Performance Evaluation of Evolutionary Class of Algorithms— An Application to 0-1 Knapsack Problem

P. DUTTA AND D. DUTTAMAJUMDER
Electronics and Communication Sciences Unit
Indian Statistical Institute
203, B. T. Road
Calcutta 700035, India
res9314@isical.ernet.in

(Received and accepted October 1997)

Abstract—The 0-1 knapsack [1] problem is a well-known NP-complete problem. There are different algorithms in the literature to attack this problem, two of them being of specific interest. One is a pseudo polynomial algorithm of order $O(nK)$, K being the target of the problem. This algorithm works unsatisfactorily, as the given target becomes high. In fact, the complexity might become exponential in that case. The other scheme is a fully polynomial time approximation scheme (FPTAS) whose complexity is also polynomial time. The present paper suggests a probabilistic heuristic which is an evolutionary scheme accompanied by the necessary statistical formulation and its theoretical justification. We have identified parameters responsible for the performance of our evolutionary scheme which in turn would keep the option open for improving the scheme.

Keywords—Almost sure convergence, Stochastic process, Maximum likelihood estimates, Asymptotic joint distribution, Knapsack problem, Fully polynomial time approximation scheme, Pseudo-polynomial algorithm.

1. INTRODUCTION

In recent years, enough research initiative is being directed towards exploring evolutionary class of algorithms that are primarily used for search and optimization [2–4]. Genetic algorithm is an instance of this evolutionary class. However, the theoretical foundation of these algorithms is not well explored in literature. In the present discussion, after suggesting an evolutionary scheme to solve the 0-1 knapsack problem, we try to formulate the convergence aspect of the algorithm and we have identified the parameters responsible for the performance of such an algorithm. In due course, we shall see that since the performance parameters are unknown so, the question of their statistical estimation is of great interest.

We have devoted the next section to give the idea about an evolutionary algorithm. In the following section, we describe the 0-1 knapsack problem along with some real life applications. There we also show how the problem is mapped into the evolutionary domain. In Section 4, we suggest the formulation of our new evolutionary scheme. In Section 5, we concentrate on statistically formulating the questions pertaining to the convergence of an evolutionary algorithm. Here we have shown how the convergence is ensured for any member of the evolutionary family which satisfy two necessary assumptions and as such, the convergence is strongest (almost sure

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX

convergence) in statistical sense. In this connection, we have described Kolmogorov's strong law of large number [5,6] for the sake of convenience and formulated our problem in Kolmogorov's framework. In Section 6, we located the parameters responsible for the performance of our algorithm. Since these parameters are unknown, they require statistical estimation which is covered in Section 7. The last section contains results in which we shall show that the problem being properly mapped to the domain of stochastic evolution converges in finite number of steps almost surely. In the treatment, the number of steps required to converge to an optimal solution is a nonnegative integral valued random variable. Under certain 'realistic' assumptions, the distribution function of the above random variable has been found. The expectation and variance of the random variable are derived, both of which are found to be finite. Kolmogorov's strong law of large number will then ensure the almost sure convergence of an evolutionary scheme in finite number of steps.

2. PERSPECTIVE AND MOTIVATION

Essentially, an evolutionary algorithm is a stochastic algorithm. In an evolutionary scheme, the argument(s) of the objective function (function to be optimized) is/are encoded in a suitable manner. The search mechanism takes place in the transformed space consisting of the coded version of the arguments of the objective function. This choice of coding scheme has been discussed in literature [2,4]. Our formulation is based on binary coded representation. In other words, the transformed search space is composed of binary code of the arguments. So the problem reduces to searching the transformed search space and obtaining that binary code (string) corresponding to which the objective function is optimized. It may be mentioned that there is scope of theoretical formulation where the code has more than two states in its alphabet, but that is beyond the scope of the present treatment, which we hope to treat in a separate paper. If the problem domain is sufficiently complex or the search of the domain is rather vast, even if it is not that much more complex, the problem might become very difficult to solve. Problems belonging to NP-complete or NP-hard class are the glaring examples of such eventuality [7]. To date, these problems do not have deterministic polynomial time algorithms. What is best available are the fully polynomial time algorithms and/or heuristic schemes. An evolutionary algorithm is also a probabilistic and heuristic method to solve an optimization problem, especially one which is hard in nature. These optimization problems include application areas in theoretical computer science, artificial intelligence, VLSI design and layout. The stochastic nature of evolutionary schemes have been found to play an important role with respect to these problems. However, there are other areas where probabilistic algorithms have been used with encouraging outcomes. These areas encompass pattern recognition and classification [8], scheduling [9], computational geometry [10], etc. But most of the treatment so far has been of empirical nature and little theoretical formulation about algorithms of evolutionary class has been realized so far.

2.1. Structure of an Evolutionary Algorithm

In an evolutionary algorithm, we start with a handful of coded strings of a particular length that constitute the initial population, the choice of which is usually random. However, the strings may be chosen with some contextual prior knowledge and intuition. The size of the initial population has been discussed by Goldberg in [2]. From this initial population, another new population of the same size (as that of the initial population) is generated by some transition operation(s). This new population again gives rise to another new population by the same way. This process is continued until the termination condition is reached. Since we are not aware beforehand about the string(s) that correspond(s) to the optimum value of the objective function, the termination of the algorithm and the convergence of the algorithm continue to be open problems. However, there are different proposals about the termination condition [2–4]. As for example, De Jong has suggested an *elitist model* in which the best string yet obtained is kept track of in a buffer so that it does not get lost. What he has suggested is, we terminate the algorithm if the above buffer remains unchanged across a sufficiently large number of generations. There may arise questions like how many number of generations to be set as relaxation level and what is the guarantee that within that stipulated choice of the level a global optimal is reached. In fact, it may so happen that the algorithm is stuck at some suboptimal solution. This type of a situation is typically known as the problem of 'premature convergence'.

The most important question that we face is the choice of transition operation(s). In fact, it plays the most crucial role so far as the strength of the suggested scheme is concerned. In our formulation, we use the following notations:

- P_t is the population at iteration t;
- Q is a temporary population;
- $\tau(.)$ is the transition operation of the proposed algorithm;
- s and s' are coded strings.

The abstract structure of the evolutionary algorithm is as follows:

```
$t \leftarrow 0$;
repeat {
$Q \leftarrow \phi$;
for all $s \in P_t$
{
$s' \leftarrow \tau(s)$;
$Q \leftarrow Q \cup \{s'\}$;
}
$t \leftarrow t + 1$;
$P_t \leftarrow Q$;
}
until termination condition is attained.
```

3. THE 0-1 KNAPSACK PROBLEM

3.1. Prelude

The knapsack problem is an old problem that has different versions that defines the 0-1 knapsack problem later on. As the problem is posed, it may sound rather simple, but interestingly enough, the problem being NP-complete in nature has no deterministic polynomial time algorithm till this date. But as such, the problem has been attacked from different angles and different algorithms have been proposed to this effect, of which two are the most accepted in the literature. One is a pseudo polynomial algorithm [7] and the other is a fully polynomial time approximation scheme (FPTAS) [7]. Different heuristic schemes have been proposed for this purpose. It has also been tried with artificial neural network methodology [12,13] and genetic algorithm approach [2,14–16]. Dutta and DuttaMajumder have suggested an evolutionary heuristic in [17] to this end. We have proposed a mapping of the 0-1 knapsack problem to the domain of our evolutionary scheme and have developed the algorithm on the basis of that mapping. The algorithm is applied to different randomly generated data sets and the results obtained are found to be encouraging enough. For the sake of completeness, we consider existing heuristic as well as a fully polynomial time approximation scheme (FPTAS) for this purpose and, we compare the performance with the pseudo polynomial algorithm at one hand and the FPTAS on the other.

3.2. Definition of the Problem

Given positive integers $c_1, c_2, \ldots, c_n$ and a target K, does there exist a $S \subseteq \{1, 2, \ldots, n\}$ such that $\Sigma_{j \in S} c_j = K$?

To make things clearer, let us illustrate an example. Let the integers given be $\{1,3,7,5,9\}$ and the target is 16. Then there can be solutions like $\{1,3,7,5\}$ or $\{7,9\}$. But if the target is 23 instead, there cannot be any solution.

3.3. Example of a Real Life Application

A vendor car has capacity K kg. There are some bundles having respective weights $c_1, c_2, \dots, c_n$ kg which are to be transported by that vendor car. The problem is to pick up those bundles and load them in the car so that the car capacity is maximum utilized, if not fully.

3.4. Mapping the Problem to Evolutionary Domain

We take the length of the coded string to be equal to the cardinality of the knapsack set. Our algorithm will produce a binary string in which those positions of the string would contain '1', whose corresponding numbers come in the subset S. Thus, in the above example, for target value 16, 11110, or 00101 are possible solutions.

In this context, let us define the following.

DEFINITION. *The sum of the numbers in the knapsack corresponding to the positions containing '1' in a string is called the evaluation of that string. We represent the evaluation of a string s by $e(s)$.*

The *evaluation* of the string containing '1' in all the places gives the sum of all the elements in the knapsack.

This is clear from the above definition that given a target K, the closer the *evaluation* $e(s)$ of the string s to the target, the more prospective it is. On the basis of this observation, we choose the objective function. The natural choice could be $|K - e(s)|$. The objective is to get hold of that string s^* such that the distance measure $|K - e(s^*)|$ is minimum. Now if there is any string s^* for which the above expression is zero, then it is immediately concluded that

(i) there exists an exact solution to the knapsack problem, and
(ii) s^* corresponds to a solution.

4. OUR APPROACH AND THE NEW ALGORITHM

Each location of the string may or may not change which has certain probabilities. Our proposal is to choose the probability of changing from one status to another, commonly referred to as the transition probability, on the basis of the distance measure described above. Of course, it is reasonable to choose small probability values for small values of the distance. In other words, strings with evaluation closer to the target will have a smaller probability to undergo change. Moreover, if a string s^* is reached for which the distance is zero, at a stage the transition probability becomes zero. This means that this s^* will remain in all the future populations to come. In our formulation regarding convergence of the algorithm, we shall see that this property is very essential. In that sense, the scheme is inherently elitist in nature. About the choice of the *transition probability* for a string s, we take it equal to $1 - e(s)/K$ if $e(s) < K$ and $1 - K/e(s)$ if $e(s) \geq K$. At the initial stage, the search process is extensive and as generations pass on the search process gradually becomes intensive. We formally describe the algorithm as follows.

```
 Input: $c_1, c_2, \dots, c_n, K$.
Output: a binary string of length $n$.
        begin
Step 1. To randomly create the initial population $\mathcal{P}_0$ of strings of a particular size provided
        by the user.
Step 2. $t \leftarrow 0$.
Step 3. For all the strings $s \in \mathcal{P}_0$ do
        distance $\leftarrow$ min $(|e(s) - K|)$.
```

```
Step 4. Repeat Step 5 through Step 8 until either of the following holds:
          (i) distance becomes zero,
          (ii) iteration number exceeds a preassigned positive integer $T$, the maximum allowable iteration number, i.e., $t > T$.
Step 5. $Q \leftarrow \phi$.
Step 6. For all the strings $s \in \mathcal{P}_t$ do
        {
        each location of the string $s$ undergoes change with its corresponding transition probability $p(s)$;
        the new string $s'$ thus obtained is put in some temporary population $Q$. In other words, $Q \leftarrow Q \cup \{s'\}$
        }
Step 7. $t \leftarrow t + 1$.
Step 8. $P_t \leftarrow Q$.
        end
```

5. STATISTICAL FORMULATION OF OUR NEW ALGORITHM

5.1. A Brief Review of Current Study

The evolutionary algorithms, as we have already pointed out, are essentially used for the purpose of search and optimization and till today they lack a thorough comprehensive theoretical formulation, though there have been some treatment about their convergence [18,19]. We shall try to give some theoretical justification of our proposed algorithm. In [20,21], Dutta and DuttaMajumder have presented the probability mass function of the nonnegative integer valued random variable representing the number of steps required to converge to an optimal under certain conditions. In [22], we have identified the parameters responsible for the performance of an evolutionary scheme. In the present initiative, we try to solve the statistical question of estimating these parameters.

5.2. Theoretical Treatment

Here we will treat the number of steps required to converge to an optimal string (any string that correspond to optimum value of the objective function since there could be more than one string which may give the optimum value) in an evolutionary algorithm as a nonnegative integer valued random variable. Let $X =$ number of steps required to attain an optimal. Now we shall find out the expectation $E(X)$ and the variance $V(X)$. In our following discussion, we use the notations listed below.

$$
\begin{aligned}
s^* &= \text{Optimum string (it may not be unique).}\\
G_k &= \text{Population in the } k^{\text{th}} \text{ generation.}\\
\theta &= P[s^* \in G_k \mid s^* \notin G_{k-1}].\\
p_m &= P[X = m],\ m \geq 0.\\
r_m &= P[s^* \notin G_i,\ \forall 0 \leq i \leq m],\ m \geq 0.\\
E_m &= \{s^* \notin G_i,\ \forall 0 \leq i \leq m\},\ m \geq 0.\\
P(s) &= \textstyle\sum_{m \geq 0} p_m s^m, \text{ the generating function of } \{p_m, m \geq 0\}.\\
R(s) &= \textstyle\sum_{m \geq 0} r_m s^m, \text{ the generating function of } \{r_m, m \geq 0\}.
\end{aligned}
$$

In our calculation, we take the following assumptions.

1. One step homogeneous Markov chain which means that

$$P\left[s^* \in G_k \mid s^* \notin G_i, \forall i = 0(1)k-1\right] = P\left[s^* \in G_k \mid s^* \notin G_{k-1}\right], \qquad k \geq 1. \tag{1}$$

Moreover, $\theta = P[s^* \in G_k \mid s^* \notin G_{k-1}]$ does not depend on the generation k.

2. Once an optimal string is attained at a particular generation, then all the subsequent generations would contain that

$$P[s^* \notin G_k \mid s^* \in G_{k-1}] = 0, \qquad \forall k \geq 1. \tag{2}$$

It comes out that the probability mass function of X belongs to a two parameter family and is given by

$$P[X = k] = (1 - p_0)\theta(1-\theta)^{k-1}, \qquad k \geq 1, \tag{3}$$

where $p_0 = P[X = 0]$. First, let us note that

$$\begin{aligned}
\{X = k\} &= \{s^* \in G_k, s^* \notin G_i, \forall i = 0(1)k-1\}, \\
p_k &= P[X = k] \\
&= P[s^* \in G_k, s^* \notin G_i, \forall i = 0(1)k-1] \\
&= P[s^* \in G_k \mid s^* \notin G_i, \forall i = 0(1)k-1]\, P[s^* \notin G_i, \forall i = 0(1)k-1] \\
&= P[s^* \in G_k \mid s^* \notin G_{k-1}]\, P[s^* \notin G_i, \forall i = 0(1)k-1], \qquad \text{by Assumption 1} \\
&= \theta P[s^* \notin G_i, \forall i = 0(1)k-1] \\
&= \theta \{P[s^* \notin G_i, \forall i = 0(1)k-2] - P[s^* \in G_{k-1}, s^* \notin G_i, \forall i = 0(1)k-2]\} \\
&= \theta r_{k-2} - \theta p_{k-1}.
\end{aligned}$$

Thus,

$$p_k + \theta p_{k-1} = \theta r_{k-2}. \tag{4}$$

Note that,

$$E_0 \supseteq E_1 \supseteq E_2 \supseteq \cdots \supseteq E_n \supseteq E_{n+1} \supseteq \cdots.$$

This implies that,

$$r_0 \geq r_1 \geq r_2 \geq \cdots \geq r_n \geq \cdots. \tag{5}$$

Now,

$$r_0 = P[s^* \notin G_0] = 1 - P[s^* \in G_0] = 1 - p_0. \tag{6}$$

Again observe that,

$$r_k + p_k = r_{k-1}. \tag{7}$$

Now multiplying both sides of (7) by $s^k, s \in [0,1]$ and summing over $k \geq 1$, we have

$$P(s) = 1 - (1-s)R(s). \tag{8}$$

Again multiplying both sides of (4) by $s^k, s \in [0,1]$ and summing over $k \geq 2$, we get

$$\begin{aligned}
P(s) - p_0 - p_1 s + \theta s(P(s) - p_0) &= \theta s^2 R(s), \\
(1+\theta s)P(s) - (1+\theta s)p_0 - sp_1 &= \theta s^2 R(s), \\
(1+\theta s)(1 - (1-s)R(s)) - (1+\theta s)(1 - r_0) - s(r_0 - r_1) &= \theta s^2 R(s), \\
r_0 + r_0\theta s - r_0 s + r_1 s &= \left(\theta s^2 + (1+\theta s)(1-s)\right) R(s), \qquad \text{or} \\
(1 - (1-\theta)s)R(s) &= (1 - (1-\theta)s)r_0 + r_1 s.
\end{aligned}$$

Thus,

$$R(s) = r_0 + \frac{r_1 s}{1 - (1-\theta)s}. \tag{9}$$

Since $s(1-\theta) \in [0,1)$, so expanding the right-hand side of (9), we end up at

$$r_k = r_0(1-\theta)^k, \qquad k \geq 1. \tag{10}$$

Now from (7),

$$\begin{aligned} p_k &= r_{k-1} - r_k, & k \geq 1, \\ &= r_0(1-\theta)^{k-1} - r_0(1-\theta)^k \\ &= r_0\theta(1-\theta)^{k-1}, & k \geq 1, \\ &= (1-p_0)\theta(1-\theta)^{k-1}, & k \geq 1. \end{aligned}$$

Therefore,

$$P[X = k] = (1-p_0)\theta(1-\theta)^{k-1}, \qquad k \geq 1. \tag{11}$$

Now, $\lim_{s\to 1-} R(s) = \lim_{s\to 1-}(1-P(s))/(1-s) = \lim_{s\to 1-}(-P'(s))/-1 = \lim_{s\to 1-} P'(s) = \lim_{s\to 1-} \sum_{k\geq 1} kp_k s^{k-1} = \sum_{k\geq 1} \lim_{s\to 1-}(kp_k s^{k-1}) = \sum_{k\geq 1} kp_k = E(X)$. Thus,

$$E(X) = \lim_{s\to 1-} R(s). \tag{12}$$

Here, note that

$$\theta = P[s^* \in G_1 \mid s^* \notin G_0] = \frac{P[s^* \in G_1, s^* \notin G_0]}{P[s^* \notin G_0]} = \frac{p_1}{1-p_0}. \tag{13}$$

So,

$$E(X) = \frac{1-p_0}{\theta}. \tag{14}$$

To calculate $V(X)$, we note the fact that since $P(s) = \sum_{k\geq 0} p_k s^k$ so $P''(1) = E(X(X-1))$. However, calculation gives

$$P''(1) = \frac{2(1-p_0)(1-\theta)}{\theta^2}. \tag{15}$$

This will give

$$V(X) = \frac{(1-p_0)(1+p_0-\theta)}{\theta^2}. \tag{16}$$

5.3. Almost Sure Convergence

We shall be applying the well-known **Kolmogorov's Strong Law of Large Numbers** [5,23] as the underlying conditions are satisfied.

Kolmogorov's strong law of large numbers: let $X_1, X_2, \ldots, X_n$ be independent and identically distributed random variable with $E(|X_1|) < \infty$. It immediately follows that $E(X_1)$ exists and will be finite. If $E(X_1) = \mu$, then $\bar{X}_n = 1/n(X_1 + X_2 + \cdots + X_n) \to \mu$ almost surely as $n \to \infty$. As a corollary, it follows that if $X_1, X_2, \ldots, X_n$ are nonnegative and $E(X_1)$ exists and is finite, then $\bar{X}_n \to E(X)$ almost surely as $n \to \infty$. In the present context, the number of steps required to attain an optimal being a nonnegative integer valued random variable is represented by X. Our derivation gives that $E(X)$ exists and is finite. If the present algorithm is conducted n many times on the problem and if the steps required are $X_1, X_2, \ldots, X_n$, respectively, then $1/n(X_1 + X_2 + \cdots + X_n) \to (1-p_0)/\theta$ almost surely as $n \to \infty$.

5.4. Convergence Viewed as Stochastic Process

Let us define a sequence of random variables $\{Z_k, K \geq 0\}$ as follows:

$$Z_k = \begin{cases} 1, & \text{if } s^* \in G_k, \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

So, $P(Z_k = 1) = P(s^* \in G_k) = P(X \leq k) = \sum_{r=0}^{k} P(X = r) = \sum_{r=0}^{k} p_r = 1 - r_k$. Similarly, $P(Z_k = 0) = r_k$. $\{Z_k, k \geq 0\}$ is a homogeneous Markov chain of order 1. Clearly, $Z_k \sim$ Bernoulli $(1 - r_k)$, $\forall k \geq 0$. Since each Z_k has two states 0 and 1, the transition matrix would look like

$$P = \begin{array}{c|cc} & 0 & 1 \\ \hline 0 & p_{00} & p_{01} \\ 1 & p_{10} & p_{11} \end{array}$$

where

$$p_{00} = P\left[Z_k = 0 \mid Z_{k-1} = 0\right] = P\left[s^* \notin G_k \mid s^* \notin G_{k-1}\right] = 1-\theta,$$
$$p_{01} = P\left[Z_k = 1 \mid Z_{k-1} = 0\right] = P\left[s^* \in G_k \mid s^* \notin G_{k-1}\right] = \theta,$$

for $k \geq 1$. Moreover,

$$P\left[Z_n = 1 \mid Z_0 = 0\right] = P\left[s^* \in G_n \mid s^* \notin G_0\right] \to 1, \qquad \text{as } n \to \infty.$$

Next we try to find out the modified expression of p_k. In the previous model, $p_k = P[s^* \in G_k] = P[X = k]$. There X was the minimum number of steps after which s^* belongs to the population. It was not possible for s^* to go out of any future generation once it is included in the population of a particular generation due to the assumption of inherent elitism. But let us modify the set-up where the transition from s^* being present in one generation to it being absent in the next has a strictly positive probability. Let $P[s^* \notin G_k \mid s^* \in G_{k-1}] = \tau > 0$. We shall see that under this new set-up, the probability of convergence to an optimal is strictly less than unity, though it is strictly positive

$$\begin{aligned} p_k &= P\left[s^* \in G_k\right] \\ &= P\left[s^* \in G_k \mid s^* \notin G_{k-1}\right] P\left[s^* \notin G_{k-1}\right] \\ &\quad + P\left[s^* \in G_k \mid s^* \in G_{k-1}\right] P\left[s^* \in G_{k-1}\right], \qquad k \geq 1, \\ &= \theta(1-p_{k-1}) + (1-\tau)p_{k-1} \\ &= \theta + (1-\theta-\tau)p_{k-1}. \end{aligned}$$

Solving the above recurrence relation we get the following:

$$P\left(s^* \in G_k\right) = \begin{cases} p_0, & \text{if } k = 0, \\ \dfrac{\theta}{\theta+\tau} + \dfrac{\tau}{\theta+\tau}(1-\theta-\tau)^k, & \text{if } k \geq 1. \end{cases} \tag{5.2}$$

From this expression, it is clear that $\lim_{k\to\infty} P(s^* \in G_k) = \theta/(\theta+\tau) < 1$. Now for the sake of convenience, we use the following notations:

$$\begin{aligned} t_n &= 1 - (1-\theta-\tau)^n, \\ \alpha &= \frac{\theta}{\theta+\tau}, \\ \beta &= \frac{\tau}{\theta+\tau}. \end{aligned}$$

Let us notice that

$$\begin{aligned} \alpha + \beta &= 1, \\ t_0 &= 1, \\ t_1 &= \theta + \tau, \\ \lim_{n\to\infty} t_n &= 1. \end{aligned}$$

It comes out quickly from this notation that

$$Q^n = \begin{pmatrix} 1-\alpha t_n & \alpha t_n \\ \beta t_n & 1-\beta t_n \end{pmatrix},$$

$|Q^n| = 1 - t_n > 0$, $\forall n \geq 1$. This means, the determinant goes to zero as n becomes larger. In other words,

$$Q^{(\infty)} = \lim_{n\to\infty} Q^n = \begin{pmatrix} \dfrac{\tau}{\theta+\tau} & \dfrac{\tau}{\theta+\tau} \\ \dfrac{\tau}{\theta+\tau} & \dfrac{\tau}{\theta+\tau} \end{pmatrix}.$$

Thus, Q^n is asymptotically singular.

6. PERFORMANCE ANALYSIS OF THE SCHEME

It is well known that the performance of a deterministic algorithm is judged by its computational complexity. But since an evolutionary algorithm is probabilistic in nature, so its convergence has some probability associated with it. For the present, we try to locate the parameters that control the performance of our proposed algorithm. In our present formulation, we made some assumptions, under which the performance comparison is possible. We also propose a notion of characteristic polynomial which gives a measure of the performance of an evolutionary scheme. An area under the curve of the characteristic polynomial over the range 0 to 1 has been equivalently shown to represent the performance criterion.

6.1. Performance Analysis of the Evolutionary Algorithm

We have seen that the distribution of X is characterised by two parameters p_0 and θ. Now

$$p_0 = 1 - \left(1 - \frac{|S|}{2^L}\right)^N \tag{17}$$

is found to depend on

(i) length L of the strings under the coding methodology;
(ii) size N of the population;
(iii) cardinality of S, the collection of strings present in the whole search space which correspond(s) to attaining the target value K, in case, if any solution exists and/or going as close of K as possible in case of no solution existing.

It follows that p_0 is very much problem specific and as such does not exactly guide the performance of the algorithm proposed to solve it. It is more a problem specific parameter rather than a performance parameter. On the other hand, it is the choice of θ which primarily controls the performance of the algorithm. It is clear from the expression of $E(X)$ that larger the value of θ, better is the performance expected of the algorithm.

On the basis of the particular coding scheme (which decides L) in the present problem and suitable population size N, p_0 becomes fixed. It is of course worth mentioning that the term 'suitable' is not immune to criticism and as such, it is another difficult part to ascertain N, which we are not dealing with now. We note from (14) and (16) that $E(X)$ and $V(X)$ both decrease with the increase of θ over the range $(0, 1]$. Thus, it is statistically desirable for the evolutionary algorithm to have as large θ value as possible in both the senses of

(a) expected steps to converge;
(b) consistency (low variability).

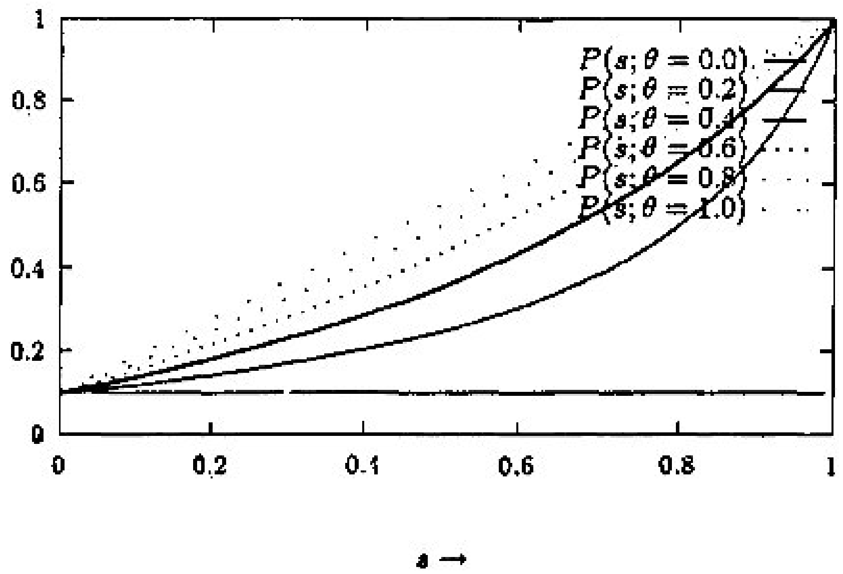

Figure 1.

6.2. Our Algorithm in Terms of Characteristic Polynomial

In the discussion onwards, $P(s)$ is assumed to denote the generating function of $\{p_k, k \geq 0\}$. It can be verified that the expression of $P(s)$ is given as follows:

$$P(s) = p_0 + s - p_0 s - \frac{(1-\theta)(1-p_0)(1-s)s}{(1-s(1-\theta))}, \tag{18}$$

then it follows that

$$P'(s) = \frac{\theta(1-p_0)}{(1-s(1-\theta))^2}. \tag{19}$$

Now $P'(s) > 0$, $\forall\theta \in (0,1]$, $s \in [0,1]$ which means that $P(s)$ is monotonically increasing in s with a particular choice of θ. Again, $P''(s) > 0$, $\forall\theta \in (0,1)$, $s \in [0,1]$, and $P'(s) = 0$ for $\theta = 1$, $s \in [0,1]$. Thus, $P(s)$ is *convex* $\forall s \in [0,1]$ and $\forall\theta \in (0,1)$, and $P(s)$ is *linear* $\forall s \in [0,1]$ and for $\theta = 1$. Also, note that $P(0) = p_0$ and $P(1) = 1$. Moreover, from equation (18), it is evident that $P(s)$ increases with θ for a particular choice of s. The curve of $P(s)$ against s for different choice of θ looks like Figure 1. We note that since the increase in θ statistically means quicker convergence which means that more the area under $P(s)$ from 0 to 1, the better the performance of the algorithm expected. This means that $\int_0^1 P(s)\,ds$ plays the measure of performance of an evolutionary algorithm. In the following discussion, we note some characteristics of $\int_0^1 P(s)\,ds$:

$$\begin{aligned}
\int_0^1 P(s)\,ds &= \int_0^1 \left(\sum_{k=0}^{\infty} p_k s^k\right) ds \\
&= \sum_{k=0}^{\infty} \left(\int_0^1 p_k s^k\,ds\right) \\
&= \sum_{k=0}^{\infty} p_k \left(\int_0^1 s^k\,ds\right) \\
&= \sum_{k=0}^{\infty} p_k \left[\frac{s^{k+1}}{k+1}\right]_0^1
\end{aligned}$$

$$= \sum_{k=0}^{\infty} \frac{p_k}{k+1}$$
$$= E\left(\frac{1}{X+1}\right).$$

On the other hand, since equation (18) gives

$$P(s) = p_0 + s - p_0 s - \frac{(1-\theta)(1-p_0)(1-s)s}{(1-s(1-\theta))},$$

therefore,

$$\int_0^1 P(s)\, ds = 1 - \left(\frac{1-p_0}{1-\theta}\right) - \frac{\theta(1-p_0)}{(1-\theta)^2} \log \theta. \tag{20}$$

Thus, $E(1/X+1)$ gives the measure of the required area

$$E\left(\frac{1}{X+1}\right) = 1 - \left(\frac{1-p_0}{1-\theta}\right) - \frac{\theta(1-p_0)}{(1-\theta)^2} \log \theta. \tag{21}$$

7. ESTIMATING THE PERFORMANCE PARAMETERS

Relation (17) gives us

$$p_0 = 1 - \left(1 - \frac{|S|}{2^L}\right)^N,$$

where

$$S = \{s \mid f(s) \geq f(t), \forall t \in \mathcal{X}_L\},$$

$f(.)$ being the fitness of a string and $\mathcal{X}_L$ being the set of all binary strings of length L. It is obvious that for $L > 0$, $S \neq \phi$, and hence, $|S| > 0$ and so $p_0 > 0$. Moreover, since $|\mathcal{X}_L| = 2^L$, so whenever $S \not\subseteq \mathcal{X}_L$, that means since the objective function is not constant so $p_0 < 1$. It is evident from the expression of p_0 that if we know $|S|$, then p_0 is known and there is nothing to estimate about p_0. But unfortunately, we do not know $|S|$. All we have is that the original search space being isomorphic to $\mathcal{X}_L$ for some L, the number of global optimal solutions in the original search space is more than one would mean $|S| > 1$. So, it is necessary to estimate p_0 statistically. Similarly, since θ is also not known beforehand, one has to estimate that also. So our problem essentially boils down to estimating both p_0 and θ. Now it is highly recommendable from statistical point of view to estimate any unknown parameter on the basis of good number of observations. But unfortunately, it is a frequent practice to conclude one algorithm (that too a probabilistic algorithm) to be performing better than another on the basis of one instance. If the algorithm is applied n times on the 0-1 knapsack problem with $X_1, X_2, \ldots, X_n$ being the respective number of steps required for convergence in these n experiments, then these n observations are statistically independent and identically distributed. As we already know, each of these random variables has a underlying probability mass function $\{p_k, k \geq 0\}$, then the following holds:

$$E\left(\overline{X}\right) = \frac{(1-p_0)}{\theta},$$
$$V\left(\overline{X}\right) = \frac{(1-p_0)(1+p_0-\theta)}{n\theta},$$

$\overline{X}$ being the sample mean of the n observations.

We consider the following transformations:

$$\beta_0 = \frac{1}{\theta}$$

and

$$\beta_1 = \frac{p_0}{\theta}.$$

Note that, $(p_0, \theta) \overset{1:1}{\leftrightarrow} (\beta_0, \beta_1)$. Expressed in terms of β_0 and β_1, we have

$$E\left(\overline{X}\right) = \beta_0 - \beta_1,$$
$$V\left(\overline{X}\right) = \frac{\left(\beta_0^2 - \beta_1^2 - \beta_0 + \beta_1\right)}{n}.$$

Going through the relevant calculations, we get the estimates as follows:

$$\beta_0 = \frac{nV\left(\overline{X}\right) + E^2\left(\overline{X}\right) + E\left(\overline{X}\right)}{2E\left(\overline{X}\right)},$$

$$\beta_1 = \frac{nV\left(\overline{X}\right) - E^2\left(\overline{X}\right) + E\left(\overline{X}\right)}{2E\left(\overline{X}\right)},$$

whence it follows that

$$p_0 = \frac{nV\left(\overline{X}\right) - E^2\left(\overline{X}\right) + E\left(\overline{X}\right)}{nV\left(\overline{X}\right) + E^2\left(\overline{X}\right) + E\left(\overline{X}\right)}, \tag{22}$$

$$\theta = \frac{2E\left(\overline{X}\right)}{nV\left(\overline{X}\right) + E^2\left(\overline{X}\right) + E\left(\overline{X}\right)}. \tag{23}$$

Thus, the estimates are obtained using the corresponding sample estimates

$$\hat{p}_0 = \frac{nV\left(\hat{\overline{X}}\right) - E^2\left(\hat{\overline{X}}\right) + E\left(\hat{\overline{X}}\right)}{nV\left(\hat{\overline{X}}\right) + E^2\left(\hat{\overline{X}}\right) + E\left(\hat{\overline{X}}\right)}, \tag{24}$$

$$\hat{\theta} = \frac{2E\left(\hat{\overline{X}}\right)}{nV\left(\hat{\overline{X}}\right) + E^2\left(\hat{\overline{X}}\right) + E\left(\hat{\overline{X}}\right)}. \tag{25}$$

7.1. Maximum Likelihood Estimates

The *likelihood* function is given by

$$L(\theta, p_0 \mid X_1, X_2, \dots, X_n) = p_0^{\sum_{i=1}^n I_{(X_i=0)}} \prod_{i=1}^{n-\sum_{i=1}^n I_{(X_i=0)}} (1-p_0)\theta(1-\theta)^{X_i-1}, \tag{26}$$

where

$$I_{(A)} = \begin{cases} 1, & \text{if } A \text{ holds}, \\ 0, & \text{otherwise}. \end{cases} \tag{7.3}$$

The log-*likelihood* function is as follows:

$$\begin{aligned} \log L(\theta, p_0 \mid X_1, X_2, \dots, X_n) &= \sum_{i=1}^n I_{(X_i=0)} \log p_0 \\ + \left(n - \sum_{i=1}^n I_{(X_i=0)}\right) [\log(1-p_0) + \log\theta] &+ \sum_{i=1}^n (X_i - 1)\log(1-\theta) I_{(X_i\neq 0)}. \end{aligned} \tag{27}$$

The maximum likelihood estimates of θ and p_0 are obtained by solving the following equations:

$$\frac{\partial \log L}{\partial p_0} = 0,$$
$$\frac{\partial \log L}{\partial \theta} = 0,$$

subject to the Hessian being negative definite evaluated at the estimates obtained by solving the above equations. A small calculation gives

$$\hat{\theta} = \frac{\sum_{i=1}^{n} I_{(X_i \neq 0)}}{\sum_{i=1}^{n} X_i I_{(X_i \neq 0)}} \tag{28}$$

and

$$\hat{p_0} = \frac{\sum_{i=1}^{n} I_{(X_i = 0)}}{n}. \tag{29}$$

Now,

$$\sum_{i=1}^{n} I_{(X_i \neq 0)} \sim \text{ Binomial } (n, 1 - p_0)$$

and

$$\sum_{i=1}^{n} I_{(X_i = 0)} \sim \text{ Binomial } (n, p_0).$$

7.2. Asymptotic Behaviour of the Performance Parameters of the Evolutionary Algorithm

Here we shall study the asymptotic joint distribution of $\begin{pmatrix} \hat{\theta} \\ \hat{p_0} \end{pmatrix}$,

$$\begin{pmatrix} \hat{\theta} \\ \hat{p_0} \end{pmatrix} \sim AN_2 \left(\begin{pmatrix} \theta \\ p_0 \end{pmatrix}, I^{-1}(\theta, p_0) \right),$$

where $I(\theta, p_0)$ is the Fisher's information matrix evaluated at (θ, p_0), and as we know is defined as

$$I(\theta, p_0) = \begin{pmatrix} E\left(-\frac{\partial^2 L}{\partial {p_0}^2}\right) & E\left(-\frac{\partial^2 L}{\partial p_0 \partial \theta}\right) \\ E\left(-\frac{\partial^2 L}{\partial p_0 \partial \theta}\right) & E\left(-\frac{\partial^2 L}{\partial \theta^2}\right) \end{pmatrix}.$$

Again, a trivial calculation will give us the following:

$$E\left(-\frac{\partial^2 L}{\partial {p_0}^2}\right) = \frac{n}{p_0(1 - p_0)}, \tag{30}$$

$$E\left(-\frac{\partial^2 L}{\partial \theta^2}\right) = \frac{n(1 - p_0)}{\theta^2(1 - \theta)}, \tag{31}$$

$$E\left(-\frac{\partial^2 L}{\partial p_0 \partial \theta}\right) = 0. \tag{32}$$

Therefore,

$$\begin{pmatrix} \hat{\theta} \\ \hat{p_0} \end{pmatrix} \sim AN_2 \left(\begin{pmatrix} \theta \\ p_0 \end{pmatrix} \begin{pmatrix} \frac{\theta^2(1-\theta)}{n(1-p_0)} & 0 \\ 0 & \frac{p_0(1-p_0)}{n} \end{pmatrix} \right).$$

If we denote the maximum likelihood estimates of θ and p_0 based on n sample points by $\hat{\theta}_n$ and $\hat{p_0}_n$, respectively. Then $\hat{\theta}_n$ and $\hat{p_0}_n$ are asymptotically independent. Moreover, $V(\hat{\theta}_n)$ and $V(\hat{p_0}_n)$ both will tend to 0 as n becomes large. This also gives their consistency.

8. RESULTS

We have applied our algorithm on different data set of several sizes enlisted and obtained the results which are summarised below:

$$
\begin{aligned}
S_1 = \{&21,91,54,69,51,46,63,67,21,45,24,48,85,79,99,68,70,25,34,71,50,26,46,87,90,\\
&27,57,44,46,24,55,32,9,63,17,16,6,29,61,56,37,55\},\\
S_2 = \{&49,79,66,45,43,14,43,67,49,41,68,64,29,71,31,96,94,17,58,31,14,38\},\\
S_3 = \{&29,15,78,73,35,34,23,67,29,89,60,80,21,15,11,24,18,9,82,91,78,98,94,11,34,\\
&39,65,100,54,24,95,96,13,7,49,80,62,17,17,96,17,19,50,59,75,86,35,4,12,55\},
\end{aligned}
$$

$$
\begin{aligned}
S_4 = \{&457,3,690,849,827,802,551,67,557,385,652,596,365,459,791,100,442,801,806,\\
&451,42,658,518,299,254,595,717,428,906,348\},\\
S_5 = \{&417,227,114,705,211,890,559,419,565,281,388,428,301,147,503,756,938,985,\\
&254,171,570,530,414,323,798,755,525,684,114,348,407,740,969,847,\\
&149,576,622,677\},\\
S_6 = \{&3045,2015,5974,7781,903,9758,8739,9919,745,6877,6080,5344,4593,639,2683,\\
&8584,5862,4077,9978,5031,8334,142,7538,8511,7570,8511,5577,4312,8366,\\
&9024,627,9872,5573,9995,4665,4108,1826,1745,729,6928,9577,3347,7926,391,8171,\\
&8746,3487,9584,7304,7235,1966,2725,6751,715,5783,1977,9009,1618,112,2269,\\
&4836,8751,3647,3903,8499,9536\},\\
s_1 &= 0110111000010011110100110011110000001111000,\\
s_2 &= 0111011101010100101100,\\
s_3 &= 00001000000010010010100101000111011000000100001111,\\
s_4 &= 110000101011101110001001000000,\\
s_{5,1} &= 00011111101100011000110011100111010011,\\
s_{5,2} &= 01010111101000010110111011111100011110,\\
s_{6,1} &= 111001101111000111000100000100100111000011001001111101000110010010,\\
s_{6,2} &= 000001010010101110101010011110101010011010001001000001100010001001.
\end{aligned}
$$

8.1. Comparison with Other Schemes

For getting some idea about the comparative performance, Table 2 is given which is more or less self-explanatory. Here the population size is 10 and the knapsack set size is 1000, whereas the elements of the set are taken randomly from 0 to 100. The number of iterations we allowed to set, the termination condition of our evolutionary algorithm has been taken to be 50. The total sum of elements comes out to be 49722. It is worth mentioning that all the three algorithms are implemented on SUN-SPARC 1.

8.2. Conclusion

We have presented an evolutionary scheme as a stochastic process with binary coded arguments of the objective functions to be optimized. Nonbinary coded schemes will be treated seperately. We have presented a theoretically sound statistical formulation of our algorithm including a treatment on the number of steps required to converge to an optimal string along with a treatment showing almost sure convergence viewed in this case a stochastic process.

We have developed a methodology for performance analysis of our algorithm in terms of characteristic polynomial. The performance parameters are estimated using maximum likelihood estimate and their asymptotic behaviours are established.

Table 1.

Set Label	Set Size	Target Provided	Population Size	Iterations Required	Target Attained	String Label
S_1	42	1102	10	14	1102	s_1
S_2	22	673	10	196	673	s_2
S_3	50	686	10	96	686	s_3
S_4	30	4855	10	345	4855	s_4
S_5	38	12680	10	1000	12678	$s_{5,1}$
S_5	38	12680	20	231	12680	$s_{5,2}$
S_6	66	160559	10	1000	160553	$s_{6,1}$
S_6	66	160559	20	316	160559	$s_{6,2}$

Table 2.

Algorithm	Target Set	Target Attained	Time Required
Pseudo Polynomial		49000	327 sec.
FPTAS	49000	48948	155 sec.
Evolutionary Algorithm		48999	94 sec.

We have applied our algorithm on different sets S_1, S_2, S_3, S_4, S_5, and S_6 of different size generated randomly. All of them have been allowed to work with population size 10. As is evident from Table 1, out of them on two occasions (S_5 and S_6) the algorithm did not converge in 1000 steps, though the departure from the target was very small in comparison to the target as well as the size of the problem. However, when these two sets are tried with the population size 20, both of them did converge, other parameters remaining the same. We did try with knapsack sets of vast cardinality, we got encouraging results. In case of not exact convergence, it is not computationally feasible to ascertain if the targets in those cases are at all attainable due to the sheer size of the problem.

REFERENCES

1. S. Bhattacharyya and G.J. Koehler, An analysis of non-binary genetic algorithms with cardinality 2^v, *Complex Systems* **8**, 227–256, (1994).
2. D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA, (1989).
3. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, (1992).
4. L. Davis, *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann, Los Altos, CA, (1987).
5. C.R. Rao, *Linear Statistical Inference and Its Applications*, Second edition, John Wiley, (1973).
6. W. Feller, *An Introduction to Probability Theory and Its Applications*, Volume I, Third edition, John Wiley and Sons, (1986).
7. M.R. Garey and D.S. Johnson, *Computers and Intractability—A Guide to the Theory of NP-Completeness*, Freeman, New York, (1979).
8. Y.H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Reading, MA, (1989).
9. J.L. Peterson and A. Silberschatz, *Operating System Concepts*, Addison-Wesley, Reading, MA, (1983).
10. F.P. Preparata and M.I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, New York, (1985).
11. C.H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Eaglewoods Cliffs, NJ, (1982).
12. M. Ohlsson, C. Peterson and B. Soderberg, Neural networks for optimization problems with inequality constraints—The knapsack problem, In *Neural Computation*, Orient Longmann, Great Britain, (1991).
13. C.R. Reeves, *Modern Heuristic Techniques for Combinatorial Problems*, Orient Longman, Great Britain, (1993).

14. D.E. Goldberg and R.E. Smith, Nonstationary function optimization using genetic algorithms with dominance and diploidy, In *Second International Conference on Genetic Algorithms*, Lawrence Erlbaum, Hillsdale, NJ, 1987, pp. 59–68.
15. J.J. Grefenstette, Editor, *Proceedings of the Second International Conference on Genetic Algorithms*, Lawrence Erlbaum Associates, Hillsdale, NJ, (1987).
16. A. Fairley, Comparison of methods of choosing the crossover point in the genetic crossover operation, Ph.D. Thesis, (1991).
17. P. Dutta and D. DuttaMajumder, Convergence of an evolutionary heuristic and its application in 0-1 knapsack problem, In *Sixth National Seminar on Theoretical Computer Science*, (India), p. 119, Allied Publishers, (1996).
18. G. Rudolph, Convergence analysis of canonical genetic algorithms, *IEEE Transactions on Neural Networks* 5 (1), 96–101, (1994).
19. U.K. Chakraborty and D. GhoshDastidar, Using reliability analysis to estimate the number of generations to converge in genetic algorithms, *Information Processing Letters* 46, 199–209, (1993).
20. P. Dutta and D. DuttaMajumder, An evolutionary algorithm and its convergence, Communicated to Complex Systems, (1996).
21. P. Dutta and D. DuttaMajumder, Convergence of an evolutionary algorithm, In *Proceedings of the Fourth International Conference on Soft Computing*, IIZUKA-96, Japan, 1996, Volume 2, pp. 515–518.
22. P. Dutta and D. DuttaMajumder, Performance comparison of two evolutionary schemes, In *Proceedings of 13th International Conference on Pattern Recognition*, 1996, Track D.
23. P. Billingsley, *Probability and Measure*, Second edition, John Wiley and Sons, (1991).