A Fast Algorithm for Joint Optimization of Capital Investment, Revenue Management and Production Scheduling in Manufacturing Systems

March 14, 2006

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

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

Abstract

Optimal decision making in 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 considers these interactions and present a simplified example involving a single dedicated-production line. We employ 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 exceedingly faster than dynamic programming at finding near-optimal solutions, finding a solution within 1% of optimal value in less than one thousandth the computation time.

1 Introduction

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

[1]Department of Industrial and Operations Engineering, The University of Michigan, 1205 Beal Avenue, Ann Arbor, Michigan 48109-2117 USA

[2]Department of Operational Sciences, Air Force Institute of Technology, 2950 Hobson Way, Wright Patterson AFB, Ohio 45433-7765 USA

[3]General Motors Research and Development Center.

capitalize on market opportunities. Management must carefully weigh these competing goals when making decisions on capital investments, pricing, and operational policies. 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 specific product depends on dynamic exogenous factors such as economic conditions, price of substitutes and complements, 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 product in each time period? These prices, combined with the dynamic exogenous economic factors, yield demands for the product. These demands in turn drive production requirements. 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, in the automotive sector, 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 units of each product 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. A very high-level understanding of the interaction between various decision making units in an OEM firm can be developed by looking at Figure 1 below. In particular, we consider four decision making units, namely, the capital investment (CI) unit, the revenue management (RM) unit, the production planning (PP) unit and the sales (S) unit, each represented by a node. The precise procedure for drawing arcs between these nodes will be described later. However, informally, we draw a solid directed arc from node i to node j if node j needs to know node i's decision to *optimally* determine its own decision. For example, the optimal number of units to be sold, which is decided by the sales unit, is affected by the price of each unit, which is decided by the revenue management unit. Therefore, there is a directed solid arc from the RM node to the S node. Similarly, we draw a bi-directional dashed arc between node i and node j if node i (node j) needs to know node j's (node i's) decision to determine *feasibility* of its own decision. For example, the feasibility of the number of units to be produced, which is decided by the production unit, is affected by the production capacity, which is decided by the capital investment unit. Hence, there is a bi-directional dashed arc between the CI unit and the PP unit. In general, the interaction between different decision making units is so involved, that an integrated approach to decision making is vital. In particular, the higher level units anticipate the impact of their decisions on the lower level units and take the response of the lower level units into consideration while making their own decisions. 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

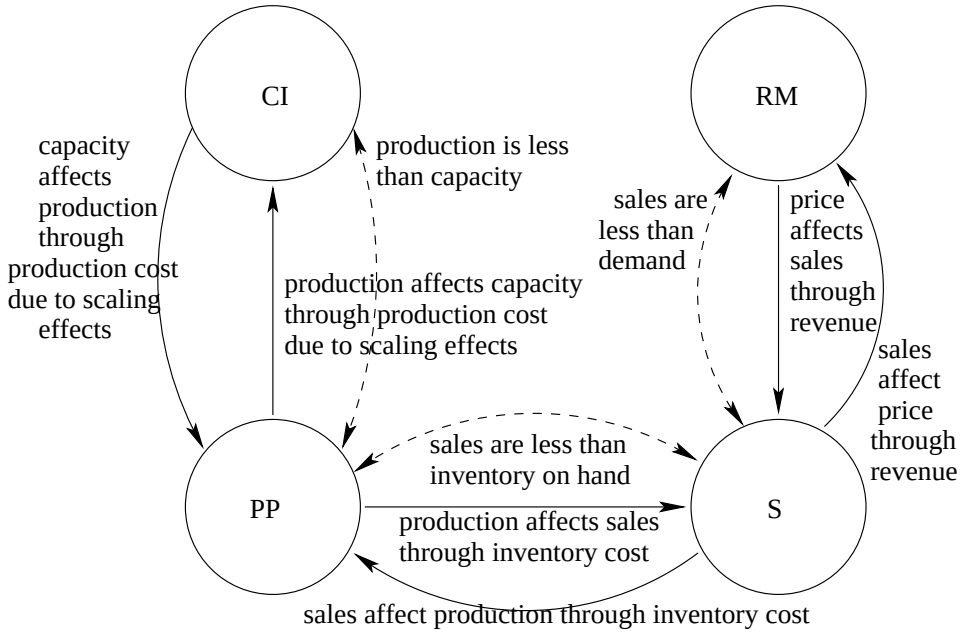

Figure 1: Interaction between different decision making units in an OEM

management and of investments in flexible production systems are not fully realized unless both are exercised together. Therefore, we will take an integrated approach to manufacturing optimization as represented in Figure 1 in this paper. In particular, we will model the joint optimization problem at hand as a game of identical interests between four players represented by the nodes in Figure 1 above.

There has been a recent boom in the revenue management-inventory control literature. See Chan et al [2] for a comprehensive survey. Researchers have begun to realize that pricing and sales decisions cannot be made independently of operational decisions such as inventory management and production scheduling [1]. Chen and Simchi-Levi [3] considered a finite horizon, single product, periodic review model with simultaneous pricing, and production/inventory decisions. In their work, product demands across periods were independent random variables dependent on price. They proved optimality of an (s, S, p) policy when the demand model is additive. Federgruen and Heching [5] considered a similar problem and its infinite horizon discounted version. Gallego and van Ryzin [7] considered the problem of optimally pricing inventory to sell a given stock of items by a deadline under uncertain demand. For an exponential family of demand functions, they were able to find an optimal pricing policy in closed form. For other, more general demand functions, they obtained an upper bound on the expected revenue. In a very recent paper, Kocabiyikoglu and Popescu [9] extended the conventional revenue management concepts to simultaneously optimize price and inventory decisions in three phases of the revenue management process: capacity investment, product design, and intertemporal segmentation. They showed that a joint optimization approach leads to significantly higher profits as compared to the traditional hierarchical decision process.

To the best of our knowledge, literature in this area mainly focuses on analytical results.

This type of work utilizes intuitive assumptions on the price-demand functions and cost structure to come up with closed form optimal policies or decision sequences. A major advantage of these results is that they provide important insight into the problem at hand. However, similar results are hard to obtain in the absence of simplifying assumptions. In that case, one may have to resort to an algorithmic approach to come up with optimal decisions. There has been relatively less research in that direction. In this paper we devise an efficient algorithm for computing near-optimal policies for a complex joint optimization problem in manufacturing that involves capital investment, pricing and production planning decisions.

To understand the problem abstractly, we will first establish a mathematical model that captures key features of 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. Employing a purely numerical approach to developing these insights is usually very hard. It requires solving multiple problem scenarios quickly. We will illustrate that such an exercise is possible using our algorithm because it is about four orders of magnitude faster than other conventional approaches.

In particular, we will use an approach based on the game theoretic concept of Sampled Fictitious Play (SFP). SFP is a modification of a well known learning paradigm called Fictitious Play from game theory. FP was proposed as a method to find Nash equilibria of finite games in strategic form [14]. In general, the FP algorithm may not converge to a Nash equilibrium. However, in a landmark paper, Monderer and Shapley [12] proved that FP converges to a Nash equilibrium if the players have identical interests. This convergence behavior is usually called *the fictitious play property*. It motivated researchers to use FP as a heuristic for solving large scale, unconstrained, discrete optimization problems. The main reason for this was that these problems can be modeled as finite games in strategic form with identical interests where the players share the common interest of optimizing the objective function. FP has one important advantage in that it can be completely parallelized. However, if the number of actions available to each player is very large, FP becomes computationally intractable very quickly.

To overcome this computational difficulty, researchers suggested using a variation of FP called Sampled Fictitious Play (SFP), which, as the name suggests, uses a sampling technique, and reported good numerical results on real world problems in dynamic traffic routing [8] and communication protocol design [10]. Some of these results were at least as good as those obtained by well known, competing discrete optimization methods such as Simulated Annealing [10]. At that point, no analytic results were known about SFP. However, Lambert et al., [11] demonstrated that SFP inherits the fictitious play property, i.e., it exhibits convergence behavior similar to FP. Further, SFP is significantly faster than FP.

As the above discussion suggests, SFP was used only on unconstrained discrete optimization problems until now. In this paper, we break this significant hurdle by applying an appropriately modified version of SFP to the joint optimization problem in manufacturing,

which is a sequential decision problem with constraints. Using a variable transformation, we convert this problem to a finite game in strategic form with identical interests and argue that SFP can indeed be applied to this problem. Our approach essentially breaks the original large scale dynamic programming formulation down into smaller subproblems, some of which are simple dynamic programs themselves, thus leading to an efficient solution method.

We consider the automotive manufacturing sector as a specific example of a manufacturing system and use real world data to perform numerical experiments using SFP. We use this case study to compare SFP's performance to more conventional dynamic programming algorithms and conclude that SFP finds near-optimal solutions significantly faster. Since SFP, just like FP, can be parallelized and hence, can be used to solve multiple problem scenarios quickly, it may be used as a managerial tool to develop important insights and rules of thumb for the problem at hand.

The paper is organized as follows. We refer the reader to appendix A for some mathematical preliminaries from game theory, and the original FP and SFP algorithms. In the second section, we formulate deterministic and stochastic versions of a manufacturing system optimization problem. We then convert these problems into finite games in strategic form with identical interests by using a variable transformation to make them amenable to SFP. In the third section, we present numerical results obtained by applying SFP to a vehicle manufacturing plant. We illustrate how SFP can be used as a managerial tool to develop key insights and rules of thumb for a manufacturing plant by solving multiple instances of the problem quickly followed by a regression analysis of how performance is influenced by other factors. We conclude in the fourth section.

2 Optimizing a Manufacturing System

We consider a manufacturing firm that produces units of a product such as computers, cars, airplanes, clothes etc. We assume that the decision making personnel in this plant are divided into different strategic, managerial, tactical and operational modules such as capital investment, pricing, production planning, and sales. We model the manufacturing process over a finite horizon as follows. The planning horizon is divided into periods of equal length. At the beginning of the horizon, the capital investment module decides what plant capacity to put in operation. At the beginning of each period, the pricing module (also called the revenue management module) decides the price of each unit of the product. This induces a demand for the product during that period through a price-demand function. This demand may be deterministic or stochastic. (Before observing the realized demand) The production planning module decides the number of units to be produced in that period. Note that the manufacturing plant may not be actually able to produce these many units because the machines could be unreliable. (Before observing the realized demand and the realized production) The sales module sets a goal as to how many units they would like to sell in that period. We assume that extra demand is lost, i.e. no back-ordering is allowed. The units left over at the end of a period can be carried to the next period as inventory and used to satisfy the demand in that and subsequent periods.

2.1 The Deterministic Problem

In the beginning, for simplicity, assume that the problem is deterministic. That is, the price-demand function is deterministic and all machines are reliable. The goal is to maximize profit where the revenue is generated by selling the product and costs are incurred for building the plant, production, and handling inventory. It will be clear soon that we do not assume any specific cost structure making this problem potentially very hard to solve by conventional methods. We will use the following notation.

- The planning horizon N is divided into periods of equal duration denoted by the indices $n = 1, 2, \ldots, N$.
- The finite set of possible plant capacities is denoted by $\mathbf{M} = \{m_1, m_2, \ldots, m_{|\mathbf{M}|}\}$.
- The amortized cost of building a plant with capacity $m \in \mathbf{M}$ is $C(m)$ incurred every year. Thus the total building cost is $NC(m)$.
- The finite set of possible unit prices in each period is denoted by $\mathbf{P}$.
- The demand induced by a price of $p \in \mathbf{P}$ per unit in period n is given by the price-demand function $D_n(p)$.
- The discount factor is γ.
- The cost of production depends on the plant capacity m, the production level x, and the period n and is denoted by $c(m, x, n)$.
- The cost of carrying an inventory of i units from period n to $n + 1$ is $h(i, n)$.

Recall that the capital investment module decides the plant capacity at the beginning of the first period, (the plant becomes operational instantaneously) and then all the other decision modules make their own decision at the beginning of every period including the first. Consider the sequential decision problem faced by the pricing, production and sales units where they decide the price per unit, number of units to be produced and number of units to be sold in each period. This problem can be formulated as the following non-linear integer program.

$$
\begin{aligned}
\text{(IP)} \quad \max \quad & - \quad NC(m) + \sum_{n=1}^{N} \{p_n \times s_n - c(m, x_n, n) - h(n, i_n)\} \\
\text{subject to} & \\
m & \in \mathbf{M} \\
p_n & \in \mathbf{P}, \quad n = 1, 2, \ldots, N \\
x_n & \leq m, \quad n = 1, 2, \ldots, N \\
s_n & \leq i_{n-1} + x_n, \quad n = 1, 2, \ldots, N \\
s_n & \leq D_n(p_n), \quad n = 1, 2, \ldots, N \\
x_n, s_n & \geq 0, \text{ integer}, \quad n = 1, 2, \ldots, N \\
i_n & = i_{n-1} + x_n - s_n, \quad n = 1, 2, \ldots, N \\
i_0 & = 0
\end{aligned}
$$

Observe that variable i_n can be eliminated by substituting

$$i_n = \sum_{k=1}^{n}(x_k - s_k) \;\; n = 1, 2, \ldots, N.$$

and then using $h(n, \sum_{k=1}^{n}(x_k - s_k))$ in place of $h(n, i_n)$. Once this has been done, the interaction between different variables in the above integer program can be illustrated in a compact manner by using a 'variable interaction diagram'.

2.1.1 Variable Interaction Diagram

The variable interaction diagram (VID) for problem (IP) is a graph with nodes and arcs as shown in Figure 2.

In particular, there is one node for each *type* of variable. Thus, there is one node for capacity variable m, one node for price variables p_n, $n = 1, 2, \ldots, N$, one node for sales variables s_n, $n = 1, 2, \ldots, N$, and one node for production variables x_n, $n = 1, 2, \ldots, N$. Observe that each of these nodes corresponds to a specific decision making unit represented by a circle in Figure 1. For simplicity, we will call these variable-types (and hence the nodes) as type-m, type-p, type-s, and type-x respectively. To emphasize that nodes type-p, type-x and type-s choose a vector each as their decision, we use curly braces to name these nodes on the VID. We will denote the objective function by $V(m, p, s, x)$ for simplicity.

There are two types of directed arcs in the VID; *value arcs* and *feasibility arcs*. The value arcs are shown by solid arrows whereas the feasibility arcs are shown by dashed arrows. While drawing the value arcs, we ignore all constraints except the simple box constraints on the variables. In other words, while drawing the value arcs we assume that any combination of variables of different types is jointly feasible as long as it satisfies the box constraints. For examples, while drawing the value arcs for problem (IP), we ignore the constraints

$$\begin{aligned} x_n &\leq m, \;\; n = 1, 2, \ldots, N \\ s_n &\leq i_{n-1} + x_n, \;\; n = 1, 2, \ldots, N \\ s_n &\leq D_n(p_n), \;\; n = 1, 2, \ldots, N \end{aligned}$$

We will explain the procedure for drawing value arcs through a specific example as opposed to providing a general algorithm. Consider the node type-m. This node corresponds to the capacity variable. In order to decide if there is an incoming directed arc into node type-m from nodes type-p,s, and x, we fix the values of these three variable types and observe which of these values are *relevant* for choosing an optimal value of capacity. In particular, there is a directed arc from node type-x to node type-m *if there exist* jointly feasible triples (ignoring all constraints other than the box constraints) $\hat{p}, \hat{s}, \hat{x}$, and $\hat{p}, \hat{s}, \tilde{x}$ such that $\tilde{x} \neq \hat{x}$ and

$$\hat{m}^* = \underset{m \in \mathbf{M}}{\operatorname{argmax}} \;\; V(m, \hat{p}, \hat{s}, \hat{x}) \neq \tilde{m}^* = \underset{m \in \mathbf{M}}{\operatorname{argmax}} \;\; V(m, \hat{p}, \hat{s}, \tilde{x}).$$

Similarly for directed arcs from node type-p and type-s. Existence of value arcs coming into nodes p,s, and x can be decided similarly. When the exact form of the objective function is not known, there will be a lot more value arcs in the VID as opposed to when it is known.

Feasibility arcs are determined by considering the constraints that were ignored while drawing the value arcs. We draw a feasibility arc from node type-i to node type-j $j \neq i$ if there exist a value of variable type-j (feasible to the box constraints) and (at least) two distinct values of variable type-i such that such that one of these (two) values makes variable type-j feasible to the constraints (other than the box constraints) and then other value(s) makes variable type-j infeasible. Feasibility arcs are many a time symmetric in the sense that there is a feasibility arc from node type-i to node type-j if and only if there is a feasibility arc from node type-j to node type-i, in which we draw one bi-directional arc between the two nodes. Specifically, consider the type-m and type-x nodes above. There is a feasibility arc from node x to node m if there exist a capacity $\hat{m} \in \mathbf{M}$ and two distinct production plans $\bar{x}$ and $\hat{x}$ such that

$$\begin{aligned} \bar{x}_i &\leq \hat{m} \;\; i = 1, 2 \ldots, N \\ \hat{x}_j &> \hat{m} \;\; \text{for some} \;\; 1 \leq j \leq N \end{aligned}$$

The VID below was drawn using the procedure described above. Note that the VID

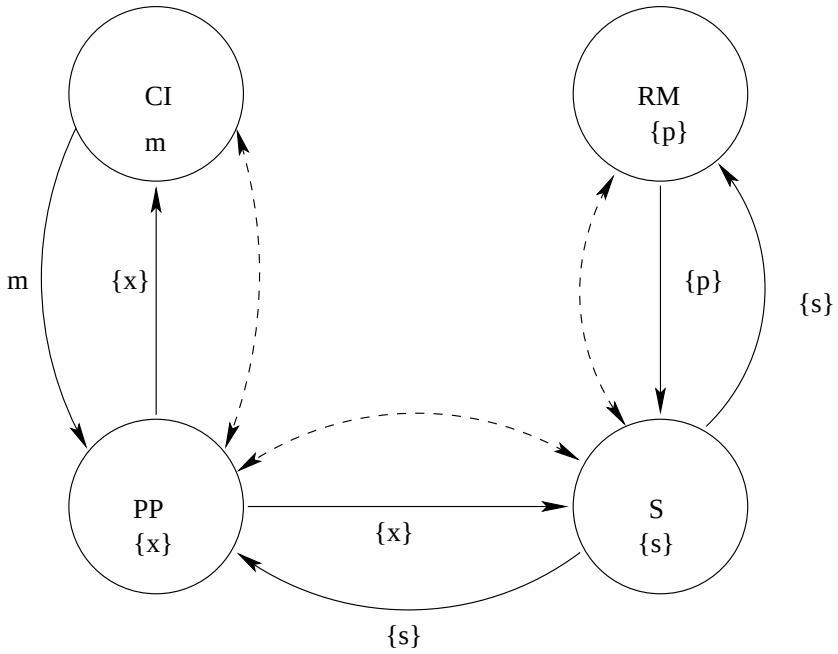

Figure 2: The Variable Interaction Diagram for problem (IP)

conveys a lot of information about problem (IP). In particular, it gives us a very high-level view of the feasibility structure of the problem as well as how decisions of one decision making module affect those of another. We will return to the VID later in section 2.1.4. We will present a dynamic programming formulation of the problem (IP) in the next section.

2.1.2 A Dynamic Programming Formulation

With some thought it becomes clear that the state is defined by (m, n, i) where m is the capacity chosen by the capital investment module at the beginning of the first period, and we

are starting period n with inventory i on hand. Let $F(m,n,i)$ be the set of feasible 3-tuples (p_n, x_n, s_n) in state (m,n,i). Let $f(m,n,i)$ be the maximum profit obtainable when the plant capacity is m and we begin period n with inventory i on hand. Then, the manufacturing system optimization problem can be formulated as

$$\begin{aligned}(\text{PD})\ \max_{m\in\mathbf{M}} & f(m,1,0) - NC(m) \\ & \text{with}\ \ f(m,n,i) = \max_{(p_n,x_n,s_n)\in F(m,n,i)} \{p_n \times s_n \\ & - c(m,x_n,n) - h(n,i) + \gamma f(m,n+1,i+x_n-s_n)\}, \\ & \text{for all}\ \ m\in M,\ i,\ \text{and}\ \ n = 1,2,\ldots,N-1 \\ & \text{where},\ \ f(m,N,i) = 0\ \ \text{for all}\ \ m\in M,\ \text{and}\ \ i\end{aligned}$$

and the feasible set $F(m,n,i)$ is defined by the constraints

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

In the presence of restrictive structural assumptions on the price-demand function such as stationarity over time, the production and other costs such as linearity, concavity, or convexity, one may be able to obtain some analytical results regarding an optimal decision sequence for the above problem. However, we will not make such assumptions at this point, and resort to a purely algorithmic approach to compute optimal decisions. In the absence of simplifying assumptions, a large number of possible decision choices in each period makes the problem very challenging to conventional optimization techniques. Further, even though specially designed algorithms that exploit cost and demand structure to obtain near-optimal solutions may be developed in some cases, our main goal here is to devise a general purpose algorithmic framework that works on manufacturing problems of the above form without regard to cost and demand structure. Moreover, we would like such an algorithm to be very efficient in practice so that it can be used to solve multiple problem scenarios quickly to develop managerial insight. In particular, we will modify the Sampled Fictitious Play algorithm to make it applicable to the above problem.

2.1.3 Solution by SFP: Divide and Conquer

SFP has been investigated ([8], [10], [11]) as a heuristic for optimization problems of the form

$$\begin{aligned} & \max u(y^1, y^2, \ldots, y^{|\mathcal{N}|}) \\ \text{subject to}\ \ & y^1 \in \mathcal{Y}^1,\ \ y^2 \in \mathcal{Y}^2, \ldots, y^{|\mathcal{N}|} \in \mathcal{Y}^{|\mathcal{N}|}\end{aligned} \tag{1}$$

where $\mathcal{Y}^i$, $i = 1,2,\ldots,|\mathcal{N}|$ are finite sets. One motivation for using SFP as an optimization heuristic is that the above problem can be associated with a finite game in strategic form, and an optimal solution of this problem is a pure strategy Nash equilibrium of that game.

(Unfortunately, a Nash equilibrium of that game may not be an optimum solution of problem (1)). (For a brief but complete description of relevant concepts in game theory, please see the appendix). In absence of any structural assumptions on the form of the objective function $u(\cdot)$, many conventional discrete optimization algorithms may be too slow even in obtaining close to optimal solutions. This situation is exasperated by the curse of dimensionality. SFP has emerged as a good heuristic to solve problems of the form (1) in the recent past. In particular, it has been successfully applied to very complex discrete optimization problems in traffic routing [8] and communication protocol design [10]. A significant advantage of SFP is that it can be easily parallelized. Researchers have only recently begun taking full advantage of this feature [4]. In this paper, we will extend SFP to problems with constraints that are more complicated than those in problem (1) in section 2. More specifically, observe that in the above problem, every variable appears in only one constraint, essentially only constraining it to take values from a finite set. In that sense, problem (1) is *unconstrained*.

It is clear from the inequality constraints for problem (PD) (or problem (IP)) that the feasible set is not in the box-constraint format required by SFP as in problem (1). However, the first inequality constraint on production level x_n can be rewritten as $x_n = \alpha_n m$ where $\alpha_n \in \{0, \epsilon, 2\epsilon, \ldots, 1\} \equiv \mathbf{A}$ for some $0 < \epsilon \leq 1$. Similarly, the inequality constraint on sales can be rewritten as $s_n = \beta_n \min\{i + x_n, D_n(p_n)\}$ with $\beta_n \in \{0, \delta, 2\delta, \ldots, 1\} \equiv \mathbf{B}$ for some $0 < \delta \leq 1$. Then a transformed version of the original manufacturing system optimization problem can be written by substituting for x_n and s_n in terms of α_n and β_n. We will not repeat it here for brevity. Note that the feasible set $F(m, n, i)$ can be redefined by the constraints

$$p_n \in \mathbf{P}, \quad \alpha_n \in \mathbf{A}, \quad \beta_n \in \mathbf{B} \text{ for } n = 1, 2, \ldots, N \tag{2}$$

Observe that constraints (2) are precisely in the form required for SFP. In particular, writing the constraints in this form eliminates all feasibility arcs from the VID in Figure 2. Thus, the decision making modules in the firm now interact only through the objective function and not through the constraints. We are now ready to discuss precisely how SFP can be applied to this problem.

2.1.4 Choice of Players and Actions

In order to model the manufacturing problem as a finite game in strategic form with identical interests, we must first identify the players involved in this game, and their actions sets. Comparing the constraints in equation (2) with those in the unconstrained optimization problem (1), the most naive way to choose players is to assign one player to each variable. In that case, we would have one player to decide the plant capacity and one player each to decide the price, production quantity and sales in each period. We would then have $1 + 3N$ players in the game. Some thought is required to see that if these were the players, the best response problem for each of these players in each iteration of SFP would be a one variable problem that can be solved by pure enumeration. Thus, these players are not very 'intelligent', deciding their best actions by pure enumeration in every iteration. Further, this choice is also not very 'natural', since in a real manufacturing firm, one decision module makes decisions regarding a specific aspect of the manufacturing process in every period. In that sense, there is a very natural choice for players in the manufacturing problem that corresponds to the real world situation. In particular, we consider four players: a Capital

Investment (CI) player, a Revenue Management (RM) player, a Production Planning (PP) player and a Sales (S) player. Thus, there is one player corresponding to every node in the VID in figure 2. We will see later that the best response problems of three of these players are dynamic programs, making these players 'very smart'. The CI player decides the capacity, and hence its actions set is $\mathbf{M}$. The RM player decides price per unit in each period, i.e., $(p_1, p_2, \ldots, p_N)$, hence its action set is $\mathbf{P}^N \equiv \mathbf{P} \times \mathbf{P} \ldots \times \mathbf{P}$. The PP player decides the production level in each period, i.e., $(\alpha_1, \alpha_2, \ldots, \alpha_N)$, hence its action set is $\mathbf{A}^N \equiv \mathbf{A} \times \mathbf{A} \ldots \times \mathbf{A}$. Similarly, the S player decides the number of units to be sold in each period, i.e., $(\beta_1, \beta_2, \ldots, \beta_N)$, hence its action set is $\mathbf{B}^N \equiv \mathbf{B} \times \mathbf{B} \ldots \times \mathbf{B}$. The next step is to define the identical interest that these players share, i.e. the objective function. The formulation below includes such an objective, i.e. to maximize profit over N periods.

$$\begin{aligned} &\text{(SFP-D)}\max \quad v(m, ((p_1, p_2, \ldots, p_N)), ((\alpha_1, \alpha_2, \ldots, \alpha_N)), ((\beta_1, \beta_2, \ldots, \beta_N))) \\ \equiv &\left[-NC(m) + \sum_{n=1}^{N} \gamma^n \{p_n \times \beta_n \min\{i_n + \alpha_n m, D_n(p_n)\} - c(m, \alpha_n m, n) - h(n, i_n)\} \right] \qquad (3) \\ &\text{subject to } \; p_n \in \mathbf{P}, \;\; \alpha_n \in \mathbf{A}, \;\; \beta_n \in \mathbf{B} \;\text{ for }\; n = 1, 2, \ldots, N \text{ and } m \in \mathbf{M} \end{aligned}$$

where $i_1 = 0$ and $i_{n+1} = i_n + \alpha_n m - \beta_n \min\{i_n + \alpha_n m, D_n(p_n)\}$. Observe that this problem is of the form (1). The heuristic SFP with sample size of one can be applied to the above formulation. As explained in section A.3, at each iteration each of the four players draws a sample from its history. It is important to note that due to the transformation we have used, every joint play by the four players is feasible.

Since there are four players in the game, we have to solve four best response problems at each iteration, and the best reply of each player is added to its own history of plays. Let $\hat{m}$, $\hat{p} \equiv (\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_N)$, $\hat{\alpha} \equiv (\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_N)$, and $\hat{\beta} \equiv (\hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_N)$ be the actions sampled by CI, RM, PP, and S respectively in a particular iteration. We use m^*, p^*, α^*, and β^* to denote the best responses of these four players. We formulate the best response problems of these four players below. In particular, we show that the CI best response is a simple one variable maximization problem that can be solved by pure enumeration, whereas the other three best response problems are dynamic programming problems themselves.

2.1.5 CI Best Response

This best response problem is conceptually the easiest because the capital investment module makes a decision only at the beginning of the horizon. The best response problem for CI is simply a one variable discrete optimization problem written

$$m^* = \arg \max_{m \in \mathbf{M}} \; v(m, \hat{p}, \hat{\alpha}, \hat{\beta})$$

2.1.6 RM Best Response

The best response problem of the revenue management player can be formulated as a dynamic program. The state for this dynamic program is (n, i), which stands for beginning the n^{th} period with inventory i on hand. Let $g_{rm}(n, i)$ be the maximum profit obtainable in periods n

through N when we begin period n with inventory i. The dynamic programming functional equation can be written as follows.

$$g_{rm}(n,i) = \max_{p_n \in \mathbf{P}} \{p_n \times \hat{\beta}_n \min\{i + \hat{\alpha}_n \hat{m}, D_n(p_n)\} - c(\hat{m}, \hat{\alpha}_n \hat{m}, n) - h(n,i) +$$
$$\gamma g_{rm}(n+1, i + \hat{\alpha}_n \hat{m} - \hat{\beta}_n \min\{i + \hat{\alpha}_n \hat{m}, D_n(p_n)\})\} \text{ for all } i, \text{ and } n = 1, 2, \ldots N-1$$
$$g_{rm}(N,i) = 0 \text{ for all } i$$

This dynamic programming problem can be solved efficiently using standard dynamic programming techniques to yield the RM best response $p^* = (p_1^*, p_2^*, \ldots, p_N^*)$.

2.1.7 PP Best Response

The state for this dynamic program is the same as above, i.e., (n,i). The dynamic programming functional equation can be written as

$$g_{pp}(n,i) = \max_{\alpha_n \in \mathbf{A}} \{\hat{p}_n \times \hat{\beta}_n \min\{i + \alpha_n \hat{m}, D_n(\hat{p}_n)\} - c(\hat{m}, \alpha_n \hat{m}, n) - h(n,i) +$$
$$\gamma g_{pp}(n+1, i + \alpha_n \hat{m} - \hat{\beta}_n \min\{i + \alpha_n \hat{m}, D_n(\hat{p}_n)\})\} \text{ for all } i, \text{ and } n = 1, 2, \ldots N-1$$
$$g_{pp}(N,i) = 0 \text{ for all } i$$

where $g_{pp}(n,i)$ be the maximum profit obtainable in periods n through N when we begin period n with inventory i. This dynamic programming problem can be solved efficiently using standard dynamic programming techniques to yield the PP best response $\alpha^* = (\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*)$.

2.1.8 S Best Response

With notation similar to above, this dynamic programming formulation can be written as

$$g_s(n,i) = \max_{\beta_n \in \mathbf{B}} \{\hat{p}_n \times \beta_n \min\{i + \hat{\alpha}_n \hat{m}, D_n(\hat{p}_n)\} - c(\hat{m}, \hat{\alpha}_n \hat{m}, n) - h(n,i) +$$
$$\gamma g_s(n+1, i + \hat{\alpha}_n \hat{m} - \beta_n \min\{i + \hat{\alpha}_n \hat{m}, D_n(\hat{p}_n)\})\} \text{ for all } i, \text{ and } n = 1, 2, \ldots N-1$$
$$g_s(N,i) = 0 \text{ for all } i$$

Again, this dynamic programming problem can be solved efficiently using standard dynamic programming techniques to yield the S best response $\beta^* = (\beta_1^*, \beta_2^*, \ldots, \beta_N^*)$.

The formal SFP procedure applied to the deterministic manufacturing problem is stated below. The relevant notation is as follows. $H_{CI}(i)$ represents the *number* stored in the history of the CI player at the end of the i^{th} iteration. Similarly, $H_{RM}(i)$, $H_{PP}(i)$ and $H_S(i)$ are the N dimensional *vectors* stored in the histories of the RM, PP and S players at the end of the i^{th} iteration. Observe the difference in the way history is stored for player CI as opposed to the other players. This difference stems from the fact that CI player chooses a single action in each iteration whereas the other players choose a vector of N actions.

Algorithm 2.1. SFP for Optimizing A Deterministic Manufacturing Operation

1. *Begin with randomly chosen actions* m^0, $p^0 \equiv (p_1^0, p_2^0, \ldots, p_N^0)$, $\alpha^0 \equiv (\alpha_1^0, \alpha_2^0, \ldots, \alpha_N^0)$, *and* $\beta^0 \equiv (\beta_1^0, \beta_2^0, \ldots, \beta_N^0)$ *in the histories of the CI, RM, PP, and S players respectively. That is,* $H_{CI}(0) = m^0$, $H_{RM}(0) = (p_1^0, p_2^0, \ldots, p_N^0)$, $H_{PP}(0) = \alpha^0 \equiv (\alpha_1^0, \alpha_2^0, \ldots, \alpha_N^0)$ *and* $H_S(0) = (\beta_1^0, \beta_2^0, \ldots, \beta_N^0)$. *Set the iteration counter* $k = 1$.

2. *In iteration* k, *sample four numbers* ci, rm, pp,*and* s *uniformly randomly and independently from integers* $0, 1, \ldots, k-1$. *Let the sampled actions for the four players be given by* $\hat{m} = H_{CI}(ci)$, $\hat{p} \equiv (\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_N) = H_{RM}(rm)$, $\hat{\alpha} \equiv (\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_N) = H_{PP}(pp)$, *and* $\hat{\beta} \equiv (\hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_N) = H_S(s)$.

3. *Compute the best responses* m^*, $p^* \equiv (p_1^*, p_2^*, \ldots, p_N^*)$, $\alpha^* \equiv (\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*)$, *and* $\beta^* \equiv (\beta_1^*, \beta_2^*, \ldots, \beta_N^*)$ *by solving the best response problems for the CI, RM, PP, and S players defined above.*

4. *Add the best responses* m^*, p^*, α^*, *and* β^* *to the histories of the CI, RM, PP, and S players respectively. That is, set* $H_{CI}(k) = m^*$, $H_{RM}(k) = p^* \equiv (p_1^*, p_2^*, \ldots, p_N^*)$, $H_{PP}(k) = \alpha^* \equiv (\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*)$, *and* $H_S(k) = \beta^* \equiv (\beta_1^*, \beta_2^*, \ldots, \beta_N^*)$. *Advance the iteration counter to* $k + 1$.

5. *Continue steps 2-4 above until a stopping criterion is met.*

It is very important to note that modeling the original large scale constrained dynamic program as a finite game in strategic form after applying the variable transformation results in breaking the problem down into much smaller subproblems called the 'best response problems'. Moreover, three of these subproblems are dynamic programming problems, lending themselves to efficient solution by standard techniques. At this point, we move on and add effects of stochasticity to the above model.

2.2 The Stochastic Problem

In this section, we study the stochastic version of the above problem. In particular, as mentioned at the beginning of section 2, we allow the price demand function to be stochastic and for random machine breakdowns. We use the following approach to model stochasticity.

- We use a parameter ρ_n to model reliability of the production system. In particular, ρ_n is a random variable that indicates the fraction of planned capacity actually available for production in period n. By definition, $\rho_n \in [0, 1]$. We assume that the distribution of this random variable is stationary, and it takes values from a finite set $\mathbf{L} \equiv \{l_1, l_2, \ldots, l_{|\mathbf{L}|}\}$. More specifically, the probability that ρ_n takes a value $l_k \in \mathbf{L}$ does not depend on n, is denoted by P_{l_k}, and is independent of everything else.

- We let $\mathbf{D} = \{D_1(\cdot), D_2(\cdot), \ldots, D_{\mathbf{D}}(\cdot)\}$ be the set of possible price-demand functions. In each period, n the demand induced by a particular price p_n is $D_k(p_n)$ with probability Q_k independently of everything else. Observe that we could have used a 'different' set $\mathbf{D}_n$ for each period n relaxing the stationarity assumption inherent in the notation $\mathbf{D}$. We have not done that here only for notational simplicity. This is not a requirement forced by the algorithm.

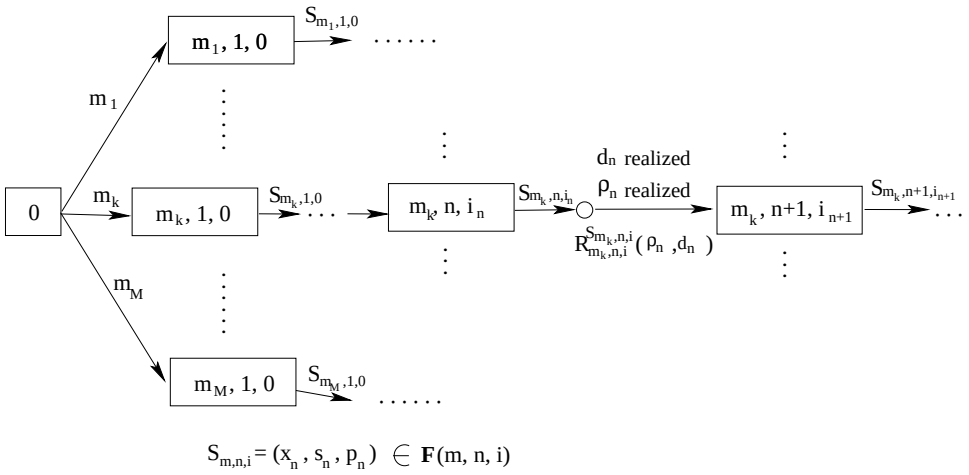

Figure 3: MDP Network for the Stochastic Manufacturing Optimization Problem

It is important to specify the order in which these random variables are realized and the exact time-points at which various decision modules make their decisions. Recall that the capital investment module decides the plant capacity at the beginning of the first period. Then, at the beginning of every period, the revenue management, production planning and sales modules decide the price, production, and sales in that period. The stochastic price-demand function and plant reliability are realized after that. In particular, these three units do not know the realized demand and plant capacity at the time of making their decisions in each period. We model this stochastic joint optimization problem as a Markov decision process. Just like the deterministic problem, the state for the stochastic problem is (m, n, i), where m is the plant capacity chosen by the capital investment unit, n is the period and i is the inventory beginning period n. The stochastic decision network for this problem is depicted in figure 3. The feasible set $F(m, n, i)$ is now defined by the constraints

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

where $\bar{D}(p_n)$ is the maximum possible demand, i.e., $\bar{D}(p_n) = \max_{k \in \{1,2,\ldots,|\mathbf{D}|\}} D_k(p_n)$. Observe that unlike the deterministic problem, we have used this upper bound on demand while defining the feasible region. This is because the stochastic demand is realized only after making all the decisions in a particular period. The state transitions in the stochastic problem are more complicated than the deterministic counterpart. Therefore, we define them explicitly here. In particular, let the current state be (m, n, i_n) and $(p_n, x_n, s_n) \in F(m, n, i_n)$ be the feasible decisions chosen by the pricing, production and sales modules respectively. Also, let $l_k \in \mathbf{L}$ be the realized plant reliability and $d_j(\cdot) \in \mathbf{D}$ be the realized price-demand function. The joint probability of this event is $P_{l_k} Q_j$. Then, the new state $(m, n+1, i_{n+1})$ is defined as

follows. Let

$$\begin{aligned} \hat{x}_n &= \min\{x_n, l_k m\} \\ \hat{s}_n &= \min\{s_n, i_n + \hat{x}_n, d_j(p_n)\} \\ i_{n+1} &= i_n + \hat{x}_n - \hat{s}_n \end{aligned}$$

We assume that the production cost depends on both the planned production x_n and the realized production $\hat{x}_n$ in addition to m and n, and denote it by $c(m, x_n, \hat{x}_n, n)$. Now let $f(m, n, i_n)$ be the maximum expected profit generated in periods n thorough N when we begin period n with inventory i_n and plant capacity m chosen by the capital investment module at the beginning of the first period. The Markov decision problem can be written using the maximum discounted expected reward criterion as follows.

$$\begin{aligned} (PS) \max_{m \in \mathbf{M}} & f(m, 1, 0) - NC(m) \\ & \text{with } f(m, n, i_n) = \max_{(p_n, x_n, s_n) \in F(m,n,i)} \mathbf{E}\{p_n \times \hat{s}_n \\ & - c(m, x_n, \hat{x}_n, n) - h(n, i_n) + \gamma f(m, n+1, i_n + \hat{x}_n - \hat{s}_n)\}, \\ & \text{for all } m \in M, \ i, \text{ and } n = 1, 2, \ldots, N-1 \\ & \text{where, } f(m, N, i) = 0 \text{ for all } m \in M, \text{ and } i \end{aligned}$$

To solve the above stochastic problem approximately, we use an approach similar to the deterministic problem. However, the implementation is significantly more complicated due to the fact that the optimal decisions are now policies.

2.3 Solution by SFP: Sampling Policies

We use the same four players to solve the stochastic problem as in the deterministic case, i.e., CI, RM, PP and S. The variable transformations are also similar. However, instead of using α_n and β_n for the PP and the S players, we use $\alpha(n, i)$ and $\beta(n, i)$. This allows the decision variables to explicitly depend on the inventory. In particular, for a specific value of plant capacity m, we write

$$\begin{aligned} p(n, i) &\in \mathbf{P} \\ x(n, i) &= \alpha(n, i) \times m \\ s(n, i) &= \beta(n, i) \times \min\{i + x(n, i), \bar{D}(p(n, i))\} \end{aligned}$$

Further, the actual state transition after making decisions $p(n, i)$, $\alpha(n, i)$ and $\beta(n, i)$ can be defined as follows. Let

$$\begin{aligned} \hat{x}(n, i) &= \min\{x(n, i), l_k m\} \\ \hat{s}(n, i) &= \min\{i + \hat{x}(n, i), s(n, i), d_j(p(n, i))\} \\ \hat{i}_{n+1} &= i + \hat{x}(n, i) - \hat{s}(n, i). \end{aligned}$$

The state is transformed from (m, n, i) to $(m, n+1, \hat{i}_{n+1})$ with probability $P_{l_k} Q_j$. Now we informally describe how SFP attempts to solve problem (PS).

Similar to the deterministic case, each of the four players CI, RM, PP and S samples one element from its history of past plays. An important point to note is that although an element in player CI's history is just a real number as in the deterministic case, an element in any of the other three players' history is a two dimensional matrix. The four players solve one best response problem each at every iteration. Again, the best response of the CI player is a real number whereas for the other three players it is a policy (stored in a two dimensional matrix). These best responses are added to the history of past plays before beginning the next iteration. In order to precisely understand the progress of this algorithm, let $\hat{m}$ be a capacity sampled by the CI player at a particular iteration and $\hat{p}$, $\hat{\alpha}$ and $\hat{\beta}$ be the matrices sampled by the RM, PP and S players respectively. Let H_{CI} be the vector of the history of the CI player's plays. In particular, $H_{CI}(i)$ is the real number that is the best response of the CI player in the i^{th} iteration for $i = 1, 2, 3, \ldots$. Let H_{RM} be the three dimensional matrix that stores the history of the RM player's plays. More specifically, $H_{RM}(i)$ is the best response policy (which is a two dimensional matrix) of the RM player in the i^{th} iteration. Similarly, for H_{PP} and H_S.

Algorithm 2.2. SFP for Optimizing A Stochastic Manufacturing Operation

1. *Begin with a randomly chosen action m^0, and randomly chosen policies p^0, α^0, and β^0 in the histories of the CI, RM, PP, and S players respectively. That is, $H_{CI}(0) = m^0$, $H_{RM}(0) = p^0$, $H_{PP}(0) = \alpha^0$ and $H_S(0) = \beta^0$. Set the iteration counter $k = 1$.*

2. *In iteration k, sample four numbers ci, rm, pp,and s uniformly randomly and independently from integers $0, 1, \ldots, k-1$. Let $\hat{m} = H_{CI}(ci)$ be the sampled action for player CI. Similarly, let $\hat{p} = H_{RM}(rm)$, $\hat{\alpha} = H_{PP}(pp)$, and $\hat{\beta} = H_S(s)$ be the sampled policies for the RM, PP and S players respectively.*

3. *Compute the best responses m^*, p^*, α^*, and β^* by solving the best response problems for the CI, RM, PP, and S players defined in section 2.4 below.*

4. *Add the best responses m^*, p^*, α^*, and β^* to the histories of the CI, RM, PP, and S players respectively. That is, set $H_{CI}(k) = m^*$, $H_{RM}(k) = p^*$, $H_{PP}(k) = \alpha^*$, and $H_S(k) = \beta^*$. Advance the iteration counter to $k + 1$.*

5. *Continue steps 2-4 above until a stopping criterion is met.*

2.4 The Best Response Problems for the Stochastic Case

As mentioned earlier, some of the best response problems are more complicated in the stochastic case as compared to the deterministic one. In particular, the best response of the RM, PP, and S players is a policy that depends on the period and the inventory. We start out with the best response problem of the CI player, which is a simple one variable maximization problem. As above, suppose that in a particular iteration the sampled actions (or policies) of the four players are $\hat{m}$, $\hat{p}$, $\hat{\alpha}$ and $\hat{\beta}$.

2.4.1 CI Best Response

Let $g(m, \hat{p}, \hat{\alpha}, \hat{\beta})$ be the expected value of profit generated when the plant capacity is $m \in \mathbf{M}$ and the RM, PP, and S players use policies $\hat{p}$, $\hat{\alpha}$ and $\hat{\beta}$ respectively. The CI best response problem is defined as

$$m^* = \arg\max_{m \in \mathbf{M}} \; g(m, \hat{p}, \hat{\alpha}, \hat{\beta})$$

2.4.2 RM Best Response

Note that the policies of the CI,PP and S players are fixed at $\hat{m}$, $\hat{\alpha}$ and $\hat{\beta}$ respectively. Analogous to the deterministic case, since the capacity is fixed at its sampled value $\hat{m}$, it is no longer a part of the state. Therefore, the state is given by (n, i). If the RM player makes a decision $p(n, i)$ in this state, the state transformation equations can be written as

$$\begin{aligned} \hat{x}(n,i) &= \min\{\hat{\alpha}(n,i) \times \hat{m}, l_k \hat{m}\} \\ \hat{s}(n,i) = \min\{i + \hat{x}(n,i), \hat{\beta}(n,i) \times \min\{i + \hat{\alpha}(n,i) \times \hat{m}, \bar{D}(p(n,i))\}, d_j(p(n,i))\} \\ i_{n+1} &= i + \hat{x}(n,i) - \hat{s}(n,i), \end{aligned}$$

where the state is transformed from (n, i) to $(n+1, i_{n+1})$ with probability $P_{l_k} Q_j$. The reward received for making this transformation is given by

$$R(p(n,i)) = p(n,i) \times s(n,i) - c(\hat{m}, \hat{\alpha}(n,i) \times \hat{m}, \hat{x}(n,i), n) - h(n+1, i_{n+1})$$

Now let $g_{rm}(n, i)$ be the maximum expected profit generated when the revenue management player starts period n with inventory i on hand and continues optimally until the end of period N. Then, the revenue management best response problem can be written as

$$g_{rm}(n,i) = \max_{p(n,i) \in \mathbf{P}} \mathbf{E}\{R(p(n,i)) + \gamma g_{rm}(n+1, i_{n+1})\}$$

2.4.3 PP Best Response

We will not write the details of this best response problem here. It is very similar to the RM best response problem with the only exception that the policies of the CI, RM, and S players are now fixed at $\hat{m}$, $\hat{p}$ and $\hat{\beta}$, whereas the objective is to find an optimal policy α for the PP player.

2.4.4 S Best Response

Again, we will not repeat the details here. The best response problem is similar to the RM best response above with the only exception that the policies of the CI, RM, and PP players are now fixed at $\hat{m}$, $\hat{p}$ and $\hat{\alpha}$, whereas the objective is to find an optimal policy β for the S player.

In the next section, we present detailed numerical results illustrating performance of the above approach on a stochastic manufacturing optimization problem.

3 Vehicle Manufacturing: A Numerical Case Study

In this section, we report detailed results of numerical experiments done using real world data from a major company in the automotive sector.

3.1 Problem Data

Recall from section 2 that the pieces of data required are the plant building costs, stochastic price-demand functions, production costs, inventory costs, and plant reliability data. The general trend in the cost data as plotted in figure 4 was established by discussions with employees of a leading automobile manufacturing corporation. The actual numbers shown in this figure have been purposefully distorted for confidentiality concerns.

- The planning horizon was assumed to be $N = 10$ periods.
- Plant building cost was assumed to be a function of the plant capacity. The cost was amortized over a finite horizon of length $N = 10$, i.e., the horizon used for the optimization problem.
- The price-demand functions were assumed to be exponential, i.e., of the form $D(p) = e^a p^b$. In order to introduce stochasticity, we parameterized demand functions $D_i(\cdot)$ in the set $\mathbf{D}$ by parameters a_i and b_i. In particular, we included three possible demand functions that indicate low demand, normal demand, and high demand. This was achieved by setting $|\mathbf{D}| = 3$, and $(a_i, b_i) \in \{(48.5573, -4.5076), (49.0478, -4.5076), (49.5383, -4.5076)\}$. In each period, the actual realized demand is chosen from one of these three functions with equal probability.
- The variable production cost per vehicle was assumed to decrease with increasing plant capacity due to economies of scale. It was also assumed to be linear in the number of units produced and stationary across time periods.
- The inventory holding cost per vehicle at the end of a period was assumed to be 20 percent of the unit production cost in that period.
- The plant reliability value ρ is assumed to be an element of the set $\mathbf{L} = \{0.6, 0.66, 0.7, 0.74, 0.8\}$. One of these values is selected with equal probability in each period.
- The time value of money was ignored, i.e. the discount factor γ was set to 1.
- α and β were assumed to take values in the set $\{0, 1/300, 2/300, \ldots, 1\}$, i.e., $\epsilon = \delta = 1/300$. To ensure fair comparison between SFP and other alternatives (e.g., a standard MDP solver), we make all solution procedures search within the space of $\mathbf{M} \times \mathbf{A}^N \times \mathbf{B}^N \times \mathbf{P}$.
- 20 iterations of SFP were run on a Pentium 4 (2.8 GHz), 1 GB RAM machine with RedHat Linux operating system.

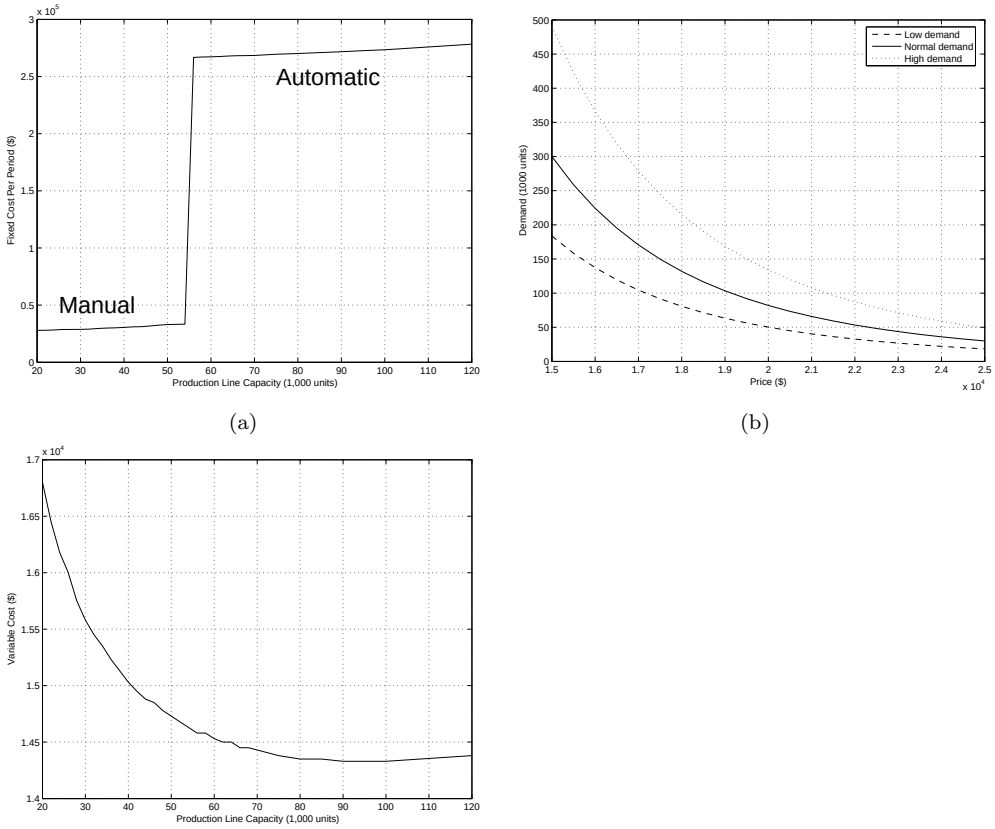

Figure 4: Important problem data: (a) Production line building cost, paid by period, as a function of capacity. (b) Demand as a function of price. (c) Variable cost as a function of capacity.

3.2 Numerical Results

In our numerical experiments, we looked at the expected values achieved by the policies obtained by both the SFP solver and a standard MDP solver. Also, we looked at computational time required to obtain above policies in both solvers. Although not mentioned earlier, SFP is numerically used as a search algorithm, and a best value and its associated policy will be kept and updated throughout algorithm execution. In our implementation, the best value and associated solution are updated at the end of each best reply evaluation in each iteration.

The comparison results are shown in table 1. Note that for the MDP solver, enumerating all possible capacities cannot be finished in a reasonable amount of time. By observing the fact that the computational effort is identical for each capacity, we can solve the problem for one specific capacity, record required time, and use it to estimate the total required time. However, since we also care about global optimum, we would like to choose this specific capacity wisely, so that while finding execution time required for one capacity, we can also find global optimum. We achieve this by choosing a specific numeric case, which by examining problem data, we see that the profit would monotonically increase in capacity

up to certain point, after which the profit would drop significantly and never climb back to comparable level. The MDP is then solved at this capacity, which from the problem data, we know a priori that it will be the optimal capacity. In our experiment, 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 required 13.1 minutes or was approximately 14,778 times faster than the (estimated) global solver execution time, and the quality of the solution was within 3% of the optimum. The evolution of best values against iterations for the SFP solver is plotted in figure 5. As

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

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.

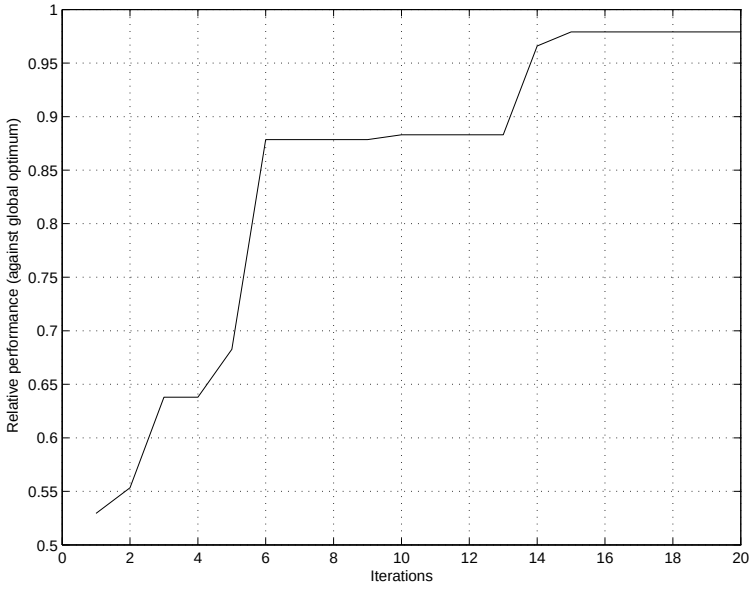

Figure 5: Best values plotted against iterations, for the SFP solver.

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 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.

3.3 Obtaining Managerial Insights via Optimization

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. Note that the

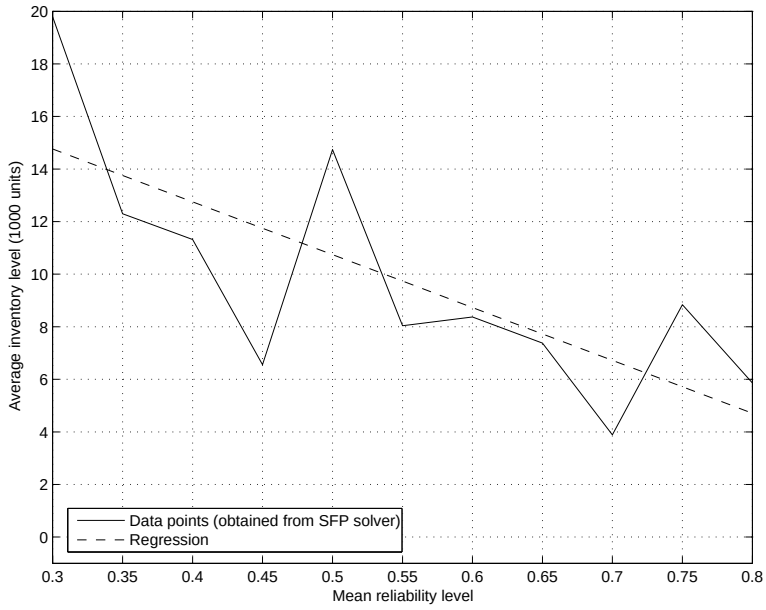

Figure 6: Average inventory levels versus mean reliability levels.

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.

4 Conclusions

In today's competitive environment in manufacturing operations, 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 Sampled Fictitious Play. SFP has emerged as an effective discrete optimization heuristic for unconstrained problems in the recent past. However, to apply it to our manufacturing optimization model, we extended it to handle constraints. This was done by applying a variable transformation to the original dynamic programming formulation to convert it into a finite game in strategic form, making it amenable to SFP. Although illustrated on a specific formulation in this paper for simplicity and concreteness, we believe that our approach can be generalized to a class of sequential decision problems. In that sense, this approach may be viewed as a heuristic for approximate dynamic programming.

We considered a case study from the automotive manufacturing sector to perform numerical experiments. SFP was able to find near optimal solutions about four orders of magnitude faster than conventional dynamic programming methods when applied to the vehicle manufacturing problem. Since SFP can be parallelized easily, this performance can be further improved. The most important utility of this approach lies in its ability to quickly solve multiple scenarios. 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 data-driven rules of thumb to guide managerial decisions in complex manufacturing operations.

Acknowledgement

This work was supported in part by the National Science Foundation under NSF grants DMI-0422752 and DMI-0322114 and by the General Motors Collaborative Research Laboratory in Advanced Vehicle Manufacturing at the University of Michigan.

A Mathematical Background

In this section we briefly review some basic concepts in game theory and formally state the FP and SFP algorithms. We also present two important convergence theorems regarding these algorithms from the literature.

A.1 Basics of Game Theory

For a detailed description of important concepts in game theory, please see Fudenberg and Tirole [6]. Following Monderer and Shapley [12], and Lambert et al, [11] let Γ be a finite game in strategic form, i.e., a game with finite number of players, with each player's action set being finite and non-empty, and the payoff function being well defined for all joint actions. Let $\mathcal{N} = \{1, 2, \ldots, |\mathcal{N}|\}$ be the set of players in Γ. For each player $i \in \mathcal{N}$, let $\mathcal{Y}^i$ be the finite set of feasible actions. We use y^i to denote an element of $\mathcal{Y}^i$. Let $\mathcal{Y} = Y^1 \times Y^2 \times \ldots Y^{|\mathcal{N}|}$ be the set of feasible joint actions by all the players. We use y to denote an element of $\mathcal{Y}$. The payoff function of player $i \in \mathcal{N}$ is denoted by $u^i : \mathcal{Y}^i \to \Re$.

For player $i \in \mathcal{N}$, let Δ^i be the set of mixed strategies, i.e.,

$$\Delta^i = \{f^i : \mathcal{Y}^i \to [0, 1] : \sum_{y^i \in \mathcal{Y}^i} f^i(y^i) = 1\}$$

Let $\Delta = \Delta^1 \times \Delta^2 \times \ldots \Delta^{|\mathcal{N}|}$. For player $i \in \mathcal{N}$, we extend u^i to be its payoff function in the mixed extension of Γ. That is, for any $f \in \Delta$,

$$u^i(f) = u^i(f^1, f^2, \ldots, f^{|\mathcal{N}|}) = \sum_{y \in \mathcal{Y}} u^i(y^1, y^2, \ldots, y^{|\mathcal{N}|}) f^1(y^1) f^2(y^2) \ldots f^{|\mathcal{N}|}(y^{|\mathcal{N}|})$$

where we have assumed that players choose their actions independently.

For $g \in \Delta$ and $f^i \in \Delta^i$, we use (f^i, g^{-i}) to denote $(g^1, g^2, \ldots, g^{i-1}, f^i, g^{i+1}, \ldots, g^{|\mathcal{N}|})$, which is a joint, mixed strategy. We say that g is a Nash equilibrium if

$$u^i(g) \geq u^i(f^i, g^{-i}) \text{ for all } f^i \in \Delta^i, \text{ for all } i \in \mathcal{N}.$$

Nash [13] proved that every finite game in strategic form has a Nash equilibrium as defined here in probably one of the most famous papers in 20^{th} century mathematics. The term *equilibrium* is often used to refer to a Nash equilibrium.

A belief path $\{f(t)\}_{t=1}^{\infty}$ is a sequence in Δ. We say that the belief path $\{f(t)\}_{t=1}^{\infty}$ converges to equilibrium if every accumulation point of $\{f(t)\}_{t=1}^{\infty}$ is an equilibrium. That is, a belief path that converges to an equilibrium is eventually arbitrarily close (in Eucledean norm in an appropriately defined Eucledean space containing Δ) to some equilibrium of the mixed extension of Γ.

A.2 The Fictitious Play Algorithm

Although the existence of an equilibrium was established by Nash for finite games in strategic form, finding one is hard in general. Fictitious Play was proposed to approximate Nash

equilibria of finite games in strategic form [14]. This algorithm can be informally described as follows. Players start out with an arbitrary initial play. At any iteration of the algorithm, the history of past plays defines an associated belief path, and each player assumes that the other players are using mixed strategies as represented by their respective belief paths. In every iteration, each player computes its pure strategy best response to these mixed strategies. The belief path is updated at the end of iteration by including the latest best response in it. In order to formally describe the FP algorithm, we first need some notation and background.

A.2.1 Notation and Background

We would like to mention that parts of this subsection are technical, and the reader may skip them and move forward to algorithm A.1 directly. A path in $\mathcal{Y}$ is a sequence $\{y(t)\}_{t=1}^{\infty}$ of elements of $\mathcal{Y}$. A belief path $\{f(t)\}_{t=1}^{\infty}$ can be naturally associated with a path $\{y(t)\}_{t=1}^{\infty}$ by letting $f_y(t) = \frac{1}{t}\sum_{s=1}^{t} y(s)$ for every $t \geq 1$, where we view $y(s)$ as an element of Δ. For any integer $k \geq 0$, it is not hard to see that

$$f_y(t+k) = f_y(t) + \frac{1}{t+k}\sum_{s=t+1}^{t+k}(y(s) - f_y(t))$$

We say that the path $\{y(t)\}_{t=1}^{\infty}$ converges in belief to equilibrium if the associated belief path $\{f(t)\}_{t=1}^{\infty}$ converges to equilibrium. We now formally define a fictitious play process. For $i \in \mathcal{N}$ and for $f \in \Delta$, let $v^i(f) = \max\{u^i(g^i, f^{-i} : g^i \in \Delta^i\}$. That is, $v^i(f)$ is the payoff of player i's best response to other players' strategies f^{-i}. Observe from the definition of $u^i(f)$ that $v^i(f)$ can always be attained by an extreme point of Δ^i, i.e., $\max\{u^i(g^i, f^{-i} : g^i \in \Delta^i\} = \max\{u^i(y^i, f^{-i}) : y^i \in \mathcal{Y}^i\}$. A path $\{y(t)\}_{t=1}^{\infty}$ is a fictitious play process if for every $i \in \mathcal{N}$,

$$u^i(y^i(t+1), f_y^{-i}(t)) = v^i(f_y(t)) \text{ for every } t \geq 1.$$

Thus, $y^i(t+1)$ is a pure strategy best response of player i to the mixed strategies of the other players as represented by the beliefs $f_y^{-i}(t)$. The formal procedure of the FP algorithm defined below is taken from Lambert et al [11].

Algorithm A.1. Formal Fictitious Play Algorithm
Initialization: *Set* $t = 1$ *and select* $y(1) \in \mathcal{Y}$ *arbitrarily; set* $f_y(1) = y(1)$.
Iteration $t \geq 1$*: Given* $f_y(t)$*, find*

$$y^i(t+1) \in \underset{y^i \in \mathcal{Y}^i}{argmax}\ u^i(y^i, f_y^{-i}(t)),\ i = 1, 2, \ldots, |\mathcal{N}| \tag{4}$$

Set $f_y(t+1) = f_y(t) + \frac{1}{t+1}(y(t+1) - f_y(t))$ *and increment* t *by* 1

Note that the algorithm generates a fictitious play process as defined in section A.2.1. The above algorithm may not converge to an equilibrium for a general finite game in strategic form. However, Monderer and Shapley [12] proved that it does indeed converge to an equilibrium if the players have common interests. More formally,

Theorem A.2. *Let Γ be a finite game in strategic form with identical payoff functions $u^1(\cdot) = u^2(\cdot) = \ldots = u^{|\mathcal{N}|}(\cdot) = u(\cdot)$. Then, any fictitious play process converges in beliefs to equilibrium.*

A naive implementation of the above algorithm may be prohibitively expensive because of the challenging best response calculation for each player. In particular, if the vector $f_y(t)$ is strictly positive, computing $u^i(y^i, f_y(t)^{-i})$ in equation (4) can require evaluating the payoff functions for all combination of pure strategies. The concept of Sampled Fictitious Play was formally introduced to avoid this difficulty (Lambert et al [11]).

A.3 The Sampled Fictitious Play Algorithm

In view of theorem A.2, we will only focus on games with identical interest from now on. The key to Sampled Fictitious Play is to realize that the components of the belief vector $f_y(t)$ can be viewed as components of a probability distribution over the pure strategies of the players. Thus, the best response problem (4) for a game of identical payoff function $u(\cdot)$ can be defined as maximizing an expectation, i.e. $\max_{y^i \in \mathcal{Y}^i} \{E_{Y^{-i}}(u(y^i, Y^{-i}))\}$, where Y^{-i} is a random vector whose j^{th} component, Y^j, $j \neq i$, has probability distribution $f_y^j(t)$. In the FP algorithm, we maximize this expectation while solving the best response problem for each player. Sampled Fictitious Play, as the name suggests, approximates this expectation by a sample mean at each iteration. The sample size is increased at an appropriate rate with the iteration counter to ensure convergence. To define this formally, we need some more notation. In particular, let $Y_j^{-i}(t)$, $j = 1, 2, \ldots, k$ be an i.i.d. sample of size k of random vectors with distribution $f_y^{-i}(t)$. Let $\bar{U}_k^i(y^i, f_y^{-i}(t))$ be the sample mean (with sample size k) of player i's payoff when using strategy y^i, i.e.,

$$\bar{U}_k^i(y^i, f_y^{-i}(t)) = \sum_{j=1}^{k} \frac{u(y^i, Y_j^{-i}(t))}{k}$$

It is important to realize that the above sample mean is a random variable. In an implementation of SFP, the sample size is increased with iterations, thus we denote the sample size at the t^{th} iteration by k_t. Moreover, we use $\bar{u}_{k_t}^i(y^i, f_y^{-i}(t))$ to denote a realization of the random sample mean $\bar{U}_k^i(y^i, f_y^{-i}(t))$. We use this realization in the best reply computation. The formal procedure is defined below.

Algorithm A.3. Formal Sampled Fictitious Play Algorithm
Initialization: *Set $t = 1$ and select $y(1) \in \mathcal{Y}$ arbitrarily; set $f_y(1) = y(1)$.*
Iteration *$t \geq 1$: Given $f_y(t)$, select a sample size $k_t \geq 1$, and draw i.i.d. random samples $Y_j(t)$, $j = 1, 2, \ldots, k_t$, from the distribution $f_y(t)$. Find*

$$y^i(t+1) \in \underset{y^i \in \mathcal{Y}^i}{argmax}\ \bar{u}_{k_t}(y^i, f_y^{-i}(t)),\ i = 1, 2, \ldots, |\mathcal{N}| \tag{5}$$

Set $f_y(t+1) = f_y(t) + \frac{1}{t+1}(y(t+1) - f_y(t))$ and increment t by 1

A key feature of SFP is that the sampled mixed strategy has a limited number of positive components, making the best reply computation significantly easier than FP. Observe that

(even with deterministic tie breaking) the sampled fictitious play process $\{y(t)\}_{t=1}^{\infty}$ is a stochastic process. The limiting behavior of the (stochastic) belief path associated with this process was formalized by Lambert et al [11], and is essentially an extension of theorem A.2.

Theorem A.4. *Let Γ be a finite game in strategic form with identical payoff functions $u^1(\cdot) = u^2(\cdot) = \ldots = u^{|\mathcal{N}|}(\cdot) = u(\cdot)$. Then every sampled fictitious play process $y(t)$ with sample size $k_t = \lceil Ct^{\beta} \rceil$ for $\beta > 1/2$, $C > 0$ converges in beliefs to an equilibrium with probability 1.*

There are some important points regarding the above theorem. It guarantees convergence to an equilibrium only in the limit with probability one. Further, the accumulation point may be a mixed strategy equilibrium with no relation to an optimal (pure strategy) equilibrium. Therefore, when applied to optimization problems of the form (1), SFP remains a heuristic. In practice, a sample size of one is used while implementing SFP, making it extremely simple to implement in many cases. However, no limiting behavior results are known for SFP with sample size one. While using SFP for obtaining near optimal solutions of discrete optimization problems, we usually only keep track of the best joint action observed so far.

References

[1] Bitran, G., and Caldentey, R., An Overview of Pricing Models for Revenue Management, *Manufacturing & Service Operations Management*, 5(3), 203-229, (2003).

[2] Chan, L., Shen, Z., Simchi-Levi, D., Swann, J.,Coordination of Pricing and Inventory Decisions: A Survey and Classication. D. Simchi-Levi, S. Wu and Z. Shen, ed. Handbook of Quantitative Supply Chain Analysis: Modeling in the E-Business Era. Kluwer Academic Publishers, 335-392, (2004).

[3] Chen, X. D., and Simchi-Levi, D., Coordinating Inventory Control and Pricing Strategies with Random Demand and Fixed Ordering Costs: The Finite Horizon Case, *Operations Research*, 52(6), 887-896, (2003).

[4] Cheng, S. F., Epelman, M., and Smith, R. L., CoSign: A Fictitious Play Algorithm for Coordinated Traffic Signal Control, Technical Report TR04-08, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor.

[5] Federgruen, A. A., and Heching, A., Combined Pricing and Inventory Control Under Uncertainty, *Operations Research*, 47(3), 454-475, (1999).

[6] Fudenberg, D., and Tirole, J., Game Theory, MIT Press, (1991).

[7] Gallego, G., and van Ryzin, G., Optimal Dynamic Pricing of Inventories with Stochastic Demand over Finite Horizons, *Management Science*, 40(8), 999-1020, (1994).

[8] Garcia, A., Reaume, D., and Smith, R. L., Fictitious Play for Finding System Optimal Routings in Dynamic Traffic Networks, *Transportation Research C*, 34 (2), 146-157, (2000).

[9] Kocabiyikoglu, A., and Popescu, I., Joint Pricing and Revenue Management with General Stochastic Demand, Technical Report, Decision Sciences Area, INSEAD, France, (2005).

[10] Lambert, T. J. and Wang, H., Fictitious Play Approach to a Mobile Unit Situation Awareness Problem, Technical Report, University of Michigan, (2003).

[11] Lambert, T. J., Epelman, M., and Smith, R. L., A Fictitious Play Approach to Large-Scale Optimization, *Operations Research*, 53 (3), 477-489, (2005).

[12] Monderer, D. and Shapley, L. S., Fictitious Play Property for Games with Identical Interests, *Journal of Economic Theory*, 68 (1), 258-265, (1996).

[13] Nash, J., Equilibrium Points in n-person Games, *Proceedings of the National Academy of Sciences*, 36, 48-49, (1950).

[14] Robinson, J., An iterative method for solving a game, *Annals of Mathematics*, 54, 296-301, (1951).