices such as welding or gripper robots. For example, one outcome signal when a robot has finished its pre-defined program is robot-program-complete, which typically last a few seconds. Consider a situation where the robot-program-complete signal is present for 2 second. If the robot faults immediately after it finishes its program, and clearing the fault takes more 2 seconds, then this signal is no longer present when recovery is initiated. If the pending operations depend on the robot-program-complete signal, a re-execution of the robot program is required. If re-welding is prohibited, a series of actions must be taken in order to resume the normal process. Although many current implementations use internal variables to latch the value of these time-sensitive effects, most implementations are not systematic and thus error prone.

There can also be time-delayed effects, where an effect takes place only after a time period has elapsed, or an event starts when a pre-specified time period has elapsed after another event started. For example, we may start a motor after water has been on for 2 seconds. To re-achieve these effects upon failure recovery, one often has to re-execute this operation, which might be many steps back. This often requires undoing some previous activities that is often difficult if not impossible on factory floors.

Mixed-initiative Plan Recovery

In light of the above findings, we are using a mixed-initiative plan recovery, where effective monitoring of plan execution is critical, and the human operators are involved in restarting the system with computer guidance.

Effective Monitoring of Plan Execution

Upon execution failure, the first thing an engineer must check is the state of the process, what activities are executed and what not. Providing effective monitoring of plan execution can help the operations personnel in quickly identifying the failure point, assessing the source of failure, and better recovery. To accomplish this, we need good models of the factory,

To this end, we have modeled the factory using Petri-net models and a partial-order plan with preconditions and effects. While each representation has its own advantages and disadvantages, domain engineers have expressed that partial order plan with explicit representation of interlock and safety conditions for each activity will be a helpful tool for monitoring plan execution on factory floor. The petri-net model is ineffective for monitoring purpose due to lack of a good visualization tool.

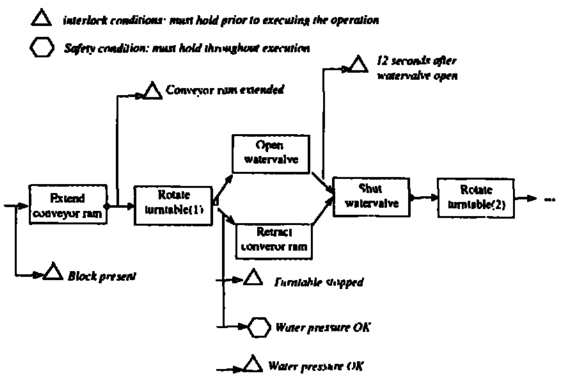

Figure 2. Monitoring process plan

For example, let's consider a factory for manufacturing widgets. A widget is a block of wood on which certain machining operations are performed at different stations. There are several stations in this example factory and a system of conveyor belts that bring unprocessed wooden blocks into the factory, move the wooden blocks out from station to station, and finally move widgets out of the factory into the warehouse. One of the processing stations is a 3-position turntable. When a block is in position, the turntable moves the block from one position to another – and finally, returns the block to the conveyor. In the first position, the block is washed with water. In the second position it is dried using high-pressure air. Finally, in position 3, the block is treated with a special chemical. A part of the process plan in the turntable can be expressed as in Figure 2. It is critical to realize that existing planning representation must be enhanced with interlock and safety preconditions (interlock condition are those that must hold at the beginning of the execution, and safety conditions are those that must hold throughout execution). These conditions can be displayed and manipulated (e.g., bypass the condition) by human operators on the factory floor.

Just as process plans are realized through RLL code, effective monitoring techniques must also be integrated with the rest of the RLL code for process monitoring in the panelViews.

Speedy Recovery from Failure

Most previous work on classical planning has focused on "ante-failure" robustness, i.e., generating robust plans that take into consideration different outcome possibilities of actions or the environment in order to avoid failure (Blythe 1994). Previous work on reactive planning has addressed plan repair issues by readjusting the initial plans (Firby 1989). In the manufacturing domain, as in all other real-world domains, failure of the process (due to uncertainty in the domain and equipment failure) is a normal complement of production line. In addition, the original process plan cannot be altered with the exception of very infrequent well-planned changes. Thus post-failure plan robustness, i.e., the ability to resume normal operations speedily upon failure, is critical in successful deployment of planning systems. In general, a recovery mechanism must be effective in handling the following two central issues:

- When one device faults or an emergency condition arises, how are other devices and machines affected? In other words, what are the behaviors of other devices/machines in the whole workcell and factory floor? Should they stop immediately, or complete the activity they are currently engaged in?
- When the fault device is repaired, what must be done in order for the manufacturing line to resume its normal operation?

Appropriate Suspension of Machines

Several approaches are used in typical manufacturing domains in handling suspension of other machines. Their effectiveness depends on several characteristics of fault or emergency situations, including factory safety requirements, inter-dependancies between the fault device and other devices, and the functionalities of each device. These approaches are:

- Stop all machines immediately upon detection of any malfunction in the production process. This is typically done when safety conditions are violated and human lives are at stake. However, if not done carefully, this often leads to lower production capabilities of the plant, as other parts of manufacturing tasks independent of the failed machine should be able to continue. In addition, if a machine is stopped in the middle of its execution, resuming is far more difficult than if the machine is allowed to complete its current operation. Furthermore, excessive machine stoppage also shortens the machine life cycle.

- Stop immediately only those devices that are affected by the failed device while allowing some devices to continue execution until they finish the current activity, and leaving the rest of the factory unaffected. This approach requires identifying the scope of each type of failure for each device, as well as the scope of each safety gate/matt. For example, when a robot in one station faults, this robot should stop immediately, while a different robot in the same station can continue execution until its current process (e.g. welding) is done, yet all other stations can be functioning in their normal operations. The challenge is to find the minimal number of devices that are affected by each type of failure for each device, and if possible, do so automatically with planning technology (these tasks are all currently manually performed.)
- Use redundant devices for uninterrupted execution upon minor failures. For example, use two sensors instead of one for the same purpose, so that when one sensor is broken, the controller can report the failure while continuing execution. The broken sensor can be fixed and re-installed in the meantime. This mechanism is only feasible if the redundant device is not expensive and can be easily placed in the workcell – it's not wise to use two robots only to have one robot idle most of the time.

Effective Resumption of Normal Operations

As we have discussed in this paper, prolonged downtime is typically due to the fact that PLC does not recognize the state of the factory, or the missing precondition for restarting is not easily achievable.

Although an automated real-time plan recovery system encoded in RLL is an attractive idea, it is extremely challenging for the following reasons.

- A planner requires a complete model of the domain. Replanning from failure in particular is very sensitive to the accuracy of precondition and effects of operators (Wang et al, 1997,Wilkins 1989). However, explicit specifications of action preconditions and effects are difficult to construct in practice. One difficulty is in specifying all the possible outcomes of each action. For example, physically, you can always close a clamp, but under normal operations, we want to ensure that clamps only close when there is a part in it, and especially only close the clamp when the part is properly seated. To represent these situations, we need to add "context-dependent" preconditions. Due to physical relationships of the set of clamps mounted together, a clamp can only be closed when other clamps are in certain states, and such constraints can be arbitrary, varying from one workcell to another. Another difficulty is determining how preconditions and effects are actually sensed on the factory floor. For example, suppose that we do decide that "part in place" is a logical precondition of close clamp. How this precondition is sensed depends on the context - differing sets of sensors, or maybe in some cases asking the operator. Fixing this up will require manual adjustment for each action.

  As another example, consider modeling robot activities as planning operators. Robot activities are implicitly specified in robot code, different for every robot. The preconditions and effects of the activities are further constrained by various specifications and protocols of the robot. Modeling robot in a planning specification proves to be challenging.
- Even if we can completely model the physical world, there are states that can not be sensed. For example, when the clamps is stuck in between open and close, we have no way of distinguishing where the clamps really are.
- A correctly modeled state-space could be huge, and the time and memory constraints do not permit exhaustive exploration.

As fully automated approach is not readily feasible, and furthermore, pre-calculating all the recovery steps for every possible state of the factory is compositionally infeasible, we are using a mixed-initiative approach for plan recovery. Preliminary work is underway.

In our approach, a factory is modeled by a list of activities (operators), a set of constraints, and a normal operation process plan. An operator is specified by its actuators, sensors, duration, and latched property (i.e., whether the actuators and sensors are transient or latched to stay on until an inverse operation is performed). Constraints specify for each activity what other activities must be on or must not be on in order to actuate this activity. Process plans are modeled by specifying the predecessors for each activity.

During recovery, the system first identifies the workcell state when the process was interrupted. Then the system computes the required sensor as well as actuator values for resuming normal operation. This must be a state that PLC recognizes. The recovery mechanism then backchains from these requirements to the current factory state while ensuring that the constraints are not violated. If a plan is found, the system suggests it to the human operator. Otherwise, it informs the human operator which relevant inputs are not in the state expected, and give the operator the choice of either manually fixing the state, or asserting that the condition is in fact okay. If the normal workcell still can not be resumed, then:

- If the missing input is a time-sensitive or time-delayed effect, advice human operators to either re-execute the activities with such effects if possible, or perform other activities (such as removing the part and load a new part in) prior to re-executing the desired activities.
- Otherwise, if the missing input is an effect that was previously achieved, but is undone, advice the human operators to undo operations prior to re-executing the operation with the undone effect.

Note that the human operators are always included in the recovery process.

Summary

Efficient process plan execution is a critical component in factory automation domain. Our field studies, analysis, preliminary work all show that not only process planning directly affects the execution performance, but also that the execution demands place requirements on process plan generation in terms of the quality of the plan. Planning and execution interplay, both must be addressed when applying AI to real-world problems.

Conclusion and Future Work

We have characterized factory automation as a challenging and promising domain for planning application and described the complexity of plan executing environment. We analyzed the issues in failure recovery and presented a mixed-intiative approach to address these issues that exploits effective monitoring of process plan and post-failure robustness.

In addition to the areas we have presented in this paper, factory automation domains in particular, and manufacturing domains in general, are rich domains with promising new avenues for research and application. Areas pertinent to planning and scheduling include, for example, integrating diagnostics with recovery, intelligent monitoring (including appropriate alarming), and data mining from floor shop execution traces (to enhance the quality of the plan). We believe that AI can play an important role in this domain.

Acknowledgment:

We thank Denise Draper for stimulating discussions and helpful comments on this paper.

References:

Blythe, J. 1994. Planning with external events, *In proceedings of the conference on Uncertainty in AI*, Menlo Park, Calif.: AAAI Press.

Chang, T. C., and Wysk, A. R, and Wan, H. P., 1998. *Computer-Aided Manufacturing*. Prentice Hall Inc, Englewood Cliffs, New Jersey 07632.

Firby, J. 1989. Adaptive Execution in Complex Dynamic Worlds, Ph.D. Thesis, Yale University Technical Report, YALEU/CSD/RR #672.

Kempenaers, J; Pinte, J.; Kruth, J.-P.; Detand, J.: A Collaborative Process Planning and Scheduling System. 14th ASME Int. Computers in *Engineering (CIE94) Conf.;* Minneapolis, September 11-14 1994

Kiritsis, D.1995. A Review of Knowledge-Based Expert Systems for Process Planning: Methods and Problems, *International Journal of Advance Manufacturing Technology, 10(4), 240*

Wang, X., and Chien, S. Replanning using hierarchical task network and operator-based planning, *4th European Conference on Planning, September 1997, Toulouse France*

Wilkins, D. *Extending the classical AI planning paradigm*, 1988, Morgan Kaufmann