Modeling Manufacturing Systems with Stochastic Petri Nets and Markov Decision Processes with Imprecise Probabilities: Theory and Application in the Automotive Industry

Monica G. Eboli, Fabio G. Cozman

Escola Politécnica da Universidade de São Paulo
Av. Prof. Mello Moraes 2231 - Cidade Universitária, São Paulo, SP - Brasil
E-mails: monica.eboli@gmail.com, fgcozman@usp.br

Abstract— This paper discusses models for manufacturing systems that are based on generalized stochastic Petri nets and on Markov decision processes. We employ generalized stochastic Petri nets to model manufacturing processes with high degree of uncertainty due to the human operator behavior. We extend the usual generalized stochastic Petri nets by allowing imprecision about probabilities to be explicitly represented. We then consider the translation of the resulting models into Markov decision processes with imprecise probabilities, so as to compute optimal policies. We introduce an algorithm that performs this translation.

Keywords— Scheduling, Automotive Manufacturing Lines, Markov Decision Processes, Petri Net, Imprecise Probabilities.

1 Introduction

The automotive industry is currently experiencing a unique situation. There is now an overall over capacity among the largest producers, which makes them dispute consumers fiercely – thus handing over to the consumers the central decision power. With this decision power, consumers have become more demanding, wanting vehicles to be custom made for them, and not willing to wait to get what they desire. As a consequence, manufacturing plants must address *mass customization* [Anderson (2004)]; that is, must have the ability to quickly and efficiently (on demand) produce customized products. This scenario has been the day-to-day experience of the first author, in her work at a major car manufacturer.

Since automotive manufacturing lines have to cope with such a huge variety of models and content options, their lines are *mixed model manufacturing lines*; that is, lines where more than one product can be produced [Miller and Park (1998)]. The sequence of vehicles produced is obtained by *scheduling* the necessary operations. The scheduling problem is basically a *sequential planning problem*; that is, to determine a sequence of operations that satisfies some given criteria.

Automotive manufacturing lines share an important characteristic: there are many workers doing the assembling at any given time. This fact inserts uncertainty into scheduling, as human workers do not have standard time patterns and therefore their tasks are not always finished in preplanned amounts of time. To be able to model the human behavior in the assembling job we must chose a method that can deal with uncertainty. Our focus here is on scheduling problems under uncertainty, in situations where probability distributions are quite hard to assess.

There are some methods that could be used I this planning problem, such as the Dempster-Shafer theory of evidence [Shafer (1976)] or the Markov Decision Process (MDP). We decided to use the former. It is widely used in planning problems under uncertainty, and is reviewed in Section 2.2. In our case, we resort to Markov Decision Process with Imprecise Probabilities (MDPIP) [Buffet & Aberdeen (2005), Harmanec (1999), Satia & Lave Jr. (1973), Whilte III & Eldeib (1994)]., as we are interested in situations where probabilities are not exactly specified. The basic theory on MDPIPs is briefly reviewed in Section 2.2.

The overall goal of this work is to develop tools so that scheduling under uncertainty (where not all probability distributions are precisely specified) can be tackled. Our starting point is to use MDPIPs as the core tool for schedule generation. However, MDPIPs have a clear downside in that modeling an MDPIP for a given manufacturing plant is far from trivial.

In order to overcome this difficulty, the ideal solution is to start with an intuitive and easy modeling language, and then to translate the resulting models to the language of MDPIPs for schedule generation. One of the first modeling tools that come to mind when discussing manufacturing systems modeling are Petri Nets (PNs), as these are broadly used in connection with discrete systems, in particular manufacturing lines [Balbo et al (1995), Desrochers & Al-Jaar (1995), Zhou & Venkatesh (1999), Haas (2004), Buzacott & Shanthikumar (1993)]. Petri nets are well-known and simple, thus easing the modeling process. In the context of scheduling under uncertainty, the appropriate kind of Petri net to use is the Stochastic Petri Net, reviewed in Section 2.3.

Thus, the specific goal of this paper is to present a method that receives a stochastic Petri net where not all probabilities are precisely specified, and translates it into a MDPIP appropriate for schedule generation. After a brief review of the automotive manufacturing lines that have motivated our study, and a review of necessary background in Section 2, we present the method that converts stochastic Petri Nets into MDPIP in Section 3. Section 4 concludes the paper, summarizing our contributions and indicating future work.

2 Background

2.1 Automotive Manufacturing Lines

In this section we briefly review how an automotive manufacturing line and its line stopping system work, as these systems have directly motivated our design choices concerning scheduling tools.

An automotive facility is usually organized in a job-shop structure [Miller & Park (1998)]. There are three main shops: the body shop, paint shop and general assembly shop. The body shop is the shop where the vehicle body is welded together. The paint shop is where it receives chemical treatment and it is painted. After going through the paint shop, the vehicle goes through an assembly line where all the other parts are installed.

Because the content of vehicles varies considerably, the existence of an assembly line for each car configuration is not viable; therefore all models and configurations are usually assembled in the same general assembly line. This type of assembly line where different models and configurations are assembled in the same line is called a *mixed model line*, and such a line is the focus of our work.

It is possible to model an assembly line regardless of the other shops since there is a buffer after the paint shop, and the models can be resequenced before going into the assembly shop.

In order to understand how a general assembly line works, it is important to know how line stopping system works. There are many reasons for a line to stop: power shortage, equipment break down and others, but in this study the only factor to be considered and the focus of this work will be worker task time. A common scheme for line stoppage is the *andom system*, which is a pull-cord system, in which the vehicles can't go to the next station until all of the stations had their tasks completed. Each worker has access to the pull-cord system, and is supposed to push them when something unexpected happens.

The andom system has three states, the first one is the *ideal* state, which is the one when everything goes as expected and the cord has not been pulled. Usually at that stage the worker has a standard behavior that can be predicted and described. The second one is the *warning* state and is differentiated from the previous state by a yellow line on the floor. When the worker crosses this line while doing its task, he pulls the cord in order to inform everyone else that he may not finish his task on time. When this state is reached usually the team leader comes to help the worker finish his task on time and the worker begins to work faster. When this state is reached, there is no known standard behavior since we can't predict how faster the worker will work and whether or not the team leader will be able to come help out. The third state, which is the one that must be avoided the most and is marked by a red line on the floor, is the *line stoppage* state. It happens when, despite the fact that the cord was pulled and the worker was helped, the task was not completed in cycle time and the whole line must stop.

2.2 Markov Decision Processes (MDP) and Markov Decision Processes with Imprecise Probabilities (MDPIPs)

Markov Decision Processes are often used to produce schedules (that is, plans) [Trevizan et al (2006), Trevizan et al (2007), Russel & Norvig (2004)]. We wish to ultimately have our manufacturing lines expressed through Markov Decision Processes so that scheduling can be automatically generated.

The structure of Markov Decision Processes (MDPs) consists of states and actions, and the functions that relate actions and states. For each state s and action a a probability function that determines the next state must be given. An MDP possesses the Markov property, in which the next state s' depends only in the present state s (therefore, a state is independent from previous states).

The main elements of an MDP are [Trevizan et al (2006), Trevizan et al (2007)]: a discrete and finite state space S, a non-empty set of initial states $S_0 \subseteq S$, a goal given by a set $S_G \subseteq S$, a non-empty set of actions $A(s) \subseteq A$ representing the applicable actions in each state s, a state transition function $F(s,a) \subseteq S$ mapping state s and action $a \in A(s)$ into a non-empty set of states ($|F(s,a)|>=1$), a positive cost $C(s,a)$ for taking $a \in A(s)$ in s; a probabilistic distribution $P_0(\cdot)$ in S_0 and s probabilistic distribution $P(\cdot \mid s,a)$ in $F(s,a)$ to all $s \in S$ and all $a \in A(s)$.

A state is the description of the system being studied. The description of the system must have all the information needed by the decision agent to make a decision, so it may vary depending on the goal and the method that will be used. In the MDP context, the space state is the finite set of all states that can be reached by the system.

To define a problem in the MDP form, it is necessary to have, besides the state space, the non-empty set of initial states, and the non-empty set of goal states. Actions are interventions that the decision making agent can make in the system in order to chance it's natural evolution of the states course. There must be a function $F(s,a)$ that maps which states s' can be reached from the present state s if the action a is taken. The last element of the MDP is the probabilistic distribution $P(\cdot \mid s,a)$ that measures the probability to reach the state s' given the present state s and the taken action a to all states mapped by $F(s,a)$.

There are some situations where there are not enough resources (people, equipment, time) to get a precise probability estimative, or the members of the team disagree on the probability value. In order to bypass these situations a level of imprecision needs to be added to the probability. One way to add this imprecision is to represent probability values by sets of probability distributions, often referred to as *credal sets* [Lexi (1980)].

A credal set formalizes the imprecision in probabilities, by listing all probability distributions that are deemed possible in a given problem. Credal set are usually taken to be *closed*, an assumption we adopt in this paper. Given a closed set of probability measures, the upper and lower bounds on any probability are called respectively *upper* and *lower* probabilities.

An extension of MDP is the MDPIP (Markov Decision Process with Imprecise Probabilities), in which the effect of the actions is described by a credal set *K* over the space state [Buffet & Aberdeen (2005), Harmanec (1999), Satia & Lave Jr. (1973), Whilte III & Eldeib (1994)]. This representation is taken to mean that the probability distribution $P(s'|s,a)$ for the next states of *S* is represented by a non-empty credal set $K_S(a)$ to all states $s \in S$. The formal definition is [Trevizan et al (2006)]:

1. A discrete and finite state space *S*;
2. A non-empty set of initial states $S_0 \subseteq S$;
3. A goal given by a set $S_G \subseteq S$;
4. A non-empty set of actions $A(s) \subseteq A$ representing the applicable actions in each state *s*;
5. A state transition function $F(s,a) \subseteq S$ mapping state *s* and action $a \in A(s)$ into a non-empty set of states ($|F(s,a)|>=1$);
6. A positive cost C*(s,a)* for taking $a \in A(s)$ in *s*;
7. A non-empty credal set $K_S(a)$ for all states $s \in S$ e actions $a \in A(s)$, representing the probability distribution $P(s'/s,a)$ over the next states of *S*.

2.3 Petri Nets, Timed and Stochastic Petri Nets

Petri nets offer a language to model and analyze discrete event systems. It allows one to represent graphically and mathematically those systems [Balbo et al (1995), Desrochers & Al-Jaar (1995), Zhou & Venkatesh (1999), Haas (2004), Buzacott & Shanthikumar (1993)]. In this section will review the basics of PN regarding only the elements needed for the modeling of an assembly line which is the focus of this method.

There are four graphical elements in Petri nets: places (circles), transitions (rectangles), oriented arcs and tokens (dots). Places represent elements of the system that suffer the action, usually representing a specific type of resource in a determined stage of the process. The transitions represent actions of the system, in our case the assembling processes. The directed arcs represent the relationship between places and transitions, connecting transitions to places and places to transitions. They define what resources are needed for the transition and what its outcomes are. The tokens are placed inside the places and represent the existence of a unit of the resource represented by the place. The disposition of tokens over the places is called the *marking* of the net.

PN evolves in time by the firing of the transitions which represent that the action represented by the transitions occurred. When it happens, tokens are moved from the input places to the output ones.

Transitions in Petri nets can have two states: enabled and disabled. In order to be fired, the transitions must be enabled, meaning its pre conditions and post conditions must be satisfied.

Pre-conditions refer to the input places. All the input places must have tokens, meaning the resources for the operation are available. Post conditions refer to all the output places, usually representing physical space to put the product of the operation represented by the transition. The output places must have space to receive the output token.

There are situations when an enabled transition may not fire. This happens when there is a conflict which will result in a non-deterministic behavior in the PN. A conflict is when two enabled transitions compete to use the same resource (token). In order to model these conflicts, a *random switch* is defined. *Random switch* is when a firing probability is associated to each transition. When there are many configuration of conflicts there will be defined as many sets of probabilities as the number of configurations.

Timed Petri Net (TPN) [Desrochers & Al-Jaar (1995)] was developed due to the need to model the amount of time taken for an event to occur. The inclusion of time can be done in two ways. The first one is by attaching time to transitions, creating *Timed Transition Petri Net* (TTPN), and the second one é by attaching time to places resulting in *Timed Place Petri Net* (TPPN). Stochastic Petri Nets are a special case of TTPN.

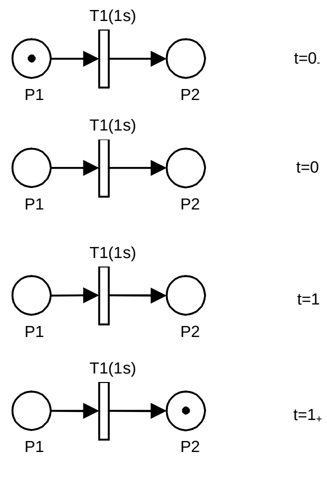


Figure 1 – Example of TTPN firing.

In a TTPN the transition is fired as soon as the inputs arrive to their places. Although the input tokens are taken right away from the input places, they will only be placed in the output places after the transition time has elapsed. Timed transitions are represented by white rectangles. An example of its dynamics can be seen in Figure 1.

Stochastic Petri Nets [Balbo et al (1995), Desrochers & Al-Jaar (1995), Zhou & Venkatesh (1999), Haas (2004), Miyagi (1996)] are special extensions of TTPNs, in which the transition time is not deterministic. This extension augments the modeling power, allowing systems that are affected by non-deterministic factors (in our case human behavior.) to be modeled. The transition firing time is usually described by a probabilistic distribution, and commonly exponential distributions are used.

The formal definition of stochastic Petri net (P, T, I, O, M_0, Λ) is given bellow [Zhou & Venkatesh (1999)]:

1. $P=\{p_1, p_2, ..., p_n\}, n>0$: finite places set;
2. $T=\{t_1, t_2, ..., t_s\}$, $s>0$: finite transitions set, given $P \cup T \neq \emptyset$ and $P \cap T = \emptyset$;
3. $I: PxT \to N$ is the input function which defines the set of arcs directed from P to T where $N=\{0,1,2,...\}$;
4. $O: PxT \to N$ is the output function which defines the set of arcs directed from T to P;
5. $M : P \to N$ *: marking in which for every place* $p \in P$ there are markings. The initial marking is denoted M_o;
6. $\Lambda: T \to R^+$ firing function in which λ_i is the firing rate of the transition t_i.

There is a specific type of SPN called Generalized Stochastic Petri Nets (GSPN) [Desrochers & Al-Jaar (1995), Zhou & Venkatesh (1999)]. GSPN have instant transitions besides the stochastic time transition. We chose this kind of PN because it allows more complex modeling.

Figure 2 shows an example of GSPN, representing the car's hang-ons assembly. Notice that the immediate transitions (T1, T2 and T5) do not represent any action; they are just parallelizing the right and left door assembly.

Each stochastic transition (T3, T4, T6 and T7) will receive a value for λ_i, which will represent the firing rate. For example, $\lambda_3=6$ means that in average the transition T3 will fire six time over one unit of time.

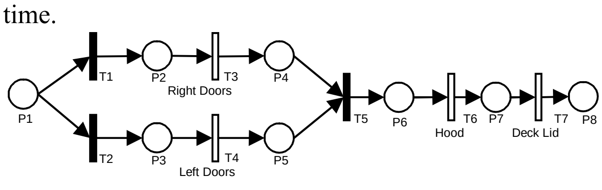


Figure 2 –Example of GSPN.

3 The Conversion Method and an Example

In this section will show how to turn a GSPN that describes an assembly line into an MDPIP. Our problem is this. Suppose one models an automotive line using a GSPN where some probabilities are partially known. In the realm of automotive lines, a person will find it difficult to determine some probabilities connected with stops in the andom system: when the operator reaches the warning state, its behavior is hardly predictable, and estimating probabilities for that state is a difficult matter. Now the challenge is to translate the GSPN with imprecise probabilities into an MDPIP.

Assume a line has already been modeled through a GSPN. Converting this GSPN into an MDPIP takes five steps:

1. Turning each transition probabilistic time distribution into a credal set;
2. Defining the states;
3. Defining of the set of actions;
4. Defining the probability distributions $P(s'/s,a)$;
5. Defining the cost of each action.

The conversion will be illustrated by an example as follows.

EXAMPLE

The GSPN in Figure 3 represents a system with two workstations W1 and W2. Suppose there are three kinds of products that have to be made (C1, C2, C3). In order to completely define the problem, besides having the Petri net, we must have the table informing the mean task time for each workstation/product combination, like shown in table 1.

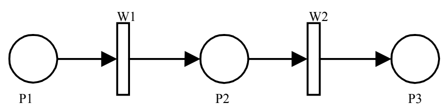


Figure 3 –GSPN of the example.

The first step consists of turning each transition probabilistic time distribution into a credal set. Notice that the credal set and the probabilistic function will represent different entities. The probabilistic function will represent the time it takes for the task to be completed, while the credal set will represent the probability that the task will be finished before the cycle time has elapsed.

Workstation	Product	Mean Task Time
W1	C1	36
W1	C2	37
W1	C3	34
W2	C1	36
W2	C2	40
W2	C3	27

Table1 –Example's firing rates.

If we were to convert the probabilistic transition time to a precise probability, it could be measured directly from the assembly time graph by looking at the probability in the y axis at the time provided by the assembly line. But this probability value does not represent what is really happening, because once the worker realizes he might not finish his work on time, he might speed up and/or get help. When this stage is reached, the worker stops acting according to the curve, his behavior becomes unknown and this uncertainty will have to me measured. We use credal sets to represent this uncertainty.

In order to model the unknown behavior we must measure how much the work can be sped up. This acceleration must be stated for each station considering how much can the worker speed up, how many other people might help and so on. This acceleration will be referred as the *maximum compression rate*.

The lower probability will be defined by the probability the work would get done if the worker kept working at the same pace, and its upper probability is defined by the probability the work would get done at the speed produced by the maximum compression rate in the warning zone.

EXAMPLE

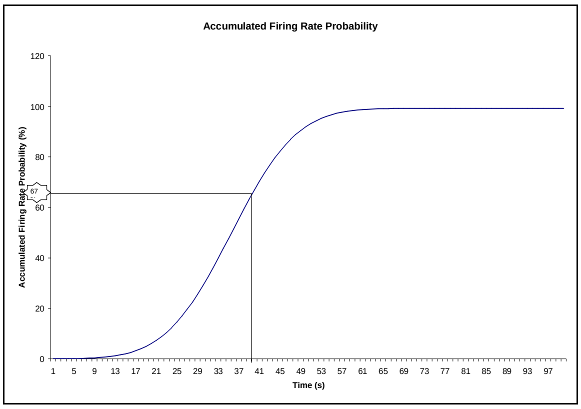
Suppose the line from the previously stated example gives 40 seconds for the task to be completed. The inferior limit of the credal set can be determined by the firing probability graph. Graph 1 represents C1 in W1, so the lower probability is approximately 67%. That means that 67%, the lower probability, is the minimum probability possible for the operator to finish its task, we can but sure that at least in 67% of the time this task will be completed.

The upper probability will be acquired from the altered probability graph, which is when it is exposed to the compression rate, beginning in the warning zone. In this problem will consider the warning zone begins at 37s. For this problem will consider that the man working at W1 works 20% faster under pressure and that he have a team leader that may come to help him. The compression rate would be (100%+20%)*2=240%. Applying this compression rate will get Graph2 and the upper probability equal to 75%. That means that in the best case scenario, there'll be a 75% chance that the task will be finished on time (upper probability).

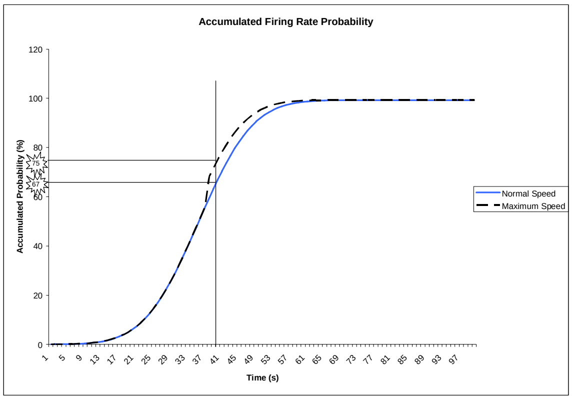
The next step is the definition of the states in the MDPIP. As it was mentioned in Section 2.3, the definition of the states depends on what we are trying to model. In this case, the main goal is to model the line stoppage, so this is one of the parameters considered while modeling. The number of states will depend on the number of vehicles to be scheduled because the number of assembly cycles is the number of vehicles plus the number of stations minus one, equation 1, and each state will be described by the combination of what happened to the line in each assembly cycle.

$$C = S + V - 1 \rightarrow \begin{cases} C = Number \quad of \quad cycles \\ S = Number \quad of \quad stations \\ V = Number \quad of \quad products \end{cases}$$

Equation 1.



Graph 1.



Graph 2.

The values given to the states, OK and NOK, represent if the line stopped or not, respectively. The combination of the cycles and their states determines the possible MDPIP states, so, there are 2^N states, where N is the number of cycles.

EXAMPLE

For our example, there are two stations and three products, so there are four cycles, as shown in Table 2. Since we have four cycles, will have 2^4 states.

Once the states have been defined, the next step is defining the set of actions that can be taken in each state. Defining the actions in this particular case is easy, since the action is choosing the next product to go into the line. The only restriction is that each product can only be chosen one time, so, in case it has been picked before it can no longer be chosen.

Cycle	Description	Values
1	1st product in M1	OK & NOK
2	1st product in M2 & 2nd in M1	OK & NOK
3	2nd product in M2 & 3rd in M1	OK & NOK
4	3rd product in M2	OK & NOK

Table 2 – States.

EXAMPLE

The actions in our case at the initial state could be C1, C2 or C3. At the next state they could be all but the one chosen at the initial, supposing we had chosen for example C2, in the second state we could only chose C1 or C3. At the third state we could chose all but the ones selected on the previous states, supposing we had chosen C3 for the second one, at the third we could only chose C1.

Having defined states and its possible actions, we need to define the probabilities distributions $P(s'/s,a)$. This is defined from the combinations of the credal sets of each station and the possibility that they will or will not stop the line. When combining those probabilities it is important to notice that it will be done as if all the stations where in series, because stations in parallel may as well stop each other since the line works as a whole.

The cost of each action will be inversely proportional to the probability of the action to stop the line. The higher the probability of stoppage the smaller will be the cost, since the method maximizes the cost; it will be minimizing the probability of line stoppage.

4 Conclusion

In this article we proposed a scheduling methodology that deals with uncertainties that are typically found in manufacturing lines, in particular in the automotive industry. In short, the methodology is to start with a model by stochastic Petri nets where some probability values may be left unspecified; to transform such a stochastic Petri net into a Markov Decision Process with Imprecise Probabilities; and finally to operate with this Markov process to produce policies if necessary. We have contributed with a method that turns a GSPN (with characteristics usually found in automotive assembling lines) into an MDPIP.

We are now starting tests with a real manufacturing line in the automotive industry, and plan to report on results shortly. Because the method was developed considering automotive assembling lines, the most pressing future work is to apply our methodology to other types of manufacturing lines.

Acknowledgements

The second author has been partially supported by CNPq. This work started in the context of Project FAPESP 04/09568-0.

References

Marsan, M. A., Balbo, G., Conte, G., Donatelli, S., Franceschinis, G. (1995). Modeling with Generalized Stochastic Petri Nets, *John Wiley and Sons*.

Miller, D. M., Park, Y. (1998). Simulation and analysis of an automotive assembly operation, SAE, pp 338-444.

Desrochers, A. A., Al-Jaar, R. Y. (1995). Applications of Petri Nets in Manufacturing Systems, *IEEE Press*

Zhou, M., Venkatesh, K. (1999). Modeling, Simulation and Control of flexible manufacturing systems – A Petri Net approach, *World Scientific*.

Hass, P. J. (2004). Stochastic Petri Nets for Modeling and Simulation, *Winter Simulation Conference*, pp 101-112.

Trevisan, F. W., Cozman, F. G., Barros, L. N. (2006). Unifying Nondeterministic and probabilistic Planning through Imprecise Markov Decision Processes, *Brazilian Symposium on Artificial Intelligence*, Ribeirão Preto, Springer, pp. 502-511.

Trevisan, F. W., Cozman, F. G., Barros, L. N. (2008). Planning under Risk and Knightian Uncertainty, *International Joint Conference on Artificial Intelligence*, pp. 2023-2028.

Shafer, G. (1976). A mathematical theory of evidence, *Princeton University Press*.

Buzacott, J. A., Shanthikumar, J. G. (1993). Stochastic models of manufacturing systems, *Prentice Hall*.

Miyagi, P. E. (1996). Controle Programavel, *Editora Edgard Blücher*.

Boutilier, C., Dean, T., Hanks, S. (1999). Decision-Theoretic Planning: Structural Assumptions and Computational Leverage, *Journal of Artificial Intelligence Research*, pg 1-94.

Hayes, R. H., Pisano, G. P. (1996). Manufacturing Strategy: at the intersection of two paradigm shifts, *Production and Operations Management*, pp 25-41.

Anderson, D. M. (2004). Build-to-Order & Mass Customization, *CIM Press*.

Russel, S., Norvig, P. (2004). Inteligência Artificial, *Editora Campus*.

Levi, I. (1980). The Enterprise of Knowledge, *MIT Press*.

Buffet, O., Aberdden, D. (2005). Robust planning with (L)RTDP, *In Proc. of the 19th IJCAI*, pp 1214–1219.

Harmanec, D. (1999). A generalization of the concept of Markov decision process to imprecise probabilities, *In ISIPTA*, pp 175–182.

Satia, J. K., Lave Jr, R. E. (1973). Markovian decision processes with uncertain transition probabilities, *Operations Research*, 21(3), pp 728–740.

White III, C. C., Eldeib, H. K .(1994). Markov decision processes with imprecise transition probabilities, *Operations Research*, 42(4), pp 739–749.