An Application of EDA and GA to Dynamic Pricing

Siddhartha Shakya
Intelligent Systems Research Centre, BT Group Chief Technology Office
Adastral Park, Ipswich
IP5 3RE, UK
sid.shakya@bt.com

Fernando Oliveira
Warwick Business School
University of Warwick
Coventry, CV4 7AL, UK
Fernando.Oliveira@wbs.ac.uk

Gilbert Owusu
Intelligent Systems Research Centre, BT Group Chief Technology Office
Adastral Park, Ipswich
IP5 3RE, UK
gilbert.owusu@bt.com

ABSTRACT

E-commerce has transformed the way firms develop their pricing strategies, producing shift away from fixed pricing to dynamic pricing. In this paper, we use two different Estimation of distribution algorithms (EDAs), a Genetic Algorithm (GA) and a Simulated Annealing (SA) algorithm for solving two different dynamic pricing models. Promising results were obtained for an EDA confirming its suitability for resource management in the proposed model. Our analysis gives interesting insights into the application of population based optimization techniques for dynamic pricing.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search
; G.3 [**Probability and statistics**]: Probabilistic algorithms, Stochastic processes

General Terms

Algorithms, Management, Performance, Design, Economics

Keywords

Estimation of Distribution Algorithms, Dynamic Pricing, Evolutionary Computation, Resource Management

1. INTRODUCTION

E-commerce has transformed the way firms price their products and interact with their customers. The increasingly dynamic nature of the e-commerce has produced a shift away from fixed pricing to dynamic pricing [3]. The basic idea of dynamic pricing (also called flexible pricing [10]) is for a firm to adjust the price of its products or services, online, as a function of its perceived demand at different times for the different price levels. Traditionally, dynamic pricing strategies have been applied in (and in fact developed by the) service industries (and industries that produce *perishable* goods). For example, the airline industry uses dynamic pricing to decide which fare class should be open and which should be closed in order to maximize their sales, and gain profit from seats which otherwise could be left unused. Restaurants use dynamic pricing to decide what portion of tables should be booked in advanced and what portion should be kept for walk in customers. Hotels use dynamic pricing to adjust the room rate depending upon demand and also to decide on how much to overbook. More recently, dynamic pricing is also being used in other non-traditional domains, such as, supply chain management, and planning and manufacturing of *non-perishable* products. A significant increase in profit has been reported by companies implementing dynamic pricing strategies. This resulted in increasing interest in this area both by academics and practitioners.

In this paper, we investigate the implications of dynamic pricing for maximizing a firm's profit and improve resource management. For this purpose, we extend the dynamic pricing model presented in [11] and implement several different Evolutionary Algorithms (EA) [1] to solve it. EA have been successfully applied in various search and optimization problems. They are inspired by Darwin's theory of evolution and use the concept of natural selection and random variation to evolve a better solution to the problem. The idea is to put pressure on the evolution of high quality solutions by means of selection, and at the same time explore more of the search space by means of variation. In particular, Genetic Algorithm (GA) [5] use crossover and mutation approach to variation. In contrast, Estimation of Distribution Algorithm (EDA) [9][8] use probabilistic approach to variation, where a probabilistic model is built and sampled to generate new solutions. EDAs are being increasingly applied to real-world optimization problems and are often reported to perform better than the traditional GAs [6][8][16][17]. This paper investigates both EDAs and GAs to solve dynamic pricing problems.

The three main objectives of this paper are:

1. To show how dynamic pricing can be used for resource management.

2. To study the performance of evolutionary techniques in resource management via dynamic pricing

3. To extend the application area of EA, and in particular, EDA, to dynamic pricing problems

The rest of the paper is organized as follows. Section 2 describes the motivation for this paper and describes how resource management can be improved by means of dynamic pricing. Section 3 derives a mathematical model for dynamic pricing. Section 4 describes how an EA can be used to solve this model and presents a penalty based approach to handling constraints involved in the model. Section 5 presents the experimental result on the performance of implemented EAs for this model. It also analyses the best solution found by these algorithm for both short-term and long-term profit. Finally, section 7 highlights future direction to this work and concludes the paper.

2. DYNAMIC PRICING FOR RESOURCE MANAGEMENT

This work was motivated by an automated resource management system, [13][21], which manages access services and provides telecommunication service to its customers. Resource management can be loosely defined as the effective workforce utilization for a given calendarised work demand profile, while minimizing and/or maximizing a set of constraints such as quality of service targets, conflict resolution schemes, such as overtime and borrowing additional workforce [12]. More formally, workforce management seeks to answer these three questions: what skills does our workforce have, where are resources located and when are they available to work. The system described in [11] integrates various Artificial Intelligence and Operational Research techniques in order to forecast demand for specific products and services at regional level, and to optimize the allocation of resources to each one of the region. Furthermore, the system incorporates a multi-agent co-ordination problem that aims to improve balancing of resources between different regions [20]. In this regard, the work presented in this paper focuses on investigating the use of dynamic pricing for improving resource management. The aim is to increase profit by optimizing the use of resources at any given time period during the life cycle of a product (or service). For this purpose, and based on [11], a dynamic pricing model has been developed which shows how the interaction between demand behavior, pricing policies and available resources interact in order to extract additional value from limited resources. Such a model can be used to analyze the effect of dynamic pricing on resource management, and ultimately on overall profit from a product by implementing different prices at different times, reducing congestion time and charging the opportunity cost of using a given resources, promoting demand transfer from congested to low demand period, developing a strategic pricing algorithm for new products and simulating the product lifecycle to analyze how price of a given product changes over its life cycle.

In particular we use dynamic pricing for two types of analysis. a. *Short-term analysis* - This can be used as a tool to take advantage of the dynamics of demand throughout a week, or even during a day. b. *Long-term analysis* - This is to model the long-term implications of short-term pricing and investment policies with the view to maximize the long-term profit from a product. Depending upon the nature of the product (or service) developed, and their expected demand behavior, a company has to choose between short-term (for example, weeks or days) profits or long-term (for example month or year) profits. In next section we describe a dynamic pricing model that can be used for analyzing both short-term and long-term profit from a product.

3. MODELLING DYNAMIC PRICING

The model presented in this section extends [11] to a stochastic setting. In this section we use the following notation:

N – Number of periods in planning horizon
t – Any given period in the planning horizon
Q_t – Number of jobs produced (demand) at period
P_t – Average price of a job (product) at period
C_t – Cost of producing one extra job (product) at period
Π – Total profit during the entire planning horizon

Expected total profit earned during the planning horizon ($E(\Pi)$) can be modeled as

$$E(\Pi) = \sum_{t=1}^{N} \left(P_t E(Q_t) - C_t E(Q_t)\right) \qquad (1)$$

Where, $E(Q_t)$ is the expected demand for jobs (which is equivalent to expected number of jobs) in period t, $P_t E(Q_t)$ is the total revenue at period t , and $C_t E(Q_t)$ is the variable cost at t. Even though, aiming to maximize the expected value during the planning horizon, a firm does not know the elasticities of demand. Therefore, it needs to learn the optimal policy by interacting with the environment. The observed profit is, therefore, give by replacing the expected demand $E(Q_t)$ in (1) with Q_t. This requires the introduction of a term representing random error in profit due to random fluctuation in demand and costs. The resulting equation for the total observed profit is given by (2), where ϵ represents the stochastic shock following a normal distribution, $\epsilon \approx N(0, \sigma)$, in which σ represents the standard deviation.

$$\Pi = \sum_{t=1}^{N} \left(P_t Q_t - C_t Q_t\right)(1 + \epsilon) \qquad (2)$$

Demand for jobs (Q_t), in period t, depends on the job price (P_t) in period t as well as the other prices within the planning horizon. We represent this, for all $t = 1, ..., N$, as

$$Q_t = b_{0t} + b_{1t}P_1 + b_{2t}P_2 + ... + b_{tt}P_t + ... + b_{Nt}P_N \qquad (3)$$

where, b_{ij} are the parameters of the model representing sensitivity of demand at time j to the price at time i. Note that, for normal products and services, $b_{tt} < 0$, since for current period, the price has negative effect to demand.

Inversely, average price for jobs (P_t), in period t, depends on the demand of the job in that period as well as the other demands within the planning horizon, which can be modeled as (4) for all $t = 1, ..., N$.

$$P_t = a_{0t} + a_{1t}Q_1 + a_{2t}Q_2 + ... + a_{tt}Q_t + ... + a_{Nt}Q_N \qquad (4)$$

Hhere, a_{ij} represent the impact of demand at time i on the price at time j. Note that, for normal jobs, $a_{tt} < 0$. Equation (4) can be represented more compactly as (5).

$$P_t = a_{0t} + \sum_{j=1}^{N} a_{jt}Q_j \qquad t = 1, ..., N \qquad (5)$$

Substituting P_t from (5) to (2), we get the general model for the total profit (6).

$$\Pi = \sum_{t=1}^{N} \left[a_{0t}Q_t + \sum_{j=1}^{N} a_{jt}Q_jQ_t - C_tQ_t)\right](1 + \epsilon) \qquad (6)$$

Now let us define some additional constraints a firm needs to impose when defining its policy for pricing a given job (product, service). Following are two of the most common constraints.

a. *Available capacity constraints* - These are number of jobs that can be produced in a given period and regulates the resources such as number of workers or machines that should be used in a given period. Available capacity has the lower bound and upper bound. For all $t = 1, ..., N$

$$M_t \leq Q_t - \text{Lower bound for the capacity constraint}$$
$$K_t \geq Q_t - \text{Upper bound for the capacity constraint} \quad (7)$$

b. *Price caps constraints* - These are the prices of a job (product) produced in a given period and regulate the value to the costumers. Low price may suggest low value so there should also be the thresholds (lower and upper bounds). For all $t = 1, ..., N$

$$\overline{P}_t \leq P_t - \text{Lower bound for the capacity constraint}$$
$$\underline{P}_t \geq P_t - \text{Upper bound for the capacity constraint} \quad (8)$$

Given the upper bound and lower bound to both capacity constraint and price cap, and also the standard deviation, σ for ϵ representing the shock in demand, our goal is to maximize the total profit, i.e. maximize Π in the equation (6). Since equation (6) is nonlinear, this problem is a nonlinear stochastic optimization problem. Also, since the goal is to maximize equation (6), and at the same time satisfy the constraints defined in equation (7) and (8), this can be seen as a constrained optimization problem.

4. AN EA APPROACH TO DYNAMIC PRICING

A general constrained optimization problem can be defined as $max_x f(x), \quad x \in S \subset \mathbb{R}^n$ subject to the linear or nonlinear constraints $g_i(x) \leq 0, \quad i = 1, ..., m$. Here m is the total number of constraints. Since problem definition in EA, generally, does not consider the constraint part, it can be thought of as an unconstrained optimization algorithm [14]. In order to apply EA for the dynamic pricing model developed in previous section, we first need to find the way to handle constraints in EA.

One of the most popular ways to solving constrained optimization problems is by using a *penalty function*. The idea is to construct a function that penalizes the original objective function for violating the constraints in the model. In order to avoid the penalty, the algorithm tries to focus its search to more of the feasible part of the search space. Let us describe one such technique adopted from [14] for our problem. We define the penalty function as

$$F(x) = f(x) - h(k)H(x), \quad x \in S \subset \mathbb{R}^n \quad (9)$$

where, $f(x)$ is the original objective function (in our case it is defined by Π in equation (6)), $h(k)$ is a dynamically modified penalty value, where k is the algorithm's current iteration number, and $H(x)$ is a penalty factor, defined as

$$H(x) = \sum_{i=1}^{m} \theta(q_i(x))q_i(x)^{\gamma(q_i(x))} \quad (10)$$

Here, function $q_i(x)$ is a relative violated function of the constraints defined as

$$q_i(x) = max\{0, g_i(x)\}, \quad i = 1, ..., m \quad (11)$$

$\theta(q_i(x))$ is known as multi-stage assignment function; $\gamma(q_i(x))$ is the power of the penalty function and $g_i(x)$ are the constraints. In our case, $g_i(x) \leq 0$ are the constraints defined by equations (7) and (8) and can be re-written as following four sets for all $t = 1, ..., N$.

$$M_t - Q_t \leq 0, \;\; Q_t - K_t \leq 0, \;\; \underline{P}_t - P_t \leq 0, \;\; P_t - \overline{P} \leq 0 \quad (12)$$

The functions $h(k)$, $\theta(q_i(x))$ and $\gamma(q_i(x))$ are problem dependent function. For the purpose of our work, we set $h(k) = 2\sqrt{k}$; Also we set, $\theta(q_i(x)) = 10000$ if $q_i(x) < 0.001$, else $\theta(q_i(x)) = 15000$ if $q_i(x) < 0.1$, else $\theta(q_i(x)) = 20000$ if $q_i(x) < 1$, else $\theta(q_i(x)) = 30000$; Furthermore, we set $\gamma(q_i(x)) = 1$ if $q_i(x) < 1$, otherwise we set $\gamma(q_i(x)) = 2$.

4.1 Solution representation for EA

A solution, x, is represented as a set $Q = \{Q_1, Q_2, ..., Q_N\}$, where each Q_t is represented by a bit-string of length l. The total length of a bit-string solution, $x = x_1, x_2, ..., x_n$, where $x_i = \{-1, +1\}$, is therefore, equal to $n = l \times N$. Since, the range for each set of l bits representing Q_t is set from M_t to K_t, the constraints (7) are always satisfied. Therefore, the penalty factor in (10) is completely estimated from the set of constraints in (8). The goal of an algorithm is to maximize the penalty function defined in (9).

4.2 Overview of the used algorithms

We use two EDAs and a GA for solving this problem. They include Population Based Incremental Learning (PBIL) algorithm [2], Distribution Estimation using Markov Random Filed with direct sampling (DEUM$_d$) algorithm [19][18] and a GA [5]. We also find it interesting to test the performance of a non-population based algorithm known as Simulated Annealing (SA) [7] for this problem.

The two EDAs implemented here, PBIL and DEUM$_d$, both fall in the category of univariate EDA. This means the model of distribution used by them assumes each variable, x_i , in the solution $x = \{x_1, x_2, ..., x_n\}$ to be independent. In other word, they do not take into account any possible interaction between variables in the solution. Other categories of EDA include, bivariate EDA, assuming at most pair-wise interaction between variables, and multivariate EDA, assuming interaction between multiple variables [8][15][4][18]. Our motivation behind using univariate EDA for this problem is two fold: firstly, they are simple and, therefore, often quickly converges to optimum resulting in higher efficiency. This is particularly important since efficiency of an algorithm matters a lot in a dynamic environment. Secondly, the number of problems that has been shown to be solved by univariate EDA is surprisingly large. Let us describe the workflow of these algorithms.

PBIL

---

1. Initialize a probability vector $p = \{p_1, p_2, ..., p_n\}$ with each $p_i = 0.5$. Here, p_i represents the probability of x_i taking value 1 in the solution

2. Generate a population P consisting of M solutions by sampling probabilities in p

3. Select set D from P consisting of N best solutions

4. Estimate probabilities of $x_i = 1$, for each x_i, as
$$p(x_i = 1) = \frac{\sum_{x \in D, x_i = 1} x_i}{N}$$
5. Update each p_i in p using $p_i = p_i + \lambda(p(x_i = 1) - p_i)$. Here, $0 \leq \lambda \leq 1$ is a parameter of the algorithm known as the learning rate
6. Go to step 2 until termination criteria are meet

DEUM$_d$

---

1. Generate a population, P, consisting of M solutions
2. Select a set D from P consisting of N best solutions, where $N \leq M$.
3. For each solution, x, in D, build a linear equation of the form
$$\eta(F(x)) = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + ... + \alpha_n x_n$$
Where, function $\eta(F(x)) < 0$ is set to $-ln(F(x))$, for which $F(x)$, the fitness of the solution x, should be ≥ 1; $\alpha = \{\alpha_0, \alpha_1, \alpha_2, ..., \alpha_n\}$ are equation parameters.
4. Solve the build system of N equations to estimate α
5. Use α to estimate the distribution $p(x) = \prod_{i=1}^{n} p(x_i)$, where
$$p(x_i = 1) = \frac{1}{1 + e^{\beta \alpha_i}}, \quad p(x_i = -1) = \frac{1}{1 + e^{-\beta \alpha_i}}$$
Here, β (inverse temperature coefficient) is set to $\beta = g \cdot \tau$; g is current iteration of the algorithm and τ is the parameter known as the cooling rate
6. Generate M new solution by sampling $p(x)$ to replace P and go to step 2 until termination criteria are meet

---

GA

---

1. Generate a population P consisting of M solutions
2. Build a breeding pool by selecting N promising solutions from P using a selection strategy
3. Perform crossover on the breeding pool to generate the population of new solutions
4. Perform mutation on new solutions
5. Replace P by new solutions and go to step 2 until termination criteria are meet

---

SA

---

1. Randomly generate a solutions $x = \{x_1, x_2, ..., x_n\}$
2. For $i = 1$ to r do
   (a) Randomly mutate a variable in x to get x'
   (b) Set $\Delta F = F(x') - F(x)$
   (c) Set $x = x'$ with probability
   $$p(x') = \begin{cases} 1 & \text{if} \quad \Delta F \leq 0 \\ e^{-\Delta F / T} & \text{if} \quad \Delta F > 0 \end{cases}$$
   Where, temperature coefficient T was set to $T = 1/i \cdot \tau$; here, i is the current iteration and τ is the parameter of the algorithm called the cooling rate
3. Terminate with answer x.

---

5. EXPERIMENTS AND RESULTS

We perform three sets of experiment modelling three different scenarios for both short-term and long-term analysis. Also, for each set of experiment, we use three different setup for the ϵ, i.e. with $\sigma = \{0.0, 0.1, 0.2\}$, making the total number of experiment equal to $3 \times 2 \times 3 = 18$.

5.1 Parameterization of the model

For short-term analysis, we assume that the production for a given day is negative function of the price on that day and a positive function of the prices on other days of the week. More specifically, we assume that at any given time t: a) Production decreases by one unit for each pound increase in price; b) an increase in production in a given day reduces the demand during other days of the week. Further, the cost of an additional unit of production was assumed to be zero (all costs are fixed) and the minimum production for each day was also assumed to be zero. Moreover, it was assumed that demand is higher during first few days of the week. These are reflected in a_{jt} shown in Table 1.

Table 1: a_{jt} for all three short-term experiments

t	a_{0t}	a_{1t}	a_{2t}	a_{3t}	a_{4t}	a_{5t}	a_{6t}	a_{7t}
1	900.0	-1.0	0.1	0.1	0.1	0.1	0.1	0.1
2	800.0	0.0	-1.0	0.1	0.1	0.1	0.1	0.1
3	800.0	0.0	0.0	-1.0	0.1	0.1	0.1	0.1
4	700.0	0.0	0.0	0.0	-1.0	0.1	0.1	0.1
5	600.0	0.0	0.0	0.0	0.0	-1.0	0.1	0.1
6	500.0	0.0	0.0	0.0	0.0	0.0	-1.0	0.1
7	400.0	0.0	0.0	0.0	0.0	0.0	0.0	-1.0

Also, for experiment(scenario) 1, maximum production capacity was set to 1000 units and maximum price was set to 250/unit, for experiment 2, maximum production capacity was set to 1000 units and maximum price was set to 1000/unit, and for experiment 3, maximum production capacity was set to 300 units and maximum price was set to 1000/unit.

For long-term analysis, we assumed that the production (demand) in a year is a negative function of the average price in that year and positive function of the product during previous year. More specifically, we assumed that at any given time t: a) production decreases by one unit for each pound increase in price; b) the company keeps a given proportion of its customers from the previous year. These are reflected in Table 2 and 3 showing the setup for all a_{jt}. Further, in all three experiments, the cost of an additional unit production was assumed to be zero. For experiment 1, the maximum production capacity was set to 3000 units during the first 4 years and 6000 units during the last 3 years, no maximum price was set and it was assumed that the company keeps 50% of its customers from the previous year. For experiment 2, the maximum production capacity was set to 3000 units during the first 4 years and 6000 units during the last 3 years, no maximum price was set and it was assumed that the company keeps 90% of its customers from the previous year. For experiment 3, the maximum production capacity was set to 3000 units during all 7 years, no maximum price was set and it was assumed that the company keeps 90% of its customers from the previous year.

Table 2: a_{jt} for long-term experiments no 1

t	a_{0t}	a_{1t}	a_{2t}	a_{3t}	a_{4t}	a_{5t}	a_{6t}	a_{7t}
1	3000.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0
2	3000.0	0.5	-1.0	0.0	0.0	0.0	0.0	0.0
3	3000.0	0.0	0.5	-1.0	0.0	0.0	0.0	0.0
4	3000.0	0.0	0.0	0.5	-1.0	0.0	0.0	0.0
5	3000.0	0.0	0.0	0.0	0.5	-1.0	0.0	0.0
6	3000.0	0.0	0.0	0.0	0.0	0.5	-1.0	0.0
7	3000.0	0.0	0.0	0.0	0.0	0.0	0.5	-1.0

Table 3: a_{jt} for long-term experiments no 2 and 3

t	a_{0t}	a_{1t}	a_{2t}	a_{3t}	a_{4t}	a_{5t}	a_{6t}	a_{7t}
1	3000.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0
2	3000.0	0.9	-1.0	0.0	0.0	0.0	0.0	0.0
3	3000.0	0.0	0.9	-1.0	0.0	0.0	0.0	0.0
4	3000.0	0.0	0.0	0.9	-1.0	0.0	0.0	0.0
5	3000.0	0.0	0.0	0.0	0.9	-1.0	0.0	0.0
6	3000.0	0.0	0.0	0.0	0.0	0.9	-1.0	0.0
7	3000.0	0.0	0.0	0.0	0.0	0.0	0.9	-1.0

5.2 Parameterization of the algorithms

In order to achieve high accuracy of the results we set the size of the bit string representing each Q_t to $l = 25$. Therefore the solution length, n, was equal to $l \times n = 175$. In each execution, the algorithm was allowed to do a fixed number of fitness evaluations. This was equal to 600000 for PBIL, DEUM$_d$ and GA, and 800000 for SA. The number of fitness evaluation for PBIL, DEUM$_d$ and GA was calculated as the product of their population size, $M = 400$, and the maximum number of generations, $G = 1500$. For all experiments, the learning rate λ for PBIL was set to 0.02 except for the short-term experiment no 3 where it was set to 0.08. Similarly, cooling rate τ for DEUM$_d$ was set to 0.02 for all the experiment except for the short-term experiment no 3 where it was set to 0.08. For SA a very small cooling rate of 0.000001 was used except for short term experiment no 3 where 0.0001 was used. 10 best solutions were selected in PBIL and DEUM$_d$ for estimating the marginal probabilities. For GA, one point crossover was used with crossover probability set to 0.7. The mutation probability was set to 0.01 for all experiments except for the short-term experiment no 3 where it was set to 0.001. All of these setups were the best performing setups for each of these algorithms, and was taken from a range of experiments conducted with wide range of setups for each of these parameters.

5.3 Results

The total of 100 execution of each algorithm was done for each experiment and the best fitness found in each execution was recorded. The average fitness (avg), the standard deviation of fitness (stdev) and the best fitness (max) out of all 100 executions for each of the algorithms are shown in Table 4 for short-term experiment 1, in Table 5 for short-term experiment 2 and in Table 6 for short-term experiment 3 for all three setup of σ . Similarly, Table 7, 8 and 9 shows the results for long-term experiments 1, 2 and 3 respectively. Note that, for the purpose of comparing the performance of the algorithm, the final solution found by the algorithm was evaluated without the random term, ϵ. Therefore, the tables below shows the expected total profit.

Table 4: Results for all four algorithms on short-term experiment no 1 with $\sigma = \{0.0, 0.1, 0.2\}$

σ	metric	SA	PBIL	DEUM$_d$	GA
0.0	avg	886697.55	934409.54	933967.11	**942269.57**
	stdev	29687.75	8319.78	8494.54	**5730.97**
	max	939530.13	947627.26	947469.07	**949237.45**
0.1	avg	877995.45	892244.99	892804.00	**908335.54**
	stdev	30307.00	**6914.63**	9712.63	11135.55
	max	**936855.87**	907074.33	914572.74	928919.86
0.2	avg	868825.40	884495.07	887476.09	**887883.34**
	stdev	31033.90	**8738.90**	10050.51	17757.45
	max	916509.97	908851.80	910667.64	**928446.67**

Table 5: Results for all four algorithms on short-term experiment no 2 with $\sigma = \{0.0, 0.1, 0.2\}$

σ	metric	SA	PBIL	DEUM$_d$	GA
0.0	avg	1161672.37	**1173298.77**	1173193.22	1173270.60
	stdev	6711.47	**4.53**	346.25	163.37
	max	1173298.86	**1173299.32**	1173299.31	1173299.27
0.1	avg	1126195.39	**1162053.61**	1156611.35	1156102.40
	stdev	24283.17	**6753.12**	10676.73	11674.64
	max	1166935.56	1170886.09	**1172134.22**	1171525.33
0.2	avg	1094432.47	**1157138.68**	1147708.79	1146043.84
	stdev	36033.63	**10212.56**	17848.49	20233.47
	max	1156489.15	**1172168.56**	1170882.22	1170239.29

Table 6: Results for all four algorithms on short-term experiment no 3 with $\sigma = \{0.0, 0.1, 0.2\}$

σ	metric	SA	PBIL	DEUM$_d$	GA
0.0	avg	969099.53	969099.83	969099.81	969099.86
	stdev	0.21	0.19	0.19	0.18
	max	969099.86	969099.98	969099.98	969099.98
0.1	avg	887459.80	**958871.67**	944537.71	958674.23
	stdev	28612.27	**3994.84**	10669.16	4525.37
	max	945854.76	**966932.38**	966435.26	965325.75
0.2	avg	851703.55	**951920.12**	931934.03	947800.78
	stdev	38267.07	**5658.27**	18910.42	11765.29
	max	928493.17	**966543.93**	959712.62	962955.78

The value for the best performing algorithm is plotted in bold. As we can see, for all of the experiments, SA has the worst performance compared to other three algorithms with lower value for all three metrics. The performance of DEUM$_d$ and GA is somewhat comparable for all three metrics, with occasionally one outperforming another and vice versa. And finally, out of all four algorithms, PBIL has the best performance for most of the problems, both in terms of the average fitness and in terms of the best fitness. It also has the lowest standard deviation making it the most predictable algorithm out of all.

Effect of the shock- in- demand to the performance of the algorithm (profit): We can also see from these tables that the stochastic term $\epsilon \approx N(0, \sigma)$, representing the shock in demand, has negative effect to the overall profit. It can be observed that as σ increases the performance of the algorithm decreases for all three matrices. This effect can be explained as algorithms cannot properly optimize the function due to the random fluctuation in the profit during fitness evaluation. For example, in Figure 1, we present a typical graph showing, how the shock in demand affects the performance of the algorithm in terms of the suggest price (Figure 1(a)) and production (Figure 1(b)). Here, the best solution found by PBIL for three different setup of for the long-term experiment 1 are decoded and plotted. This can be interpreted as follows: when $\sigma = 0.0$, i.e, when no shock in demand is assumed, the curves for both prices and production is optimal (assuming algorithm gives the optimal solution). However, as σ increases, it can be observed that the resulting curves start to depart away from the optimum. This is an intuitive result and shows that, in environments, subject to high levels of uncertainty, optimal behavior is difficult to achieve.

Table 7: Results for all four algorithms on long-term experiment no 1 with $\sigma = \{0.0, 0.1, 0.2\}$

σ	metric	SA	PBIL	DEUM$_d$	GA
0.0	avg	27903293.85	**28206185.50**	28205885.77	28205990.61
	stdev	318069.78	**0.06**	809.71	742.40
	max	28206185.44	**28206185.57**	28206185.56	28206185.17
0.1	avg	26898281.26	**28074435.76**	27900169.95	27861705.52
	stdev	592182.61	**90622.83**	169661.46	192965.15
	max	28192160.95	**28190443.96**	28182770.53	28175343.07
0.2	avg	26383883.29	**27952714.54**	27660168.06	27639052.69
	stdev	928241.18	**175564.15**	286826.18	285301.39
	max	27999222.67	**28184160.98**	28177194.93	28084393.31

Table 8: Results for all four algorithms on long-term experiment no 2 with $\sigma = \{0.0, 0.1, 0.2\}$

σ	metric	SA	PBIL	DEUM$_d$	GA
0.0	avg	52354427.68	**54786246.45**	54786143.19	54786221.39
	stdev	2127124.29	**33.67**	215.44	111.54
	max	54786270.17	**54786270.18**	54786270.18 5	4786270.18
0.1	avg	51371029.57	**54403268.03**	53853670.39	53786468.14
	stdev	1614410.12	**157566.94**	382570.85	434745.35
	max	54350942.73	**54659052.03**	54579439.95	54572058.58
0.2	avg	49446794.81	**54118494.00**	53193854.64	53190355.83
	stdev	2027148.57	**287349.48**	710165.67	664648.38
	max	54041078.37	**54627716.02**	54515661.02	54369960.24

Table 9: Results for all four algorithms on long-term experiment no 3 with $\sigma = \{0.0, 0.1, 0.2\}$

σ	metric	SA	PBIL	DEUM$_d$	GA
0.0	avg	48643880.12	**48644998.95**	48644938.55	48644998.84
	stdev	958.68	**0.00**	240.51	0.56
	max	48644998.94	**48644998.95**	48644998.94	48644998.95
0.1	avg	44947357.93	**48352358.09**	47675291.37	47651841.87
	stdev	1289223.02	**121845.47**	460598.21	442566.02
	max	47575149.77	**48582177.25**	48386805.20	48400135.28
0.2	avg	42629266.68	**48175217.03**	47178487.87	46971715.86
	stdev	1950990.60	**188331.61**	504471.41	635703.92
	max	46462454.21	**48548524.23**	48163214.83	48333998.06

5.4 Analyzing the results

Now, let us use our results to analyze the implication of dynamic pricing for both short-tem and long-term environments.

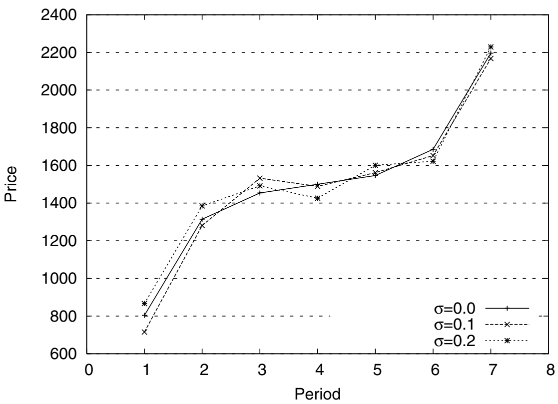
No random impact on demand or costs: We now look at the results when there is no shock in demand or costs, i.e, when $\sigma = 0.0$. Since PBIL was the overall best performing algorithm, its results are used for analysis. As we can see from Figure 2 for short-term, by using dynamic pricing, the maximum price increased significantly during the first day of the week. For instance, with smaller price caps scenario (Exp1), the prices were set to the given maximum of 250/unit for first five days of the week and the production was higher in the first few days of the week and gradually decreased towards the end of the week. Similarly, with no price caps and loose production constraints (Exp2), both price and production was increased during the first few days of the week. Finally, with no price caps and tighter production constraints (Exp3), the production was set to its maximum, 300 unit, for the first six days of the week and slightly decreased to 290 unit for the last day of the week. The low production also triggered the high price during the first few days of the week.

For long term environment, comparing the total profit shown in Table 7, 8 and 9, it can be observed that the experiment (scenario) 2 has the higher total profit. This, as expected, suggests that the increased customer loyalty results in higher profits. Now let us analyze the implication of dynamic pricing to the price and production as shown in Figure 3. First, in year 1, Exp2 and Exp3 have lower prices and higher production than Exp1. This suggests that the optimal policy is to have lower price in order to gain market share and increase future profits. Second, the capacity investment has important impacts on the pricing policy. Whereas in Exp3, under a capacity constraint of 300 units, prices tend to increase monotonically, in Exp2, with a capacity increase to 600 units, the company has non monotonic pricing policy. In this case, there is a promotional period in year 5 in order to gain enough costumers for a large price rise in period 7. Finally, as it can be seen from period seven of Exp2, higher capacity leads not only to higher production but also to higher prices.

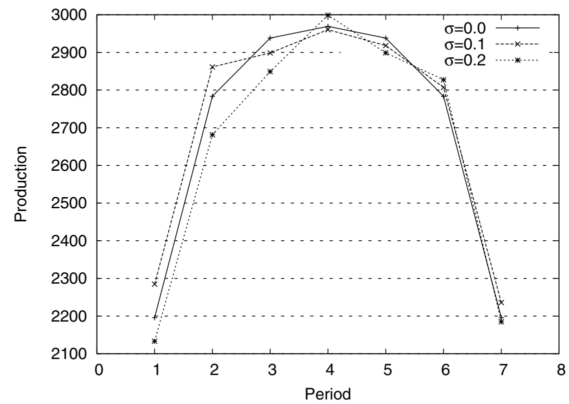
Impact of Random Shocks: An important feature of our model is the inclusion of a random shock representing uncertainty regarding the behavior of demand and production costs. As presented in Figure 1, this random function did not change significantly the production schedule; however they can have an important impact on the effective profits received by the firm during the planning horizon. The results shown here reports the expected profit for the solution reached by the algorithms, and not the actual profits. This shows that the algorithms fail to converge to the optimal solution due to the impact of the random term. This in fact tend to decrease the expected value of the policies derived for the dynamic pricing problem.

6. CONCLUSION

In this paper, we investigated the use of dynamic pricing strategy in firms with the aim to maximize the overall profit during the lifecycle of a product by means of improved resource management. We described how dynamic pricing can be useful in resource management, and developed a model

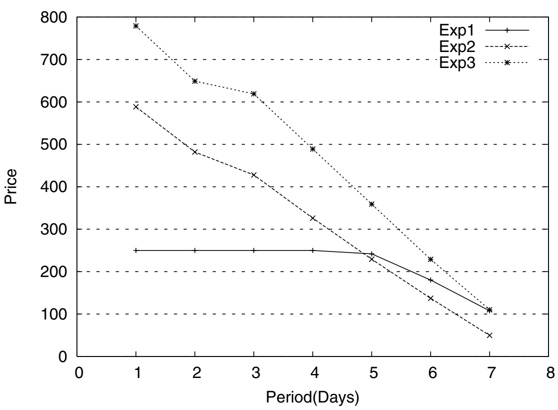


(a) Effect of shock on price per time period

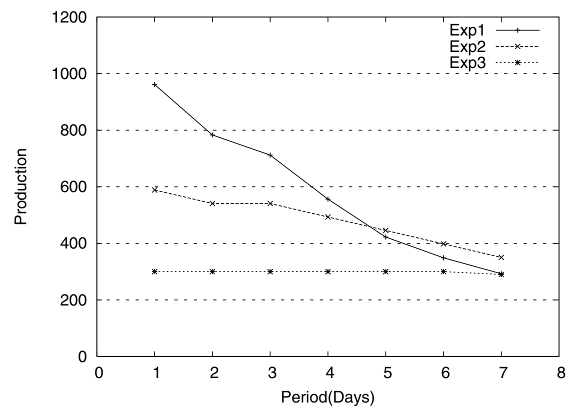


(b) Effect of shock on production per time period

Figure 1: Best solution found by PBIL for three different setup of σ for the long-term experiment 1

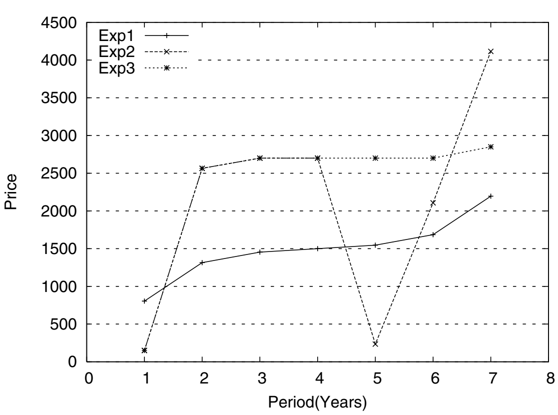


(a) Price per time period

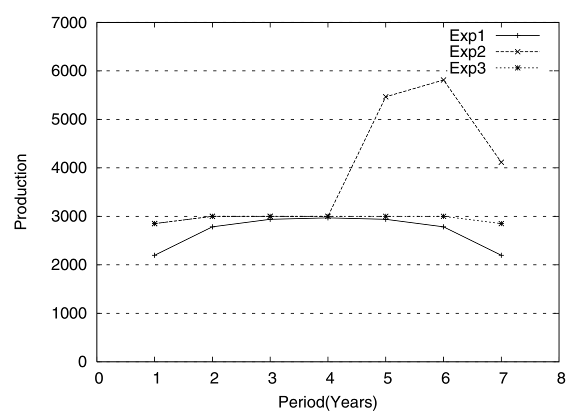


(b) Production per time period

Figure 2: Graph showing the best prices and production found by PBIL for all three short-term experiments



(a) Price per time period



(b) Production per time period

Figure 3: Graph showing the best prices and production found by PBIL for all three long-term experiments

encapsulating the effect of interaction between demand and price has in the overall profit. We implemented two different EDAs, a GA and a SA algorithm for solving the developed dynamic pricing model. The experiments were conducted for both short-term and long-term profits. The results show that the PBIL algorithm had the best performance out of all the tested algorithms. We have also used the best solution found by PBIL to analyze the implication of dynamic pricing for both short-term and long-term profit.

The assumptions made in this paper for both short-term and long-term analysis were typical but not universal. In real world scenario, however, there may be variations in the dynamics of demand and price for each individual products (or services). This therefore may affect the model and therefore effect the performance of the algorithms. However, the better performance of PBIL in different scenarios studied in this paper suggests that, the PBIL algorithm can be effectively applied for improving resource management via the proposed dynamic pricing model.

We note that the experimental results presented in this paper are based on the bit string representation of the solution. Therefore, our conclusions only apply for binary EDA and GA. The performance of real valued version of these algorithms may have different performance. Further analysis should be done in order to verify this and remains one of the areas for the future work. Also, both EDAs used in this paper assume no dependency between variables in the solution. It would be interesting to see the performance of other higher order EDAs for his problem.

7. REFERENCES

[1] T. Bäck, D. Fogel, and Z. Michalewicz. *Handbook of Evolutionary Computation*. Oxford Univ. Press., 1997.

[2] S. Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning,. Technical Report CMU-CS-94-163, Pittsburgh, PA, 1994.

[3] M. Bichler, J. Kalagnanam, K. Katircioglu, A. J. King, R. D. Lawrence, H. S. Lee, G. Y. Lin, and Y. Lu. Applications of flexible pricing in business-to-business electronic commerce. *IBM Systems Journal*, 41(2):287–302, 2002.

[4] P. A. Bosman. *Design and Application of Iterated Density-Estimation Evolutionary Algorithms*. PhD thesis, Universiteit Utrecht, Utrecht, The Netherlands, 2003.

[5] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[6] I. Inza, P. Larrañaga, and B. S. R. Etxeberria. Feature subset selection by Bayesian networks based optimization. *Artificial Intelligence*, 123(1–2):157–184, 2000.

[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983*, 220, 4598:671–680, 1983.

[8] P. Larrañaga and J. A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2002.

[9] H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions: I. binary parameters. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature – PPSN IV*, pages 178–187, Berlin, 1996. Springer.

[10] Y. Narahari, C. V. Raju, K. Ravikumar, and S. Shah. Dynamic pricing models for electronic business. *Sadhana*, 30(part 2,3):231–256, April/June 2005.

[11] F. S. Oliveira. A constraint logic programming algorithm for modeling dynamic pricing. *Informs Journal of Computing*, 2007.

[12] G. Owusu, R. Dorne, C. Voudouris, and D. Lesaint. Dynamic planner: A decision support tool for resource planning, applications and innovations in intelligent systems x. In *Proceedings of ES 2002*, pages 19–31, 2002.

[13] G. Owusu, C. Voudouris, M. Kern, A. Garyfalos, G. Anim-Ansah, and B. Virginas. On optimising resource planning in bt with fos. In *Proceedings International Conference on Service Systems and Service Management*, 2006.

[14] K. Parsopoulos and M. Vrahatis. Particle swarm optimization method for constrained optimization problems. *Intelligent Technologies–Theory and Application: New Trends in Intelligent Technologies, volume 76 of Frontiers in Artificial Intelligence and Applications*, pages 214–220, 2002.

[15] M. Pelikan. *Bayesian optimization algorithm: From single level to hierarchy*. PhD thesis, University of Illinois at Urbana-Champaign, Urbana, IL, 2002. Also IlliGAL Report No. 2002023.

[16] M. Pelikan and D. E. Goldberg. Hierarchical BOA solves Ising spin glasses and MAXSAT. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2003)*, pages 1271–1282, 2003. Also IlliGAL Report No. 2003001.

[17] A. Petrovski, S. Shakya, and J. McCall. Optimising cancer chemotherapy using an estimation of distribution algorithm and genetic algorithms. In *proceedings of Genetic and Evolutionary Computation COnference (GECCO 2006)*, seattle, USA, 2006. ACM.

[18] S. Shakya. *DEUM: A Framework for an Estimation of Distribution Algorithm based on Markov Random Fields*. PhD thesis, The Robert Gordon University, Aberdeen, UK, April 2006.

[19] S. Shakya, J. McCall, and D. Brown. Using a Markov Network Model in a Univariate EDA: An Emperical Cost-Benefit Analysis. In *proceedings of Genetic and Evolutionary Computation COnference (GECCO2005)*, pages 727–734, Washington, D.C., USA, 2005. ACM.

[20] B. Virginas, M. Ursu, C. Voudouris, and G. Owusu. Distributed versus centralized resource allocation: An interplay of power in an agent-based system. In *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA2005)*, pages 12–17, Innsbruck, February, 2005. Acta Press.

[21] C. Voudouris, G. Owusu, R. Dorne, and A. Mccormick. Fos: An advanced planning and scheduling suite for service operations. In *Proceedings International Conference on Service Systems and Service Management*, 2006.