Computers & Operations Research 39 (2012) 521–540

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Inequality constraint handling in genetic algorithms using a boundary simulation method

Xiang Li *, Gang Du

School of Management, Tianjin University, Tianjin 300072, People's Republic of China

ARTICLE INFO

Available online 9 April 2011

Keywords:
Genetic algorithms
Constrained optimization
Constraint handling
Inequality constraints
Equality constraints
Boundary simulation method
Binary search method

ABSTRACT

Constraint handling is one of the major concerns when applying genetic algorithms (GAs) to solve constrained optimization problems. This paper proposes a boundary simulation method to address inequality constraints for GAs. This method can efficiently generate a feasible region boundary point set to approximately simulate the boundary of the feasible region. Based on the results of the boundary simulation method, GAs can start the genetic search from the boundary of the feasible region or the feasible region itself directly. Furthermore, a series of genetic operators that abandon or repair infeasible individuals produced during the search process is also proposed. The numerical experiments indicate that the proposed method can provide competitive results compared with other studies.

1. Introduction

The constrained optimization technique is a hot topic in operations research area. Generally, there are three types of constraints, namely linear inequality, nonlinear inequality and nonlinear equality constraints, as linear equality constraints can be easily converted and added into other type constraints. In the remainder of this paper, inequality constraints will refer to linear and nonlinear inequality constraints, while equality constraints will only refer to nonlinear equality constraints. The aim of this paper is to provide a new constraint handling method for genetic algorithms (GAs) to address inequality constraints that normally exist in practical problems.

In this paper, we concentrate on constrained optimization problems that can be represented as the following expression:

$$\begin{aligned} &\text{Minimize} \quad f(x) \\ &\text{Subject to} \quad \begin{cases} g_j(x) \le 0, & j=1,2,\ldots,m, \\ x_i^l \le x_i \le x_i^u, & i=1,2,\ldots,n, \end{cases} \end{aligned} \tag{1}$$

where $x=(x_1, x_2,\ldots,x_n)$ is the solution vector, $f(x)$ is the objective function, $g_j(x)$ is the j-th inequality constraint, while x_i^l and x_i^u are the lower and upper bounds of the independent variable x_i, respectively.

It should be emphasized that the following conditions are assumed.

1. The feasible region is connected.
2. The problem only involves inequality constraints.
3. All the independent variables have an lower and upper bounds.

When solving constrained optimization problems, traditional deterministic optimization techniques generally require the problem under consideration to possess certain mathematical properties, such as continuity, differentiability and convexity, which may be difficult to satisfy in practical problems. These requirements severely limit the applicability of these traditional approaches. Furthermore, these traditional approaches usually lack global search capabilities for non-convex problems, as any local optimal point can satisfy their convergence condition.

GAs are a stochastic search technique inspired by natural selection and natural genetics [1,2]. In the past decades, GAs have been extensively used for solving a wide variety of problems, in which traditional approaches may not work adequately. Compared with traditional approaches, GAs have the following advantages.

1. GAs do not require the objective function to be continuous or differentiable.
2. GAs have good robustness for many applications.
3. GAs have outstanding global search capabilities for convex and non-convex problems.
4. GAs have inherent parallel processing capabilities.
5. GAs are easy to implement.

However, on the other hand, GAs also have some limitations. GAs are essentially an unconstrained optimization technique. Although GAs perform well for unconstrained or simple constrained optimization problems, they may encounter some difficulties when applied to highly constrained problems. It is very

* Corresponding author.
E-mail address: tju.som.li.xiang@gmail.com (X. Li).

0305-0548/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2011.03.019

likely that individuals generated by traditional genetic operators (i.e., crossover and mutation) would violate certain constraints. A special treatment of constraints is usually required to efficiently find the feasible region and then to prevent the genetic search from leaving it.

In this paper, we propose a boundary simulation method to address inequality constraints for GAs. This method allows GAs to start the genetic search from the boundary of the feasible region or the feasible region itself directly, even for highly constrained problems that always have a very small feasible region. Furthermore, a series of genetic operators that abandon or repair infeasible individuals produced during the search process is also proposed to ensure that the genetic search stays in the feasible region.

The structure of this paper is as follows. Section 2 is a literature review, which provides a brief introduction to status of current research on this topic. Section 3 describes the proposed boundary simulation method in detail. Section 4 combines the boundary simulation method and GAs to form a new constrained optimization algorithm called boundary simulation genetic algorithm (BSGA). Section 5 applies BSGA to 12 test problems to investigate the feasibility and efficiency. Our conclusions are presented in Section 6. Finally, some possible directions for future research are discussed in Section 7.

2. Literature review

2.1. Previous constraint handling techniques

In order to handle constraints, many techniques have been proposed in the past decades. According to Michalewicz [3–5], Eiben [6], Coello [7] and Salcedo-Sanz [22], the existing constraint handling techniques can be classified into five different types, namely penalty methods, special representation and operator methods, repair methods, separation of objective and constraints methods, and hybrid methods.

2.1.1. Penalty methods

The main idea behind the penalty method is to transform a constrained optimization problem into an unconstrained one by adding a constraint violation measure to the objective function as a penalty term. Due to the simplicity of its implementation, this approach is very popular. However, its performance is not always satisfactory. Furthermore, in most cases, it is difficult to set the penalty coefficients properly. As reported by Deb [15], a large penalty value may cause convergence of the algorithm to some local optimal points; in addition, a small penalty value may cause the algorithm to converge to an infeasible point or to spend time exploring the infeasible region. Richardson [8] proposed some guidelines to set the penalty coefficients properly. However, these guidelines are still difficult to apply in some cases.

In order to overcome this obstacle, several techniques have been proposed that use a dynamic or an adaptive penalty scheme [9–14]. In these methods, the penalty coefficients could be adapted according to the degree of constraint violation, the overall performance of the genetic search or the generation number. For some problems, the dynamic or adaptive penalty method works well. However, these methods usually require another set of parameters to tune the penalty coefficients automatically.

Deb [15] proposed a penalty method that uses a binary tournament selection operator to handle constraints. In his paper, the following rules were adopted when comparing two individuals.

1. Any feasible individual is preferred to any infeasible individual.
2. Among two feasible individuals, the one having better objective function value is preferred.
3. Among two infeasible individuals, the one having smaller constraint violation is preferred.

This method does not require any penalty coefficients. However, it requires other techniques to maintain the population diversity, which always causes some extra computation costs.

2.1.2. Special representation and operator methods

This method essentially consists in using a specially designed coding scheme to preserve the feasibility of individuals. This method is very efficient for its intended applications [16–20]. However, it strongly depends on some special characteristics of the problem under consideration. Due to the special design of the representation scheme, the specially designed genetic operators are always required to work properly. The proposed representation scheme and the corresponding operators may only be applicable to the intended problem. Furthermore, prior knowledge of the problem is generally required to design an appropriate representation scheme and the corresponding genetic operators. Sometimes the development of the special representation scheme and the corresponding genetic operators may be difficult or even impossible.

2.1.3. Repair methods

The main idea behind this method is to repair infeasible individuals by using a specially designed repair procedure. In other words, an infeasible individual will be replaced by a nearest feasible individual [21]. For some problems that only involve simple constraints, such as non-negativity or simple bounds, the infeasible individuals can be repaired easily. However, practical problems may contain some complex constraints. In such circumstances, repairing the infeasible individuals can be as complex as solving the original problem. Sometimes the repairing cost may be expensive, computationally speaking. Furthermore, prior knowledge of the problem is generally required to design an efficient repair procedure. For a comprehensive survey of repair methods, the reader is referred to [22].

2.1.4. Separation of objective and constraints methods

This method essentially consists in using co-evolution, multi-objective or other optimization techniques to handle the objective and the constraints separately. Paredis [23] proposed a co-evolutionary method, in which there are two populations. The first population contains the constraints to be satisfied, while the second contains potential solutions. Using the analogy of the predator–prey model, the selection pressure on members of one population depends on the fitness of the members of the other population.

Multi-objective optimization approaches treat the feasibility as another objective of the problem, i.e., they transform the original constrained single-objective problem into an unconstrained multi-objective problem. Consequently, all the well-developed multi-objective optimization techniques can be used to rank the individuals. Based on the ranking result, the selection and other genetic operators can be performed [24–28].

2.1.5. Hybrid methods

The main idea behind the hybrid method is to combine GAs with other numerical optimization techniques (e.g., Lagrangian multiplier, fuzzy logic or simulated annealing) to handle constraints. Like penalty methods, these methods generally require several parameters to work properly [29–32].