Journal of Constructional Steel Research 67 (2011) 1453–15462

Contents lists available at ScienceDirect

Journal of Constructional Steel Research

A unified approach to parameter selection in meta-heuristic algorithms for layout optimization

A. Kaveh [a,*], N. Farhoudi [b]

[a] Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran-16, Iran
[b] Khajeh Nasir Toosi University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 17 May 2010
Accepted 16 March 2011
Available online 30 April 2011

Keywords:
Layout optimization
Steel braced frames
Convergence factor
Heuristic algorithms
Genetic algorithm
Ant colony optimization
Particle swarm
Big Bang–Big Crunch

ABSTRACT

Meta-heuristic optimization algorithms have attracted many researchers in the last decade. Adjustment of different parameters of these algorithms is usually a time consuming task which is mostly done by a trial and error approach. In this study an index, namely convergence factor (CF), is introduced that can show the performance of these algorithms. CF of an algorithm provides an estimate of the suitability of the parameters being set and can also enforce the algorithm to adjust its parameters automatically according to a pre-defined CF.

In this study GA, ACO, PSO and BB–BC algorithms are used for layout (topology plus sizing) optimization of steel braced frames. Numerical examples show these algorithms have some similarities in common that should be taken into account in solving optimization problems.

1. Introduction

Scarcity of structural materials and the need for efficiency in today's competitive world have forced engineers to evince greater interest in economical designs for structures. The meta-heuristic algorithms provide efficient tools for performing structural optimum designs and this is why these methods have been extensively employed in the field of structural engineering.

The first meta-heuristic algorithm was genetic algorithm (GA) which was introduced in 1957 [1]. This algorithm uses the Darwin's theory of evaluation and has been employed in solving various types of engineering problems as well as structural design [2–4].

On the other hand, ant colony optimization (ACO) is principally inspired by the rules governing the behavior of real ants in finding their roots. Such a natural process was first simulated in numerical methods in the pioneering work of Dorigo et al. [5]. Since then, its performance has been studied in several optimization problems [6–8].

Particle swarm optimization (PSO), developed in 1995 [9], is another meta-heuristic algorithm that mimics the rules that bees and birds obey, finding their ways to the food source. This algorithm has been used in structural design in the works of [10–15].

Big Bang–Big Crunch (BB–BC) is one of the most recent algorithms developed by [16]. It simulates the Big Bang theory and recently it is employed in structural design [17–19].

Most of the works applied to structural optimization are on sizing of cross sections of trusses and frames. The present study deals with the layout optimization of dual system of moment frame together with X-bracing. Optimizing such systems, adds two complications to the problem that are not present in the other types of structural systems:

1. In this kind of structural system, two types of elements have to be optimized, bars and beam-columns, where each has its own requirements to be satisfied.
2. Degree of statical indeterminacy is higher compared to the moment frames and trusses. Increase in the degree of indeterminacy may cause the internal forces to be much dependent to each other. Thus when the problem is to optimize topology and sizing together, a small change in topology will cause big changes in the internal forces and hence size of elements. Therefore the ability of the optimization algorithm in finding the globally best answer rather than local one will be important.

On the other hand adjusting the algorithm parameters for proper optimization is an important step dealing with GA, ACO, PSO or BB–BC. However, no important suggestion is made up to now for adjusting these parameters and it is mainly done by trial and error approaches. In the present study an index is presented that is called

* Corresponding author.
E-mail address: alikaveh@iust.ac.ir (A. Kaveh).

0143-974X/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jcsr.2011.03.019

the *convergence factor* (CF). It is shown how CF helps one to set the optimization parameters and how each of the algorithms can adjust themselves properly.

In the first part of this paper optimization algorithms and their parameters are introduced. In the second part, formulation for the optimization is presented, and the subsequent part is devoted to the method used for implementing some building code requirements. In the fifth part, the CF that is the main contribution of the present work is introduced. In the sixth part, numerical examples are presented. This part is followed by implementation of the results and conclusion.

2. Meta-heuristic algorithms

2.1. Genetic algorithm

The main steps of genetic algorithm are as follows:

1. Initiation: Answers that are mimicked as chromosomes are selected randomly all around the search space.
2. Fitness based selection: Chromosomes that have better fitness are selected for the production of the next generation.
3. Crossover: Selected chromosomes of the previous step as parents make two new chromosomes of the next generation in a way that each gene of every new chromosome is from one of the parents.
4. Mutation: Genes of a chromosome change randomly with a possibility that is called here as “Mutation Rate”.
5. Convergence check: If the convergence criterion is achieved, the algorithm will end.

2.2. Ant colony optimization

The main steps of the ant colony optimization algorithm are as follows:

1. Initiation: Answers that are mimicked as the path that ants choose from nest to food source are selected randomly all around the search space. Here it should be noticed that ants trace a kind of hormone called pheromone on their way and as the pheromone on a specific path increases the possibility of the path to be chosen becomes higher. In this algorithm for representing the intensity of pheromone on each path, the τ matrix is defined. The matrix has m rows and n columns where m is the number of alternatives that can be chosen for each path and n is the number of paths. τ is the filling from the first loop of the algorithm to the end and it acts as the memory of the algorithm. In this step, in order to have equivalent chance for each alternative, all arrays of τ matrix are filled by an equivalent initial value of τ_0 using the following relationship:

$$\tau_{ij}(t) = (1-\varphi)\cdot\tau_{ij}(t) + \varphi\tau_0 \tag{1}$$

where φ is a parameter between 0 and 1.
2. Evaporation: The pheromone trail evaporation is performed on each edge after it is passed by an ant using the relationship (2):
3. Fitness based pheromone trail intensity adjustment: τ is filled in a way that higher fitness gets higher possibility to be chosen in the next step. Thus the value of $\Delta\tau_{ij}$ is defined for each ant's crossed way, according to the related answer, in a way that better answers achieve greater $\Delta\tau_{ij}$. Therefore the trial intensity is adjusted using the following formula:

$$\tau_{ij}(t+n) = \rho\cdot\tau_{ij}(t) + \Delta\tau_{ij} \tag{2}$$

where ρ is a parameter between 0 and 1, chosen such that $(1-\rho)$ represents the evaporation of pheromone between the time t and t + n (the amount of time required to complete a cycle).

4. Probability calculation: a_{ij}, or ant decision table is calculated as follows:

$$a_{ij}^{k}(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha}\cdot\left[v_{ij}\right]^{\beta}}{\sum_{l\in allowed}^{n}\left[\tau_{ij}(t)\right]^{\alpha}\cdot\left[v_{ij}\right]^{\beta}} \tag{3}$$

where α and β are optimization parameters, and ν_{ij} is the visibility that is defined according to the following formula:

$$v_{ij} = \frac{1}{d_{ij}}. \tag{4}$$

The probability p_{ij}^{k} of ant k choosing a path from i to j at time t is:

$$p_{ij}^{k} = \frac{a_{ij}(t)}{\sum_{l\in allowed_k} a_{il}(t)}. \tag{5}$$

5. Convergence check: Once the convergence criterion is satisfied, the algorithm ends.

2.3. Particle swarm

The main steps of particle swarm optimization algorithm are:

1. Initiation: Answers that are mimicked as the position of each particle in the swarm are selected randomly all around the search space.
2. Fitness based position update: Velocity vector is defined to update the current position of each particle in the swarm. The velocity vector is updated based on the “memory” gained by each particle, conceptually resembling an autobiographical memory, as well as the knowledge gained by the swarm as a whole. Thus the position of each particle in the swarm is updated based on the social behavior of the swarm which adapts to its environment by returning to promising regions of the space previously discovered and searching for better positions over time. Numerically, the position x of a particle i at iteration $k+1$ is updated as

$$x_{k+1}^{i} = x_{k}^{i} + v_{k+1}^{i}\Delta t \tag{6}$$

where v_{k+1}^{i} is the corresponding updated velocity vector, and Δt is the time step value. Throughout the present work a unit time step is used. The velocity of each particle is calculated as defined by Eq. (7),

$$v_{k+1}^{i} = \omega v_{k}^{i} + c_1 r_1\frac{\left(P_k^i - x_k^i\right)}{\Delta t} + c_2 r_2\frac{\left(P_k^g - x_k^i\right)}{\Delta t} \tag{7}$$

where v_k^i is the velocity vector at iteration k, r_1 and r_2 represent random numbers between 0 and 1; p_k^i represents the best position of particle i in the kth iteration, and p_k^g corresponds to the global best position in the swarm up to iteration k. ω is a dynamic parameter that is linearly decreased with each algorithm iteration as the following [20]:

$$\omega_{k+1} = \omega_{max} - \frac{\omega_{max}-\omega_{min}}{k_{max}}k. \tag{8}$$

c_1 and c_2 are parameters that have the constant sum of 4.
3. Convergence check. Once the convergence criterion is achieved the algorithm ends.