Computers and Structures 182 (2017) 137–148

Contents lists available at ScienceDirect

Computers and Structures

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

Efficient truss optimization using the contrast-based fruit fly optimization algorithm

Stratis Kanarachos *, James Griffin, Michael E. Fitzpatrick

Faculty of Engineering, Environment and Computing, Coventry University, Priory Street, Coventry CV1 5FB, United Kingdom

ARTICLE INFO

Article history:
Received 29 April 2016
Accepted 16 November 2016

Keywords:
Fruit fly optimization
Multi-parameter
Truss optimization

ABSTRACT

A recent biological study shows that the extremely good efficiency of fruit flies in finding food, despite their small brain, emerges by two distinct stimuli: smell and visual contrast. "contrast-based fruit fly optimization", presented in this paper, is for the first time mimicking this fruit fly behaviour and developing it as a means to efficiently address multi-parameter optimization problems. To assess its performance a study was carried out on ten mathematical and three truss optimization problems. The results are compared to those obtained using twelve state-of-the-art optimization algorithms and confirm its good and robust performance. A sensitivity analysis and an evaluation of its performance under parallel computing were conducted. The proposed algorithm has only a few tuning parameters, is intuitive, and multi-faceted, allowing application to complex n-dimensional design optimization problems.

1. Introduction

Design optimization is a powerful tool widely utilised by engineers to produce better performing, more reliable and cost-effective products. It originated from the aircraft industry and rapidly expanded in multiple domains like structural and mechatronics engineering [1–3]. Its success is mainly due to its inherent merit, delivered in combination with a significant increase in computational power and accessibility to practitioners through commercial engineering software [4]. The mathematical formulation of an optimization problem can be expressed as:

$$\begin{aligned} &\textit{minimize } \mathrm{f}(\mathbf{x}) \\ &\textit{subject to}: \ \mathrm{h}_{\mathrm{eq\_j}}(\mathbf{x}) = 0, \quad eq\_j = 1, 2, \ldots, n_{eq} \\ &\qquad\qquad\quad \mathrm{g}_{\mathrm{ineq\_j}}(\mathbf{x}) \leqslant 0, \quad ineq\_j = 1, 2, \ldots, n_{ineq} \end{aligned} \tag{1}$$

where $\mathrm{f}(\mathbf{x})$ is the objective function that expresses the performance of a system, $\mathbf{x}$ is a vector comprised out of m design variables, n_{eq} and n_{ineq} are the number of equalities and inequalities described by the functions $\mathrm{h}_{\mathrm{eq\_j}}$ and $\mathrm{g}_{\mathrm{ineq\_j}}$ respectively.

Initially, optimization technology – including the steepest descent, the conjugate gradient, the Newton and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) methods – was based on mathematical formulations involving the calculation of first and/or second derivatives [5,6]. For example, in the gradient descent method, one starts from an initial point $\mathbf{x}_0$ where the function value $\mathrm{f}(\mathbf{x}_0)$ is calculated and then takes a step in a downward direction, where the function value will be lower. To make such a step, one utilises local information $\nabla f^T(\mathbf{x}_0)$ and explores the immediate vicinity of the current point. The search for the optimum design vector $\boldsymbol{x}^*$ is expressed by the following iterative formula:

$$\mathbf{x}[\mathrm{k}+1] = \mathbf{x}[\mathrm{k}] - a[k] \cdot \nabla f^T(\mathbf{x}[\mathrm{k}]) \tag{2}$$

where $a[k]$ is a scaling parameter, k is the iteration number, $\mathbf{x}[k]$ is the design vector in kth iteration and $\mathbf{x}[\mathrm{k}+1]$ is the new design vector. It is highlighted that gradient-based optimization methods converge to the optimum value in only a few iterations and are considered to be the best approach for solving many optimization problems, at least in a local context.

Although mathematically rigorous, gradient-based algorithms get trapped in local minima in the case of noisy or highly nonlinear problems. By contrast, meta-heuristic optimization algorithms like the *Genetic Algorithm* [7], *Particle Swarm Optimization* [8] and *Harmony Search* [9] do not use gradient information and are better suited for global optimization problems. On the downside, the performance of non-gradient algorithms depends on a number of tuning parameters which are not known *a priori*.

Although, in some cases, for the meta-heuristic algorithms empirical rules exist, they are not always adequate. There is a need for intuitive meta-heuristic algorithms with a minimum number of tuning parameters.

* Corresponding author at: Engineering & Computing Building - EC 4-07, Faculty of Engineering, Environment and Computing, Coventry University, 3, Gulson Road, Coventry CV1 2JH, United Kingdom.

E-mail addresses: stratis.kanarachos@coventry.ac.uk (S. Kanarachos), ac0393@coventry.ac.uk (J. Griffin), ab6856@coventry.ac.uk (M.E. Fitzpatrick).

http://dx.doi.org/10.1016/j.compstruc.2016.11.005
0045-7949/

2. Brief literature review on truss optimization

Trusses are fundamental in structural engineering and applications can be found from nano to macro levels [10,11]. Truss optimization problems are usually multi-parameter optimization problems owing to the large number of members comprising the truss. They are also highly nonlinear because of the multiple constraints considered, including displacement, stress and natural frequency, and the complex interaction between the structural members. In the general case, the truss optimization problem is formulated as a mathematical optimization problem:

$$
\begin{aligned}
&\textit{find design vector } \mathbf{x}\\
&\textit{that minimizes } \mathrm{R}(\mathbf{x}) = \sum_{q=1}^{n} \rho_q \cdot A_q \cdot L_q\\
&\textit{subject to}: \ \delta_{qmin} \leqslant \delta_q \leqslant \delta_{qmax}, \quad q = 1,2,\ldots,n_n\\
&\qquad \sigma_{qmin} \leqslant \sigma_q \leqslant \sigma_{qmax}, \quad q = 1,2,\ldots,n_n\\
&\qquad \mathrm{A}_{qmin} \leqslant \mathrm{A}_q \leqslant \mathrm{A}_{qmax}, \quad q = 1,2,\ldots,n\\
&\qquad \omega_{qmin} \leqslant \omega_q \leqslant \omega_{qmax}, \quad q = 1,2,\ldots,n_\omega
\end{aligned}
\tag{3}
$$

where R is the mass of the truss, n is the number of truss members, n_n is the number of nodes, and n_ω the number of desired natural frequencies. ρ_q, A_q and L_q are the density, cross sectional area and length of qth member respectively. δ_q and σ_q are the displacement and stress at the qth node. ω_q is the qth natural frequency. δ_{qmin} and δ_{qmax} are the lower and upper displacement bounds for the qth node, σ_{qmin} and σ_{qmax} are the lower and upper normal stress bounds for the qth node, A_{qmin} and A_{qmax} are the lower and upper cross sectional area bounds for the qth structural member and ω_{qmin} and ω_{qmax} are the lower and upper bounds for the qth natural frequency.

There is an increasing interest in developing efficient algorithms for large-scale truss optimization. The algorithms are mainly meta-heuristic and broadly classified into three categories.

The first category encompasses the Evolutionary Algorithms (EA). EAs use mechanisms inspired by biological evolution, such as reproduction, mutation, recombination, and selection for calculating new candidates. EAs usually suffer from premature convergence and weak exploitation capabilities. Both drawbacks are compensated by choosing bigger populations, however, this leads to larger computational cost. Wei et al. [12] proposed, as a solution to this problem, the Niche Hybrid Parallel Genetic Algorithm (NHPGA). NHPGA aims to effectively combine the robust and global search characteristics of the genetic algorithm, the strong exploitation ability of Nelder-Mead's simplex method, and the computational speed of parallel computing.

The *second category* encompasses physical algorithms that resemble an employed physical process. For example, Kaveh and Bakhshpoori [13] developed an algorithm that mimics the evaporation of a tiny amount of water molecules on a solid surface with different wettability. The "*Water Evaporation Optimization Algorithm*" was tested and analysed in comparison to other existing methods on a set of 17 benchmark unconstrained functions, a set of 13 classical benchmark constraint functions, and three benchmark constraint engineering problems. The results obtained indicate that the proposed technique is highly competitive. The performance of the algorithm depends on a number of parameters, including the assumption of a monolayer and droplet evaporation phase, the number of water molecules and the minimum and maximum values of monolayer and droplet evaporation probabilities. Another example is the modified *Teaching-Learning-based optimization* (*TLBO*) algorithm [14]. *TLBO* mimics the two types of pedagogy in a classroom to find the optimum solution: class-level learning from a teacher, and individual learning between students. *TLBO* uses a relatively simple algorithm with no intrinsic parameters controlling its performance.

The third category includes population-based algorithms, such as *Particle Swarm Optimization* (*PSO*) [15]. *PSO* is formulated by mathematically modelling the social behaviour of birds and fish colonies in finding food resources or escaping from predators. In the standard *PSO* each member of the swarm finds its way based on their own experience and the best particle's position: particles do not exchange any information. This causes *PSO* to get trapped into local optima. In a recent publication by Mortazavi and Toğan [16] a new version of *PSO* was proposed. In this version, the concept of a weighted particle, created by exploiting all particles' experiences, is introduced. This helps to avoid premature convergence.

Other popular population-based algorithms are the artificial bee colony, the ant colony and the bacterial algorithm. In [17] the *Artificial Bee Colony* algorithm (*ABC*) is applied to truss optimization problems. *ABC* models the honeybee foraging behaviour in the natural environment. In a bee colony, the female bees start to search for food randomly. After finding a food source, the bee returns to the hive and informs her nest mates about her findings. The information concerns the food source; the direction in which it can be found; distance from the hive; and its quality. In a decentralised and intelligent manner, some of the bees follow their nest mates to the food source, while others search for food independently.

The ant colony optimization method (*ACO*) is employed in [18] for solving truss optimization problems with cardinality constraint. *ACO* models the food search behaviour of real ants. Ants deposit pheromone on the ground to mark their path and to inform other ants about the food location. The more ants concentrate in an area, the more pheromone is laid, and this will attract even more ants. By contrast, locations with no food have lower levels of pheromone, which diminish over time owing to evaporation.

In [19] three different variants of the bacterial foraging optimization algorithm are presented for solving a 10-bar truss structural optimization problem. The bacterial foraging algorithm models the foraging behaviour of *Escherichia coli* bacteria, which is characterised by three phases: chemotaxis, reproduction, and elimination-dispersal. In the chemotaxis phase, every bacterium moves a single step towards a random position. If an improvement in the objective function is achieved, the bacterium continues moving in the same direction till a stopping criterion is met. In the reproduction phase the bacteria with bad performance are eliminated while those with good performance are replicated. The performance is determined by the sum of all chemotaxis steps performed. In the dispersion phase randomly-selected bacteria are substituted by other randomly generated bacteria.

Fruit fly optimization (*FOA*) is a recently developed population-based optimization algorithm [20]. Fruit flies can very effectively find food at very long distances. This, in combination with the fact that their brain is very simple (it has only 100,000 neurons compared to house fly brains that have 300,000 neurons and human brains which have 100 billion), makes them very interesting from a biological and optimization perspective [21,22]. It is well-known that the main food search mechanism of fruit flies is based on smell. However, a recent biological study reveals that the search mechanism is stimulated also by visual contrasts, which are irrelevant to smell. Furthermore, their motion is described by standardised distinct sensory-motor reflexes, independent of each other. The contrast-based *fruit fly optimization algorithm*, proposed in this paper, mimics those recently discovered elements of fruit fly food search behaviour. The algorithm is, first, evaluated on a set of standard mathematical benchmark tests and then applied to structural truss design benchmark problems.

It is highlighted that fruit fly algorithms have never been tested in structural optimization before. The results show that the algo-