J Glob Optim (2017) 67:135–149
DOI 10.1007/s10898-015-0398-9

Interactive model-based search with reactive resource allocation

Yue Sun[1] · **Alfredo Garcia**[1]

Received: 12 April 2015 / Accepted: 28 December 2015 / Published online: 7 January 2016

Abstract We revisit the interactive model-based approach to global optimization proposed in Wang and Garcia (J Glob Optim 61(3):479–495, 2015) in which parallel threads independently execute a model-based search method and periodically interact through a simple acceptance-rejection rule aimed at preventing duplication of search efforts. In that paper it was assumed that each thread successfully identifies a locally optimal solution every time the acceptance-rejection rule is implemented. Under this stylized model of computational time, the rate of convergence to a globally optimal solution was shown to increase exponentially in the number of threads. In practice however, the computational time required to identify a locally optimal solution varies greatly. Therefore, when the acceptance-rejection rule is implemented, several threads may fail to identify a locally optimal solution. This situation calls for reallocation of computational resources in order to speed up the identification of local optima when one or more threads repeatedly fail to do so. In this paper we consider an implementation of the interactive model-based approach that accounts for real time, that is, it takes into account the possibility that several threads may fail to identify a locally optimal solution whenever the acceptance-rejection rule is implemented. We propose a modified acceptance-rejection rule that alternates between enforcing diverse search (in order to prevent duplication) and reallocation of computational effort (in order to speed up the identification of local optima). We show that the rate of convergence in real-time increases with the number of threads. This result formalizes the idea that in parallel computing, exploitation and exploration can be complementary provided relatively simple rules for interaction are implemented. We report the results from extensive numerical experiments which are illustrate the theoretical analysis of performance.

Keywords Model-based search · Parallel algorithms · Reactive resource allocation

✉ Alfredo Garcia
alfredo.garcia@ufl.edu

Yue Sun
ys6dn@virginia.edu

[1] Department of Industrial and Systems Engineering, University of Florida Gainesville , FL, USA

1 Introduction and literature review

We consider a parallel computing scheme for global optimization that combines multi-start local search with the dynamic reallocation of computational resources (e.g. processing time). Our work builds upon the interactive model-based approach to global optimization proposed in Wang and Garcia [1] in which parallel threads independently execute a model-based search method (see [2]) and periodically interact through a simple acceptance-rejection rule aimed at preventing duplication of search efforts. In a model-based search method, the distribution of re-start points is adjusted at each iteration upon evaluating local search results which informs the selection of a new "model" (i.e. probability distribution) over the feasible region. This model is in turn used to randomly generate new re-start points. The degree to which the new probability distribution (or "model") is concentrated around the best solutions identified so far reflects the relative emphasis on exploitation versus exploration. Diversity in multiple re-start points (i.e. exploration) is a desirable trait as it provides a form of insurance against operating with a poor model. However, too much diversity may slow down the identification of globally optimal solutions which could be accelerated by selecting models the lead to increased search effort in promising areas as determined by history (i.e. exploitation). This description encapsulates a wide variety of stochastic methods in the literature based upon a multi-start strategy featuring different resolutions to the exploration vs. exploitation tradeoff [3,4,12,13]. Invariably, in single-thread approaches to global optimization based on stochastic multi-start search, exploitation and exploration are substitutes. Several multi-start approaches aim to infer the underlying structure of the problem by statistical learning and then use this information in real time in order to strike an acceptable trade-off between exploitation and exploration (see e.g. [12]). In this paper we take a different approach. Rather than attempting to learn the structure of the problem, the main point of [1] is that in a parallel computing environment when duplication of search effort is prevented (or limited), exploitation and exploration are complements and not substitutes. This is shown to be the case as when the models governing each thread's multi-start local search strategy are subject to an acceptance-rejection rule. Assuming each thread successfully identifies a locally optimal solution every time the acceptance-rejection rule is implemented, it was shown in [1] that the rate of convergence to a globally optimal solution was shown to increase exponentially in the number of threads.

In practice however, the computational time required to identify a locally optimal solution varies greatly. Therefore, when the acceptance-rejection rule is implemented, several threads may fail to identify a locally optimal solution. Thus the main result in [1] relies on a highly stylized model of computational time. In this paper we consider an implementation of the interactive model-based approach that accounts for real time, that is, it takes into account the possibility that several threads may fail to identify a locally optimal solution whenever the acceptance-rejection rule is implemented. We propose a modified acceptance-rejection rule that alternates between enforcing diverse search—in order to prevent duplication—and reallocation of computational effort—in order to speed up the identification of local optima—when one or more threads repeatedly fail to do so. We show that the rate of convergence in real-time increases with the number of threads. This result formalizes the idea that in parallel computing, exploitation and exploration are complements and not substitutes (as in most single-thread approaches to global optimization).

The promise of parallel computing for global optimization is more then ever a reality as computer manufacturers have continued to introduce more cores per chip and graphics processing units (GPUs) are increasingly popular. This trend implies significant multi-thread processing power is readily accessible to optimization practitioners which no longer need sophisticated or overly expensive infrastructure to run parallel algorithms for solving global optimization problems. This has motivated recent studies aiming to develop paralleled implementation of well-known global optimization algorithms (see for example, [5,6] for simulated annealing and [7] for particle swarm algorithm).

Parallel computing approaches to global optimization vary depending upon the level of coordination amongst threads (see [8] for a survey). Without any coordination amongst threads, a judicious choice of stopping rules is needed to fully accrue the benefits of parallelization (see [9]). Often some degree of coordination is desirable as real-time information by different threads can be used to improve performance at the expense of overhead. This is for example the case of a parallel implementation of simulated annealing (see [5,6]). Some degree of coordination also enables the real time re-allocation of computational resources among several instances of search algorithms (see for example, [10–12]) in order to improve performance. The optimal real-time allocation of computational resources can be modeled as a non-stationary multiple-armed bandit problem which—in and of itself—may be as complex as the underlying global optimization problem. For example, in [10] certain regularity assumptions are needed to obtain asymptotic bounds on algorithm's performance, and in [12] statistical inference on the underlying problem structure is used to avoid duplication of search efforts.

In this paper, we develop a real-time reallocation strategy that is based upon historical performance. There is no attempt at using sophisticated algorithmic variations in order optimally react to search outcomes. Instead, the main idea is to leverage relatively simple ideas such as (i) continuing searches that are promising because the end-points have lower objective values than all other solutions found so far and (ii) avoiding duplication of failed searches and/or search effort across threads. The relatively small gains afforded by these simple ideas are then shown to be magnified by parallelization. Indeed, we show that the rate of convergence for an interactive model-based search increases with the number of threads for a wide-class of local search techniques (i.e. model-based) when compared to independent parallel implementation. The structure of this paper is as follows. In Sect. 2 we review the single-thread model based search and provide a characterization of improved performance when the algorithm reacts to incomplete search outcomes. In Sect. 3, we analyze an interactive multi-thread approach that in addition to reacting to incomplete searches (at each thread) incorporates a way to avoid duplication of failed searches and/or search effort across threads. In our main result we show that the interactive scheme speeds up the search for global optimal solutions, i.e. the time needed to identify a global solutions decreasing with the number of threads. In Sect. 4 we illustrate this effect by means of a computational testbed.

2 Single thread model-based search in real-time

For bounded set $\Omega \subset \mathbb{R}^n$, consider a general optimization problem $\min\{f(x) : x \in \Omega\}$ where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is continuous and $X^* = \arg\min_{x\in\Omega} f(x)$ is well-defined. Assume further f has finite N local (non-global) minimas, say $X = \{x_1, x_2, \ldots, x_N\}$, that is,

$$f(x_i) \leq f(x) \quad \forall x \in N(x_i, \epsilon_i)$$

for some $\epsilon_i > 0$ where $N(x_i, \epsilon_i) = \{x \in \mathbb{R}^n \mid \|x - x_i\| \leq \epsilon_i\}$.

The multi-start method that we shall describe later makes use of a local search algorithm. A local search algorithm takes an initial solution or "seed" as input, say $x \in \Omega$ and produces an output in the form of a local minimum, say $y \in X \cup X^*$. The local search algorithm determines a map $\bar{\ell} : \Omega \rightarrow X \cup X^*$. The "basin of attraction" of local minimum $y \in X \cup X^*$ $B(y)$ is defined as:

$$B(y) := \{x \in \Omega \mid \bar{\ell}(x) = y\}$$

The properties of the local search algorithm (e.g. the size of basins of attraction) are unknown. However, we assume the basins of attractions partition the solution space Ω:

Assumption 1 $B(x_i) \cap B(x_j) = \varnothing$, for all $x_i \neq x_j \in X \cup X^*$ and

$$\bigcup_{x_i \in X \cup X^*} B(x_i) = \Omega.$$

This assumption states that given any initial input on the solution space, the local search algorithm will produce one and only one local minimum as the output. Additionally, the local search algorithm is deterministic, i.e. the same output is always obtained when provided the same input.

In what follows we revisit the basic iteration scheme in [1] so that each iteration is equivalent to $T_0 > 0$ units of computational time. Let $T(x)$ be the time required by local search $\bar{\ell}$ from x to identify x_i if $x \in B(x_i)$. Define operation ℓ for real time as: if $T(x) \leq T_0$, $\ell(x)$ returns local optima x_i when $x \in B(x_i)$; if $T(x) > T_0$, $\ell(x)$ returns ending location of local search from x at time T_0.

2.1 Basic single-thread computation

Let $\mathfrak{G}$ denote a class of probability density functions (with respect to Lebesgue measure) on the sample space Ω. Define $J \subseteq X \bigcup X^*$ as the state set of detected local optimas. We shall refer to each $g(\cdot, J) \in \mathfrak{G}$ as a "model". The goal is to find a "model" that speeds up the identification of globally optimal solution when the starting solutions to be used by the local search are drawn from such density. Assume that any $g \in \mathfrak{G}$, $g(x, J) \neq 0$ for $x \in \Omega$ almost everywhere. Taking into account the computational time limit T_0, the basic iteration for each thread is as follows:

1. A sample x from the current "model" $g(\cdot, J)$ is drawn and a local search algorithm is launched.
2. At time T_0, the resulting state of information is
$$J' = \begin{cases} J \cup \ell(x) & \text{if } T(x) \leq T_0 \\ J & \text{otherwise} \end{cases}$$
3. A new model $g(\cdot, J') \in \mathfrak{G}$ is selected as follows:
$$g(x, J') = \arg\min_{g \in \mathfrak{G}} D_{KL}(h(x; J'), g) \tag{1}$$
where
$$h(x; J') = \frac{I(f(x), J')U(x)}{\int_{\Omega} I(f(x), J')U(x)dx}$$
where D_{KL} is the Kullback–Leibler divergence, U is the uniform probability density function on Ω and the reference function I is defined as:
$$I(f(x), J') = \begin{cases} 1 & f(x) \leq \min_{x \in J'} f(x) + \epsilon \\ 0 & f(x) > \min_{x \in J'} f(x) + \epsilon \end{cases}$$

for $\epsilon > 0$.

The lower the value of ϵ the more probability mass the reference density function posits around the best locally optimal solutions in the state of information J' and thus the selection of a new model emphasizes exploitation over exploration. To account for the possibility that the local search procedure may fail to identify a locally optimal solution within the allotted time T_0 we define the set $\mathfrak{C}(T_0) = \{x : T(x) \leq T_0\}$. We obtain a Markov chain model for $\{J^{nT_0} : n > 0\}$ with transition probabilities:

$$\Pr(J' \mid J) = \begin{cases} \int_{B(x)\cap\mathfrak{C}(T_0)} g(y, J)dy & J' = J \cup \{x\} \text{ and } x \in \{X \cup X^*\}\backslash J \\ \sum_{x\in J} \int_{B(x)\cap\mathfrak{C}(T_0)} g(y, J)dy \int_{\Omega\backslash\mathfrak{C}(T_0)} g(y, J)dy & J' = J \end{cases}$$

Define $\mathcal{J}^*$ as the class of states with at least a globally optimal solution, i.e. $J \in \mathcal{J}^*$ if and only if $J \cap X^* \neq \emptyset$. It follows that $J \notin \mathcal{J}^*$,

$$\Pr(\mathcal{J}^* \mid J) = \int_{B(X^*)\cap\mathfrak{C}(T_0)} g(y, J)dy$$

For transition probability matrix, arrange state J by (1) lowest local minima value of state; (2) for states with same lowest local minima value, by number of local minima in states. The transition probability matrix for the Markov chain $\{J^{nT_0} : n > 0\}$ is upper triangular. The eigenvalues of this matrix are $\lambda_J = \Pr(J|J)$ and $\Pr(\mathcal{J}^*|\mathcal{J}^*) = 1$. Let $\pi_J^{nT_0} = \Pr(J^{nT_0} = J|J^0)$ be the distribution at time nT_0. From convergence speed of finite-state Markov chains, we have

$$\left|\pi_{\mathcal{J}^*}^{nT_0} - 1\right| \leq C\lambda_{[2]}^n$$

where $\lambda_{[2]} \in (0, 1)$ is the second-largest eigenvalue.

2.2 Reacting to incomplete searches

We consider a variation in which an incomplete search leading to a "high quality" end-point is allowed to continue. Specifically, if the objective function value associated with the end-point of an incomplete search is lower than the values associated with *all* discovered local optima, this end-point must be in the attraction basin of an undiscovered local optima. Hence, the search should be allowed to continue. The iteration of modified single-thread model based search is modified as follows:

1. A sample x from the current "model" g is drawn and a local search algorithm is launched.
2. At time T_0, the resulting state of information is

$$J' = \begin{cases} J \cup \{\ell(x)\} & T(x) \leq T_0 \\ J & T(x) > T_0 \end{cases}$$

   Let y denote the end-point if search is incomplete.
3. A new model $g' \in \mathfrak{G}$ is selected as follows:

$$g' = \begin{cases} g(\cdot, J') & f(y) \geq \min_{x\in J'} f(x) \\ \mathbf{1}_y & f(y) < \min_{x\in J'} f(x) \end{cases} \tag{2}$$

$g(\cdot, J')$ is computed as in (1) and $\mathbf{1}_y$ is Dirac's density on y.

Let y^{kT_0} denote the end point if search is incomplete at time kT_0 where $y^{kT_0} = \emptyset$ if the search is completed. The modified single-thread model based search is no longer a *stationary* Markov-chain. The one-step transition probability matrices $\{P^{nT_0} : n > 0\}$ is a stochastic process adapted to the filtration generated by the process $\{y^{nT_0} : n > 0\}$. Note that P^{nT_0} maintains upper triangular structure so that its eigenvalues (entries along diagonal) are of the form:

$$\lambda_J^{nT_0} = \Pr(J^{nT_0} = J | J^{(n-1)T_0} = J)$$

Let $\bar{P}^{nT_0}$ denote the product of one-step transition probability matrices:

$$\bar{P}^{nT_0} = \prod_{k>0}^{n} P^{kT_0}$$

Note that $\bar{P}^{nT_0}$ is also upper triangular so that its eigenvalues are the entries along the diagonal, i.e. they are of the form:

$$\bar{\lambda}_J^{nT_0} = \Pr(J^{nT_0} = J | J^0 = J) = \prod_{k>0}^{n} \lambda_J^{kT_0}$$

In our main result of this section we show that reacting to incomplete local searches provides an improved rate of convergence with probability 1. We will assume that the computational time required to identify any locally optimal solution is uniformly bounded.

Assumption 2 $\bar{n} = \sup_{y \in \Omega} \lfloor \frac{T(y)}{T_0} \rfloor < \infty.$

Theorem 1 *For all $n \geq \bar{n}$ and all $J \in 2^X$,*

$$(\lambda_J)^{n-\bar{n}} \geq \bar{\lambda}_J^{nT_0}$$

with probability one.

Proof We start by characterizing the process $\{\lambda_J^{nT_0} : n > 0\}$ as a function of the history of the process $\{y^{nT_0} : n > 0\}$. Conditional upon $J^{(n-1)T_0} = J$ and y^{nT_0}:

$$\lambda_J^{nT_0} = \begin{cases} \lambda_J & f(y^{nT_0}) \geq \min_{x \in J} f(x) \text{ or } \quad y^{nT_0} = \emptyset \\ 1 & f(y^{nT_0}) < \min_{x \in J} f(x) \text{ and } T(y^{nT_0}) > T_0 \\ 0 & f(y^{nT_0}) < \min_{x \in J} f(x) \text{ and } T(y^{nT_0}) \leq T_0 \end{cases}$$

To see why this is, recall that if the search is complete (i.e. $y^{nT_0} = \emptyset$) or the end-point is not of "high quality" (i.e. $f(y^{nT_0}) \geq \min_{x \in J} f(x)$ the single-thread model is not affected by the search outcome. The remaining cases are associated with an unsuccessful (respectively, successful) continuation of an incomplete search.

By Assumption 2, conditional upon y^{T_0}(the initial search outcome), we have

$$\bar{\lambda}_J^{nT_0} = \begin{cases} 0 & f(y^{T_0}) < \min_{x \in J} f(x) \\ \lambda_J \bar{\lambda}_J^{(n-1)T_0} & \text{otherwise} \end{cases}$$

for $n \geq \bar{n}$. Hence, by induction

$$\bar{\lambda}_J^{nT_0} \leq (\lambda_J)^{n-\bar{n}} \prod_{k>\bar{n}}^{n} \lambda_J^{kT_0} \leq (\lambda_J)^{n-\bar{n}}.$$

□

3 Interactive model-based search in real time

Having modified the single-thread model based search to react to incomplete searches we now consider a multiple-thread implementation, with M threads and denoting by $\mathbf{J} = (J_1, \ldots, J_M)$ the joint state of information. To avoid duplication of search efforts, all threads report their search outcome after T_0 units of computational time and the selection of a new model associated is subject to an acceptance-rejection test. The basic iteration of interactive model based search is as follows:

1. A sample y_i is drawn from the current model $g_i(J_i)$ and a local search algorithm is launched for each thread i.
2. After T_0 units of computational time, the current state J_i is updated as

$$J_i' = \begin{cases} J_i \cup \ell(y_i), & \text{if } T(y_i) \leq T_0 \\ J_i, & \text{otherwise} \end{cases}$$

3. A tentative new model $g_i(\cdot, J_i') \in \mathfrak{G}$ is selected by each thread i as in (1). The acceptance-rejection test is implemented so that the new model $g_i' \in \mathfrak{G}$ is determined as follows:

$$g_i' = \begin{cases} g\left(\cdot, J_i'\right) & D_{KL}\left(g\left(\cdot, J_i'\right), g\left(\cdot, J_\ell'\right)\right) > \eta \quad \forall \ell > i \\ \tilde{g}\left(\cdot, J_i'\right) & \text{otherwise} \end{cases}$$

where $\tilde{g}(\cdot, J_i') \in \arg\min_{g \in \mathfrak{G}} D_{KL}(\tilde{h}(J_i'), g)$ and

$$\tilde{h}(J) = \frac{I(x, J)U(x)}{\int_\Omega I(x, J)U(x)dx}$$

$$I(x, J) = \begin{cases} 1, & x \notin B(J) \cap \mathfrak{C}(T_0) \\ 0, & \text{otherwise} \end{cases}$$

Here the intent of the reference density $\tilde{h}$ is to choose a new model positing probability in areas not in the basin of attraction of the locally optimal solution in the state J_i'. In practice, an approximate solution to KL divergence minimization problem is needed. This is done by using moments of all empirical distributions. For example, when the class of sampling densities is normal with fixed variance then the empirical mean is used to find the distribution that approximately minimizes the KL divergence.

In what follows we shall use an independent multi-thread implementation as a benchmark. The following lemma which is adapted from [1] characterizes the performance of this scheme.

Lemma 1 *The interactive model based search can be modeled as a Markov chain* $\{\mathbf{J}^{nT_0} : n > 0\}$ *with an upper triangular transition probability matrix so that the eigenvalues are of the form* $\lambda_{\mathbf{J}}^* = \Pr(\mathbf{J}|\mathbf{J})$. *The process with M independent threads is also a Markov chain with eigenvalues of the form* $\lambda_{\mathbf{J}} = \prod_{i \leq M} \Pr(J_i|J_i)$. *Moreover,*

$$(\lambda_{[2]})^M \geq \lambda_{\mathbf{J}} \geq \lambda_{\mathbf{J}}^*$$

where $\lambda_{[2]}$ denotes the second largest eigenvalue of the single-thread model-based search.

Proof See [1] Theorem 3 and replace all $B(J_i)$ terms to $B(J_i) \cap \mathfrak{C}(T_0)$. □

3.1 Reactive resource allocation: avoiding duplication of incomplete search

In this section we propose a modification to the interactive model-based search scheme to reallocate resources in response to real-time information. We will use a relatively simple reactive strategy based upon the following ideas. Start points (or seeds) that have led to incomplete local searches should not continue to be used. We will show that the effect of this reallocation strategy is to speed up the identification of local optima when one or more threads repeatedly fail to do so. The reactive reallocation strategy outlined above operates in a different (slower) time-scale $\{nT_0 : n > 0\}$.

To motivate the analysis, recall that for single-thread implementation of model-based search the eigenvalues of the associated Markov chain are of the form:

$$\Pr(J_i \mid J_i) = \sum_{x \in J_i} \int_{B(x) \cap \mathfrak{C}(T_0)} g(y, J_i)dy + \int_{\Omega \backslash \mathfrak{C}(T_0)} g(y, J_i)dy$$

The reallocation strategy outlined above reduces the magnitude of this eigenvalue by modifying $g(\cdot, J_i)$ in order to reduce the probability to have new sample in $\Omega \backslash \mathfrak{C}(T_0)$ where local search cannot finish in time T_0 In what follows, we shall describe how to achieve the goal by introducing a “repulsive” force to the start location of unfinished local search.

We will describe how to incorporate the need to avoid duplication in incomplete search effort into the overall interactive model-based search scheme. Suppose that upon executing the interactive model-based algorithm we keep track of the set (say K_s) of starting points that have led to incomplete searches. In light of Assumption 1 we have:

$$\Omega \backslash \mathfrak{C}(T_0) = \cup_{x_i \in X \cup X^*} \tilde{B}(x_i)$$

where $\tilde{B}(x_i) = B(x_i) \cap \Omega \backslash \mathfrak{C}(T_0)$ Hence, for each $y \in K_s$ there exists a unique $x_i \in X \cup X^*$ such that $y \in B(x_i) \cap \Omega \backslash \mathfrak{C}(T_0)$. With a slight abuse of notation, in what follows, we shall refer to $\tilde{B}(x_i)$ as $\tilde{B}(y)$. Consider the modified reference density H defined as follows:

$$H(x, J, K_s) = \frac{I(x, J, K_s)U(x)}{\int_\Omega I(x, J, K_s)U(x)dx}$$

where

$$I(x, J, K_s) = \begin{cases} 1, & x \notin \cup_{y \in K_s} \tilde{B}(y) \cup B(J), \\ 0, & \text{otherwise} \end{cases}$$

In order to minimize the likelihood of sampling a starting point that would lead to either no new locally optimal information (state J) or another incomplete search outcome (the state K_s) the new candidate model $\tilde{g}$ is computed as follows:

$$\tilde{g}(\cdot, J, K_s) \in \arg\min_{g \in \mathfrak{G}} [D_{KL}(H(x, J, K_s), g)]$$

Thus, sampling from this density gives high likelihood to starting points or “seeds” that are more likely to result in new locally optimal solutions identified while avoiding incomplete searches.

We can now formally describe the modified interactive model-based search. Let $t = nT_0$, for M threads, with $\mathbf{K}_s^t = (K_{s,1}^t, \ldots, K_{s,M}^t)$ and $\mathbf{J}^t = (J_1^t, \ldots, J_M^t)$ as the current states, the interactive model-based search scheme with reactive allocation can be succinctly described as follows:

1. A sample y_i is drawn from the current model g_i^t and a local search algorithm is launched for each thread.
2. At time $t + T_0$, the result state is updated as

$$J_i^{t+T_0} = \begin{cases} J_i^t \cup \ell(y_i), & \text{if } T(y_i) \leq T_0 \\ J_i^t, & \text{otherwise} \end{cases}$$

   The set of starting points leading to incomplete searches is updated as

$$K_{i,s}^{t+T_0} = K_{i,s}^t \cup \{y_i \mid T(y_i) > T_0\}$$

   and record end-point $y_i^{t+T_0}$

3. A new model $g(\cdot, J_i^{t+T_0}) \in \mathfrak{G}$ is selected as in (2). If $y_i^{t+T_0} \geq \min_{x \in J_i^{t+T_0}} f(x)$, the acceptance-rejection test is implemented and the resulting model $g_i^{t+T_0}$ is computed as follows:

$$g_i^{t+T_0} = \begin{cases} g\left(\cdot, J_i^{t+T_0}\right) & D_{KL}\left(g\left(\cdot, J_i^{t+T_0}\right), g_\ell\left(\cdot, J_\ell^{t+T_0}\right)\right) > \eta \quad \forall \ell > i \\ \tilde{g}_i\left(\cdot, J_i^{t+T_0}, K_{i,s}^{t+T_0}\right) & \text{Otherwise} \end{cases}$$

   where $\tilde{g}_i(\cdot, J_i^{t+T_0}, K^{t+T_0}) \in \arg\min_{g \in \mathfrak{G}} D_{KL}(H(J_i, K_{i,s}^{t+T_0}), g)$.

3.2 Analysis

Having enlarged the state of information to include points leading to incomplete searches, the interactive model based search can no longer be modeled as a *stationary* Markov-chain. In fact, the sequence of one-step transition probability matrix $\{\mathbf{P}^{nT_0} : n > 0\}$ is a stochastic process adapted to the history of search outcomes, i.e. the processes $\{\mathbf{K}_s^{nT_0} : n > 0\}$ and $\{y_i^{nT_0} : n > 0\}$ for each thread i. Note that $\mathbf{P}^{nT_0}$ maintains upper triangular structure so that its eigenvalues (entries along diagonal) are of the form:

$$\lambda_{\mathbf{J}}^{nT_0} = \Pr(\mathbf{J}^{nT_0} = \mathbf{J} | \mathbf{J}^{(n-1)T_0} = \mathbf{J})$$

Let $\bar{\mathbf{P}}^{nT_0}$ denote the product of one-step transition probability matrices given the history of starting points resulting in incomplete local searches, i.e.:

$$\bar{\mathbf{P}}^{nT_0} = \prod_{k>0}^{n} \mathbf{P}^{kT_0}$$

Note that $\bar{\mathbf{P}}^{nT_0}$ is also upper triangular so that its eigenvalues are the entries along the diagonal, i.e. they are of the form:

$$\bar{\lambda}_{\mathbf{J}}^{nT_0} = \Pr(\mathbf{J}^{nT_0} = \mathbf{J} | \mathbf{J}^0 = \mathbf{J}) = \prod_{k>0}^{n} \lambda_{\mathbf{J}}^{kT_0}$$

In the following result we show in finite time the probability of identifying a globally optimal solution is increased when the finite history of incomplete local searches is used to avoid incomplete search.

Theorem 2 *For any $n > \bar{n}$ and any $\mathbf{J} \in (2^X)^M$, $(\lambda^*_{\mathbf{J}})^{n-\bar{n}} \geq \bar{\lambda}^{nT_0}_{\mathbf{J}}$ with probability one.*

Proof For any time nT_0 and any thread $i \leq M$,

$$\lambda^{nT_0}_{J_i} = \begin{cases} \varLambda_{J_i}\left(\hat{g}^{nT_0}_i, T_0\right) & f\left(y^{nT_0}_i\right) \geq \min\limits_{x \in J_i} f(x) \text{ or } \quad y^{nT_0}_i = \emptyset \\ 1 & f\left(y^{nT_0}_i\right) < \min\limits_{x \in J_i} f(x) \text{ and } T\left(y^{nT_0}_i\right) > T_0 \\ 0 & f\left(y^{nT_0}_i\right) < \min\limits_{x \in J} f(x) \text{ and } T\left(y^{nT_0}\right) \leq T_0, \end{cases}$$

where

$$\hat{g}^{nT_0}_i = \begin{cases} g\left(\cdot, J^{t+T_0}_i\right) & D_{KL}\left(g\left(\cdot, J^{t+T_0}_i\right), g_\ell\left(\cdot, J^{t+T_0}_\ell\right)\right) > \eta \quad \forall \ell > i \\ \tilde{g}_i\left(\cdot, J^{t+T_0}_i, K^{t+T_0}_{i,s}\right) & \text{Otherwise} \end{cases}$$

and $g(\cdot, J^{t+T_0}_i) \in \mathfrak{G}$ is selected as in (1) instead of (2). Define $\hat{\lambda}^{nT_0}_{\mathbf{J}} = \prod_{i \leq M} \varLambda_{J_i}(\hat{g}^{nT_0}_i, T_0)$, from Theorem 1,

$$\prod_{k=1}^{n-\bar{n}} \hat{\lambda}^{kT_0}_{\mathbf{J}} \geq \bar{\lambda}^{nT_0}_{\mathbf{J}}, \quad \text{with probability one.}$$

For any $n > \bar{n}$, one sufficient condition of $(\lambda^*_{\mathbf{J}})^{n-\bar{n}} \geq \bar{\lambda}^{nT_0}_{\mathbf{J}}$ with probability one is: $\lambda^*_{\mathbf{J}} \geq \hat{\lambda}^{nT_0}_{\mathbf{J}}$, $\forall n > 0$ with probability one.

Consider first the case with two threads ($M = 2$), recall that

$$\lambda^*_{\mathbf{J}} = \begin{cases} \varLambda_{J_1}(g_1, T_0)\varLambda_{J_2}(g_2, T_0) & D_{KL}(g_1, g_2) > \eta \\ \varLambda_{J_1}(\tilde{g}_1(J_1), T_0)\varLambda_{J_2}(g_2, T_0) & D_{KL}(g_1, g_2) \leq \eta \end{cases}$$

and

$$\hat{\lambda}^{nT_0}_{\mathbf{J}} = \begin{cases} \varLambda_{J_1}(g_1, T_0)\varLambda_{J_2}(g_2, T_0) & D_{KL}(g_1, g_2) > \eta \\ \varLambda_{J_1}(\tilde{g}_1(J_1, K^{nT_0}_{s,1}), T_0)\varLambda_{J_2}(g_2, T_0) & D_{KL}(g_1, g_2) \leq \eta \end{cases}$$

where

$$\varLambda_J(g, T_0) = \int_{B(J) \cap \mathfrak{C}(T_0)} g(y, J)dy + \int_{\varOmega \setminus \mathfrak{C}(T_0)} g(y, J)dy$$

and

$$\tilde{g}_1(J_1, K_{s,1}) = \arg\min_{g \in \mathfrak{G}} D_{K,L}(H(x, J_1, K_{s,1}), g).$$

From the definition of $\tilde{g}_1(J_1, K_{s,1})$ and $\tilde{g}_1(J_1)$, it follows that

$$\int_\varOmega \ln \frac{\tilde{g}_1\left(x, J_1, K^{nT_0}_{s,1}\right)}{H\left(x, J_1, K^{nT_0}_{s,1}\right)} H\left(x, J_1, K^{nT_0}_{s,1}\right) dx \geq \int_\varOmega \ln \frac{\tilde{g}_1(x, J_1)}{H\left(x, J_1, K^{nT_0}_{s,1}\right)} H\left(x, J_1, K^{nT_0}_{s,1}\right) dx$$

$$\int_\varOmega \ln \frac{\tilde{g}_1\left(x, J_1, K^{nT_0}_{s,1}\right)}{H(x, J_1)} H(x, J_1)dx \leq \int_\varOmega \ln \frac{\tilde{g}_1(x, J_1)}{H(x, J_1)} H(x, J_1)dx$$

which implies

$$\int_{[B(J_1)\cap\mathfrak{C}(T_0)]\cup\tilde{B}\left(K_{s,1}^{nT_0}\right)} \tilde{g}_1\left(J_1, K_{s,1}^{kT_0}, x\right)dx \le \int_{[B(J_1)\cap\mathfrak{C}(T_0)]\cup\tilde{B}\left(K_{s,1}^{nT_0}\right)} \tilde{g}_1(J_1, x)dx$$

$$\int_{B(J_1)\cap\mathfrak{C}(T_0)} \tilde{g}_1\left(J_1, K_{s,1}^{kT_0}, x\right)dx \ge \int_{B(J_1)\cap\mathfrak{C}(T_0)} \tilde{g}_1(J_1, x)dx$$

When $K_{s,1}^{nT_0} = \emptyset$, $\tilde{g}_1(J_1, K_{s,1}^{nT_0}) = \tilde{g}_1(J_1)$ and $\lambda_{\mathbf{J}}^* = \hat{\lambda}_{\mathbf{J}}^{nT_0}$. When $K_{s,1}^{nT_0} \neq \emptyset$ then:

$$[B(J_1)\cap\mathfrak{C}(T_0)] \subsetneq \left\{[B(J_1)\cap\mathfrak{C}(T_0)]\cup\tilde{B}\left(K_{s,1}^{nT_0}\right)\right\} \subseteq \{[B(J_1)\cap\mathfrak{C}(T_0)]\cup[\Omega\backslash\mathfrak{C}(T_0)]\}$$

and

$$\int_{[(B(J_1)\cap\mathfrak{C}(T_0))]\cup[\Omega\backslash\mathfrak{C}(T_0)]} \tilde{g}_1\left(J_1, K_{s,1}^{nT_0}, x\right)dx \le \int_{[(B(J_1)\cap\mathfrak{C}(T_0))]\cup[\Omega\backslash\mathfrak{C}(T_0)]} \tilde{g}_1(J_1, x)dx$$

thus

$$\Lambda_{J_1}\left(\tilde{g}_1\left(J_1, K_{s,1}^{nT_0}\right), T_0\right) \le \Lambda_{J_1}\left(\tilde{g}_1(J_1), T_0\right)$$

Hence, $\lambda_{\mathbf{J}}^* \ge \hat{\lambda}_{\mathbf{J}}^{nT_0}$.

We now prove the case in which $M > 2$ by induction. Assume the result holds for any $M-1$ dimensional state of information, say $\mathbf{J}'$. Let us construct an M dimensional state $\mathbf{J}$ as follows:

$$\mathbf{J} = \{J_1\} \times \mathbf{J}'$$

where $\mathbf{J}' = (J_2, \ldots, J_M)$ is a $M-1$ dimension state variable. The induction hypothesis implies $\lambda_{\mathbf{J}'}^* \ge \hat{\lambda}_{\mathbf{J}'}^{nT_0}$. Recall that the interaction between threads is hierarchical: any given thread only interacts with *higher-indexed* threads. Thus, adding a new thread with say index 1 (as in the construction of $\mathbf{J}$ above) has no impact on threads 2 to M. The eigenvalues $\lambda_{\mathbf{J}'}^*$ and $\hat{\lambda}_{\mathbf{J}'}^{nT_0}$ are independent of thread 1. Thus,

$$\lambda_{\mathbf{J}}^* = \begin{cases} \Lambda_{J_1}(g_1, T_0)\cdot\lambda_{\mathbf{J}'}^* & D_{KL}(g_1, g_j) > \eta\ \forall j \in \{2, \ldots, M-1\} \\ \Lambda_{J_1}(\tilde{g}_1(J_1), T_0)\cdot\lambda_{\mathbf{J}'}^* & \text{otherwise} \end{cases}$$

and

$$\hat{\lambda}_{\mathbf{J}}^{nT_0} = \begin{cases} \Lambda_{J_1}(g_1, T_0)\cdot\lambda_{\mathbf{J}'}^{nT_0} & D_{KL}(g_1, g_j) > \eta\ \forall j \in \{2, \ldots, M-1\} \\ \Lambda_{J_1}\left(\tilde{g}_1\left(J_1, K_{s,1}^{nT_0}\right), T_0\right)\cdot\lambda_{\mathbf{J}'}^{nT_0} & \text{otherwise.} \end{cases}$$

Using the same argument as above we have:

$$\Lambda_{J_1}\left(\tilde{g}_1\left(J_1, K_{s,1}^{nT_0}\right), T_0\right) \le \Lambda_{J_1}(\tilde{g}_1(J_1), T_0)$$

which together with $\lambda_{\mathbf{J}'}^* \ge \hat{\lambda}_{\mathbf{J}'}^{nT_0}$ implies $\lambda_{\mathbf{J}}^* \ge \hat{\lambda}_{\mathbf{J}}^{nT_0}$ with probability one. □

Remark In Theorems 1 and 2, we have shown that (with probability 1), the proposed parallel algorithmic scheme reduces *all* eigenvalues of the associated Markov chain. Therefore, the expected time to reach X^* which is a function of all eigenvalues is also reduced by the proposed scheme.

4 Numerical experiments

In this section we report the results from a series of numerical tests aimed at illustrating the improved performance enabled by a an interactive approach to model-based search. The proposed algorithm is compared with " Independent Model Reference Adaptive Search" from [2] and "Interactive Model-based search" from [1]. The models used in the experiment correspond to class of multivariate normal distributions with the fixed covariance matrix $\Sigma = 0.03I$. The local search method set to BFGS Quasi-Newton method.

4.1 Ackley problem

Ackley's Problem [14] is to find $x \in \mathbb{R}^n$, with $x_i \in (-32.768, 32.768)$, that minimizes the following function:

$$H(x) = -20 \cdot \exp\left(-0.2\sqrt{\frac{1}{n}\cdot\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\cdot\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \exp(1).$$

For interactive model based search method. We determine the center of rejected model distribution by sampling a finite set of trial random selected points (choosing 20 in this experiment). Calculate distance between the trailing points and the set of detected local optimas and take the one with largest total distance as the mean of reference distribution which is an approximation of distribution with maximum KL divergence.

For the modified interactive model search method, we set the iteration with cap 350 function evaluation. 20 trial points are sampled to determine new sample by choosing the one with largest total distance to the set which contains both starting points of incomplete search and detected local minimas.

To measure real local search time, we use the number of objective function evaluations as a standardized time ticks. We report the average evaluations required before reaching global optima in Table 1. The proposed algorithm requires less function evaluations to find global optima than independent model reference search and interactive model-based search, the improvement increases with the number of threads.

According to [11], we also included two finite time performance metrics as (1) the probability of finding global optimum after 5000 evaluations in Table 2 and (2) average lowest objective function value found after 5000 evaluations in Table 3 and Fig. 1. The modified interactive model achieve lower objective function value and high probability to find global optima than independent and interactive model search. The acceleration becomes more significant as number of threads increases.

Table 1 Average number of evaluations before reaching global optima

Number of threads	5	10	25	50
Independent model search	13732	10698	8223	7194
Interactive model search	13388	10491	8046	7123
Modified interactive model search	12827	9236	7032	5820

Table 2 Probability of finding global optima within 5000 evaluations

Number of threads	5 (%)	10 (%)	25 (%)	50 (%)
Independent model search	7.6	10.8	15.8	24.0
Interactive model search	7.0	12.6	21.4	26.2
Modified interactive model search	10.2	18.8	28.8	43.6

Table 3 Average lowest function value found within 5000 evaluations

Number of threads	5	10	25	50
Independent model search	6.725	4.593	2.023	1.115
Interactive model search	6.703	4.448	1.927	1.031
Modified interactive model search	6.888	3.904	1.645	0.385

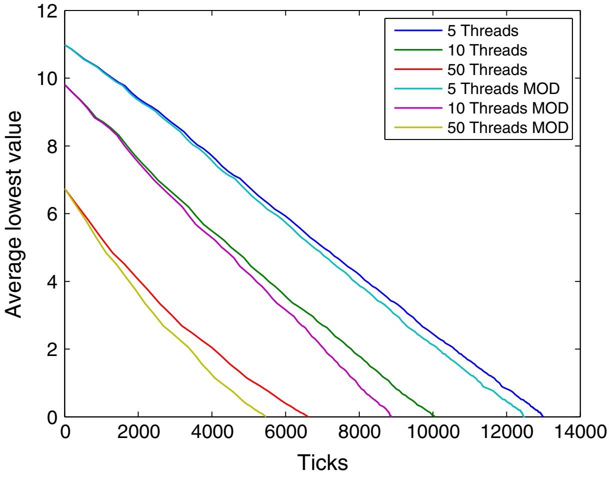

Fig. 1 Original and modified interactive model search method

4.2 Rastrigin problem

The Rastrigin problem is the minimization of the function defined as follows:

$$F(x) := 10n + \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i)\right] \quad x_i \in [-5.12, 5.12], i = 1, 2, \ldots, n.$$

we have run numerical experiment on 3-D Rastrigin function with covariance matrix $\Sigma = 0.03I$.

We take same 20 trial points to generate reference distribution mean of rejected threads. For modified interactive model search method, we set the iteration cap as 60 function evaluation. Using number of objective function evaluations as time measurement ticks, the average number of evaluations before reaching global optima is present in Table 4. The finite time

Table 4 Average number of ticks before reaching global optima

Number of threads	5	10	25	50
Independent model search	5784.5	4664.6	3363.4	2406.6
Interactive model search	6354.9	4866.6	2925.5	1671.4
Modified interactive model search	6253.6	4304.4	2486.4	1335.2

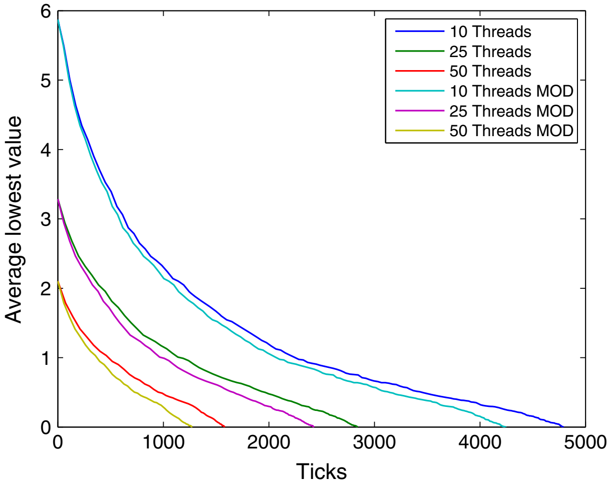

Fig. 2 Original and modified interactive model search method

Table 5 Probability of finding global optima within 2000 evaluations

Number of threads	5 (%)	10 (%)	25 (%)	50 (%)
Independent model search	11.0	15.2	28.8	44.6
Interactive model search	10.6	16.2	37.6	62.0
Modified interactive model search	9.4	19.2	38.6	70.2

Table 6 Average lowest function value found within 2000 evaluations

Number of threads	5	10	25	50
Independent model search	4.294	2.762	1.385	0.549
Interactive model search	4.103	2.336	0.927	0.000
Modified interactive model search	3.892	2.150	0.614	0.000

(within 2000 evaluations) performance metrics are presented in Fig. 2 and Tables 5 and 6. We can also notice that the modified interactive model search outperform independent and interactive model search.

5 Conclusions

In this paper we consider a parallel computing scheme for global optimization that combines multi-start local search with the dynamic reallocation of computational resources (e.g. processing time). Our work builds upon the interactive model-based approach to global optimization proposed in [1] in which parallel threads independently execute a model-based search method (see [2]) and periodically interact through a simple acceptance-rejection rule aimed at preventing duplication of search efforts.

While sophisticated algorithmic variations can be designed in order to optimally react to search outcomes our focus is to leverage relatively simple ideas such as (i) continuing searches that are promising because the end-points have lower objective values than all other solutions found so far and (ii) avoiding duplication of failed searches and/or search effort across threads. The relatively small gains afforded by these simple ideas are shown to be magnified by parallelization: the rate of convergence for an interactive model-based search increases with the number of threads for a wide-class of local search techniques (i.e. model-based) when compared to independent parallel implementation.

References

1. Wang, Y., Garcia, A.: Interactive model-based search for global optimization. J. Glob. Optim. **61**(3), 479–495 (2015)
2. Hu, J., Fu, M., Marcus, S.: A model reference adaptive search algorithm for global optimization. Oper. Res. **55**(3), 549–568 (2007)
3. Schoen, F.: Stochastic techniques for global optimization: a survey of recent advances. J. Glob. Optim. **1**(3), 207–228 (1991)
4. Martí, R., Moreno-Vega, J., Duarte, A.: Advanced Multi-start Methods. Handbook of Metaheuristics, 2nd edn. Springer, New York (2010)
5. Onbasglu, E., Ozdamar, L.: Parallel simulated annealing algorithms in global optimization. J. Glob. Optim. **19**, 27–50 (2001)
6. Ferreiro, A., Garcia, J.A., Lopez-Salas, J.G., Vazquez, C.: An efficient implementation of parallel simulated annealing algorithm in GPUs. J. Glob. Optim. **57**(3), 863–890 (2013)
7. Schutte, J.F., Reinbolt, J.A., Fregly, B.J., Haftka, R.T., George, A.D.: Parallel global optimization with the particle Swarm algorithm. Comput. Sci. Res. Dev. **61**(13), 2296–2315 (2004)
8. D'Apuzzo, M., Marino, M., Migdalas, A., Pardalos, P.M., Toraldo, G.: Parallel computing in global optimization. In: Kontoghiorghes, E.J. (ed.) Handbook of Parallel Computing and Statistics, pp. 225–258. Chapman and Hall/CRC, London (2005)
9. Boender, C.G.E., Rinnooy Kan, A.H.G.: Bayesian stopping rules for multistart global optimization methods. Math. Program. **37**, 59–80 (1987)
10. Gyorgy, A., Kocsis, L.: Efficient multi-start strategies for local search algorithms. J. Artif. Intell. Res. **41**, 407–444 (2011)
11. Calvin, J.M., Zilinskas, A.: On a global optimization algorithm for bivariate smooth functions. J. Optim. Theory Appl. **163**(2), 528–547 (2014)
12. Zielinski, A.: A statistical estimate of the structure of multi-extremal problems. Math. Program. **21**(3), 348–356 (1981)
13. Zhigljavsky, A.: Semiparametric statistical inference in global random search. Acta Appl. Math. **33**(1), 69–88 (1993)
14. Ackley, D.H.: A Connectionist Machine for Genetic Hillclimbing. Kluwer, Boston (1987)