A First Runtime Analysis of Parallel Tempering in Quadratic Optimization

Tiago Paixão[1] , **Jorge Pérez Heredia**[2] , **Dirk Sudholt**[3] and **Andrew M. Sutton**[4]
[1]Gulbenkian Institute for Molecular Medicine
[2]Nextail Labs SLU
[3]University of Passau
[4]University of Minnesota Duluth
tiago.paixao@gimm.pt, jorge.perez@nextail.co, dirk.sudholt@uni-passau.de, amsutton@d.umn.edu

Abstract

Parallel tempering, or the replica exchange method, is a Markov Chain Monte Carlo (MCMC) sampling technique for finding low-energy states in complex landscapes by executing multiple processes at different temperatures and allowing for states to migrate between parallel processes based on the Metropolis criterion. Despite a growing interest in the technique as a randomized search heuristic, it is not clear when and why parallel tempering is effective. We conduct a first runtime analysis of the parallel tempering algorithm on a simple quadratic unconstrained binary optimization problem that induces a rugged energy landscape with tunable parameters. We prove that a single-temperature Metropolis process fails to efficiently optimize the problem at any temperature. In sharp contrast, a two-state parallel tempering approach using a low and a high temperature typically solves the problem in $O(n^2 \log n)$ iterations. The low-temperature process ensures stability, as it is unlikely to accept worsenings, whereas the high-temperature process facilitates exploration by traversing energy barriers. Improvements are discovered through exploration in the high-temperature process and transferred to the low-temperature process, where they are permanently retained. This effective interplay between processes demonstrates the power of parallel tempering and lends credence to its design.

1 Introduction

Parallel tempering, also known as the replica exchange method, is a Markov Chain Monte Carlo (MCMC) technique for finding low-energy states in complex landscapes by executing multiple processes at different temperatures and allowing for states to migrate between parallel processes based on the Metropolis criterion. The intuition behind this approach is that configurations at low temperatures can efficiently exploit landscape features by stochastic hill-climbing, whereas configurations at high temperatures are more likely to escape local minima to allow for adequate exploration. Allowing processes at different temperatures to periodically exchange states results in a robust ensemble that balances intensification and diversification during the search process.

Originally introduced as a technique in statistical mechanics for simulating physical systems in order to efficiently compute thermodynamical properties such as specific heat [Swendsen and Wang, 1986; Geyer, 1991], parallel tempering has recently gained interest as a randomized search heuristic for optimization. For example, Zhu et al. 2020 pointed out the relationship between the statistical physics of Ising spin glasses and hard Boolean optimization problems and observed that parallel tempering can be used to tackle this kind of problem. Their `borealis` solver, based on parallel tempering, was ranked first for random MaxSAT instances in the Incomplete Solver track at the Eleventh Max-SAT Evaluation at SAT 2016.

Implementations of parallel tempering have recently emerged as digital annealing hardware for solving quadratic unconstrained optimization (QUBO) problems [Aramon *et al.*, 2019], as well as an API for a multi-core implementation [Almeida *et al.*, 2025]. The approach has been successfully applied for decades on a wide range of problems such as overcoming local minima in numerical simulations of molecules with rough energy landscapes [Hansmann, 1997], Bayesian data analysis [Habeck *et al.*, 2005], and makespan minimization on instances of various NP-hard scheduling problems [Almeida *et al.*, 2025].

Despite a long and successful history in various applications and many solid experimental results, so far a rigorous theoretical foundation for parallel tempering is missing. In particular, it is not clear when and why parallel tempering is more powerful than the Metropolis algorithm maintaining a single state.

In this paper, we make a first step towards establishing a rigorous theoretical foundation for parallel tempering by conducting a first runtime analysis on a simple QUBO problem with a tunably rugged landscape. We prove that the classical Metropolis algorithm fails to efficiently optimize the problem when the temperature is set too low, in which case it cannot easily escape local minima, or when the temperature is set too high, in which case its behavior is too chaotic to effectively leverage the global structure of the energy landscape. We furthermore show that, for certain settings of energy barrier parameters in the landscape, these temperature regimes will overlap and consequently there is no ideal temperature

setting that guarantees success. In contrast, we prove that a two-state parallel tempering approach using a low and high temperature and asymmetric migration rather than symmetric exchanges typically optimizes the problem in $O(n^2 \log n)$ iterations.

Our analysis gives insights into the working principles of parallel tempering and reveals why the interplay of two states running at different temperatures is crucial to optimize the benchmark efficiently. Our results demonstrate the power of parallel tempering and lend credence to its design.

1.1 Related Work

Several runtime analyses have been conducted for the Metropolis algorithm, which maintains a single state and a single (fixed) temperature. Sasaki and Hajek [1988] showed that the Metropolis algorithm can approximate a maximum matching on graphs with n vertices within a $(1-\epsilon)$ factor in expected time at most $\frac{6}{\epsilon^2} 2^{2/\epsilon} n^{3+2/\epsilon}$. Jerrum and Sorkin [1998] proved that the Metropolis algorithm computes an optimal graph bisection in quadratic time.

Allowing the system temperature to vary with time results in the simulated annealing algorithm [Kirkpatrick *et al.*, 1983; Černý, 1985], and a variety of runtime analyses investigate the utility of allowing temperature fluctuations in different search landscapes. Wegener [2005] gave the first example where simulated annealing outperforms the Metropolis algorithm with any fixed temperature. Our negative results for the Metropolis algorithm are inspired by his work. Jansen and Wegener [2007] proved that the Metropolis algorithm solves the ONEMAX benchmark function in $O(n \log n)$ time with a suitably low temperature setting. They also compared Metropolis, simulated annealing and a simple evolutionary algorithm called the (1+1) EA on a number of benchmark functions and proved the existence of functions that Metropolis solves in polynomial time, but the (1+1) EA needs at least exponential time. In contrast, Lissovoi, Oliveto and Warwicker [2023] proved that the Metropolis algorithm fails to efficiently solve the simple multi-modal CLIFF function, while Doerr et al. [2023] proved that the (1+1) EA is comparatively faster than Metropolis on CLIFF.

Oliveto et al. [2018] investigated the Metropolis algorithm and a related model from population genetics called SSWM and proved that they are able to optimize the rugged VALLEY function more efficiently than simple evolutionary algorithms by exploiting their ability to accept worsening moves. Wang, Zheng and Doerr [2024] proved that the Metropolis algorithm solves the DECEPTIVELEADINGBLOCKS problem in quadratic time, while comparable algorithms that do not accept inferior moves require at least cubic time. Zheng et al. [2024] analyzed the Metropolis algorithm in the context of multi-objective optimization, considering a bi-objective variant of DECEPTIVELEADINGBLOCKS.

Migration is a key concept in parallel evolutionary algorithms such as island models, in which several populations evolve separately, but coordinate their search processes by migrating solutions between them [Luque and Alba, 2011; Alba *et al.*, 2013; Sudholt, 2015]. For these algorithms, a theoretical foundation has been established, cf. [Sudholt, 2015] and the references therein. Most relevant to our work is [Lässig and Sudholt, 2013], which showed that migration can yield exponential speedups for a constructed benchmark problem. In addition, heterogeneous island models, in which islands solve different aspects of a problem, were proven to be effective for the SETCOVER problem [Mambrini *et al.*, 2012].

2 Preliminaries

A pseudo-Boolean function is a real-valued mapping $f\colon \{0,1\}^n \to \mathbb{R}$. A pseudo-Boolean function f is called *quadratic* when it can be written in the form of a quadratic multi-linear polynomial:

$$f(x_1, x_2, \ldots, x_n) = c_0 + \sum_{i=1}^{n} c_i x_i + \sum_{1 \le i < j \le n} q_{ij} x_i x_j. \quad (1)$$

In *quadratic unconstrained binary optimization* (QUBO), given a quadratic pseudo-Boolean function f, the objective is to solve

$$\min_{\boldsymbol{x} \in \{0,1\}^n} f(\boldsymbol{x}).$$

We introduce a simple QUBO benchmark that allows for tuning the height of energy barriers, but is elementary enough so that the behavior of algorithms is better clarified.

Definition 1. *Let n be a positive even integer and $\delta_L < \delta_G$ be the energy barrier parameters with $1 \le \delta_L = O(1)$. We define the quadratic pseudo-Boolean function* $\text{RUGGEDLANDSCAPE}_{n,\delta_L,\delta_G}$ *as follows. The quadratic terms following* (1) *are*

$$q_{ij} = \begin{cases} -(\delta_G + \delta_L) & \text{if } i \equiv 1 \pmod 2 \text{ and } j = i+1, \\ 0 & \text{otherwise.} \end{cases}$$

The linear terms are $c_i = \delta_L$ for $i > 0$ and the constant term is $c_0 = \frac{n}{2}(\delta_G - \delta_L)$.

The class $\text{RUGGEDLANDSCAPE}_{n,\delta_L,\delta_G}$ contains functions over length-n binary vectors where quadratic interactions are "partitioned" into $n/2$ disjoint bit-pairs $(x_1, x_2), (x_3, x_4), \ldots, (x_{n-1}, x_n)$, and no interactions occur between pairs. Moreover, bits within a pair exhibit so-called reciprocal sign epistasis [Poelwijk *et al.*, 2011] in the sense that the contribution to the energy from a $(0,0)$-pair is $c_0/(n/2) = \delta_G - \delta_L$, the contribution from a $(1,0)$ or $(0,1)$ pair is $c_0/(n/2) + \delta_L = \delta_G$, while the contribution from a $(1,1)$ pair is $c_0/(n/2) + 2\delta_L - (\delta_G + \delta_L) = 0$.

Let $\boldsymbol{x} \in \{0,1\}^n$ be a binary vector and $\pi = (x_i, x_{i+1})$ a bit pair such that $n > i \equiv 1 \pmod 2$. When $x_i = x_{i+1} = 0$, we say π is a *metastable* pair. When $x_i + x_{i+1} = 1$, we say π is an *unstable* pair. Finally, when $x_i = x_{i+1} = 1$, we say π is a *stable* pair.

To propose new states, the algorithms we study sample from the Hamming neighborhood, i.e., one bit is flipped at a time. This induces a topology on the energy function that makes it possible to define structural properties such as local minima and energy barriers. A state $\boldsymbol{x}$ is a local minimum with respect to Hamming neighbors if and only if it has no unstable pairs. If $\boldsymbol{x}$ is a local minimum with metastable pairs, it is a *metastable* state (non-global local minimum). If a local minimum has no metastable pairs, it is a global minimum

with zero energy. The landscape of RUGGEDLANDSCAPE thus contains $2^{n/2} - 1$ metastable states separated by states with unstable pairs. In order to escape a metastable state, a process must pass through an unstable state with higher energy. Let $\boldsymbol{x}$ be a metastable state and $\boldsymbol{y}$ a Hamming neighbor of $\boldsymbol{y}$. If $\boldsymbol{y}$ has one fewer metastable pair than $\boldsymbol{x}$, then $f(\boldsymbol{y}) = f(\boldsymbol{x}) + \delta_L > f(\boldsymbol{x})$. Similarly, if $\boldsymbol{y}$ has one fewer stable pair, then $f(\boldsymbol{y}) = f(\boldsymbol{x}) + \delta_G > f(\boldsymbol{x})$. In any case, to escape a metastable state, the processes must transition through an "energy barrier" of height at least δ_L.

Our proofs makes use of the multiplicative drift theorem, which connects the expected change in a random process to the first hitting time of that process to a certain state of interest. The multiplicative drift theorem can be stated as follows.

Theorem 1 (Doerr and Goldberg, 2013)**.** *Let $(X_t)_{t\in\mathbb{N}}$ be a non-negative stochastic process over a state space $\mathcal{S} \subseteq \{0\} \cup [s_{\min}, \infty)$ where $s_{\min} > 0$, and let $T \coloneqq \inf\{t \geq 0 \mid X_t = 0\}$. Suppose that $X_0 = s_0$ and there exists a $\delta > 0$ such that for all $s \in \mathcal{S} \setminus \{0\}$ and all $t \geq 0$*

$$\mathrm{E}\left[X_t - X_{t+1} \mid X_t = s\right] \geq \delta s.$$

Then the following tail bound holds for the first hitting time of the process to zero:

$$\Pr\left(T \geq \left\lceil \frac{r + \ln(s_0/s_{\min})}{\delta} \right\rceil\right) \leq e^{-r}.$$

We also use the negative drift theorem with self-loops presented in [Rowe and Sudholt, 2014] (an extension of the negative drift theorem [Oliveto and Witt, 2011; Oliveto and Witt, 2012] to stochastic processes with large self-loop probabilities). It is stated in the following for the sake of completeness.

Theorem 2 (Rowe and Sudholt, 2014)**.** *Consider a Markov process $X_0, X_1, \ldots$ on $\{0, \ldots, m\}$ with transition probabilities $p_{i,j}$ and suppose there exist integers a, b with $0 < a < b \leq m$ and $\varepsilon > 0$ such that for all $a \leq k \leq b$ the drift towards 0 is*

$$\mathrm{E}\left[k - X_{t+1} \mid X_t = k\right] < -\varepsilon \cdot (1 - p_{k,k}) \tag{2}$$

where $p_{k,k}$ is the self-loop probability at state k. Further assume there exist constants $r, \delta > 0$ (i. e. they are independent of m) such that for all $k \geq 1$ and all $d \geq 1$

$$p_{k,k+d} \leq \frac{r(1-p_{k,k})}{(1+\delta)^d} \quad \text{and} \quad p_{k,k-d} \leq \frac{r(1-p_{k,k})}{(1+\delta)^d}. \tag{3}$$

Let T be the first hitting time of a state at most a, starting from $X_0 \geq b$. Let $\ell = b - a$. Then there is a constant $c > 0$ such that

$$\Pr\left(T \leq 2^{c\ell/r}\right) = 2^{-\Omega(\ell/r)}.$$

3 MCMC Algorithms for Optimization

Markov chain Monte Carlo (MCMC) sampling is a technique for exploring a state space $\mathcal{X}$ by defining a Markov process on $\mathcal{X}$ whose stationary distribution, under certain balance conditions, converges in the limit to a target distribution. Many of these techniques are useful in the context of optimization when considered as randomized search heuristics in which the state space is the set of candidate solutions and the target distribution is proportional to the objective function.

One of the first such methods is the Metropolis algorithm [Metropolis *et al.*, 1953; Hastings, 1970]. This algorithm maintains a single state, and iteratively performs a biased random walk. In each iteration, a random neighbor $\boldsymbol{y}$ of the current state $\boldsymbol{x}$ is produced. If $f(\boldsymbol{y}) \leq f(\boldsymbol{x})$, then the state $\boldsymbol{y}$ is accepted as the new current state. Otherwise, $\boldsymbol{y}$ is accepted with probability $e^{-\Delta f/T}$ where $\Delta f \coloneqq f(\boldsymbol{y}) - f(\boldsymbol{x})$ is the difference in objective function value and T is a temperature parameter that governs how likely a move to a higher energy state is. The Metropolis algorithm is listed in Algorithm 1.

The temperature T is one of the most important parameters in Metropolis and similar MCMC sampling techniques. The process can permeate an energy barrier with height at most T, but it cannot easily escape from a valley with depth greater than T. Furthermore, the process cannot easily follow gradients with magnitude smaller than T, so the precision of the result depends heavily on how the temperature is set.

One way to address this is to also adapt the temperature. Simulated annealing [Kirkpatrick *et al.*, 1983; Černý, 1985] takes a "cooling" schedule and decreases the temperature monotonically during the run of the algorithm. Simulated tempering treats the temperature as another state to sample [Marinari and Parisi, 1992]. Given a set of temperatures $T_1 < \cdots < T_\kappa$, a single state $(\boldsymbol{x}, i)$ is maintained. The $\boldsymbol{x}$ update is a Metropolis update with temperature T_i, and $i \to i \pm 1$ is also updated stochastically. The idea is to use a wide range of temperatures without converging too soon into a low temperature suboptimal configuration.

Rather than maintaining a single state that explores the temperature space, parallel tempering also takes a sequence of temperatures $T_1 < \cdots < T_\kappa$, but additionally maintains a pool of κ parallel states $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\kappa$, each performing a Metropolis update with its respective temperature value [Swendsen and Wang, 1986; Geyer, 1991]. Periodically, neighboring states are exchanged using a Metropolis criterion based on the temperature ratios. In particular, an exchange between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ is accepted with probability

$$\min\left\{1, e^{\left(-(f(\boldsymbol{x}_{i+1}) - f(\boldsymbol{x}_i))\left(\frac{1}{T_i} - \frac{1}{T_{i+1}}\right)\right)}\right\}. \tag{4}$$

The pseudocode for parallel tempering is listed in Algorithm 2. Varying approaches exist for implementing the frequency of exchanges, e.g., by performing an exchange every M steps, or in each step but only with a fixed probability q.

In the classical approach, the exchange operation is *symmetric* so no configuration is ever lost during an exchange. In particular if $\boldsymbol{x}_i$ exchanges with $\boldsymbol{x}_j$ in a time step, then $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ remain in the ensemble, but in different processes with different temperatures. In this paper, we employ a slight modification to this approach by allowing for *asymmetric* exchanges. This means it is possible for a state to be lost in an exchange if it is overwritten by an incoming state. For that reason, we call the procedure a *migration* rather than an exchange. This gives rise to Algorithm 3 in which, during a migration step, process $i + 1$ migrates to process i with prob-

ability

$$\min\left\{1, \exp\left(\frac{-(f(\boldsymbol{x}_{i+1}) - f(\boldsymbol{x}_i))}{T_i}\right)\right\}, \tag{5}$$

and process i migrates to process $i+1$ with probability

$$\min\left\{1, \exp\left(\frac{-(f(\boldsymbol{x}_i) - f(\boldsymbol{x}_{i+1}))}{T_{i+1}}\right)\right\}. \tag{6}$$

As we will see later, this adaptation allows the algorithm to carefully leverage the different temperatures without losing valuable state information to an unlucky exchange. Moreover, a symmetric exchange occurs with the joint probability of both events, the product of which recovers the classical exchange probability in Equation (4).

Algorithm 1: Metropolis Algorithm

Input: state $\boldsymbol{x}$
Input: temperature $T > 0$
repeat
  $\boldsymbol{y} \leftarrow$ `neighbor(`$\boldsymbol{x}$`)`;
  $p \leftarrow \exp\left(\frac{-(f(\boldsymbol{y}) - f(\boldsymbol{x}))}{T}\right)$;
  if `random()` $\leq p$ **then** $\boldsymbol{x} \leftarrow \boldsymbol{y}$;
until *stop*;

Algorithm 2: Parallel Tempering

Input: κ states $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\kappa$
Input: κ temperatures $0 < T_1 < \cdots < T_\kappa$
repeat
  `/* Perform Metropolis updates */`
  for $i \in \{1, \ldots, \kappa\}$ **do**
    $\boldsymbol{y} \leftarrow$ `neighbor(`$\boldsymbol{x}_i$`)`;
    $p \leftarrow \exp\left(\frac{-(f(\boldsymbol{y}) - f(\boldsymbol{x}_i))}{T_i}\right)$;
    if `random()` $\leq p$ **then** $\boldsymbol{x}_i \leftarrow \boldsymbol{y}$;
  `/* Pair exchange */`
  periodically
    Choose $i \in \{1, \ldots, \kappa - 1\}$ u.a.r.;
    $p \leftarrow \exp\left(-(f(\boldsymbol{x}_{i+1}) - f(\boldsymbol{x}_i))(\frac{1}{T_i} - \frac{1}{T_{i+1}})\right)$;
    if `random()` $\leq p$ **then**
      $(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}) \leftarrow (\boldsymbol{x}_{i+1}, \boldsymbol{x}_i)$;
until *stop*;

4 Runtime Analysis

We now analyze the runtime of the Metropolis algorithm and parallel tempering with asymmetric exchanges on RUGGEDLANDSCAPE. We assume the initial state is selected uniformly at random from $\{0,1\}^n$, and the periodic exchange step occurs with constant probability $0 < q < 1$. Given a temperature $T > 0$, we use the notation $\beta = \beta(T) = 1/T$ to denote the inverse temperature.

Algorithm 3: Parallel Tempering (Asymmetric Exchange)

Input: κ states $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\kappa$
Input: κ temperatures $0 < T_1 < \cdots < T_\kappa$
Input: migration probability q
repeat
  `/* Perform Metropolis updates */`
  for $i \in \{1, \ldots, \kappa\}$ **do**
    $\boldsymbol{y} \leftarrow$ `neighbor(`$\boldsymbol{x}_i$`)`;
    $p \leftarrow \exp\left(\frac{-(f(\boldsymbol{y}) - f(\boldsymbol{x}_i))}{T_i}\right)$;
    if `random()` $\leq p$ **then** $\boldsymbol{x}_i \leftarrow \boldsymbol{y}$;
  `/* Asymmetric pair exchange */`
  if `random()` $\leq q$ **then**
    Choose $i \in \{1, \ldots, \kappa - 1\}$ u.a.r.;
    $(\hat{\boldsymbol{x}}_{\boldsymbol{i}}, \hat{\boldsymbol{x}}_{\boldsymbol{i+1}}) \leftarrow (\boldsymbol{x}_{\boldsymbol{i}}, \boldsymbol{x}_{\boldsymbol{i+1}})$;
    $p_i \leftarrow \exp\left(\frac{-(f(\boldsymbol{x}_{i+1}) - f(\boldsymbol{x}_i))}{T_i}\right)$;
    $p_{i+1} \leftarrow \exp\left(\frac{-(f(\boldsymbol{x}_i) - f(\boldsymbol{x}_{i+1}))}{T_{i+1}}\right)$;
    if `random()` $\leq p_i$ **then** $\hat{\boldsymbol{x}}_i \leftarrow \boldsymbol{x}_{i+1}$;
    if `random()` $\leq p_{i+1}$ **then** $\hat{\boldsymbol{x}}_{i+1} \leftarrow \boldsymbol{x}_i$;
    $(\boldsymbol{x}_{\boldsymbol{i}}, \boldsymbol{x}_{\boldsymbol{i+1}}) \leftarrow (\hat{\boldsymbol{x}}_{\boldsymbol{i}}, \hat{\boldsymbol{x}}_{\boldsymbol{i+1}})$;
until *stop*;

4.1 Metropolis Algorithm

Theorem 3. *Consider the Metropolis algorithm (Algorithm 1) optimizing* RUGGEDLANDSCAPE *starting from a random initial state with inverse temperature $\beta \geq \frac{c}{\delta_L} \ln n$ for any constant $c > 0$. With probability $1 - e^{-\Omega(n)}$, the algorithm does not find the optimum before $n^{c+1}/(2e)$ steps. Furthermore, when $\beta = \omega(\log n)$, the algorithm requires $n^{\omega(1)}$ steps with probability $1 - e^{-\Omega(n)}$.*

Proof. During the run of the algorithm, any metastable pair must be transformed into an unstable pair at least once before it can become a stable pair. The probability that a metastable pair becomes unstable in any iteration is $2e^{-\beta\delta_L}/n$, since the pair is changed with probability $2/n$ and the corresponding energy increase is accepted with probability $e^{-\beta\delta_L}$.

Let $\mathcal{E}$ be the event that the random initial state has at least $n/16$ metastable pairs. By a Chernoff bound [Mitzenmacher and Upfal, 2017, Theorem 4.5], $\Pr(\mathcal{E}) \geq 1 - e^{-n/64}$. Conditioning on this event, the optimum is not found until at least $n/16$ metastable pairs are removed. Applying a union bound, for any $t > 0$, the probability that a metastable pair becomes unstable before t iterations is at most $2te^{-\beta\delta_L}/n$. Thus with probability at most $(2te^{-\beta\delta_L})^{n/16}$, all of the metastable states have transitioned at least once in t iterations, which is a necessary condition for hitting the optimum. For $t < n^{c+1}/(2e)$ this is at most $\exp\left(-\frac{n}{16}\left(\beta\delta_L - c\ln n + 1\right)\right) \leq e^{-n/16}$ when $\beta \geq \frac{c}{\delta_L} \ln n$. When $\beta = \omega(\log n)$, for any $t < \frac{1}{2}\left(n^{\frac{\beta\delta_L}{2\ln n}}\right) = n^{\omega(1)}$, this probability is at most $e^{-\beta\delta_L n/32} = e^{-\Omega(n)}$.

In either case, by the law of total probability, all of the metastable pairs have transitioned within the first t iterations with probability at most $e^{-\Omega(n)} + (1 - \Pr(\mathcal{E})) = e^{-\Omega(n)}$,

which proves the claim. □

The following theorem shows that high temperatures lead to chaotic behavior and exponential optimization times.

Theorem 4. *Consider the Metropolis algorithm (Algorithm 1) optimizing* RUGGEDLANDSCAPE *starting from a random initial state with inverse temperature $\beta \leq c/(\delta_G - \delta_L)$ for an arbitrary constant $c > 0$. The number of steps until the algorithm finds the optimum is at least 2^{dn} with probability $1 - 2^{-\Omega(n)}$ for some constant $d > 0$ that may depend on c.*

Proof. Theorem 2 will be applied to the count of bit pairs that are not in a stable state when this count is in the interval $[0, \alpha n]$ for a small constant $0 < \alpha < 3/4$ to be determined later. Note that the optimum is only found when the number of bit pairs that are not stable has reached 0. Again applying a Chernoff bound, the Metropolis algorithm starts with at least αn non-optimal bit pairs with probability $1 - 2^{-\Omega(n)}$. We assume in the following that this happens; the claimed probability bound then follows from a union bound of the failure probability $2^{-\Omega(n)}$ and a failure probability $2^{-\Omega(n)}$ which will result from an application of the drift theorem.

Note that every unstable pair (i.e., of the form $(0,1)$ or $(1,0)$) will eventually transition into a metastability $(0,0)$ or a stability $(1,1)$, and this transition will always happen with probability one since unstable pairs contribute the highest energy. Conditioning on this transition, the probability of reaching a metastability is $1/2$ and the probability of reaching a stability is $1/2$.

We consider a modified process to simplify the analysis: we assume that every unstable pair immediately transitions to a metastability or a stability, with the aforementioned conditional probabilities. One way to envision this is that for a particular pair, the random decision about its next transition is made in advance and stored. When the time comes that an unstable pair is mutated, we use the stored transition. Since the algorithm flips only one bit at a time, all pairs can be considered independently, which justifies this early-decision mechanism. Intuitively, this speeds up a transition that would be made in the future. As a consequence, in the modified process, all bit pairs are always metastabilities or stabilities. Since we eliminate the time a bit pair spends in the unstable state, we claim that the optimization time of the modified process is stochastically dominated by that of the original process (though we do not give a formal proof).

Fix one bit pair. If that bit pair is in a metastable state, the probability that it will mutate into a stable state in one step is $(1/n)e^{-\beta\delta_L}$ (since the original process moves to an unstable state with probability $(2/n)e^{-\beta\delta_L}$ and subsequently moves to a stable state with probability $1/2$). Similarly, a pair in a stable state will transition to a metastable state with probability $(1/n)e^{-\beta\delta_G}$.

Consider the stochastic process $(X_t)_{t\in\mathbb{N}}$ that counts the number of pairs that are not in a stable state at time t and denote as $\mathcal{E}_t$ the event $\{X_t \leq \alpha n\}$. The self-loop probability $p_{k,k}$ in the statement of Theorem 2 is the probability that the state count remains unchanged in a single iteration. Thus we have $\Pr(X_{t+1} \neq X_t \mid X_t, \mathcal{E}_t) = 1 - p_{k,k}$, and we bound this as follows.

$$\begin{aligned} 1 - p_{k,k} &= \frac{X_t}{n} \cdot e^{-\beta\delta_L} + \frac{n/2 - X_t}{n} \cdot e^{-\beta\delta_G} \\ &< e^{-\beta\delta_L}/2, \end{aligned} \tag{7}$$

since $\delta_G > \delta_L$. The drift of the X_t process below αn can be bounded as

$$\begin{aligned} \mathrm{E}\left[X_t - X_{t+1} \mid X_t, \mathcal{E}_t\right] &= \frac{X_t}{n} \cdot e^{-\beta\delta_L} - \frac{n/2 - X_t}{n} e^{-\beta\delta_G} \\ &\leq \frac{X_t}{n} e^{-\beta\delta_L} - \left(\frac{1}{2} - \frac{X_t}{n}\right) e^{-\beta\delta_L - c} \\ &= e^{-\beta\delta_L}\left(\frac{X_t}{n}\left(1 + e^{-c}\right) - \frac{e^{-c}}{2}\right) \\ &\leq e^{-\beta\delta_L}\left(\alpha\left(1 + e^{-c}\right) - \frac{e^{-c}}{2}\right). \end{aligned}$$

Choose α small enough so that $2\left(\alpha(1 + e^{-c}) - e^{-c}/2\right) < 0$. By (7), we have $-e^{-\beta\delta_L} < -2(1 - p_{k,k})$, so

$$\mathrm{E}\left[X_t - X_{t+1} \mid X_t, \mathcal{E}_t\right] < 2(1 - p_{k,k}) \cdot \left(\alpha(1 + e^{-c}) - \frac{e^{-c}}{2}\right),$$

satisfying the first condition (2) of Theorem 2 for $\varepsilon := -2\left(\alpha(1 + e^{-c}) - e^{-c}/2\right)$. The second condition (3) follows trivially since the number of non-optimal components only changes by at most 1 in any step. Invoking the negative drift theorem implies the claim. □

We now show that the low temperature and high temperature regime can overlap, and the performance of the Metropolis algorithm can degrade significantly as a function of energy barrier parameters in RUGGEDLANDSCAPE.

Corollary 1. *Let $\delta_G = \delta_L + \ln^{-1} n$. For any temperature setting, the Metropolis algorithm requires $\Omega(n^{\delta_L + 1})$ steps to optimize* RUGGEDLANDSCAPE *with probability exponentially close to one.*

Proof. For all $\beta \leq \ln n = 1/(\delta_G - \delta_L)$, Theorem 4 with $c = 1$ yields an exponential optimization time for Algorithm 1 with probability $1 - 2^{-\Omega(n)}$. For all $\beta \geq \ln n$, Theorem 3 with $c = \delta_L$ yields an optimization time of $\Omega(n^{\delta_L + 1})$ with probability $1 - e^{-\Omega(n)}$. □

4.2 Parallel Tempering

In this section we show that parallel tempering can be highly effective, and already two parallel processes suffice to demonstrate an advantage over a single-temperature process. Our setting employs one process at a low temperature $1/\beta_1$ and one at a high temperature $1/\beta_2$. The low-temperature process ensures stability, as it is unlikely to accept worsenings, whereas the high-temperature process facilitates exploration by traversing energy barriers. We show that improvements are discovered through exploration in the high-temperature process and transferred to the low-temperature process, where they are permanently retained. This guarantees global monotonicity.

Theorem 5. *Consider a run of a $\kappa = 2$ state parallel tempering process with asymmetric exchanges (Algorithm 3) and a constant migration probability $q \in (0,1)$ on the quadratic* RUGGEDLANDSCAPE *function with any inverse temperatures $\beta_1 \geq (2+\varepsilon)\ln n$ and $\beta_2 = O(1)$ for any constant $\varepsilon > 0$. The algorithm optimizes the function in $O\big(n^2 \log n\big)$ steps with probability $1 - o(1)$.*

Proof. We consider the parallel stochastic processes $(X_t^{(1)}, X_t^{(2)})_{t\in\mathbb{N}}$ on $\mathbb{R}$ that correspond to the value of $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$ in the t-th step, respectively. We begin by arguing that for all times $t < n^{2+\varepsilon/2}$, $X_t^{(1)} \leq X_{t-1}^{(1)}$ with probability $1 - e^{-\Omega(\beta_1)}$. At any time step t, $X_t^{(1)} > X_{t-1}^{(1)}$ only if a higher energy state was accepted in line 4 of Algorithm 3, or if a higher energy state was swapped in from $\boldsymbol{x}_2$ in line 10 (or both). This happens with probability at most $2e^{-\beta_1\delta_L}$. Taking a union bound, the probability that $X_t^{(1)}$ does not increase during any of the first $n^{2+\varepsilon/2}$ steps is at least $1 - 2n^{2+\varepsilon/2}e^{-\beta_1\delta_L} = 1 - e^{-\Omega(\beta_1)}$ since we have assumed $\beta_1 \geq (2+\varepsilon)\ln n$. For the remainder of the proof, we condition on the event that the energy does not increase at state $\boldsymbol{x}_1$ before $n^{2+\varepsilon/2} = \omega(n^2 \log n)$ steps.

We now argue that $\Pr(X_{t+2}^{(1)} < X_t^{(1)}) \geq \frac{2q^2(1-q)e^{-\beta_2\delta_L}}{n^2}$. We make a case distinction on the relative values of $X_t^{(1)}$ and $X_t^{(2)}$. If $X_t^{(2)} < X_t^{(1)}$, a sufficient condition for $X_{t+1}^{(1)} < X_t^{(1)}$ is that the lower energy state from $\boldsymbol{x}_2$ was swapped in to $\boldsymbol{x}_1$ at line 10. This occurs with probability q, and since we assume that the energy does not increase in state $\boldsymbol{x}_1$, the new lower energy is preserved, and the claim also holds for $X_{t+2}^{(1)}$.

The case $X_t^{(2)} \geq X_t^{(1)}$ is slightly more involved. A sufficient condition to decrease the energy at $\boldsymbol{x}_1$ in three steps is for exactly one metastable or unstable pair in $\boldsymbol{x}_t$ to transform into a stable pair. This requires relying on the higher temperature process $\boldsymbol{x}_2$ to traverse an energy barrier. Fix an index pair π that corresponds to a pair that is not stable at time t in $\boldsymbol{x}_1$. If π is unstable, then it becomes stable at time $t+1$ with probability $(1-q)/n$, which results in $X_{t+1}^{(1)} < X_t^{(1)}$ and the claim holds as before.

Otherwise, assume that π is a metastable pair. In order for π to become stable, it suffices that three consecutive events occur. Let $\mathcal{E}_1$ denote the event that $\boldsymbol{x}_1$ is swapped into $\boldsymbol{x}_2$ at line 11 at time t. Let $\mathcal{E}_2$ denote the event that π transitions from metastable to unstable at line 4 in $\boldsymbol{x}_2$ at time $t+1$, and is not overwritten by an exchange during that iteration. Finally, let $\mathcal{E}_3$ denote the event that π transitions from unstable to stable during iteration $t+2$ and is subsequently copied back to $\boldsymbol{x}_1$ by an exchange (or that the energy at $\boldsymbol{x}_1$ has already decreased, in which case we are done).

It remains to bound the probability of this event sequence. Since $X_t^{(2)} \geq X_t^{(t)}$, we have $p_2 \geq 1$ at line 11, and it suffices that the exchange occurs, so we have $\Pr(\mathcal{E}_1) = q$. For $\mathcal{E}_2$, the Metropolis update at time $t+1$ for $\boldsymbol{x}_2$ must flip one of the zero bits in π to a one with probability $2/n$, after which the resulting state must be accepted, which happens with probability $e^{-\beta_2\delta_L}$. This update is not overwritten by a subsequent exchange during step $t+1$ with probability at least $1-q$. Thus we have $\Pr(\mathcal{E}_2 \mid \mathcal{E}_1) \geq (1-q)2e^{-\beta_2\delta_L}/n$. For $\mathcal{E}_3$, the probability that π is transformed into a stable pair in the update step is $1/n$, and this proposal results in a strict energy decrease, which is accepted with probability 1. The resulting state is copied back to $\boldsymbol{x}_1$ in the exchange step of iteration t with probability q, yielding $\Pr(\mathcal{E}_3 \mid \mathcal{E}_2 \cap \mathcal{E}_1) \geq q/n$. Event $\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3$ is sufficient for $\{X_{t+2}^{(1)} < X_t^{(1)}\}$ so we have $\Pr(X_{t+2}^{(1)} < X_t^{(1)}) \geq \frac{2q^2(1-q)e^{-\beta_2\delta_L}}{n^2}$ as claimed.

To complete the proof, consider the stochastic process $(Y_t)_{t\in\mathbb{N}}$ where Y_t counts the pairs in $\boldsymbol{x}_1$ that are not stable at time t. Since the preceding arguments hold independently for all such pairs, it follows that

$$\mathrm{E}\left[Y_t - Y_{t+2} \mid Y_t\right] \geq Y_t \left(\frac{2q^2(1-q)e^{-\beta_2\delta_L}}{n^2}\right). \quad (8)$$

We are interested in the first hitting time of $(Y_t)_{t\in\mathbb{N}}$ to zero, $T := \inf\{t \geq 0 \mid Y_t = 0\}$, which corresponds to the first time that $f(\boldsymbol{x}_1) = 0$, i.e., the optimal solution. Clearly $Y_0 \leq n/2$ and Theorem 1 can be adapted to supply a tail bound on T by setting s_0 and $s_{\min}$ accordingly. Note that the timestep lag in the drift condition in (8) only slows the drift by a factor of two. We therefore conclude that for all $r > 0$ the probability that the process requires more than $\frac{n^2}{q^2(1-q)}e^{\beta_2\delta_L}\left(\ln(n/2) + r\right)$ steps until the process hits the optimal solution is at most e^{-r}. Since this outcome is conditioned on the event that the energy of $\boldsymbol{x}_1$ never increases during a polynomial number of steps, the final tail bound is $1 - e^{-\Omega(\beta_1)} - e^{-r}$. As $\beta_1 \geq (2+\varepsilon)\ln n$, setting $r = (2+\varepsilon)\ln n$ yields the claimed result. □

5 Experiments

To complement our theoretical results, we report on the results of a selection of experimental studies. In particular, we are interested in the behavior of the algorithms on RUGGEDLANDSCAPE $_{n,\delta_L,\delta_G}$ for concrete parameter settings. We fix the problem size at $n = 100$. Our proofs show that when δ_G is only marginally larger than δ_L, the Metropolis algorithm fails to distinguish metastable and stable states; thus we set $\delta_L = 100$ and $\delta_G = 105$. We consider a sweep of 5000 temperature values $T \in \{0.01, 0.02, \ldots, 50.00\}$ and run the Metropolis algorithm 100 times with the corresponding temperature setting for 10^6 iterations each. The mean final energy discovered at each temperature value is plotted in Figure 1. The minimum energy found over all 500000 runs was 35 (7 metastable states) discovered in two individual runs at temperatures 12.87 and 18.32. In general, runs with an energy less than 50 occurred only for $T \in [10, 30]$.

To study a typical Metropolis run at a promising temperature, we track the count of metastable, unstable and stable pairs at temperature $T = 12$ in Figure 2. At the beginning of the run, each pair-type occupies roughly a $1/4$ fraction of the total count of $n/2 = 50$ pairs, as expected from a uniform random initial state. In the first 100 iterations, the two unstable pair-types $(0,1)$ and $(1,0)$ quickly vanish. Subsequently, the count of metastable and stable pairs transition into an equilibrium around $2/5$ and $3/5$, respectively.

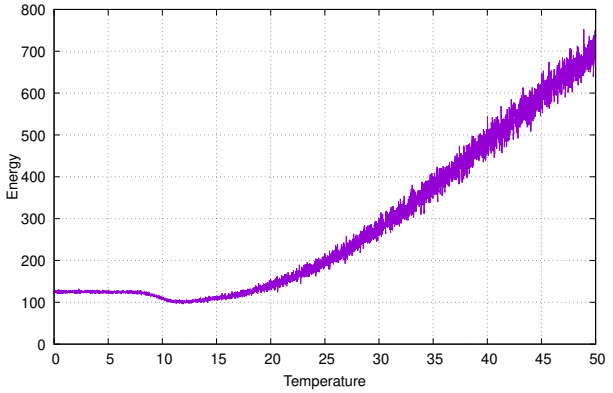

Figure 1: Metropolis energy at 10^6 iterations for different temperatures on RUGGEDLANDSCAPE.

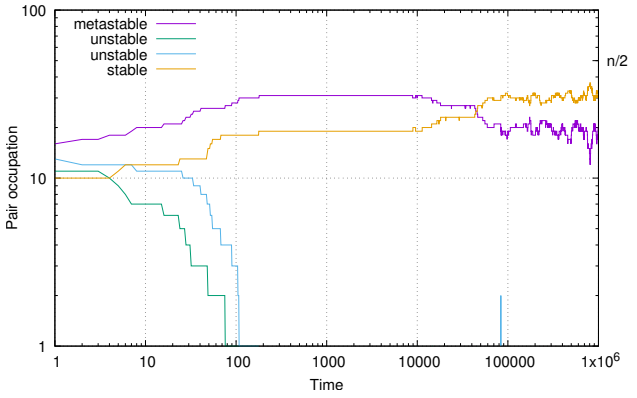

Figure 2: Pair occupation for Metropolis at temperature $T = 12$.

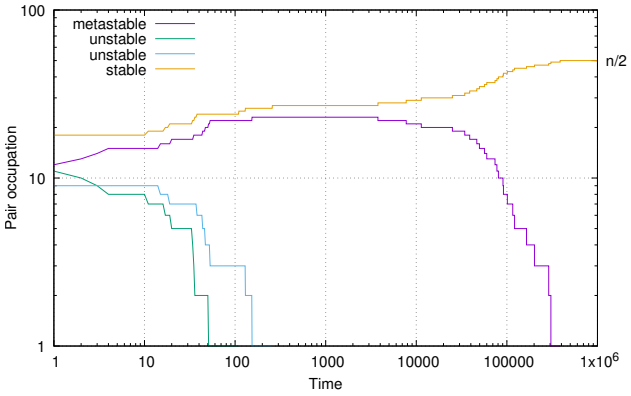

Figure 3: Pair occupation for two-state parallel tempering at $T_1 = 0.001, T_2 = 100$.

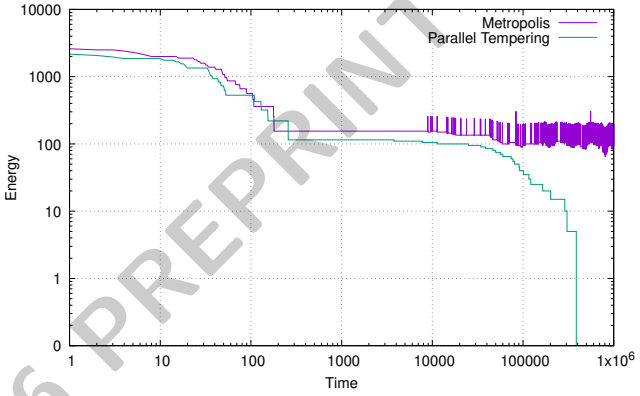

Figure 4: Energy over time for both algorithms on RUGGEDLANDSCAPE.

We repeat this for a two-state parallel tempering process with temperatures $T_1 = 1/1000$ and $T_2 = 100$ and migration probability $q = 0.1$ and plot the results in Figure 3. We observe similar behavior early in the process, whereas the number of metastable pairs vanish steadily later in the process. Finally, we compare the energy as a function of time step for both runs in Figure 4. We observe that the Metropolis algorithm stagnates at energy values around 100, whereas parallel tempering continues to make progress and eventually finds the global optimum.

6 Conclusions

We gave the first proof that parallel tempering can be more effective than a single instance of the Metropolis algorithm in the context of optimization. Our proposed QUBO class RUGGEDLANDSCAPE is highly rugged as it contains an exponential number of local optima. We showed that a single-temperature Metropolis algorithm fails if the temperature is too small or too large. In the former case, the algorithm is unable to escape from local optima, and in the latter case, the search dynamics are too chaotic to approach the global optimum. We furthermore showed that the temperature regimes of these two failure modes can overlap, precluding any ideal temperature setting that admits efficient behavior.

In contrast, we proved that a parallel tempering algorithm with one low-temperature process and one high-temperature process is effective on RUGGEDLANDSCAPE$_{n,\delta_L,\delta_G}$. The low-temperature process typically retains the best solution found so far. Improvements can be discovered by migrating this solution to the high-temperature process, which can traverse an energy barrier and find improvements that are migrated back to the low-temperature system. Hence, the interplay between these parallel processes of different temperatures is crucial for optimization. Our work lends credence to the algorithmic design of parallel tempering and takes a first step towards establishing a rigorous theoretical foundation that explains when and why parallel tempering is efficient, and how it can be used most effectively.

We also introduced a minor modification to the algorithm in which exchanges are allowed to be asymmetric. This adjustment may be particularly useful in cases where the algorithm is applied to solving optimization problems, as it allows the process a reasonable chance to maintain promising candidate solutions without the risk of being overwritten during an exchange. Two natural future directions are to extend the analysis to many parallel states, and to investigate other energy landscapes, such as those induced by combinatorial optimization or multi-objective problems.

Acknowledgments

This research was funded by NSF grant 2144080.

References

[Alba *et al.*, 2013] Enrique Alba, Gabriel Luque, and Sergio Nesmachnow. Parallel metaheuristics: recent advances and new trends. *International Transactions in Operational Research*, 20(1):1–48, 2013.

[Almeida *et al.*, 2025] André Luís Barroso Almeida, Joubert de Castro Lima, and Marco Antonio Moreira Carvalho. Revisiting the parallel tempering algorithm: High-performance computing and applications in operations research. *Computers & Operations Research*, 178:107000, June 2025.

[Aramon *et al.*, 2019] Maliheh Aramon, Gili Rosenberg, Elisabetta Valiante, Toshiyuki Miyazawa, Hirotaka Tamura, and Helmut G. Katzgraber. Physics-Inspired Optimization for Quadratic Unconstrained Problems Using a Digital Annealer. *Frontiers in Physics*, 7, April 2019.

[Černý, 1985] Vladimir Černý. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51, January 1985.

[Doerr and Goldberg, 2013] Benjamin Doerr and Leslie Ann Goldberg. Adaptive drift analysis. *Algorithmica*, 65:224–250, 2013.

[Doerr *et al.*, 2023] Benjamin Doerr, Taha El Ghazi El Houssaini, Amirhossein Rajabi, and Carsten Witt. How well does the Metropolis algorithm cope with local optima? In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '23, pages 1000–1008, New York, NY, USA, July 2023. Association for Computing Machinery.

[Geyer, 1991] Charles J. Geyer. Markov chain Monte Carlo maximum likelihood. In Elaine M. Keramidas and Selma M. Kaufman, editors, *Proceedings on the Twenty-Third Symposium on the Interface of Computing Science and Statistics*, pages 156–163, 1991.

[Habeck *et al.*, 2005] Michael Habeck, Michael Nilges, and Wolfgang Rieping. Replica-exchange Monte Carlo scheme for Bayesian data analysis. *Physical Review Letters*, 94(1), January 2005.

[Hansmann, 1997] Ulrich H. E. Hansmann. Parallel tempering algorithm for conformational studies of biological molecules. *Chemical Physics Letters*, 281(1):140–150, December 1997.

[Hastings, 1970] Wilfred K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[Jansen and Wegener, 2007] Thomas Jansen and Ingo Wegener. A comparison of simulated annealing with a simple evolutionary algorithm on pseudo-boolean functions of unitation. *Theoretical Computer Science*, 386(1):73–93, October 2007.

[Jerrum and Sorkin, 1998] Mark Jerrum and Gregory B. Sorkin. The Metropolis algorithm for graph bisection. *Discrete Applied Mathematics*, 82(1):155–175, March 1998.

[Kirkpatrick *et al.*, 1983] Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983.

[Lässig and Sudholt, 2013] Jörg Lässig and Dirk Sudholt. Design and analysis of migration in parallel evolutionary algorithms. *Soft Computing*, 17(7):1121–1144, 2013.

[Lissovoi *et al.*, 2023] Andrei Lissovoi, Pietro S. Oliveto, and John Alasdair Warwicker. When move acceptance selection hyper-heuristics outperform Metropolis and elitist evolutionary algorithms and when not. *Artificial Intelligence*, 314:103804, January 2023.

[Luque and Alba, 2011] Gabriel Luque and Enrique Alba. *Parallel Genetic Algorithms: Theory and Real World Applications*. Studies in Computational Intelligence. Springer, 2011.

[Mambrini *et al.*, 2012] Andrea Mambrini, Dirk Sudholt, and Xin Yao. Homogeneous and heterogeneous island models for the set cover problem. In Carlos A. Coello Coello, Vincenzo Cutello, Kalyanmoy Deb, Stephanie Forrest, Giuseppe Nicosia, and Mario Pavone, editors, *Proceedings of the Twelfth International Conference on Parallel Problem Solving from Nature (PPSN XII)*, pages 11–20, Berlin, Heidelberg, 2012. Springer.

[Marinari and Parisi, 1992] Enzo Marinari and Giorgio Parisi. Simulated tempering: A new Monte Carlo scheme. *Europhysics Letters (EPL)*, 19(6):451–458, July 1992.

[Metropolis *et al.*, 1953] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, June 1953.

[Mitzenmacher and Upfal, 2017] Michael Mitzenmacher and Eli Upfal. *Probability and Computing*. Cambridge University Press, second edition, 2017.

[Oliveto and Witt, 2011] Pietro S. Oliveto and Carsten Witt. Simplified drift analysis for proving lower bounds in evolutionary computation. *Algorithmica*, 59(3):369–386, 2011.

[Oliveto and Witt, 2012] Pietro S. Oliveto and Carsten Witt. Erratum: Simplified drift analysis for proving lower bounds in evolutionary computation. *CoRR*, abs/1211.7184, 2012.

[Oliveto *et al.*, 2018] Pietro S. Oliveto, Tiago Paixão, Jorge Pérez Heredia, Dirk Sudholt, and Barbora Trubenová. How to escape local optima in black box optimisation: When non-elitism outperforms elitism. *Algorithmica*, 80(5):1604–1633, May 2018.

[Poelwijk *et al.*, 2011] Frank J. Poelwijk, Sorin Tănase-Nicola, Daniel J. Kiviet, and Sander J. Tans. Reciprocal sign epistasis is a necessary condition for multi-peaked fitness landscapes. *Journal of Theoretical Biology*, 272(1):141–144, 2011.

[Rowe and Sudholt, 2014] Jonathan E. Rowe and Dirk Sudholt. The choice of the offspring population size in the

(1,λ) evolutionary algorithm. *Theoretical Computer Science*, 545:20–38, 2014.

[Sasaki and Hajek, 1988] Galen H. Sasaki and Bruce Hajek. The time complexity of maximum matching by simulated annealing. *J. ACM*, 35(2):387–403, April 1988.

[Sudholt, 2015] Dirk Sudholt. Parallel evolutionary algorithms. In Janusz Kacprzyk and Witold Pedrycz, editors, *Springer Handbook of Computational Intelligence*, pages 929–959. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[Swendsen and Wang, 1986] Robert H. Swendsen and Jian-Sheng Wang. Replica Monte Carlo simulation of spin-glasses. *Physical Review Letters*, 57(21):2607–2609, November 1986.

[Wang *et al.*, 2024] Shouda Wang, Weijie Zheng, and Benjamin Doerr. Choosing the right algorithm with hints from complexity theory. *Information and Computation*, 296:105125, January 2024.

[Wegener, 2005] Ingo Wegener. Simulated annealing beats Metropolis in combinatorial optimization. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Automata, Languages and Programming*, pages 589–601, Berlin, Heidelberg, 2005. Springer.

[Zheng *et al.*, 2024] Weijie Zheng, Mingfeng Li, Renzhong Deng, and Benjamin Doerr. How to use the Metropolis algorithm for multi-objective optimization? In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence*, volume 38 of *AAAI'24*, pages 20883–20891. AAAI Press, February 2024.

[Zhu *et al.*, 2020] Zheng Zhu, Chao Fang, and Helmut G. Katzgraber. borealis—A generalized global update algorithm for Boolean optimization problems. *Optimization Letters*, 14(8):2495–2514, November 2020.