

Optimizing Production Manufacturing using Reinforcement Learning

Sridhar Mahadevan and Georgios Theodorou

Department of Computer Science
Michigan State University
East Lansing, MI 48824
(mahadeva,theochar)@cps.msu.edu

Abstract

Many industrial processes involve making parts with an assembly of machines, where each machine carries out an operation on a part, and the finished product requires a whole series of operations. A well-studied example of such a factory structure is the transfer line, which involves a sequence of machines. Optimizing transfer lines has been a subject of much study in the industrial engineering and operations research fields. A desirable goal of a *lean* manufacturing system is to maximize demand, while keeping inventory levels of unfinished product as low as possible. This problem is intractable since the number of states is usually very large, and the underlying models are stochastic. In this paper we present an artificial intelligence approach to optimization based on a simulation-based dynamic programming method called reinforcement learning. We describe a reinforcement learning algorithm called SMART, and compare its performance on optimizing manufacturing systems with that of standard heuristics used in industry.

Introduction

Industrial manufacturing usually involves making parts with an assembly of flexible machines. The machines are programmable in some way in that their operations can be selected from a repertoire of basic operations, including doing one of several operations on a part, or doing maintenance. Optimizing manufacturing requires making parts with the lowest cost, which is usually a function of the number of parts stored in inventory (not yet finished), maintenance and failure costs of the machines involved etc. Although there are well-known stochastic models for optimization of such machine assemblies, these models are intractable to solve for large numbers of machines (usually three or more) (Gershwin 1994).

An alternative approach to optimization is through the use of simulation models, which are a time-honored approach to modeling complex systems (Law & Kelton 1991). Many software tools are available to simulate

a wide range of systems including manufacturing, process control systems, and robotic control. These tools, however, rely on a human decision-maker to supply a fixed procedure or policy, and only provide a statistical profile on the quality of the policy. In this paper, we describe a new approach to automatically find good policies to intelligently control a complex simulation model. Our approach is based on a machine learning framework for autonomous agents called *reinforcement learning* (RL) (Mahadevan & Kaelbling 1996; Kaelbling, Littman, & Moore 1996; Sutton & Barto 1998). This framework models the sequential decision making problem faced by an agent (i.e., what action should the agent do in a particular state) as a Markov Decision Process (MDP) (Puterman 1994). The aim of the agent is to learn a policy (mapping states to actions) that maximizes its performance on the task.

Reinforcement learning is an ideal approach to optimize simulation models, since these generate sample trajectories through the state space as the agent experiments with its current policy. Classical optimization methods, such as dynamic programming (Bertsekas 1995), cannot be applied here because they require a *transition* model to predict the next state distribution, given the current state and action. Simulation models can generate sample next states, but cannot directly provide the information needed by dynamic programming.

In this paper, we are interested in studying a broad class of optimization problems in the general area of manufacturing, such as flexible product manufacturing, product scheduling, maintenance, inventory control, and transfer line optimization. In particular, we will focus on the problem of optimizing a transfer line (as shown in Figure 1). Transfer line optimization is a well-studied problem (Gershwin 1994), but the analytical models are intractable to analyze for lines with more than 2-3 machines. Transfer lines with 20 machines are standard in industry, but generate state spaces with 10^{20} states or more, which seems out of

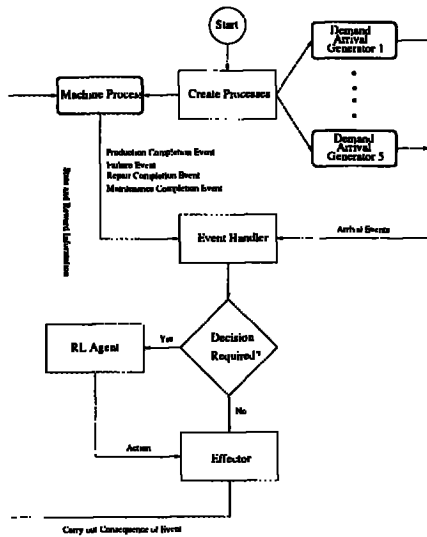


Figure 3: The SMART reinforcement learning algorithm is implemented using CSIM, a commercial discrete-event simulator. CSIM allows construction of simulation models of arbitrary size, since it is directly based on the C language.

part. This machine then picks up its raw materials from the buffer immediately behind it, which releases the Kanban card for that part. This way, information flows back upstream to all the machines for them to produce one additional part.

The following cost model for optimizing a transfer line was adopted. Repairs are modeled as incurring a cost of -1000. Each unit of satisfied demand is worth a positive reward of 10. In addition, there is a continual rate cost based on the average inventory levels, which is scaled by 0.1. In our experiments, each of the three machines receives the same reward.

Table 1 compares the performance of the SMART algorithm with the Kanban heuristic. The table shows the total inventory levels needed by SMART and Kanban for various target demand levels. In each case, the fill rate (percentage of demand satisfied) was about 98%, indicating that most of the demand was satisfied. As the table shows, SMART outperforms the Kanban heuristic in needing much fewer items in the inventory.

Target Demand	SMART	Kanban
0.2	106.7	135.27
0.5	100.69	117.3
0.6	85	101.15

Table 1: Comparison of Total Average Inventory Levels of SMART with Kanban heuristic.

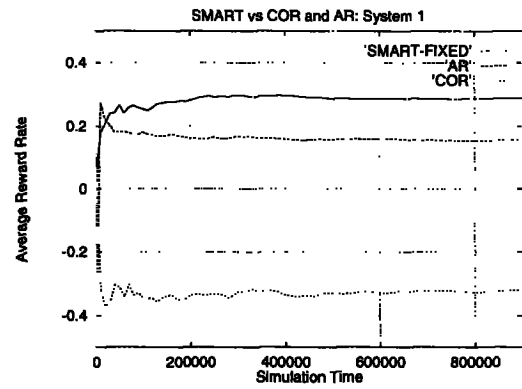


Figure 4: All curves show median-averaged costs of production of 5 parts using an unreliable machine. The top curve corresponds to the policy learned by the reinforcement learning algorithm. The bottom two correspond to fixed policies representing two heuristic methods.

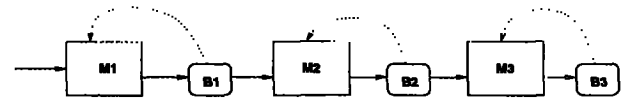


Figure 5: Diagram showing the Kanban heuristic policy for optimizing a transfer line.

Inventory levels are of course, only part of the overall improvement offered by SMART. In particular, because SMART actually learn a maintenance policy in addition to a production policy, the total number of failures is far fewer when using SMART. Table 2 compares the number of failures incurred under SMART and under Kanban. Note that since the Kanban heuristic does not incorporate any maintenance policy, it will obviously incur a far higher number of failures, as the table shows. The higher number of failures implies that the Kanban heuristic will result in a much lower average reward, as shown in Figure 6.

Target Demand	SMART	Kanban
0.2	843	2878
0.5	1880	6458
0.6	2992	8980

Table 2: Comparison of Total Number of Machine Failures for SMART with Kanban heuristic.

Limitations and Future Work

The results described in this paper should be viewed as preliminary, particularly those for optimizing a transfer line. We are currently running additional experi-

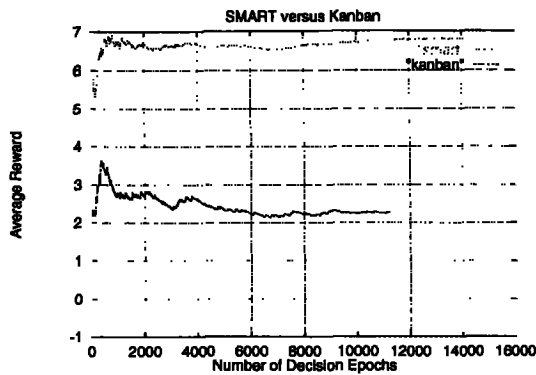


Figure 6: This figure compares the average reward of the policy learned by SMART with that produced by the Kanban heuristic.

ments, which we expect to report during the presentation of this paper. Here, we briefly discuss the nature of the additional experiments being planned.

There are several additional heuristics that have been developed, such as the CONWIP (constant work-in-process) strategy (Spearman, Woodruff, & Hopp 1990), as well as hybrid methods (Bonvik & Gershwin 1996) which combines a kanban with a CONWIP strategy. We plan to compare SMART to these other heuristics.

Another research direction is to investigate hierarchical reinforcement learning methods to scale to large transfer lines, as well as generalizations of transfer lines to job shops and flow shops. Hierarchical optimization of such assemblies has been theoretically investigated in depth (Sethi & Zhang 1994), but however, the computational study of such optimization methods is limited. There is currently much interest in the reinforcement learning literature on hierarchical methods (Parr & Russell 1998), but there has yet been no demonstrable results on interesting large scale problems. We believe the factory optimization domain to be an excellent testbed for hierarchical reinforcement learning algorithms. Finally, each machine is given the same global reward, and we plan to conduct comparative experiments where each machine is given different local rewards.

Acknowledgements

This research is supported in part by an NSF CAREER Award Grant No. IRI-9501852.

References

Bertsekas, D. 1995. *Dynamic Programming and Optimal Control*. Belmont, Massachusetts: Athena Scientific.

Bonvik, A., and Gershwin, S. 1996. Beyond kanban: Creating and analyzing lean shop floor control problems. In *Conference on Manufacturing and Service Operations Management*.

Boutillier, C.; Dearden, R.; and Goldszmidt, M. 1995. Exploiting structure in policy construction. In *Proceedings of the Fourteenth IJCAI*.

Crites, R., and Barto, A. 1996. Improving elevator performance using reinforcement learning. In *Neural Information Processing Systems (NIPS)*.

Gershwin, S. 1994. *Manufacturing Systems Engineering*. Prentice Hall.

Kaelbling, L.; Littman, M.; and Moore, A. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4.

Law, A., and Kelton, W. 1991. *Simulation Modeling and Analysis*. New York, USA: McGraw-Hill.

Mahadevan, S., and Kaelbling, L. P. 1996. The NSF workshop on reinforcement learning: Summary and observations. *AI Magazine* 1(12):1-37.

Mahadevan, S.; Marchallick, N.; Das, T.; and Gosavi, A. 1997. Self-improving factory simulation using continuous-time average reward reinforcement learning. In *Proceedings of the Fourteenth International Machine Learning Conference*. Morgan Kaufmann. 202-210.

Parr, R., and Russell, S. 1998. Reinforcement learning with hierarchies of machines. In *Advances in Neural Information Processing Systems (NIPS)*.

Puterman, M. L. 1994. *Markov Decision Processes*. New York, USA: Wiley Interscience.

Sethi, S., and Zhang, Q. 1994. *Hierarchical Decision Making in Stochastic Manufacturing Systems*. Birkhauser.

Shingo, S. 1989. *A Study of the Toyota Production System from an Industrial Engineering Viewpoint*. Productivity Press.

Simmons, R., and Koenig, S. 1995. Probabilistic robot navigation in partially observable environments. In *Proceedings of the IJCAI*, 1080-1087.

Spearman, M.; Woodruff, D.; and Hopp, W. 1990. CONWIP: An alternative to kanban. *International Journal of Production Research* 28(5):879-894.

Sutton, R., and Barto, A. 1998. *Reinforcement Learning: An Introduction*. MIT Press.