Joint Optimization of Capital Investment, Revenue Management, and Production Planning in Complex Manufacturing Systems

Stephen Baumert[1] • Shih-Fen Cheng[2] • Archis Ghate[2]
Daniel Reaume[3] • Dushyant Sharma[2] • Robert L. Smith[2]

Stephen.Baumert@afit.edu • *chengsf@umich.edu* • *archis@umich.edu*
daniel.reaume@gm.com • *dushyant@umich.edu* • *rlsmith@umich.edu*

2005-7-27 14:32

Optimal decision making in automotive manufacturing requires integrated consideration of capital investment, revenue management, production scheduling and sales planning. Traditionally these decisions are made independently at different times and organizational levels, which may lead to sub-optimal results. In this paper, we develop a joint optimization model that includes these considerations and present a simplified example involving a single dedicated-production line. We apply a novel game-theoretic algorithm called Sampled Fictitious Play (SFP) to solve this example. Although this instance can be solved via traditional optimization approaches such as dynamic programming, we demonstrate that SFP is five orders of magnitude quicker than dynamic programming at finding near-optimal solutions.

1. Introduction

Automotive original equipment manufacturers (OEMs) are faced with the challenge of minimizing costs in the face of intense global competition while still maintaining the operational flexibility to capitalize on market opportunities. Management must carefully weigh these competing goals when making decisions on capital investments, pricing, and operational policies. In this paper, we focus on addressing the problem of optimally investing capital in new production facilities and equipment. Thus, the first key decision to be made is: 1) What equipment to install? This involves determining the number, capacity, and flexibility of production lines. These decisions are governed by constraints on available capital and must factor in forecasts of future demand patterns. Although demand for a vehicle model depends on dynamic exogenous factors such as economic conditions, fuel costs, and consumer trends, it can be partially controlled by adjusting the selling price. This introduces the second key decision: 2) What should be the selling price of each vehicle model in each time period? These prices, combined with the dynamic exogenous economic factors, yield demands for each vehicle model. These demands in turn drive production requirements.

[1]Department of Operational Sciences, Air Force Institute of Technology
[2]Department of Industrial and Operations Engineering, The University of Michigan
[3]General Motors Research and Development Center

Thus the third key decision is: 3) What are the production targets for each time period? Note that even if production meets or exceeds demand, it may not always be optimal to immediately fulfill all demands. For example, it may be preferable to stockpile convertibles in anticipation of higher selling prices during warm-weather months. Thus the fourth key decision is 4) How many vehicles of each model should be sold in each time period? Note: Although an OEM generally does not directly hold inventory nor control sales to customers, these may be conceptually viewed as being under OEM control via dealer and customer incentives.

The optimization problem described above is hierarchical in nature, involving decisions at strategic, tactical, and operational levels by different decision-makers. Higher-level decisions constrain and set the context for lower-level decisions, while the potential results of lower-level decisions in turn impact higher-level decisions. Consequently, an integrated approach to decision making is vital. For example, in response to a spike in demand for a model, it is generally neither optimal to drastically increase prices nor to drastically increase production. Instead, a hybrid response is generally preferable, where prices are moderately increased while simultaneously leveraging flexible manufacturing assets to moderately increase production. The benefits of revenue management and of investments in flexible production systems are not fully realized unless both are exercised together.

Due to the difference in levels in the decision hierarchy, each decision in the integrated optimization problem may have its own horizon, ranging from very long for strategic decisions such as capital investment to quite short for operational decisions such as weekly production levels. The joint optimization problem is extremely complicated, and it is not clear how to make an integrated set of optimal decisions.

To understand the problem abstractly, we will first establish a mathematical model that approximates the joint optimization problem. It should be noted that when formulating this problem, a high level of fidelity is not our top priority as this would require consideration of an inordinate number of uncertainties as well as numerous exogenous, qualitative, and strategic factors. Even if such an optimization problem were tractable, the required data - much of it stochastic in nature - would be exceedingly difficult to collect. Instead, we propose a simpler model for which data can actually be obtained, with the goal of generating strategic and operational insights that may be effectively used by decision-makers to improve performance.

In order to understand the relationships between key system parameters and such insights, the problem must be repeatedly solved with different sets of input data. Consequently, it is essential to be able to quickly solve instances of the integrated decision problem. Unfortunately, as we will show in later sections, even a simplified model is quite difficult to solve exactly. Moreover as more and more features are added to the model, it will eventually become impossible to describe the model analytically and instead a simulator will have to be used. We must therefore develop an algorithm capable of efficiently optimizing the model even if it is expressed via a black-box simulator.

There has been a recent boom in the revenue management-inventory control literature. Research in the past has considered different forms of revenue management. For a recent review on this topic, please refer to Swaminathan and Tayur (2003). Various researchers have considered adaptive pricing and stocking problems (Alpern and Snower (1988), Subrahmanyan and Shoemaker (1996), and Burnetas and Smith (2000)). Petruzzi and Dada (2002) considered deterministic demand parameterized by one parameter. Chen and Simchi-

Levi (2004a,b) considered coordinating pricing and inventory decisions in the presence of stochastic demand over a finite as well as an infinite horizon. Federgruen and Heching (1999), and Feng and Chen (2003, 2004) considered similar problems. However, to the best of our knowledge there is no literature that focuses on joint optimization of investment, pricing, production and sales. In this paper we propose to use the game-theoretic paradigm of sampled fictitious play to partly address this issue. To precisely capture the effectiveness of the algorithm in reality, we will include major features of the manufacturing system, but only to the extent that the problem can still be solved to optimality, so that we can compare the result of the algorithm to the global optimum.

The paper is organized as follows. In section 2, we formulate the joint optimization problem as a Markov decision process. In section 3, we present the motivation for applying a game-theoretic approach to the problem and outline an intuitive description of our methodology. In section 4, we formally introduce necessary background for our game-theoretic approach. In section 5, we formally state how the game-theoretic approach can be applied to solve the original Markov decision process. In section 6, we discuss the results of numerical experiments and how we can use our approach to develop managerial guidelines. Finally, section 7 presents conclusions and discusses future work.

2. The Joint Optimization Problem

As described in the introduction, the joint optimization problem is composed of four important decisions. These four decision modules are formally introduced in 2.1. The modelling assumptions and the model are described in 2.2. The problem data used is explained in 2.3.

2.1 Decision Modules

Following the description in Section 1, four important decision modules are defined as follows. Note that for simplicity, we assume that the planning horizon is discretized into periods $0, 1, 2, \ldots$ with equal length.

- **Capital Investment (CI)**: In general, the CI module will decide the type (dedicated, reconfigurable, or flexible) and the capacity of the production line. However, to simplify the analysis, we assume that we can only build a dedicated production line that produces only one type of vehicle. Thus, the only decision for CI is the production line capacity. Unlike all other modules, where decisions are made at each epoch $n = 1, 2, \ldots, N$, the CI decision is only made at the beginning of the planning horizon, i.e., epoch 0.

- **Revenue Management (RM)**: At the n^{th} epoch ($n = 1, 2, \ldots, N$), the unit price of the vehicle will be decided by the RM module. Note that in the general case where we have multiple vehicle types, a price should be specified for each type. However, since we limit ourself to a dedicated production line that produces only one type of vehicle, our decision for the RM module is just a scalar (instead of a price vector). The pricing decision will then generate the demand for the vehicles through a price-demand function (may be deterministic or stochastic).

- **Production Scheduling (PS)**: at the n^{th} epoch ($n = 1, 2, \ldots, N$), the production goal for the current period is decided by the PS module. Note that the production goal cannot exceed the production line capacity decided by the CI module.

- **Sales Planning (SP)**: at the n^{th} epoch ($n = 1, 2, \ldots, N$), the projected sales goal is decided by the SP module. Notice that our sales goal may exceed the real demand in the market, in this case, our real sales will be up to the demand.

2.2 The Markov Decision Process

When formulating the model, we would like to include most important features of the problem, while at the same time avoid unnecessary complications. In our investigation, we choose to focus on the stochasticity of the reliability of the production line and the demand function. These two features alone will make the problem non-trivial and difficult numerically.

2.2.1 Assumptions

- The planning horizon is discretized into $N+1$ periods, $0, 1, \ldots, N$. The capital investment decision is made at period 0. All other decisions, including revenue management, production scheduling and sales, are made at the beginning of all subsequent periods, $n = 1, 2, \ldots, N$.

- We assume that the capacity of the production line can only be chosen from a fixed finite set, and a fixed building cost is associated with each capacity choice. This cost can either be paid by a lump sum deducted in period 0, or it can be paid in installments. In the latter case, we assume that same amount of installment is charged in each period n ($n = 1, 2, \ldots, N$). In our model, we assume that the building cost is always paid in installments. The advantage of paying building cost by installments is that we can solve the version of the problem with horizon shorter than the life of the production line. We will emphasize this again when we describe the notation used in our model.

- All the problem data and decision variables related to the volume of the production are for one shift (8 hours) only. In practice, multiple shifts (usually three, but in the case where additional capacity is needed, a fourth shift can be arranged using weekend time) can be arranged at the production facility, therefore the actual production output may be several times its capacity. However, multiple shifts will only complicate the computation of the cost and production output without providing much insights into the problem. Therefore, we assume only one shift is used in our model.

- The production line is assumed to be unreliable. Reliability of the production line can be modelled at various operation levels, from micro level to macro level. At micro level, the reliability is modelled at station-level, and the actual production output in each period is collectively decided by the statuses of all stations. Since the interaction among stations can be extremely complicated, in practice we have to use Monte Carlo simulation in order to obtain production output. At macro level, we consider the production line as a whole and assume that its reliability (and thus production output) is governed by a known probability distribution. Since we would like to have

an analytical expression for the operation of the production line, we will model the reliability at macro level.

- Since the production line is unreliable and breakdown actually happens, we will need to staff maintenance crews and decide proper maintaining schedule. However, since we choose to take a macro view regarding the reliability of the production line, the detail on the maintenance will not be considered in our model.

- The demand function is assumed to be stochastic, reflecting the fact that the demand for the product as a function of price cannot be precisely predicted when the pricing decision is made. To simplify the formulation, we assume that we have a finite set of possible demand functions, and for each period, one function will be randomly selected from this set. This set is assumed to be known to the planner.

- No backlog is allowed. If the current inventory plus production is not enough to satisfy the demand in some period, the demand is lost.

- The manufacturing cost depends both on the line capacity and the period when the production occurs.

- The holding cost of carrying i vehicles in the inventory in period n is a fixed fraction of the cost for manufacturing i units of products in period n.

2.2.2 Notation

- $\mathbf{N} = \{0, 1, \ldots, N\}$: set of time periods.

- $\mathbf{M} = \{m_1, m_2, \ldots, m_{|\mathbf{M}|}\}$: set of feasible production line capacities.

- $\mathbf{P} = \{p_1, p_2, \ldots, p_{|\mathbf{P}|}\}$: set of feasible pricing decisions.

- γ: the discount rate.

- $C(m), m \in \mathbf{M}$: the installment to be paid in each period for the initial investment of building a production line with capacity m. $C(m)$ is computed so that if production line is designed to operate for L periods, the discounted sum of L payments equals the lump sum payment of the building cost. i.e., $\sum_{n=1}^{L} \gamma^{n-1} C(m) =$ cost for building line with capacity m.

- $c(n, x, \hat{x}, m), n \in \mathbf{N}, m \in \mathbf{M}, \hat{x} \leq x \leq m$: the cost of producing $\hat{x}$ units of products in period n, with original production goal x and the capacity of m. The portion of production that is planned but cannot be realized due to machine breakdown will not incur material and component cost. However, since the staffing of workers is arranged a priori, the labor cost will still be charged during the breakdown. In order to account for this, we model the production cost as the sum of two costs: the labor cost, $c_l(n, x, m)$, and the material and component cost, $c_m(n, \hat{x}, m)$. $c(n, x, \hat{x}, m) = c_l(n, x, m) + c_m(n, \hat{x}, m)$.

- ρ_n: As stated in our assumption, the reliability of the production line is modelled in a macro manner. Here we use ρ_n to represent the fraction of available production capacity in period n. By definition, $\rho_n \in [0, 1]$. We assume that in each period, the production line can be operated at one of service levels listed in set $\mathbf{L}$, where $\mathbf{L} = \{l_1, l_2, \ldots, l_{|\mathbf{L}|}\}$. We further assume that the probability that the production line operates under certain service level l_k is the same for all period, and will be denoted as P_{l_k}.

- $\mathbf{D} = \{D_1(\cdot), \ldots, D_{|\mathbf{D}|}(\cdot)\}$: the set of possible demand functions. In our model, we assume that each element in $\mathbf{D}$ is chosen with equal probability. Furthermore, we assume that the general form of the demand function is exponential, with constant elasticity (we are using similar modelling assumptions as in Hagerty et al. (1988)). To simplify the pricing part of the problem, we assume that the only factor that influences the demand is our own pricing decision (thus excluding competitor's pricing and exogenous variables from the demand function). $D_i(p)$ can be formally represented as follows:

$$\begin{aligned} D_i(p) &= e^{a_i} p^{b_i} \\ \text{hence, } \log D_i(p) &= a_i + b_i \log p \\ \text{where, } (a_i, b_i) &\in \{(a_1, b_1), \ldots, (a_{|\mathbf{D}|}, b_{|\mathbf{D}|})\}. \end{aligned} \tag{1}$$

- $d_n(\cdot) \in \mathbf{D}$: the realized demand function in period n.

- $h(n, i)$: the cost of holding i units of inventory from period n to period $n+1$. According to the earlier assumption, $h(n, i) = \lambda \cdot c(n, i, i, m)$, where $\lambda \in [0, 1]$ is a pre-specified constant. As the notation suggests, λ does not depend on the period n.

2.2.3 The Model

The problem is a natural sequential decision process, with decisions being made sequentially from period 0 to period N. In period 0, we make capital investment decision m, where $m \in \mathbf{M}$. In period $n \geq 1$, the decisions for RM, PS and SP are made at each epoch. Just as in traditional production control problem, the information required to make optimal decisions for PS, RM and SP is current period and the level of inventory beginning that period. In addition, since decision for CI sets the upper bound on the production, its decision, m should also be required in each period. This enables us to define the state space for period $n \geq 1$ as the triple, (m, n, i), where m is the capacity of the production line, n is the current period, and i is the inventory entering period n.

After defining the states for the problem, we will define the feasible decisions at each state, the state transition function, the reward function, and finally the functional equation. These important elements of the model are illustrated in Figure 1. In Figure 1, $\mathbf{F}(m, n, i)$ is the set of feasible decisions and $S_{m,n,i}$ is the decision being made, both at state (m, n, i). The demand function, d_n, and the available fraction of the capacity, ρ_n, are both random variables, and will be realized after we make the decision. This reflects the fact that available capacity of the production line and the demand function cannot be known for certain when the decision is planned. The state transition is completed as these two random variables

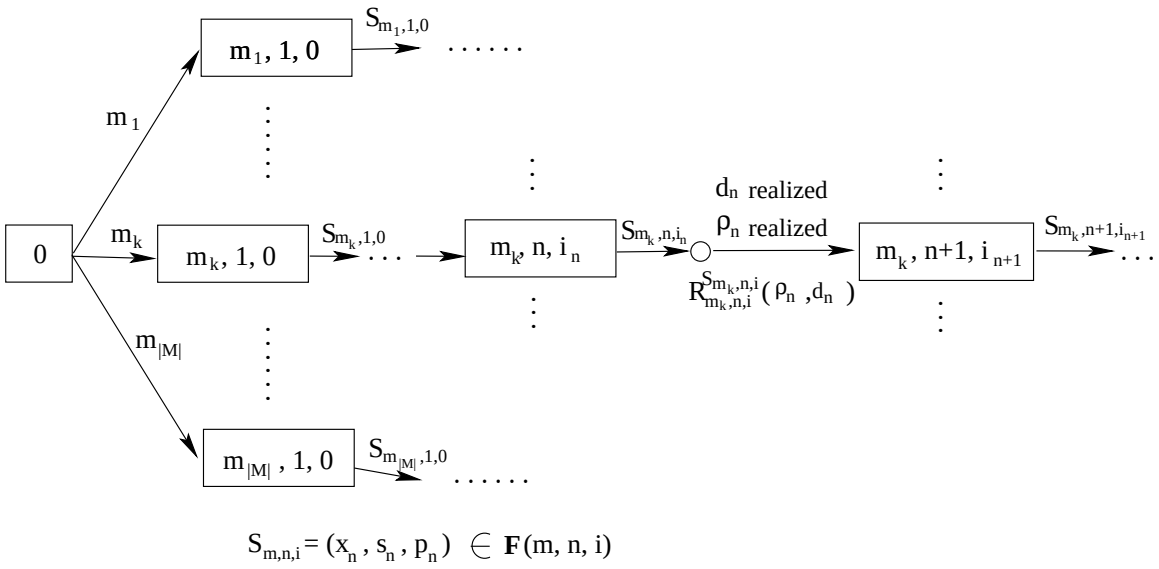

Figure 1: The Markov decision process describing the joint optimization problem.

are realized. Also, as ρ_n and d_n are realized, the reward, $R^{S_{m,n,i}}_{m,n,i}(\rho_n, d_n(\cdot))$, is generated and accumulated. The model is formally stated as follows:

- At any given state (m, n, i), the set of feasible decisions, $\mathbf{F}(m, n, i)$, is defined by following constraints:

$$\begin{aligned} x_n &\leq m \\ s_n &\leq \min\{i + x_n, \max_{D_j \in \mathbf{D}}\{D_j(p_n)\}\} \\ x_n, s_n &\geq 0, \text{ integer} \\ p_n &\in \mathbf{P}, \end{aligned} \tag{2}$$

  where x_n is the decision on production and s_n is the decision on sales.

- The state transition at state (m, n, i), with action (x_n, s_n, p_n), after the realization of ρ_n and $d_n(\cdot)$, is defined by:

$$\begin{aligned} \hat{x}_n &= \min\{x_n, \rho_n m\} \\ \hat{s}_n &= \min\{s_n, i + \hat{x}_n, d_n(p_n)\} \\ i_{n+1} &= i_n + \hat{x}_n - \hat{s}_n. \end{aligned} \tag{3}$$

  As mentioned in section 2.2, we know that $\rho_n \in \mathbf{L}$, $P(\rho_n = l_k) = P_{l_k}$, and each element within $\mathbf{D}$ is chosen with equal probability. With these definitions and equation 3, we can compute the transition probability, $P^a_{A_1,A_2}$ (the probability of transiting from state A_1 to A_2, if action a is taken), accordingly.

- The reward function at state (m, n, i), with action $S_{m,n,i} = (x_n, s_n, p_n)$, after realizations of ρ_n and $d_n(\cdot)$, is defined by:

$$R^{S_{m,n,i}}_{m,n,i}(\rho_n, d_n(\cdot)) = \hat{s}_n \cdot p_n - c(n, x_n, \hat{x}_n, m) - h(n, i), \tag{4}$$

where $\hat{x}_n$ and $\hat{s}_n$ are as defined in equation 3.

- Functional equation $f(\cdot)$:

  For $n = 0$,

$$f(0) = \max_{m \in \mathbf{M}} \{f(m, 1, 0) - N \cdot C(m)\}. \tag{5}$$

  For $n \geq 1$,

$$\begin{aligned} f(m, n, i_n) &= \max_{a \in \mathbf{F}(m,n,i_n)} \mathbf{E}_{\rho_n, d_n(\cdot)} \left\{ R^a_{m,n,i_n}(\rho_n, d_n(\cdot)) + \gamma f(m, n+1, i_{n+1}) \right\} \qquad (6) \\ &= \max_{a \in \mathbf{F}(m,n,i_n)} \sum_{\rho_n \in \mathbf{L}} \sum_{d_n(\cdot) \in \mathbf{D}} \frac{P_{\rho_n}}{|\mathbf{D}|} \left\{ R^a_{m,n,i_n}(\rho_n, d_n(\cdot)) + \gamma f(m, n+1, i_{n+1}) \right\}, \end{aligned}$$

  where i_{n+1} can be computed by using equation 3.

2.3 Problem Data

Three important sets of problem data are necessary in driving the model: building costs of production lines with different capacities, the set of demand functions, and the manufacturing cost as a function of capacities. The details on the problem data are described later in section 6, with illustrations.

3. Motivation for a Game-Theoretic Approach

As mentioned previously, one of the major difficulties in solving the global problem is the explosion of combinations of decisions at each given state. To counter this issue, we can decompose the problem along the dimension of the decision space. This kind of decomposition scheme is actually quite intuitive from the model, since the decision space is composed of decision modules described in section 2.1.

To be more specific, instead of trying to collectively come up with the optimal decision at each state (implying that we must enumerate all possible combinations of decisions), let each decision module independently make its decision by assuming that other decision modules will follow certain pattern.

However, to effectively approximate the original problem, it is important that each decision module's expectation on other decision modules' behaviors should be reasonably modelled. This type of decentralized decision making problem is widely researched in the field of game theory. Thus we will try to model the interaction of these decision modules through a game-theoretic approach. The notion of a solution to a game is that of a *Nash equilibrium*, which can be viewed as a coordinate-wise local optimum for our problem. Intuitively, a joint decision is a Nash equilibrium if no individual decision maker can improve its objective value by unilaterally deviating from the original joint decision.

It is well-known that finding Nash equilibria is a hard problem (McKelvey and McLennan, 1996). One of the earliest algorithms used to find Nash equilibria is an iterative process called *fictitious play* (Brown, 1951; Robinson, 1951). The primary pitfall of fictitious play (FP) is that in general it does not converge to an equilibrium. However, Monderer and Shapley

(1996) showed that for a special class of games, namely games of identical interest (i.e., all decision makers share same objective function), FP will converge to equilibrium. Since almost all unconstrained discrete optimization problems can be represented as games of identical interest, this result has recently inspired researchers in optimization to introduce FP as an optimization tool (Lambert et al., 2004).

In the next section, we will formally introduce minimal necessary game-theoretic terminology required to understand the rest of the paper. In addition, we will also describe the sampled fictitious play (SFP) algorithm.

4. Game Theory and the Fictitious Play Algorithm

In this section, we formally define a game and the solution concept of a Nash equilibrium, and discuss how one can use fictitious play (FP) to find a Nash equilibrium of a game.

4.1 Game Theory Fundamentals

Game theory studies how independent decision makers would act under the assumption that individual's payoff will be determined by the joint actions. We now define the components of a game.

- **Players:** Each independent decision maker in the game is defined as a player. Every player has a finite set of decisions called **strategies** that it can choose from. A **(mixed) strategy** is a probability distribution over the set of the player's strategies. A **joint strategy** is a specification of (mixed) strategies for all players.
- **Payoff function:** For every player, its associated payoff function is defined as a mapping from joint strategies to the corresponding payoffs this player will get were these joint strategies played (or expected payoffs, if mixed strategies are played). In general, players may have different payoff functions. However, in this paper, all players will be assumed to have identical payoff functions.
- **Best reply function:** Given an arbitrary joint strategy, a player's best reply function will return the strategy that gives this player its highest payoff value, assuming that all other players use the strategies specified in this joint strategy. The concept of a best response is very important in Fictitious Play and will be used multiple times later.
- **Nash equilibrium:** A joint strategy is a Nash equilibrium if no individual player can improve its payoff by unilaterally deviating from its play in the original joint strategy. More precisely, a joint decision is a Nash equilibrium if for every player, its best reply against this joint strategy is its current decision. In other words, Nash equilibrium is a fixed point of the best reply function.

The first important existence theorem, proposed by Nash (1950), stated that every finite game in strategic form[1] has a mixed strategy equilibrium.

[1] A game is said to be in strategic form if it has a finite set of players, each player has a nonempty strategy set, and each player's payoff functions are properly defined for all joint strategies. A strategic game is finite if each player has a finite strategy set.

For complete treatment of these introductory terms and concepts, please refer to Fudenberg and Tirole (1991).

4.2 FP and SFP Algorithms

Computing Nash equilibria can be a difficult task. McKelvey and McLennan's work on GAMBIT is an excellent source for various computational methods for finding Nash equilibria. In this paper, we will use a simple-to-implement iterative algorithm called Sampled Fictitious Play Lambert et al. (2004), which is a variation of the well known Fictitious Play method.

The convergence results for the FP algorithm and its variants are stated in Monderer and Shapley (1996) and Lambert et al. (2004). Since in this paper we are mainly interested in solving the joint optimization problem, most technical details are omitted here. We refer interested readers to Lambert et al. (2004) for a complete treatment.

The intuition behind FP lies in the theory of learning in games. In an FP process, every player assumes that other players are playing unknown stationary mixed strategies, and tries to learn them iteratively. The estimates of the unknown stationary mixed strategies are represented as *belief distributions*, or *beliefs*, and are shared among all players. The belief distribution for player i is a mixed strategy calculated by finding the relative frequency of all strategies from the history of its past plays. During each iteration, each player will try to find a *best reply* against the belief distribution of other players (i.e., his belief of how they will play). These best replies are then included in the history of past plays and the beliefs are updated accordingly. To start the FP process, an arbitrary joint strategy is used.

The FP algorithm does not converge to equilibrium in general. However, for games with identical interest, as in our case, the sequence of beliefs generated by the FP algorithm are guaranteed to converge to equilibrium (Monderer and Shapley, 1996).

FP is computationally expensive to be implemented in practice. Sampled Fictitious Play (SFP) is a variant of FP that is computationally much more efficient. SFP is very similar to FP except the best reply evaluation in each iteration is done against samples randomly drawn from the belief distribution instead of the belief distribution itself. The SFP algorithm was first implemented and applied to a dynamic traffic routing assignment problem by Garcia et al. (2000). When compared to previously established methods, SFP algorithm was able to obtain a solution of same quality significantly faster. Lambert and Wang (2003) further demonstrated the superiority of the SFP algorithm over simulated annealing and random search in a communication protocol design problem. Lambert et al. (2004) extended the convergence result of Moderer and Shapley to SFP. In particular, they showed that SFP inherits convergence properties of FP if the sample size grows with iteration number at a particular rate. In practice, samples of size one are used at each iteration although no convergence results are available for that case at this time.

The SFP algorithm, with sample size one, is described as follows:

1. **Initialization:** An initial joint strategy profile is chosen arbitrarily. It is then stored in the history.

2. **Sample:** A strategy is independently drawn from the history of each player (i.e., for each player, each past play in the history is selected with equal probability).

3. **Best Reply:** For every player, the best reply is computed by assuming that all other players play the strategies drawn in step 2).

4. **Update:** The best replies obtained in step 3) are stored in the history.

5. **Stop?** Check if the stopping criterion is met, if not go to step 2), otherwise stop.

The pseudo-code for the SFP algorithm is as follows. In this pseudo-code, $\mathbf{D}$ and $\mathbf{B}$ are vectors, and $\mathbf{H}$ is a history matrix, where $\mathbf{H}(i, j)$ represents player j's best reply in i^{th} iteration. Notation $\mathbf{H}(i, :)$ represents i^{th} row of matrix $\mathbf{H}$ (i.e., joint best replies in i^{th} iteration).

Algorithm 1 Sampled fictitious play algorithm

SFP()

```
1: H(0,:) ← InitialSolution()
2: while StopCriterion() is false do
3:    D ← Sample(H, i)
4:    B ← BestReply(D)
5:    H(i,:) ← B^T
6: end while
```

D = Sample(H, i)

```
1: for p = 1 to P do
2:    u ← DiscreteUniform(0, i − 1)
3:    D(p) ← H(u, p)
4: end for
5: return D
```

Above pseudo-code listing implements the SFP algorithm in a straightforward way. Line 1 generates an initial solution by calling function InitialSolution and puts it into the history matrix $\mathbf{H}$. Note that except for $r = 0$, each row r of matrix $\mathbf{H}$ stores the best reply in iteration r. Line 3 performs uniform sampling from each player's history independently. Line 4 computes the best reply $\mathbf{B}$ by taking sampled decision $\mathbf{D}$. Line 5 appends $\mathbf{B}$ at the end of the history matrix. Above three lines are then repeated until StopCriterion returns **true**. It is important to note that BestReply is just the collection of P one-dimensional optimization problems, and the inputs required by these problems are just the sampled decision $\mathbf{D}$, therefore they can be solved in parallel. Also, although the original purpose of FP algorithm was to find a Nash equilibrium of a game, we use SFP algorithm for finding an optimal solution. Therefore, we will keep track of the pure strategy with best performance we have seen so far as the solution to deliver.

5. Game-Theoretic Model for the Joint Optimization Problem

With the same notations as defined in the previous sections, we can formulate the problem as a game:

- **Players:** each decision module (CI, RM, PS, and SP) is defined as a player. We will use P_{CI}, P_{RM}, P_{PS}, and P_{SP} to represent the player for each of the decision module.
- **Strategy Space:** in the game-theoretic model we proposed, each player must have some probabilistic beliefs about all other players' behaviors. Each realization from such belief on other players' behaviors will create a reduced MDP, where the decision variables are only this player's decision. Therefore the strategy space for each player should be the policy space of this player. However, as defined in equation 2, there are interdependencies between players' decision. To be more specific, from equation 2, we can see that P_{PS}'s decision relies on P_{CI}'s decision. Similarly, P_{SP}'s decision relies on both P_{PS}'s and P_{RM}'s decisions. Unfortunately, one important requirement for modelling our joint optimization problem as a game is the assumption that all players select their actions simultaneously. Therefore, no matter what kind of beliefs each player has for all other players, it is possible that combined decisions from all players may be infeasible. We will deal with this issue later.
- **Payoff function:** the assignment of payoff values require feasible joint decisions. Therefore, in the case where joint decision is infeasible, the payoff function is not defined.

The feasibility issue mentioned above originates from our attempt to model a constrained problem with an unconstrained model. Thus, we must transform the constrained optimization problem into an unconstrained problem before putting it in the game-theoretic framework. In the following section we will describe how to design a proper transformation in order to turn our constrained optimization problem into an unconstrained problem that can be modelled by the game-theoretic framework.

5.1 Ensuring Feasibility

While implementing a game-theoretic method such as fictitious play or sampled fictitious play to solve a complex system optimization problem, one has to ensure feasibility of joint actions by the players. In the case of sampled fictitious play, the concern for feasibility arises from the fact that the algorithm assumes that each player has a finite strategy set that does not depend on actions of other players and therefore, a joint strategy corresponds to a point in a fixed hyper-rectangular subset of the integer lattice. Moreover, players sample their individual actions independently, without knowing what other players have sampled from their respective strategy spaces. This may cause a serious problem if feasibility of a particular action by a player depends on what other players have played. In terms of an optimization problem, the above observations mean that the only allowable constraints are box-constraints, i.e. fixed lower and upper bounds on the variables. However, this is rarely the case in most constrained optimization problems including the production system optimization problem at hand. In particular, the PS player may never decide to produce more than the capacity chosen by the CI player. Similarly, the SP player may never sell more than the minimum of the demand decided by the RM player and the total inventory on hand including the most recent production decided by the PS player. To handle the

feasibility issues outlined here, we propose a variable transformation called the *proportional transformation*.

The main idea behind this transformation is quite intuitive. Instead of having the PS and the SP players choose the actual production level and the actual sales in a period, we let them choose the fraction of maximum allowable production and the maximum possible sales. Mathematically, instead of letting $x(m,n,i)$ be the decision variable for the PS player, we let $\alpha(m,n,i)$ be the decision variable, where $\alpha(m,n,i)$ is the fraction of capacity that is utilized for production. Similarly, we let $\beta(m,n,i)$ be the decision variable for the SP player, where $\beta(m,n,i)$ is the fraction of the minimum of the inventory at hand after production and the realized demand. It is clear from the definition that these two decision variables lie in the interval $[0,1]$ and no matter what the actual capacity, or inventory or the period is. This helps us transform the optimization problem at hand with complicated side constraints into a problem with box-constraints. Such a problem can be handled by sampled fictitious play after discretizing the interval $[0,1]$.

One main benefit of the proportional transformation is that it decomposes the decision spaces of various players. In particular, players can choose their own policies without regard to choices made by other players. This can be illustrated as follows. For simplicity, assume that the optimization problem is deterministic, i.e. the machines are reliable and the demand is deterministically set by the price. The joint optimization problem at hand before applying the proportional transformation can then be represented schematically as a decision tree shown in Figure 2. The hollow circle represents the decision epoch for CI, each solid circle represents a decision epoch for RM in each period, a cross represents a decision epoch for PS in each period and a square represents a decision epoch for SP in each period. In this tree, the decisions feasible at a node may depend on actions of other nodes preceding it.

However, after applying the proportional transformation, this is no longer true and therefore, the optimization problem can be schematically represented by aggregating decision nodes corresponding to the same player across different periods. This is shown in Figure 3 where the initial and terminal nodes are now denoted by filled squares. It is important to note however that even though this representation resolves feasibility issues, the reward of a specific policy employed by a particular player still depends on policies of other players. However, this is relevant only while computing the best replies and not while sampling policies from the empirical distributions.

From the above description on the proportional transformation, we can derive the equation for computing the production for the PS player and the sales for the SP player at each state (m,n,i):

$$\begin{aligned} \tilde{x}(m,n,i) &= \alpha(m,n,i)\cdot m \\ \tilde{s}(m,n,i) &= \beta(m,n,i)\cdot \min\left\{ i+\tilde{x}(m,n,i), \max_{D_j\in\mathbf{D}}\{D_j(p(m,n,i))\}\right\}, \end{aligned} \tag{7}$$

where $\tilde{x}(m,n,i)$ and $\tilde{s}(m,n,i)$ are the corresponding production and sales from decision $\alpha(m,n,i)$ and $\beta(m,n,i)$. Note that after the realization of available capacity, ρ_n, and the demand function, $d_n(\cdot)$, $\tilde{x}(m,n,i)$ and $\tilde{s}(m,n,i)$ may be higher than the maximal allowed values, and must be truncated. The reason why we don't include both ρ_n and $d_n(\cdot)$ when we perform the proportional transformation is the same as in subsection 2.2.3, i.e., available

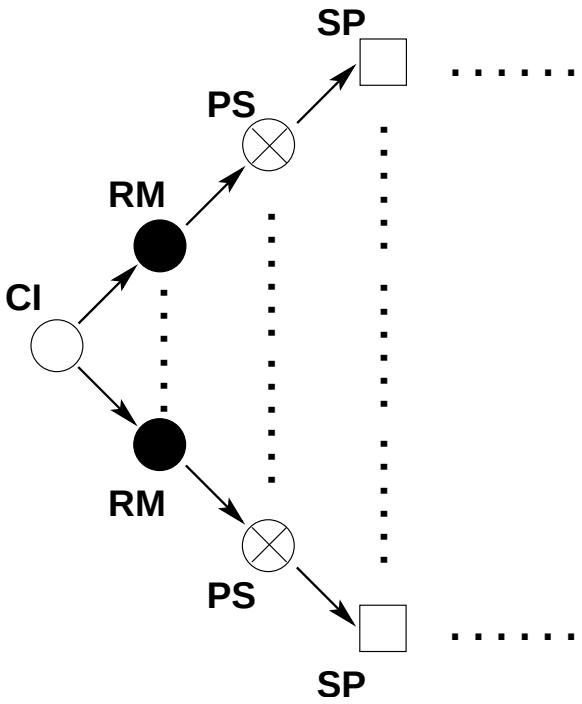

Figure 2: Decision tree illustrating the decision sequence by the players.

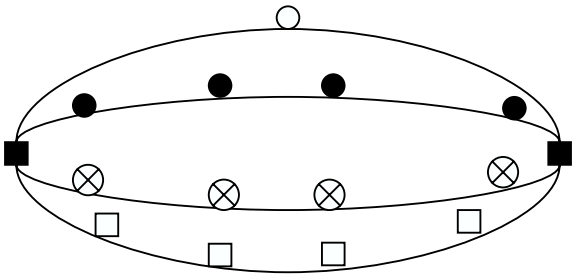

Figure 3: The decision tree after aggregation.

capacity of the production line and the demand function cannot be known for certain when the decision is planned.

Note however that since player CI makes decision on m, during each iteration, when a decision is sampled from player CI's history, it is a specific capacity. This suggests that if we only care about player PS, RM, and SP's best replies against this specific capacity, state variable m is really not necessary and can be removed from the state space. The benefit of doing so is that the computational efforts of computing best replies are reduced by a factor of M (except for player CI). However, if m is removed from the state space, player PS, RM, and SP's best replies are not dependent on m and in the subsequent iterations, it is very likely that the sampled decisions are computed under different capacity than the current sampled capacity from player CI. This constitutes a tradeoff between execution speed and the quality of best replies. While performing the numerical experiments, we tried both approaches. However, in this paper, we consider only the case where m is removed from the state space. The proportional transformation, after m is removed from the state space, can be written as:

$$\begin{aligned} \tilde{x}(n,i) &= \alpha(n,i) \cdot m \\ \tilde{s}(n,i) &= \beta(n,i) \cdot \min\left\{ i + \tilde{x}(n,i), \max_{D_j \in \mathbf{D}} \{D_j(p(n,i))\} \right\}. \end{aligned} \tag{8}$$

The best reply problem for each module is presented in the following sections. In these sections, we only include the version without capacity m in the state space.

Note that in each best reply subproblem, the best reply is computed with the assumption that other players fix their decisions. To compute the value when other players fix their decisions, we need a value function that is similar to the functional equation defined in equation 6:

For $n \geq 1$,

$$g(m,n,i_n;a(n,i_n)) = \mathbf{E}_{\rho_n,d_n(\cdot)} \left\{ \begin{array}{l} R^{a(n,i_n)}_{m,n,i_n}(\rho_n, d_n(\cdot)) + \\ \gamma g(m,n+1,i_{n+1};a(n+1,i_{n+1})) \end{array} \right\}, \tag{9}$$

where $a(n,i_n) = (\tilde{x}(n,i_n), \tilde{s}(n,i_n), p(n,i_n))$, and $i_{n+1} = i_n + \tilde{x}(n,i) - \tilde{s}(n,i)$. Note that $\tilde{x}(\cdot)$ and $\tilde{s}(\cdot)$ are as defined in equation 8.

5.2 Best Reply Problem for the Capital Investment Module

From Figure 1, we can see that CI only makes a decision at the beginning of the horizon. In the case where capacity is not part of the state variable, for each $m_k \in \mathbf{M}$ we can compute $g(m_k,1,0)$ with other players' policies fixed at $(\alpha(n,i), \beta(n,i), p(n,i))$ at each state (n,i). We can compute the transformed point $(\tilde{x}(n,i), \tilde{s}(n,i), p(n,i))$ at each state by using equation 8. In this case, CI's problem is just a one-dimensional maximum finding problem that reduces to pure enumeration over all m_k's.

$$m^* = \arg\max_{m_k \in \mathbf{M}} \{g(m_k,1,0;a(1,0)) - N \cdot C(m_k)\}, \tag{10}$$

where $g(m_k,1,0;a(1,0))$ can be computed by using Equation 9, with $a(n,i) = (\tilde{x}(n,i), \tilde{s}(n,i), p(n,i))$.

5.3 Best Reply Problem for the Production Scheduling Module

Assume that other players' decisions are fixed at $(m, \beta(n,i), p(n,i))$ at each state (n,i). With this given decision and some α, we can compute the transformed point $(\tilde{x}(n,i), \tilde{s}(n,i), p(n,i))$ at each state by using equation 8. The state transition and the reward function remain the same. The best reply at each state (n,i) is then:

$$\alpha(n,i_n) = \arg\max_{\alpha\in[0,1]} \mathbf{E}_{\rho_n,d_n(\cdot)}\left\{R^{a(n,i_n)}_{(m,n,i_n)}(\rho_n,d_n(\cdot)) + \gamma g(m,n+1,i_{n+1};a(n+1,i_{n+1})\right\}, \quad (11)$$

where $a(n,i) = (\tilde{x}(n,i), \tilde{s}(n,i), p(n,i))$, with $\tilde{x}(n,i)$ computed from α, and $i_{n+1} = i_n + \tilde{x}(n,i) - \tilde{s}(n,i)$. Note that we need a finite variable domain, therefore $\alpha \in [0,1]$ is actually replaced in implementation with $\{0, \delta, 2\delta, \ldots, 1\}$. This subproblem can be viewed as a reduced MDP from the original MDP, and can be solved by backward recursion.

5.4 Best Reply Problem for the Revenue Management Module

Assume that other players' decisions are fixed at $(m, \alpha(n,i), \beta(n,i))$ at each state (n,i). With this given decision and some p, we can compute the transformed point $(\tilde{x}(n,i), \tilde{s}(n,i), p)$ at each state by using equation 8. The state transition and the reward function remain the same. The best reply at each state (n,i) is then:

$$p(n,i_n) = \arg\max_{p\in\mathbf{P}} \mathbf{E}_{\rho_n,d_n(\cdot)}\left\{R^{a(n,i_n)}_{(m,n,i)}(\rho_n,d_n(\cdot)) + \gamma g(m,n+1,i_{n+1};a(n+1,i_{n+1})\right\}, \quad (12)$$

where $a(n,i_n) = (\tilde{x}(n,i_n), \tilde{s}(n,i_n), p)$, and i_{n+1} can be computed as in previous problem. Similarly, this subproblem can be viewed as a reduced MDP and can be solved by backward recursion.

5.5 Best Reply Problem for the Sales Planning Module

Assume that other players' decisions are fixed at $(m, \alpha(n,i), p(n,i))$ at each state (n,i). With this given decision and some β, we can compute the transformed point $(\tilde{x}(n,i), \tilde{s}(n,i), p(n,i))$ at each state by using equation 8. The state transition and the reward function remain the same. The best reply at each state (n,i) is then:

$$\beta(n,i_n) = \arg\max_{\beta\in[0,1]} \mathbf{E}_{\rho_n,d_n(\cdot)}\left\{R^{a(n,i_n)}_{(m,n,i_n)}(\rho_n,d_n(\cdot)) + \gamma g(m,n+1,i_{n+1};a(n+1,i_{n+1})\right\}, \quad (13)$$

where $a(n,i) = (\tilde{x}(n,i), \tilde{s}(n,i), p(n,i))$, with $\tilde{s}(n,i)$ computed from β, and i_{n+1} can be computed as in previous problem. Note that we need a finite variable domain, therefore $\beta \in [0,1]$ is actually replaced in implementation with $\{0, \delta, 2\delta, \ldots, 1\}$. Similarly, this subproblem can be viewed as a reduced MDP and can be solved by backward recursion.

6. Numerical Experiment

In our numerical experiment, we would like to compare the performances of MDP solver and SFP solver, therefore the size of the problem is carefully calibrated so that MDP solver can

actually finish and return a solution. In the comparison, we will focus both on the quality of the solution and also the computational time required. Based on these performance analysis, we would like to extend our efforts to the use of SFP solver as a tool in discovering operational insights.

6.1 Experiment Setup

- Program parameters:
  - Planning Horizon: 10 periods.
  - $\rho \in \mathbf{L} = \{0.6, 0.66, 0.7, 0.74, 0.8\}$, where each value in $\mathbf{L}$ is selected with equal probability.
  - $\delta = 1/300$, i.e. $\alpha, \beta \in \{0, 1/300, 2/300, \ldots, 1\}$.
  - $\gamma = 1$, in other words, we ignore the discounting factor.
  - $\lambda = 0.2$, meaning that the holding cost is 20% of the manufacturing cost.
  - Number of SFP Iterations: 20
- Testing environment: Pentium 4 (2.8GHz), 1GB RAM, running RedHat Linux.

As mentioned earlier, three sets of curves - demand vs. price, investment vs. capacity, and variable cost vs. capacity - are required to drive the model. These are plotted in Figure 4. The general shapes of these curves were established via communication with Randall Urbance, Robert Bordley, and Daniel Reaume of General Motors. For confidentiality reasons, the values in these charts are randomly generated for illustrative purposes and do not represent actual values.

Here are some notes about the problem data:

- For simplicity, the data used in our study will be stationary, meaning that demand functions and variable costs are not time dependent. Note that this is just for simplicity, not a modelling assumption.
- Note that the capacity vs. capital investment curve plotted in Figure 4(a) includes a discrete step. This corresponds to a capacity at which the reduction in variable cost via automation outweighs the amortized cost of investment in this automation.
- The demand functions, as plotted in Figure 4(b), have three varieties that represent low demand, normal demand, and high demand respectively. Demand functions take the general form of $D_i(p) = e^{a_i} p^{b_i}$. In our experiments, $|\mathbf{D}| = 3$, $(a_i, b_i) \in \{(49.0478 \cdot 0.99, -4.5076), (49.0478, -4.5076), (49.0478 \cdot 1.01, -4.5076)\}$.
- The variable cost plotted in Figure 4(c) includes both labor cost and material cost.

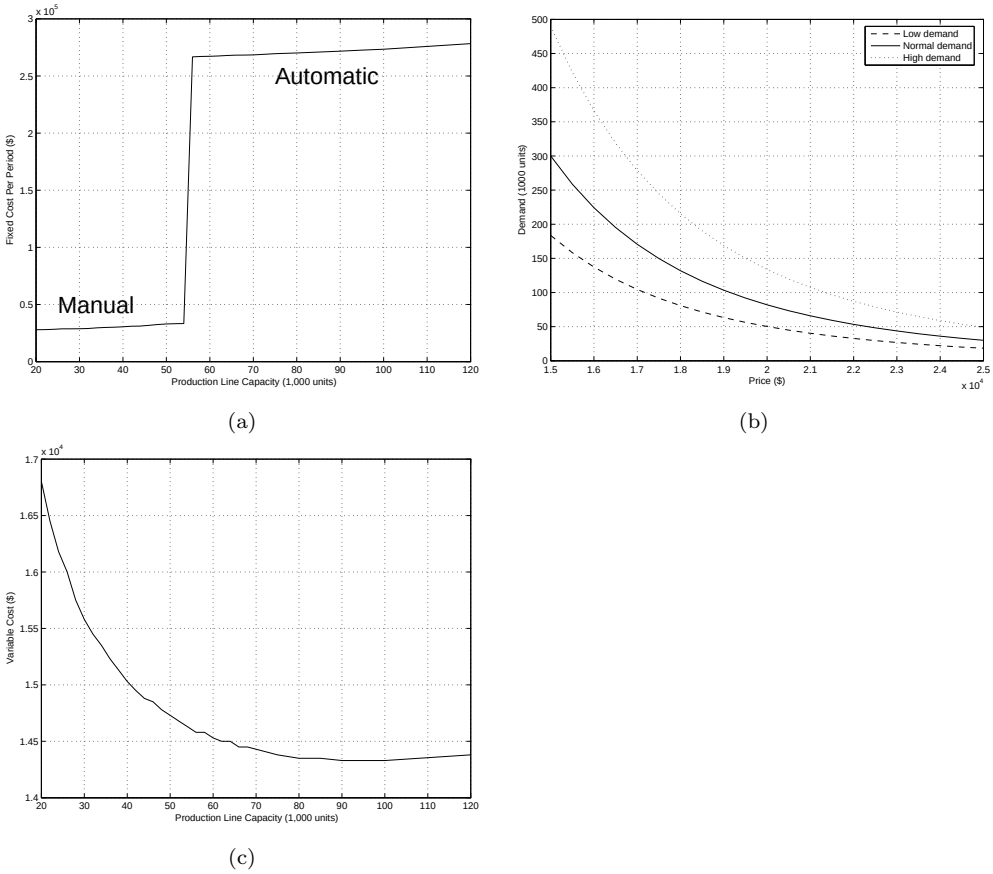

Figure 4: Important problem data: (a) Production line building cost, paid by period, as a function of capacity. (b) Demand as a function of price. (c) Variable cost as a function of capacity.

6.2 Experiment Result and Analysis

The execution results are summarized in Table 1. Note that for the MDP solver, enumerating all possible capacities cannot be finished in a reasonable amount of time. Therefore, we handpick a capacity which is made to be the optimal capacity by manipulating problem data and try to solve the single-capacity problem. Since the computational effort is identical for each capacity, we can estimate the total time required to enumerate all possible capacities. The time required to compute the optimal value for a single capacity is 5,866.3 minutes (or 4.07 days), since we have 33 capacities, the estimated execution time is 193,587.9 minutes (or 134.44 days). SFP solver is approximately 14,778 times faster than the global solver, and the quality of the solution is within 3% of the optimum.

The evolution of best values against iterations for the SFP solver is plotted in Figure 5. As plotted in Figure 5, we can see that the SFP solver makes most improvements during early iterations. In fact, it stops improving after 15^{th} iteration. This empirical finding is why we use 20 iterations as the stopping criterion for the SFP solver.

Notice that since we initiate the SFP solver with some arbitrary initial solution, we can repeatedly restart the SFP solver several times (with different initial solutions) and just keep

Algorithm	Execution time	Objective value ratio (versus global optimum)
MDP solver	134.44 days*	1.0
SFP solver	13.1 min.	0.9715

*This execution time is estimated.

Table 1: Performances of the MDP solver and the SFP solver

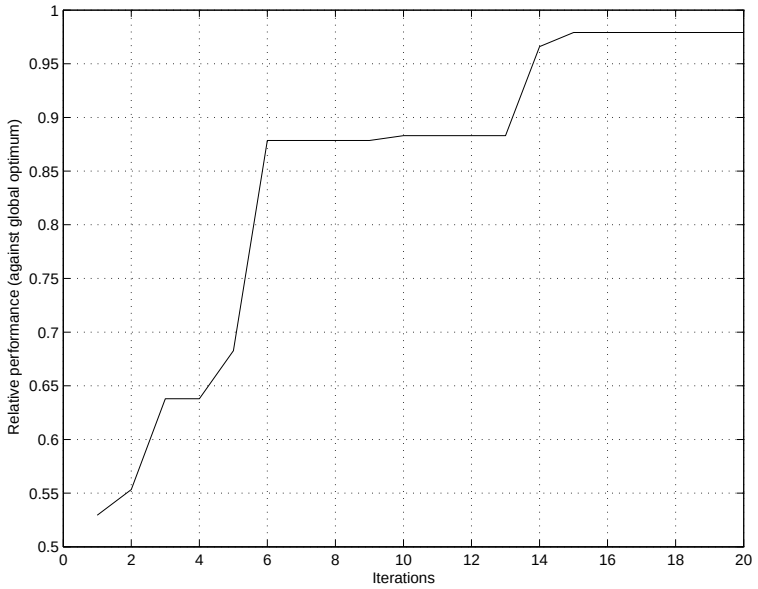

Figure 5: Best values plotted against iterations, for the SFP solver.

the best solution in these runs. As an example, if we restart the algorithm 10 times, and randomly generate the initial solution each time, the best objective value can be brought to within 1% of the global optimum. Even in this case, the SFP solver is still about 1,477 times faster than the global solver.

6.3 Obtaining Managerial Insights via Optimizations

As mentioned in the introduction, the ultimate goal of this research effort is to take advantage of the speed of the SFP optimization algorithm to develop the understandings on the impacts of key decisions by quickly considering multiple problem scenarios.

As an example, imagine the scenario where we are the production line manager, and we would like to find out the relationship between the reliability of the production line and the associated inventory stocking level.

We may accomplish this by solving the integrated problem via the SFP solver for a variety of different reliability levels. Specifically, suppose we consider several different average reliability levels. To reliability level i, we associate the set of service levels $L_i = \{0.20, 0.26, 0.30, 0.34, 0.40\} + 0.05i$. For each reliability level, we approximate an optimal policy by running the SFP solver. With these policies, we can run multiple instances of Monte Carlo simulations on ρ_n and $d_n(\cdot)$, and observe the resulting inventory level in each

case. To be more specific, we will run 1,000 instances of Monte Carlo simulations for each reliability level, and compute the average inventory level. Plotting the resulting relationship between mean service level and inventory, we can fit a linear regression equation and use it to predict the average inventory level for a given reliability. Figure 6 illustrates the result of such an analysis, where to speed up execution we set $\mathbf{D}$, the collection of demand functions, to be a singleton that includes only the normal demand function. In this case, the computed regression equation is: $I = -20.10r + 20.7924$, where r is the mean reliability level, and I is the average inventory level.

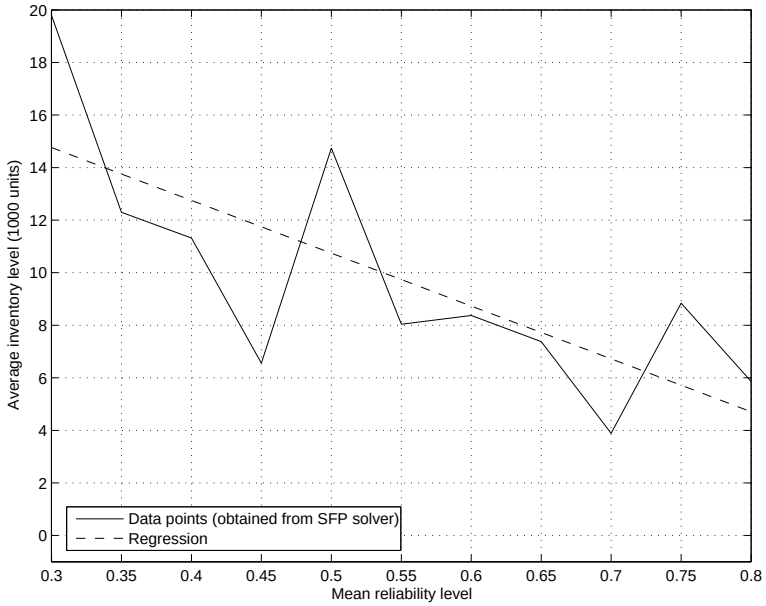

Figure 6: Average inventory levels versus mean reliability levels.

Note that the policy used above is selected from a pool of candidate policies, all generated by the SFP solver with different initializations. The selection criterion is the objective function value. In other words, we just pick the policy that returns highest expected profit. However, when comparing the average inventory levels of these policies with that of the global optimal policy, we observe that the closeness of objective function values does not imply the closeness of resulting average inventory levels. Furthermore, the policies found by the SFP solver, even with almost identical expected profits, can have very different inventory stocking patterns. This suggests that the inventory stocking level may not be a crucial factor when the expected profit is optimized. As expected, one can see that the inventory level grows almost linearly as the reliability of the production line drops. Also, as reliability level goes over certain level, it becomes optimal to implement a zero-inventory policy.

7. Conclusion and Future Work

In today's competitive environment in the automobile manufacturing sector, it is important to make coordinated, near optimal decisions at managerial, strategic and operational

levels such as capital investment, revenue management and production planning. The mathematical model of this decision problem is extremely complicated and potentially involves a multitude of exogenous as well as endogenous factors. In this paper, we presented a simplified model that captures many of these factors - capital investment, revenue management, production planning, random machine failures, and stochastic demand - yet remains computationally tractable, though still challenging to traditional optimization methods such as dynamic programming. To overcome this computational difficulty, we used the game-theoretic optimization paradigm of fictitious play. In particular, we successfully employed a recently developed algorithm called sampled fictitious play to solve the joint optimization problem to near optimality within a few minutes. The potential of using this tool as a way to develop managerial guidelines is demonstrated in the final section of this paper. We hope that in the future, this technique can be used to develop more data-driven rules of thumb to guide managerial decisions in complex manufacturing operations.

8. Acknowledgements

This work was partially supported by the National Science Foundation under Grant DMI-0422752 and General Motors Collaborative Research Laboratory at the University of Michigan.

References

Steve Alpern and Dennis J. Snower. A search model of optimal pricing and production. Discussion paper 224, Center for Economic Policy Research. London, United Kingdom, 1988.

George W. Brown. Iterative solution of games by fictitious play. In *Activity Analysis of Production and Allocation*, pages 374–376. John Wiley, New York, 1951.

Apostolos N. Burnetas and Craig E. Smith. Adaptive ordering and pricing for perishable products. *Operations Research*, 48(3):436–443, 2000.

Xin Chen and David Simchi-Levi. Coordinating inventory control and pricing strategies with random demand and fixed ordering cost: The finite horizon case. *Operations Research*, 52 (6):887–896, 2004a.

Xin Chen and David Simchi-Levi. Coordinating inventory control and pricing strategies with random demand and fixed ordering cost: The infinite horizon case. *Mathematics of Operations Research*, 29(3):698–723, 2004b.

Awi Federgruen and Aliza Heching. Combined pricing and inventory control under uncertainty. *Operations Research*, 47(3):454–475, 1999.

Youyi Feng and Youhua Chen. Joint pricing and inventory control with setup costs and demand uncertainty. Working paper, 2003.

Youyi Feng and Youhua Frank Chen. Optimality and optimization of a joint pricing and inventory-control policy for a periodic-review system. Working paper, 2004.

Drew Fudenberg and Jean Tirole. *Game Theory*. MIT Press, 1991.

Alfredo Garcia, Daniel Reaume, and Robert L. Smith. Fictitious play for finding system optimal routings in dynamic traffic networks. *Transportation Research C*, 34(2):146–157, February 2000.

Michael R. Hagerty, James M. Carman, and Gary J. Russell. Estimating elasticities with PIMS data: Methodological issues and substantive implications. *Journal of Marketing Research*, 25(1):1–9, 1988.

Theodore J. Lambert and Hua Wang. Fictitious play approach to a mobile unit situation awareness problem. Technical report, University of Michigan, 2003.

Theodore J. Lambert, Marina A. Epelman, and Robert L. Smith. A fictitious play approach to large-scale optimization. *Operations Research*, 2004. Forthcoming.

Richard D. McKelvey and Andrew McLennan. Computation of equilibria in finite games. In *Handbook of Computational Economics*, volume 1. Elsevier, 1996.

Dov Monderer and Lloyd S. Shapley. Fictitious play property for games with identical interests. *Journal of Economic Theory*, 68(1):258–265, 1996.

John Nash. Equilibrium points in n-person games. In *Proc. of the National Academy of Sciences*, volume 36, pages 48–49, 1950.

Nicholas C. Petruzzi and Maqbool Dada. Dynamic pricing and inventory control with learning. *Naval Research Logistics*, 49(3):303–325, 2002.

Julia Robinson. An iterative method of solving a game. *Annals of Mathematics*, 54:296–301, 1951.

Saroja Subrahmanyan and Robert W. Shoemaker. Developing optimal pricing and inventory policies for retailers who face uncertain demand. *Journal of Retailing*, 72(1):7–30, 1996.

Jayashankar M. Swaminathan and Sridhar R. Tayur. Tactical planning models for supply chain management. In A. G. de Kok and S. C. Graves, editors, *Supply Chain Management: Design, Coordination and Operation*, volume 11 of *Handbooks in Operations Research and Management Science*, chapter 8. Elsevier, Amsterdam, 2003.