Survey of Software Quality Project Management through Genetic Algorithm with Backtracking Algorithm

R. Rajesh
M.C.A Department, RVS College of Engineering & Technology, Dindigul, Tamilnadu, INDIA

ABSTRACT
This paper discusses the essence of Software Quality Project Management through genetic algorithm and Backtracking algorithm. It identifies the actual quality of the software or an application or a project approximately by applying Back Tracking concepts in genetic algorithm. At first a simple survey is taken out, about GA and Backtracking algorithm. Which is compared and given an individual analysis report for the survey.

Keywords: Software quality project management, Backtracking, Genetic algorithms.

I. INTRODUCTION

Software Quality Project Management:
Quality:
1.1.1. Measurement is an ideal mechanism for feedback and evaluation that are needed by any engineering process. Some people thought that measurement must be focused upon goals and models.
1.1.2. Quality requirements for University construction projects are established first by The Regents, next by the President, and then by the Facilities. Quality is measured as conformance to requirements established by facilities management personnel. Project quality is ultimately appraised.
Background:
In The recent years of rapid computer growth, software development has not enjoyed the same degree of success as hardware development. Large software development projects have frequently failed to meet expectations and are often completed over schedule, over budget or both. Furthermore, once the product is delivered it frequently fails to meet the expectations of the user. What has not occurred during this growth period is the involvement of methodologies to adequately manage the software development process. The software development problem can largely be attributed to this lack of evolving methodologies in software development. Since software has now passed hardware as the critical component in the success of many computer systems, better techniques in the management of software development become imperative. [8]

1.1.3. The concept of genetic algorithms
Genetic algorithms mimic natural evolution, by acting on a population to favor the creation of new individuals that ‘perform’ better than their predecessors, as evaluated using some criteria, such as an objective function. At any given generation (that is, population), the algorithm has a pool of trial solutions. A population can consist of from as low as 20 to several hundred individuals. These individuals compete for an opportunity to reproduce. Reproduction will propagate some of an individual’s characteristics (traits) into the next generation. Candidates for reproduction are chosen probabilistically, but in a manner that should favor individuals whose offspring will perform well. [1]

1.1.4. BACKTRACKING APPROACHES
a) Backtracking, first named by D. H. Lehmer, is a well-known algorithm design technique for the solution of a class of problems that deal with searching for a set of feasible solutions or an optimal solution satisfying some constraints [2]. It incrementally attempts to extend a partial solution that specifies consistent values for some of the variables, towards a complete assignment by repeatedly choosing a value for another variable consistent with the values in the current partial solution [3]. A general algorithm for backtracking [2] is given below.
Backtrack(\(k\))

\{
    \text{for (each } x[k] \in \mathcal{T}(x[1], \ldots, x[k-1]) \text{ do )}
    \{
        \text{if } (Bk(x[1], x[2], \ldots, x[k]) \neq 0) \text{ then }
        \{
            \text{if } (x[1], x[2],\ldots, x[k] \text{ is a path to answer node) then write}(x[1:k]);
            \text{if } (k<n) \text{ then Backtrack}(k+1);
        \}
    \}
\}
In this algorithm, $T$ is a function that accepts a partial solution to the problem $(x[1], \ldots, x[k-1])$ and returns a set of values consistent with this partial solution for variable $x[k]$. Also, $B_k$ is a bounding function that returns a non-zero value if the newly formed partial solution, i.e., $x[1], x[2], \ldots, x[k]$, can lead to a feasible solution to the problem. The chronological backtracking is a simple implementation of backtracking in which the algorithm always chronologically backtracks to the most recently instantiated variable. Thus, the search is similar to a tree traversal. The major problems with this approach are lot of redundant work and late detection of conflicts. This often leads to poor performance and thrashing. Several variations of chronological backtracking algorithm are proposed in the past to speed up the search process. These include backjumping, backmarking and forward checking. They are explained below in brief. Kondrak [4] has provided more details about these techniques. Several researchers have further improved these approaches. Prosser [5, 6] has proposed hybrid backtracking algorithms by combining two or more basic algorithms. Kondrak [4] has proposed theoretical method for evaluation of selected basic and hybrid backtracking algorithms. Bartak [2] has reported different search techniques for solving constraint satisfaction problem such as systematic search algorithm, consistency techniques, constraint propagation, variable and value ordering and reducing search.

1.1.5. Genetic algorithms:

In this article we use a GA to solve the PSP, and thus a discussion of this kind of metaheuristic is appropriate in order to make this work self contained. Genetic Algorithms (GAs) are stochastic search methods that have been successfully applied in many search, optimization, and machine learning problems in the past [1]. Unlike other optimization techniques, GAs maintain a population of encoded tentative solutions that are competitively manipulated by applying some variation operators to find a global optimum. To achieve this goal the problem variables are encoded (binary or floating point, for example) into what are called the chromosomes, which are merged and manipulated by the genetic operators to improve their associated quality (called the fitness). Thus, one individual is composed of one chromosome and its associated fitness, and the set of individuals forms the population used by the algorithm. Population-based algorithms contrast with trajectory-based ones (like simulated annealing) in that they search from multiple points at the same time, thus reducing the probability of getting stuck in local optima; in addition, they can offer multiple optima to the same problem, an interesting feature that the researchers can use to have an assorted set of solutions to the problems at hand. After creating an initial set of solutions (randomly or by using a seeding algorithm) GAs normally apply a crossover operation to combine the contents of two parents forming a new solution. This will be modified later by the mutation operation which alters some of the contents of the individual. Not all the individuals participate in the reproduction, only the fittest ones (elitism is very common) are selected from the population by a

1.1.6. Development of QA Scheme

The GASD model produced a large number of solutions that were better when compared to other methods. Since the DE-A is a well-studied project, it is possible to draw comparisons with other solutions that have been developed to this problem. A comparison of the solution arrived at by the GA with other experiments is contained in Table 4-1 [8:p. 73]. The manually derived method involved the user input of QA schemes into the DYNAMICA simulation with manual perturbations introduced. The prototype expert simulator used an expert system module incorporating heuristic rules that is interfaced with the DYNAMICA simulation. The pattern search expert simulator is a refinement of the prototype expert simulator that identifies patterns to make further refinements in improvement. [9:pp. 17-20,37]
II. GENERATIONS

The GA evolves from generation to generation by processing a population of strings during each new generation to produce a successive population. Each new generation is produced by using operations that are modeled after the Darwinian principals of reproduction incorporating survival of the fittest techniques [Ref. 10:p. 10]. It is evolutionary in nature and in each generation, several distinct activities occur in a step by step process. These activities include reproduction, crossover and mutation. It is through these processes that the GA is able to arrive at a solution. [Ref 11:p. 10]

III. REPRODUCTION

The mechanics of a genetic algorithm are not difficult to understand. The first step in the process is to build a string of characters. The string will be representative of some value or condition that can exist (describes the problem variable) and in genetic terms would be the equivalent of a chromosome. To illustrate the mechanics of a genetic algorithm, an example will be used [Ref 11:pp. 18-30]. This example will be artificially simple so the functioning of the algorithm can be clearly presented. In this example, the aim will be to maximize the number from a string 3 characters in length. As is traditional with the GA, binary representation of these strings will be used. Given these parameters, the strings would have the makeup of binary characters ranging from 000 to 111.

IV. IMPLEMENTATION

Classification by implementation
An algorithm may be implemented according to different basic principles.
Recursive or iterative
A recursive algorithm is one that calls itself repeatedly until a certain condition matches. It is a method common to functional programming. Iterative algorithms use repetitive constructs like loops. This GA with BA generates a population of points at each iteration. The best point in the population approaches an optimal solution. Selects the next population by computation which uses random number generators.
Logical or procedural
An algorithm may be viewed as controlled logical deduction. A logic component expresses the axioms which may be used in the computation and a control component determines the way in which deduction is applied to the axioms. This is the basis of the logic programming. In pure logic programming languages the control component is fixed and algorithms are specified by supplying only the logic component.
Serial or parallel
Algorithms are usually discussed with the assumption that computers execute one instruction of an algorithm at a time. This is a serial algorithm, as opposed to parallel algorithms, which take advantage of computer architectures to process several instructions at once. They divide the problem into sub-problems and pass them to several processors. Iterative algorithms are generally parallelizable. Sorting algorithms can be parallelized efficiently.

Deterministic or non-deterministic
Deterministic algorithms solve the problem with a predefined process whereas non-deterministic algorithm must perform guesses of best solution at each step through the use of heuristics.

GA with BA Implementation
Implementation of Genetic Algorithm (GA) with Backtracking Algorithm (BA). It is a method for solving both constrained and unconstrained optimization problems based on a natural selection process that mimics biological evolution. The algorithm repeatedly modifies a population of individual solutions. At each step, the GA with BA randomly selects individuals from the current population and uses them as parents to produce the children for the next generation. Over successive generations, the population "evolves" toward an optimal solution. We can apply the GA with BA to solve problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, no differentiable, stochastic, or highly nonlinear. Diversity refers to the average distance between individuals in a population. A population has high diversity if the average distance is large; otherwise it has low diversity. In the following figure, the population on the left has high diversity, while the population on the right has low diversity.

V. CONCLUSION

It is clearly said that Genetic algorithmic are process for finding out the best quality for the software. With the help of GA algorithm and implementing Backtracking algorithm inside GA algorithm that is GA with BA has gained the same result of the same results of global minima for highly nonlinear problems. Thus these algorithms can be implemented for the analysis of the Quality Software Project Management phases and it even be enhanced better for the same.
REFERENCES