Applied Soft Computing Journal 186 (2026) 114066

Contents lists available at ScienceDirect

Applied Soft Computing

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

The effect of elitist fitness-based selection on the escape from local optima

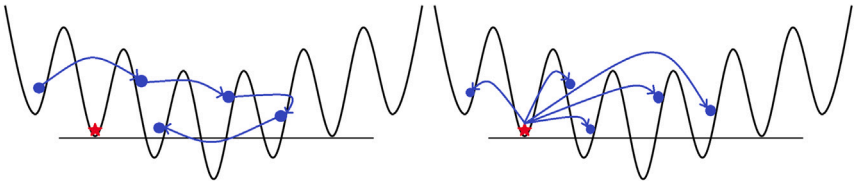
Stephen Chen

School of Information Technology, York University, 4700 Keele Street, Toronto, M3J 1P3, ON, Canada

HIGHLIGHTS

- A formal definition for escaping from local optima is developed for continuous domain search spaces.
- The effect of elitist fitness-based selection is shown to hinder the ability of many metaheuristics to escape from local optima in continuous domains.
- Key differences between combinatorial and continuous domains are presented.

GRAPHICAL ABSTRACT



The concept of escape from local optima is formally defined in continuous domains to mean that a previously found locally optimal solution is replaced by a search solution (i.e., it has survived selection) that is in a different attraction basin. Exploration that does not lead to an update is not considered to have achieved an escape from a local optimum. The search solutions (blue circles) are unable to update the previously found local optimum (red star) and thus allow the metaheuristic to escape from the local optimum.

ARTICLE INFO

Keywords:
Elitist fitness-based selection
Local optima
Attraction basin
Exploration
Continuous domain search spaces

ABSTRACT

Random Search is the baseline that a metaheuristic must improve upon to be worth its added complexity. Random Search, in the form of Hill Climbing, cannot escape from local optima. A key claim of many metaheuristics is that they are able to escape from local optima. However, these claims are poorly tested and often based on imprecise definitions of what it means to escape from a local optimum in continuous domain search spaces. A practical and precise definition for an escape from a local optimum is developed. It is then shown how elitist fitness-based selection can lead to the rejection of exploratory search solutions, and this can cause many popular metaheuristics to degrade into (localized) Random Search in their attempts to escape from local optima. The explosion of new metaheuristics has often been just a repeated re-invention of localized Random Search for the key task of escaping from local optima.

1. Introduction

There are many terms used within the field of metaheuristics that are often vague and/or generic. The generic nature of these terms is somewhat justified by the broad range of problem classes in which the same metaheuristic framework can be used. For example, evolutionary algorithms (EAs) can be applied to combinatorial optimization problems, numerical optimization in the continuous domain, multi-objective optimization problems, dynamic optimization problems, etc. It would be fair to say that all EAs perform exploration and exploitation, but the concepts of exploration and exploitation can be slightly different for each of these problem classes.

Another key term and concept is that of a local optimum. In combinatorial search spaces, local optima are operator dependent. For example, a solution which is a local optimum with respect to the 2-opt operator may not be a local optimum with respect to the 3-opt operator [1]. Since the concept of a local optimum can be dependent on specific components of a metaheuristic, concepts related to local optima such as attraction basin, exploration, exploitation, etc, can also display a necessary degree of variance. This variance should force us to be vigilant in using precise

Email address: sychen@yorku.ca.

https://doi.org/10.1016/j.asoc.2025.114066
Received 10 June 2025; Received in revised form 15 September 2025; Accepted 10 October 2025
Available online 11 October 2025

definitions, as opposed to being complacent with the use and acceptance of vague and generic terms.

The intermingling of generic and specific usages of various terms can lead to miscommunication and misunderstandings in the field of metaheuristics. Here is a generic statement that should be generally acceptable to all members of the community. “Metaheuristics are stochastic methods which differ from simple random search through their use of selection to focus their search efforts towards more promising regions of the search space.” It should thus be obvious that selection is a necessary and critical component of any metaheuristic.

Contrasting statements on the existence and importance of selection occur for Particle Swarm Optimization (PSO) [2]. The following statement by Shi and Krohling [3] is accurate for current positions: “Since in the PSO algorithm, all the particles (or individuals) survive into the next generation, there is no selection mechanism involved.” A common confusion is to apply this statement to PSO in its entirety. For example, Wang et al. [4] make the broader statement that “no selection, recombination, and mutation occur in the PSO frame.” It is noted, however, that particle movements are influenced by personal best (*pbest*) positions which are updated by using elitist, fitness-based selection: “Each agent ‘remembered’ the best value and the XY position which had resulted in that value.” [2]

The use and acceptance of vague and generic terms in the field of metaheuristics have helped to obscure the usage of elitist fitness-based selection in Particle Swarm Optimization. By using a series of precise definitions for local optima, attraction basin, exploration, and escape from local optima, we will show how metaheuristics such as PSO that use elitist fitness-based selection can often become trapped in local optima. This situation will be contrasted with the explicit claims made by the developers of a metaheuristic that it has been specifically developed to be able to escape from local optima.

The concept of escape from local optima has been poorly defined, especially for continuous domain search spaces. A simple and practical definition is introduced. Its application to popular metaheuristics such as Particle Swarm Optimization [2], Grey Wolf Optimizer [5], and Differential Evolution [6] shows a general omission in addressing this critical task. The explosion of new metaphors [7] tends to focus on new descriptions for the creation of search solutions which is effectively a reinvention of localized Random Search for the key task of escaping from local optima.

Our understanding of how localized Random Search achieves escapes from local optima depends on precise definitions for key terms such as local optimum, exploration, exploitation, and selection which are provided in Section 2. We will then present a new, formal definition of what it means to escape from local optima in Section 3. A study on the ability of Random Search to escape from local optima is reviewed in Section 4. The effect of elitist fitness-based selection on the ability of Swarm Intelligence (Section 5) and Evolutionary Algorithms (Section 6) to escape from local optima is then presented. The previous claims that these methods can escape from local optima are further discussed in Section 7 before a brief summary is provided.

2. Important definitions

The precise concept of “escaping from a local optimum” depends on precise concepts for exploration, exploitation, selection, and local optimum. The concept of a local optimum is relatively clear, but the exact definition differs in combinatorial and continuous domains, and this can lead to confusion and/or imprecision. The concepts of exploration and exploitation have historically been quite vague, so a set of precise definitions is provided for the continuous domain. The concept of selection can also cause confusion because the same term is applied to both the selection of the mating population and the selection of surviving offspring.

2.1. Local optimum

The global optimum is the best solution in the entire search space. A local optimum is the best solution in a given subset or “neighbourhood” of the search space. A neighbourhood is defined by a search operator as all of the solutions it can reach in a series of steps with monotonically improving fitness. For example, in the Traveling Salesman Problem (TSP), a 2-opt swap picks two distinct cut points in a (permutation) solution and inverts the order of the cities between these two points.

A TSP with n cities will have a total search space of size $\mathbf{O}(n!)$ while a 2-opt swap can create $\mathbf{O}(n^2)$ solutions. A TSP solution is a local optimum with respect to the 2-opt operator if all of the solutions that can be created from this solution by a single 2-opt swap have a worse fitness (i.e., longer length). It is noted that different operators (e.g., insertion, 3-opt, etc) create different neighbourhoods, so a solution that is a 2-opt local optimum may not be a 3-opt local optimum. For example, a single 4-opt move can be used in iterated Lin-Kernighan to escape from its previous local optimum [1].

The above formal definition of a local optimum is often replaced by a functional definition based on Hill Climbing and/or Greedy Search. In Greedy Search, operators are applied to a current solution which is only replaced when the operator leads to a solution of better fitness. The termination point of Greedy Search is a local optimum that cannot be improved upon by any of the operators in use. Hill Climbing is often used as an interchangeable term for Greedy Search, but its use should be restricted to a single operator which leads to a well-defined neighbourhood with well-defined “hill tops”.

This distinction between Greedy Search and Hill Climbing becomes important in continuous domains. Specifically, Hill Climbing implies a single hill whereas Greedy Search could encompass a region of multiple hills. Greedy Search encompassing multiple hills is not intuitive, but the precise mathematical definition for a local optimum in continuous domains is based on ϵ: any point in $\mathbb{R}^n$ that has a better fitness value than all other points within a distance of ϵ is defined to be a local optimum. However, Hill Climbing based on ϵ is almost never implemented in practice, so what is referred to as Hill Climbing is actually Greedy Search which uses a non-infinitesimal step size of σ.

The above functional definition of a local optimum as the solution that is reached by Greedy Search leads to great confusion in the development of Simulated Annealing for continuous domains. Simulated Annealing was first developed for combinatorial domains, and it incorporates the Metropolis procedure into Hill Climbing/Greedy Search to enable escapes from local optima [8]. Since Hill Climbing/Greedy Search cannot escape from local optima, the ability of Simulated Annealing to escape from local optima and thereby achieve better performance than Hill Climbing/Greedy Search must be attributed to the Metropolis procedure.

This assumption has been poorly transferred to an implementation of Simulated Annealing for continuous domains developed by Corana et al. [9]. For example, a typical explanation for the operation of

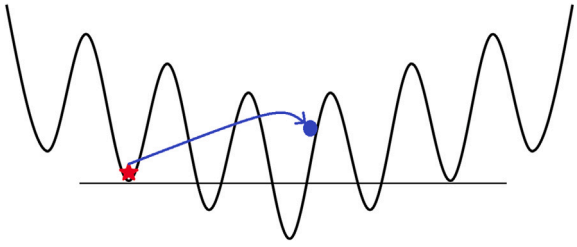


Fig. 1. Simulated annealing uses the metropolis procedure to escape from local optima through the acceptance of a non-improving step. However, practical implementations in the continuous domain show no significant difference from localized Random Search.

Simulated Annealing in continuous domains is shown in Fig. 1. However, the shown step is based on σ – a step size so large that it traverses multiple hill tops would never be accepted as an example of Hill Climbing. Thus, typical implementations of Simulated Annealing for the continuous domain are actually Greedy Search plus the Metropolis procedure.

This distinction between Hill Climbing and Greedy Search is important because Greedy Search based on a step size of σ can reach multiple hill tops/local optima as defined by ϵ. Therefore, the ability of these implementations of Simulated Annealing to escape from local optima may not be based on the addition of the Metropolis procedure. Specifically, Simulated Annealing is conceptualized as Hill Climbing plus the Metropolis procedure, so Simulated Annealing minus the Metropolis procedure should revert back to Hill Climbing. Comparisons of standard implementations of Simulated Annealing for the continuous domain (e.g., [10]) with its derived version of Hill Climbing (which is actually a localized Random Search created by deleting the Metropolis procedure) show insignificant differences in overall performance and negligible differences in the observed movements through the search space [11,12].

The poor testing of Simulated Annealing in its development [9] is an example of the importance of using precise definitions. Since Hill Climbing cannot escape from local optima, it has been widely assumed that Simulated Annealing for continuous domains has been able to escape from local optima because of the addition of the Metropolis procedure. However, the actual implementation of Simulated Annealing adds the Metropolis procedure to Greedy Search, and the different operators of Hill Climbing (i.e., ϵ sized steps) and Greedy Search (i.e., σ sized steps) lead to different local optima. The Greedy Search typically used in Simulated Annealing can already escape from local optima defined by ϵ, so the Metropolis procedure is actually a superfluous addition.

This poor definition of local optima for continuous domain search spaces is pervasive in the research community. Fitness landscapes for (benchmark) functions are often shown in two or three dimensions (e.g., [13]), and the local optima observed from these figures are (implicitly) based on a step size of ϵ. However, there are no known attempts to visualize the local optima for a continuous domain search space based on a step size of σ. The difference between local optima based on step sizes of ϵ and σ is an important distinction for our forthcoming analysis on escapes from local optima.

2.2. Exploration and exploitation

Exploration and exploitation are critical concepts for the design and analysis of metaheuristics in multi-modal search spaces. However, a broad survey of over 100 papers led Črepinšek et al. [14] to the unexpected conclusion that "exploration and exploitation have only been implicitly defined in EAs." Many of these definitions are described in relation to the metaheuristic. A specific example from Particle Swarm Optimization is provided by Bosman and Engelbrecht [15]: "Diversity is related to the notions of exploration and exploitation: the more diverse a swarm is, the more its particles are dispersed over the search space, and the more the swarm is exploring."

There is no need for metaheuristic-dependent definitions of exploration and exploitation in $\mathbb{R}^n$. The following definitions of exploration and exploitation are based on search space properties, so they can apply to any metaheuristic performing optimization in the continuous domain. First, for a minimization problem, any point in $\mathbb{R}^n$ that has a better/lower fitness value than all other points within a distance of ϵ is defined to be a local optimum. Then, the search space is divided into attraction basins where each attraction basin has exactly one local optimum. An attraction basin around an optimum includes all of the points in the search space that can reach (only) that optimum by following a path on which every successive point has a monotonically decreasing fitness (for minimization problems).

These definitions for local optima and attraction basins then lead to operator-independent definitions for exploration and exploitation. A search solution is defined to be exploratory if it is in a "new" attraction basin, and it is exploitative if it is in a "known" attraction basin. For example, in Particle Swarm Optimization with a ring topology [16], a current position would be deemed to be performing exploitation if it is in the same attraction basin as either its personal best (*pbest*) or local best (*lbest*) positions, and it would be performing exploration if it is in any other attraction basin.

The above definitions for exploration and exploitation have fewer logical contradictions than many common definitions of these terms. For example, with diversity in PSO, it is noted that a popular recommendation for initial velocities is zero [17]. The initial population of the swarm, which can be its most diverse, will have for its best particle the same personal best and global best positions (i.e., $pbest = gbest$). Since $pbest - gbest = 0$ and $v_0 = 0$, an exact copy of this solution is created during the first iteration in these implementations of PSO. The previous diversity-based description of exploration is so imprecise that it can include an exact copy of the best-known solution as being considered as exploratory.

A metaheuristic must perform both exploration and exploitation, regardless of whether vague or precise definitions are being used. Without exploration, a method with only exploitation becomes Local Search or Greedy Search. Without exploitation, a method with only exploration becomes a coarse Random Search with no ability to refine its final solution to a local optimum. The need to "balance" exploration and exploitation has thus been a key topic in the field of metaheuristics [18].

The vague usage of "exploitation" to refer to the leveraging of any known information is what leads to this confusion. For example, "The dilemma within an optimization procedure is whether to search around the best-so-far solutions… or explore some totally different regions of the search space" from [19] equates a targeted, but still exploratory, search with exploitation. The precise definitions of exploration and exploitation instead present these two concepts as distinct activities which each play an important role, but which do not necessarily need to be "balanced".

2.3. Selection

Evolutionary computation is based on Darwinian evolution, which is also known as evolution by natural selection. This selective pressure, which favours certain (random) variations over others, leads to increasing fitness of the population over time. It was first modeled in genetic algorithms by using roulette wheel selection of parents [20,21].

Roulette wheel selection means that a solution's proportion in the mating pool is related to the proportion of its fitness to the total fitness of all solutions in the population. In addition to the extreme cases of when a single parent dominates the entire mating pool or when there is negligible selection pressure to differentiate from random selection, the calculation of fitness proportion can represent unnecessary computational overhead. A popular alternative is tournament selection [22]. For example, in 2-tournament, two solutions are picked at random from the population, and the fitter solution becomes a parent.

The original development of genetic algorithms used generational replacement where every offspring solution is added to the next mating pool. Selection pressure must then be applied during the selection of parents from this mating pool. An alternative to generational replacement is steady-state replacement where solutions are added and removed from the population one at a time [23]. This type of system allows selection pressure to be applied to both how population members are selected for mating and how they are selected for removal.

The default for removal has since become elitist fitness-based selection – the fittest solutions always survive and/or the least fit solutions are always removed from the population [24]. Elitist fitness-based selection is now often used in a generational format in which all population members can produce offspring solutions (i.e., no selection applied to parents), but these offspring solutions do not replace their parents in the population if they are less fit (i.e., elitist fitness-based selection for survival). This population control strategy is specifically used by Differential Evolution [6].

Swarm intelligence methods do not apply selective pressure as part of a population control strategy. However, some type of direction must be provided to the swarm members to guide them towards better solutions. Ant Colony Optimization (which was first developed for combinatorial, path-based problems) uses pheromone trails with fitness-based updates to guide the ants (swarm members) [25]. Particle Swarm Optimization (which was first developed for optimization in the continuous domain) adds attraction to best-known positions [2]. A new position that is less fit does not lead to an update of a personal best position, and this is functionally equivalent to the use of elitist fitness-based selection in Differential Evolution to restrict updates to parent/stored solutions.

3. Escape from local optima

The preceding definitions for local optimum, attraction basin, and exploration allow us to develop a formal definition for an escape from a local optimum. First, a locally optimal solution must be found. To be trapped in a local optimum implies being unable to escape from its attraction basin. To escape from a local optimum then means that the previous local optimum is replaced by a search solution (i.e., that survives selection) which is in a different attraction basin (i.e., it is an exploratory search solution with respect to the local optimum).

We note that Random Search (across the full search space) has the ability to escape from a local optimum as has been defined for $\mathbb{R}^n$. Specifically, any search operator that can cover the entire search space has no local optima. For the ϵ-based definition of local optima in $\mathbb{R}^n$, Fig. 2 shows a multi-modal search space in which a local optimum has been found (red star). Random Search can update this best-known solution if it finds a better solution, and these better solutions are highlighted in yellow.

The idealized concept of Hill Climbing is that it examines solutions from only a single attraction basin or hill. However, typical implementations of Hill Climbing in the continuous domain are based on a maximum step size that defines a potentially larger search neighbourhood (e.g., s_{max} [26]). When this maximum step size is larger than the value of ϵ used to define local optima and attraction basins in $\mathbb{R}^n$, then these s_{max}-based implementations of Hill Climbing are actually a form of localized Random Search. Most practical implementations of Hill Climbing in the continuous domain are actually implementations of localized Random Search which have the ability to escape from ϵ-sized local optima. Specifically, in reference to Fig. 2, it is possible that an s_{max}-based implementations of Hill Climbing can reach the yellow regions of the search space and thus escape from the ϵ-based local optimum shown by the red star.

We now make the key claim of this paper. Many metaheuristics have no mechanism that is meaningfully different from the above mechanism used by localized Random Search to escape from a local optimum in $\mathbb{R}^n$. Specifically, these metaheuristics depend on a stochastically generated (i.e., random) exploratory search solution to be fitter than the current local optimum to allow an escape. If the exploratory search solution is less fit than the local optimum, and the metaheuristic uses elitist fitness-based selection, then the search solution will be discarded.

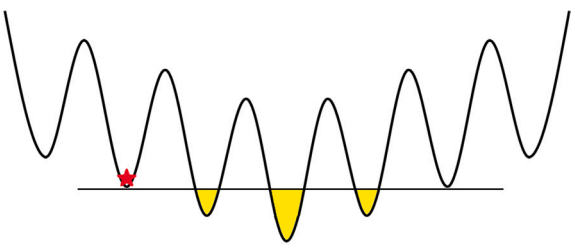


Fig. 2. A local optimum has been reached – red star. Random Search can escape from this local optimum if a fitter solution is found – shaded areas in yellow.

Most metaheuristics are of course designed with search mechanisms (e.g., particle trajectories in PSO [27]) that are more interesting than simple Random Search. However, these mechanisms often require large parameter values to create exploratory search solutions in new attraction basins, and the opportunity to refine these exploratory search solutions with exploitation is not available if these solutions are immediately eliminated by elitist fitness-based selection. Thus, a successful escape from a local optimum requires that the (stochastically generated) search solution is both located in a new attraction basin with a fitter local optimum and that it is a near local optimum in this attraction basin (see Fig. 2). The probability of this occurrence is similar to that of (localized) Random Search, so the summary of our claim is that metaheuristics in the continuous domain that use elitist fitness-based selection can become no more effective than (localized) Random Search in their attempts to escape from local optima.

4. Random search

The ability of Random Search to escape from local optima in the continuous domain has been studied by Chen et al. [28]. The Rastrigin function is based on a sinusoid super-imposed over a sphere base function. To simplify the effects of the sphere base function, the study on Random Search used sinusoids and a linear step function. To escape from a local optimum, Random Search has to find a solution in the adjacent attraction basin that is closer to the local optimum than the step difference in the fitness between two adjacent local optima.

The ability of (localized) Random Search to escape from a local optimum is based entirely on the proportion of the attraction basins in the search space that are sufficiently close (in fitness) to the local optimum. For example, if the shape of the attraction basin is linear, then 10 % of the attraction basin's area has a fitness within 10 % of an attraction basin's local optimum. An attraction basin with a sinusoidal shape (which is flat at the top and bottom) will have more than 10 % of its area with this level of fitness. However, this proportion decreases exponentially with increasing dimensionality.

Let us assume that a sufficient fitness to escape from a local optimum requires a search point to be within 10 % of the width of the attraction basin in each dimension d. This entirely acceptable 1 in 10 chance in one dimension degrades exponentially to be 1 in 10^d in d dimensions. The study on Random Search showed successful escapes from local optima stopping after $d = 10$ dimensions [28]. A subsequent study [29] showed that the exploratory search solutions created by PSO and DE have similar fitness profiles to Random Search solutions. This study also showed that successful escapes from local optima for PSO and DE also stopped around $d = 10$ dimensions on the Rastrigin function [29] meaning that PSO and DE did not show any meaningful additional capability at escaping from local optima compared to Random Search.

It is a false assumption that Random Search and/or Greedy Search cannot escape from a local optimum in $\mathbb{R}^n$. This false assumption propagates into the belief that a metaheuristic which is more capable than Random Search must be able to escape from local optima better than Random Search. Adding the equivalence that Random Search is Hill Climbing and that Hill Climbing cannot escape from local optima, it is generally assumed that any escape from a local optimum is due to the specific mechanisms employed by a given metaheuristic. There is no consideration that escapes from local optima could instead be achieved strictly through sheer luck and (localized) Random Search.

5. Swarm intelligence

The basic format of a swarm intelligence-based metaheuristic is that a set of entities (e.g., birds, ants, bees, wolves, etc) move around the search space under various influences. In many natural swarm situations, e.g., people walking in a crowd, there is no global objective, and each entity can move based on local information only, e.g., move towards the exit while avoiding collisions. The use of swarms for global optimization involves the gathering and communication of global information

(e.g., the best-known position). The storage of a best-known position in a swarm intelligence-based metaheuristic is equivalent to stating that elitist fitness-based selection is used to update the best-known position through comparisons with the current position of every entity in the swarm.

Two swarm intelligence-based metaheuristics for the continuous domain are studied in depth. Original Particle Swarm Optimization (PSO), which was developed for a unimodal search space (the cornfield vector), does not address escaping from local optima [2]. Grey Wolf Optimizer (GWO) [5], which is described as a derivative of basic swarm mechanisms by Camacho Villalón et al. [30], makes claims of being able to avoid and/or escape from local optima, but a precise definition of what this means and how it occurs is not provided.

5.1. Particle swarm optimization

An analysis of the effects of selection in PSO benefits from thinking of PSO as a metaheuristic with two populations: a population of current positions and a population of personal best (*pbest*) positions. In this model, a current position represents a search solution which is compared to a paired personal best position, and this *pbest* position acts as a reference solution that guides and influences the search performed by current positions. Selection occurs here in that potential updates of the *pbest* positions can be accepted or rejected, and this selection mechanism is specifically elitist fitness-based selection.

The attention of the broader PSO research community as summarized by Bonyadi and Michalewicz [31] is often focused on the motions and behaviours of the current positions, e.g., their elliptical trajectories [27]. All of the current positions of a swarm are usually only viewed as being trapped within a local optimum at convergence – all of the current positions and personal best positions are in the same attraction basin or even at the same optimum. A swarm that has current positions which can visit multiple attraction basins is not normally considered to be trapped in local optima.

The novel Randomized Particle Swarm Optimizer (RPSO) developed by Liu et al. [32] represents a typical example of this line of research. The statement that "the RPSO algorithm not only maintains the population diversity but also enhances the possibility of escaping the local optima trap" [32] is consistent with the general focus on current positions and the assumption that their continued motion is sufficient to prevent the overall system from becoming entrapped by local optima. Specifically, RPSO maintains the standard update procedures for *pbest* and *gbest* positions with the belief that "the dynamical behavior of the RPSO algorithm becomes more complicated than the basic PSO algorithm and the particles are therefore allowed to expand their search space, which leads to a more thorough exploration of the problem space with less trapping possibility in the local optima" [32].

The convergence of the current positions is not required for the personal best positions of PSO to be trapped in local optima. This situation is described by the stall condition recently introduced by Yadollahpour and Chen [33]. In stall, "clusters of exploiting particles are separated by an exploring particle that has its *pbest* position in one cluster/attraction basin and its *lbest* position in the other" [33]. Further, "there is no movement in *pbest* positions that could cause previously (near) stationary particles to begin moving" [33]. The stall condition matches the current definition of not being able to escape from a local optima in that the *pbest* positions have reached (near) local optima, and the swarm is "unable to find better solutions because all of its exploratory search solutions are rejected" [33].

Fig. 3 shows how a *pbest* position in (a variant of) PSO can become trapped in a local optimum, independent of the communication topology [34]. The personal best position (red star) is a local optimum. The creation of current positions (blue circles) is based on adding a movement term to the previous position. Typical implementations of PSO use random weights to generate the movement term – e.g., ϵ in Standard PSO [16].

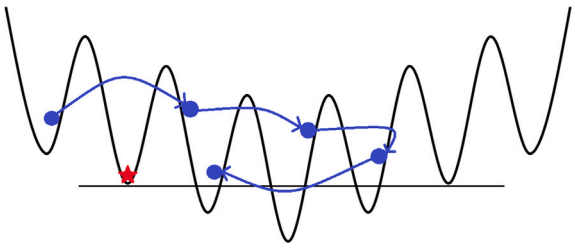


Fig. 3. The current positions (blue circles) are unable to update the personal best position (red star) that is a local optimum.

The movements between current positions in Fig. 3 do not perform a local search since the *pbest* position which influences these movements is in a different attraction basin. The resulting attraction vector is likely several times larger than the size of an attraction basin. The attraction basins visited by the current positions thus have a solution generated in them that is effectively random (e.g., due to the random weights in ϵ).

Original PSO [2] was designed for a unimodal search space (the cornfield vector), so it did not consider the need to escape from local optima. All variations of PSO that use the same update rules for *pbest* positions often do not include a new mechanism for these *pbest* positions to escape from local optima. It should thus be no surprise that the ability of (these variations of) PSO to escape from local optima is similar to that of (localized) Random Search.

5.2. Grey Wolf Optimizer

Grey Wolf Optimizer (GWO) is a popular swarm intelligence-based metaheuristic developed by Mirjalili et al. [5]. Following the advice of Sörensen [7] to ignore the underlying metaphor, the most interesting algorithmic component of GWO in comparison to PSO is that it doesn't store personal best positions. The three best solutions in the population (i.e., the alpha, beta, and delta wolves) are calculated after each iteration. These three best solutions guide the search trajectories of all the wolves through their use in the update equations.

During their development of GWO, Mirjalili et al. [5] describe escapes from local optima in a manner that is not consistent with the specific definition provided in Section 3. They instead state that "metaheuristics have superior abilities to avoid local optima compared to conventional optimization techniques. This is due to the stochastic nature of meta-heuristics which allows them to avoid stagnation in local solutions and search the entire search space extensively." Specifically, it is sufficient for their definition of avoiding/escaping from local optima that exploratory search solutions (e.g., omega wolves) can visit new attraction basins.

It is true that GWO includes a specific mechanism to encourage exploration. In GWO, "half of the iterations are devoted to exploration ($|A| \geq 1$) and the rest to exploitation ($|A| < 1$). This mechanism assists GWO to provide very good exploration, local minima avoidance, and exploitation simultaneously." [5] This exploration can indeed allow escapes from local optima. However, Mirjalili et al. [5] state themselves that "It can be observed that the final position would be in a random place within a circle which is defined by the positions of alpha, beta, and delta in the search space." The superposition of this circle over an attraction basin of unknown location means that its chances of successfully finding a solution that allows for an escape from a local optimum (similar to that shown in Fig. 2) are not meaningfully different from that for (localized) Random Search.

6. Evolutionary algorithms

Hill Climbing can be implemented as a (1 + lambda)-ES [35]. This is not to claim that all evolutionary algorithms are just elaborate forms of hill climbing, but it is a useful frame of reference for analyzing how

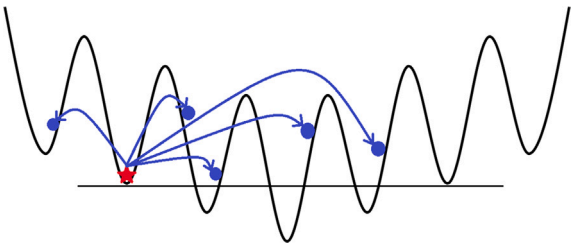


Fig. 4. The shown offspring (blue circles) that are generated from a parent that is a local optimum (red star) would all be rejected by elitist fitness-based selection.

these methods attempt to escape from local optima. Fig. 4 shows a series of search/offspring solutions generated by localized Random Search as part of its attempt to escape from a local optimum. If all these solutions were created during a single generation for a population-based evolutionary algorithm, then this figure (which is essentially the same as Fig. 3 for swarm intelligence-based metaheuristics) shows how EAs can become functionally similar to localized Random Search in their attempts to escape from local optima.

Differential Evolution (DE) developed by Storn and Price [6] is one of the most popular evolutionary algorithms for multi-modal optimization problems in the continuous domain [36]. Its use of elitist fitness-based selection is quite explicit – "If the trial vector yields a lower cost function value than the target vector, the trial vector replaces the target vector in the following generation. This last operation is called selection" [6]. If the "target vector" is a local optimum (red star), then Fig. 4 shows how "trial vectors" (the search solutions shown as blue circles) can attempt to escape from this local optimum.

The search motions implied by the blue arrows in Fig. 4 are more representative of Evolution Strategies (ES) [35] in which offspring solutions are created by adding a mutation vector to a parent solution. The trial vector in DE is created from three distinct random solutions. The DE method for producing offspring solutions can make it easier to explore attraction basins far from the current local optimum. However, for a new attraction basin with an unknown location, there can be no expectation that this offspring solution will be more likely than one generated by localized Random Search to be an improvement on an existing local optimum.

Many variations of DE [36] and many variations of evolutionary algorithms [7,37] often retain elitist fitness-based selection (and other population control strategies) unchanged and only modify the equations by which offspring solutions are generated. Without specific information on where the fittest solutions will reside in a new unexplored attraction basin, the new offspring solutions will essentially be random samples of these attraction basins. The fundamental problem of many metaheuristics in the continuous domain is that their ability to escape from local optima is not meaningfully different from localized Random Search.

7. Discussion

A random solution from an attraction basin with a fitter local optimum can be better than a current locally optimal reference solution. Each random search solution has a non-zero chance of facilitating an escape from a local optimum. Metaheuristics that increase their exploration of new attraction basins can thus increase their chances of escaping from local optima. However, using methods similar to (localized) Random Search to escape from local optima is extremely limiting.

The ability of (localized) Random Search to escape from a local optimum is related to the proportion of an attraction basin that has the required fitness. A detailed study by Chen et al. [28] on attraction basins with sinusoidal shapes (e.g., the Rastrigin function) shows that this proportion can decrease exponentially with increasing dimensionality. Metaheuristics with mechanisms similar to (localized) Random Search can indeed show an (improved) ability to escape from local optima in lower dimensional search spaces. However, their decreasing performance with increasing dimensionality (i.e., the Curse of Dimensionality [38]) may be due more to their decreasing ability to escape from local optima than to the effects of an exponentially increasing search space volume [39].

Metaheuristics need more than just an ability to perform exploration to discover new attraction basins. They also need the ability to perform local search (i.e., exploitation) in these attraction basins. This local search, which has a better chance than (localized) Random Search to find the fittest solutions in a new attraction basin, can then meaningfully improve the ability of metaheuristics to escape from local optima.

The following is a specific example of how to add local search of new attraction basins to Grey Wolf Optimizer. The omega wolves use $|A| \geq 1$ to expel themselves away from the attraction basin(s) of the alpha, beta, and delta wolves. This search motion does not promote local search in any new attraction basins found by the omega wolves. The periodic exploitation phase using $|A| < 1$ is more useful for local search in the attraction basin(s) of the alpha, beta, and delta wolves than that of the omega wolves. A (new) periodic exploitation phase should instead use (only) personal best information of the omega wolves to improve local search within their new attraction basins.

It is noted that the proposed mechanism for GWO to escape from local optima in the continuous domain is more complex than many methods that are used in combinatorial optimization problems. Simple kick operators like those used by Lin and Kernighan [40] to escape from local optima on the Traveling Salesman Problem are often sufficient in combinatorial domains. It is thus important to be aware of the differences in search operators and attraction basins for combinatorial and continuous domain search spaces.

A key insight into these differences can be obtained by comparing the search trajectories of Tabu Search [41] and Grey Wolf Optimizer. Tabu Search can go through phases of intensification and diversification. The alternating cycles of exploration and exploitation with $|A| \geq 1$ and $|A| < 1$ in GWO appear to support a similar concept. However, intensification in Tabu Search is guided by the current search state and not by the global best positions (e.g., the alpha, beta, and delta wolves). Tabu Search can perform local search in any new attraction basin that is discovered while this is not ensured in GWO.

Local search is a critical component in allowing metaheuristics in the continuous domain to escape from local optima. The limit of more local search is to reach the local optimum in every attraction basin, which becomes similar to a memetic algorithm [42]. Alternatively, a metaheuristic could perform a search for points from which to start a local optimization method (e.g., the EZopt optimization framework proposed by Hendtlass [43]). Both of these research directions move towards specific and separate methods for exploration and exploitation as opposed to attempting to find some form of balance between them.

8. Summary

The concept of escaping from local optima needs a formal definition. The proposed definition requires that the population member which is a local optimum is updated to become a solution in a new attraction basin. Typical definitions for escaping from local optima appear to be satisfied with the creation of search solutions in new attraction basins, even if all of these search solutions are rejected by elitist fitness-based selection.

The generation of search solutions in new attraction basins can lead to escapes from local optima. The stochastic nature of how these search solutions are generated by many metaheuristics can make this process functionally similar to (localized) Random Search. A key goal of metaheuristic design is to develop a process that is meaningfully better than Random Search. Metaheuristic design needs to consider the effects of elitist fitness-based selection and potentially add a local search procedure to achieve better than random chances of escaping from local optima.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been supported by Natural Sciences and Engineering Research Council of Canada (NSERC) through a Discovery Grant – RGPIN-2022-04524.

Data availability

No data was used for the research described in the article.

References

[1] G. Reinelt, The Traveling Salesman: Computational Solutions for TSP Applications, vol. 840, Springer, 2003.
[2] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, 1995, vol. 4, IEEE, 1995, pp. 1942–1948.
[3] Y. Shi, R.A. Krohling, Co-evolutionary particle swarm optimization to solve min-max problems, in: Proceedings of the 2002 Congress on Evolutionary Computation, CEC'02 (Cat. No. 02th8600), vol. 2, IEEE, 2002, pp. 1682–1687.
[4] L. Wang, B. Yang, J. Orchard, Particle swarm optimization using dynamic tournament topology, Appl. Soft Comput. 48 (2016) 584–596.
[5] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey Wolf Optimizer, Adv. Eng. Softw. 69 (2014) 46–61.
[6] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (4) (1997) 341–359.
[7] K. Sörensen, Metaheuristics—the metaphor exposed, Int. Trans. Oper. Res. 22 (1) (2015) 3–18.
[8] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680.
[9] A. Corana, M. Marchesi, C. Martini, S. Ridella, Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm, ACM Trans. Math. Softw. 13 (3) (1987) 262–280.
[10] SciPy Developers, Scipy reference guide: dual_annealing (2018) https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.dual_annealing.html, (Accessed:2023).
[11] S. Chen, S. Islam, S. Lao, The danger of metaphors for metaheuristic design, in: 2023 IEEE Latin American Conference on Computational Intelligence (LA-CCI), IEEE, 2023, pp. 1–6.
[12] D. Gulta, S. Chen, Improvements to dual annealing in SciPy, in: 2024 IEEE Latin American Conference on Computational Intelligence (LA-CCI), IEEE, 2024, pp. 1–6.
[13] J.J. Liang, B.Y. Qu, P.N. Suganthan, A.G. Hernández-Díaz, Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China And Nanyang Technological University, Singapore, Technical Report 201212 (2013) 3–18.
[14] M. Črepinšek, S.-H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: a survey, ACM Comput. Surv. 45 (3) (2013) 35.
[15] P. Bosman, A.P. Engelbrecht, Diversity rate of change measurement for particle swarm optimisers, in: International Conference on Swarm Intelligence, Springer, 2014, pp. 86–97.
[16] D. Bratton, J. Kennedy, Defining a standard for particle swarm optimization, in: Swarm Intelligence Symposium, 2007, IEEE, 2007, pp. 120–127.
[17] A. Engelbrecht, Particle swarm optimization: velocity initialization, in: 2012 IEEE Congress On Evolutionary Computation, IEEE, 2012, pp. 1–8.
[18] B. Morales-Castañeda, D. Zaldivar, E. Cuevas, F. Fausto, A. Rodríguez, A better balance in metaheuristic algorithms: does it exist?, Swarm Evol. Comput. 54 (2020) 100671.
[19] A.E. Eiben, M. Schoenauer, Evolutionary computing, Inf. Process. Lett. 82 (1) (2002) 1–6.
[20] J.H. Holland, Adaptation in natural and artificial systems. An introductory analysis with applications to biology, control and artificial intelligence, University Of Michigan Press, 1975, Ann Arbor, 1975.
[21] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley, Reading: MA, 1989.
[22] J. Smith, F. Vavak, Replacement strategies in steady state genetic algorithms: static environments, Found. Genet. Algorithms 5 (1999) 219–233.
[23] L.D. Whitley, et al., The GENITOR algorithm and selection pressure: why rank-based allocation of reproductive trials is best, in: ICGA, vol. 89, 1989, pp. 116–123.
[24] L.J. Eshelman, The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination, in: Foundations of Genetic Algorithms, vol. 1, Elsevier, 1991, pp. 265–283.
[25] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. Part B Cybern. 26 (1) (1996) 29–41.
[26] L. Nolle, On a hill-climbing algorithm with adaptive step size: towards a control parameter-less black-box optimisation algorithm, in: Computational Intelligence, Theory and Applications: International Conference 9th Fuzzy Days in Dortmund, Germany, Sept. 18–20, 2006 Proceedings, Springer, 2006, pp. 587–595.
[27] F. Van den Bergh, A.P. Engelbrecht, A study of particle swarm optimization particle trajectories, Inf. Sci. 176 (8) (2006) 937–971.
[28] S. Chen, S. Islam, A. Bolufé-Röhler, J. Montgomery, T. Hendtlass, A random walk analysis of search in metaheuristics, in: 2021 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2021, pp. 2323–2330.
[29] S. Chen, A. Bolufé-Röhler, J. Montgomery, D. Tamayo-Vera, T. Hendtlass, Measuring the effects of increasing dimensionality on fitness-based selection and failed exploration, in: 2022 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2022, pp. 1–8.
[30] C.L. Camacho Villalón, T. Stützle, M. Dorigo, Grey wolf, firefly and bat algorithms: three widespread algorithms that do not contain any novelty, in: International Conference on Swarm Intelligence, Springer, 2020, pp. 121–133.
[31] M.R. Bonyadi, Z. Michalewicz, Particle swarm optimization for single objective continuous space problems: a review, Evol. Comput. 25 (1) (2017) 1–54.
[32] W. Liu, Z. Wang, N. Zeng, Y. Yuan, F.E. Alsaadi, X. Liu, A novel randomised particle swarm optimizer, Int. J. Mach. Learn. Cybern. 12 (2) (2021) 529–540.
[33] N. Yadollahpour, S. Chen, Methods to detect and address stall in particle swarm optimization, in: 2022 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2022, pp. 1–8.
[34] L. Deng, S. Liu, Collective dynamics of particle swarm optimization: a network science perspective, Physica A: Stat. Mech. Appl. 675 (2025) 130778.
[35] H.-G. Beyer, H.-P. Schwefel, Evolution strategies–a comprehensive introduction, Nat. Comput. 1 (1) (2002) 3–52.
[36] S. Das, S.S. Mullick, P.N. Suganthan, Recent advances in differential evolution–an updated survey, Swarm Evol. Comput. 27 (2016) 1–30.
[37] L. Deng, S. Liu, Metaheuristics exposed: unmasking the design pitfalls of arithmetic optimization algorithm in benchmarking, Appl. Soft Comput. 160 (2024) 111696.
[38] R.E. Bellman, Dynamic Programming, Princeton Univ. Press, 1957.
[39] E. Keogh, A. Mueen, Curse of dimensionality, in: Encyclopedia of Machine Learning, Springer, 2011, pp. 257–258.
[40] S. Lin, B.W. Kernighan, An effective heuristic algorithm for the traveling-salesman problem, Oper. Res. 21 (2) (1973) 498–516.
[41] F. Glover, Tabu search—part I, ORSA J. Comput. 1 (3) (1989) 190–206.
[42] P. Moscato, On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms, Caltech Concurrent Computation Program, C3P Report 826 (1989) 1989.
[43] T. Hendtlass, The EZopt optimisation framework, in: 2019 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2019, pp. 3110–3117.