Effective Inversely Proportional Hypermutations for Unimodal and Multimodal Optimisation

Anonymous authors

***Abstract*—Artificial Immune Systems (AIS) employing hypermutations with linear static mutation potential have recently been shown to be very effective at escaping local optima of combinatorial optimisation problems at the expense of being slower during the exploitation phase compared to standard evolutionary algorithms. In this paper, we prove that considerable speed-ups in the exploitation phase may be achieved with dynamic inversely proportional hypermutations (IPH) with mutation potentials and argue that the potential should decrease inversely to the distance to the optimum rather than to the difference in fitness as in the literature. Afterwards, we define a simple (1+1) Opt-IA that uses IPH and ageing for realistic applications where optimal solutions are unknown. The aim of this AIS is to approximate the ideal behaviour of the IPH better and better as the search space is explored. We prove that such desired behaviour and related speed-ups occur for unimodal functions as well as for well-studied bimodal benchmark functions which are hard to optimise for traditional evolutionary algorithms and for AIS using static mutation potentials.**

***Index Terms*—Artificial immune systems, heavy-tailed mutations, inversely proportional hypermutations, theory, runtime analysis**

I. Introduction

Three recent hot topics in the theory of randomised search heuristics for optimisation in general and in artificial intelligence in particular have been the benefits of mutation operators with higher mutation rates than traditionally employed and recommended [1]–[8], the automatic adaptation of the mutation rate [9]–[13] and the analysis of when non-elitist search heuristics are beneficial compared to elitist ones [14]–[20].

Artificial Immune Systems (AIS) are a class of bio-inspired algorithms which naturally use both high mutation rates via so-called hypermutation operators, often automatically adapt the mutation rate, and use non-elitism implicitly through the use of ageing operators, characteristics that are typical in the immune system of vertebrates from which AIS draw inspiration. In this paper we analyse the performance of an adaptive mutation operator called Inversely Proportional Hypermutation (IPH) that aims to increase the mutation rate with (an estimate of) the distance from the optimum, and propose a considerably improved one for efficient unimodal and multimodal optimisation.

AIS for optimisation [21], [22] are generally inspired by the clonal selection principle [23]. For this reason they are also often referred to as *clonal selection* algorithms [21]. In the literature two key features of clonal selection algorithms have been identified [24]:

1) The proliferation rate of each immune cell is proportional to its affinity with the selective antigen: the higher the affinity, the higher the number of offspring generated (clonal selection and expansion).
2) The mutation suffered by each immune cell during reproduction is inversely proportional to the affinity of the cell receptor with the antigen: the higher the affinity, the smaller the mutation (affinity maturation).

Indeed well-known clonal selection algorithms employ mutation operators applied to immune cells (i.e., candidate solutions) with a rate that decreases with their similarity to the antigen (i.e., global optima) during affinity maturation. Often such operators are referred to as *inversely proportional hypermutations* (IPH). Popular examples of such clonal selection algorithms are Clonalg [25] and Opt-IA [26].

The ideal behaviour of the IPH operator is that the mutation rate is minimal in proximity of the global optimum and increases as the difference between the fitness of the global optimum and that of the candidate solution increases. However, achieving such behaviour in practice may be problematic because the fitness of the global optimum is usually unknown. As a result, in practical applications information about the problem is used to identify bounds on (or estimates of) the value of the global optimum[1]. Thus, the closer is the estimate to the actual value of the global optimum, the closer should the behaviour of the IPH operator be to the desired one. On the other hand, if the bound is much higher (e.g., for a maximisation problem) than the true value, then there is a risk that the mutation rate is too high in proximity of the global optimum i.e., the algorithm will struggle to identify the optimum.

Previous theoretical analyses, though, have highlighted various problems with IPH operators from the literature even when the fitness value of the global optimum is known. Zarges analysed the effects of mutating candidate solutions with a rate that is inversely proportional to their fitness for the OneMax problem [27]. She considered two different rates for the decrease of the mutation rate as the fitness increases: a linear decay (i.e., each bit flips with probability $\frac{\text{OneMax(x)}}{\text{Opt}}$ where Opt is the optimum value) and an exponential decay (i.e., each bit flips with probability $e^{-\rho\frac{\text{OneMax(x)}}{\text{Opt}}}$ where ρ is called the *decay* parameter). The motivation behind these choices are that the former operator flips in expectation exactly the number of bits that maximises the probability of reaching the optimum in the next step while the latter is the operator used in the Clonalg algorithm. She showed that if the optimum of OneMax is known, then an algorithm employing such a

[1] Alternatively, the fitness of the best candidate solution is sometimes used and the mutation rate of the rest of the population is inversely proportional to the best.

mutation operator will require exponential time to optimise ONEMAX with overwhelming probability (w.o.p.) in both cases of linear or exponential decays of the mutation rate. The reason is that the initial random solutions that have roughly half the fitness (i.e., $n/2$) of the global optimum (i.e., n) have very high mutation rates. Such rates do not allow the algorithm to make any progress with any reasonable probability even if the decay parameter ρ is chosen very carefully (i.e., $\rho = \ln n$ leads to reasonable mutation rates between $1/n$ and $1/2$ during the optimisation of ONEMAX). Hence, the behaviour of the mutation operator leads to very inefficient performance even for the simple OneMax function when the optimum is assumed to be known.

On the other hand, the Opt-IA clonal selection algorithm, that also uses very high mutation rates (called hypermutations with mutation potential), has been proven to be efficient by employing a selection strategy called *stop at first constructive mutation* (FCM). The strategy evaluates fitness after each bit is flipped and interrupts the hypermutation immediately if an improvement is detected. Indeed, the operator using a static mutation potential (i.e., a linear number of bits are flipped unless an improvement is found along the way) has been proven to be very effective at escaping local optima of standard multimodal benchmark functions [28] and at finding arbitrarily good approximations for the Number Partitioning NP-Hard problem [29], [30] at the expense of being slower at hillclimbing during the exploitation phases (it is up to a linear factor slower than the standard bit mutation (SBM) used by evolutionary algorithms for easy unimodal functions such as OneMax and LeadingOnes). Hence, differently from other clonal selection algorithms (such as Clonalg and Opt-IA without FCM), hypermutations with mutation potential coupled with FCM can cope with the desired behaviour of inversely proportional mutation rates by being capable of making progress efficiently with high mutation rates.

In this paper we first consider whether Opt-IA may become faster during the exploitation phases if inversely proportional mutation potentials are applied rather than static ones. The reason to believe this is that the hypermutations waste many fitness function evaluations during exploitation because, as the optimum is approached, the probability of flipping bits that lead to an improvement decreases. Hence, with high probability the operator flips wrong bits at the beginning of a hypermutation and ends up wasting a linear number of fitness function evaluations in each hypermutation. On the other hand, if the mutation potential decreases as the optimum is approached, then the amount of wasted evaluations should decrease accordingly.

A previous runtime analysis for OneMax of the inversely proportional hypermutations with mutation potential used by Opt-IA has shown that the mutation rate is always in the range $[(c/2)n, cn]$ where $M = cn$ is the highest mutation potential [31]. Hence it does not decrease proportionally with the decrease in distance to the optimum as desired and, as a result, no asymptotic speed-ups are achieved compared to static mutation potentials.

In this paper we first show that considerable speed-ups in exploitation (hillclimbing) phases may be achieved compared to static hypermutations, if the mutation rates decrease appropriately with either the fitness or the distance to the optimum. To show this we analyse the IPH operators for standard unimodal benchmark functions where the performance of static mutation potentials is well-understood [28]. We first identify what speed-ups may be hoped for by analysing the operators in the ideal situation where the location of the optimum is known. A result of this first analysis is that a mutation potential that increases exponentially with the Hamming distance to the optimum is the most promising out of three considered inverse potentials since it provides the larger speed-ups. Furthermore, using the Hamming distance rather than the fitness as the measure to quantify proximity to the optimum makes the operator robust to fitness function scaling. Hence we consider this operator, called M_{expoHD}, in the rest of the paper.

Afterwards we propose a clonal selection algorithm that we call (1+1) Opt-IA that uses IPH and ageing[2] [26] to be applied in practical unimodal and multimodal applications where the optimum is not known. The algorithm uses the best solution it has encountered to estimate the mutation rates. In the literature it has been shown that ageing allows algorithms to escape from local optima either by identifying a new slope of increasing fitness or by completely restarting the optimisation process if it cannot escape [28], [30]–[33]. The idea behind our proposed AIS is that the more of the search space that is explored, the better the ideal behaviour of the IPH is approximated through the discovery of better and better local optima.

Our analysis reveals that such a strategy does not produce the desired effect, and a further improvement to the algorithm is required. Since the mutation rate decreases with the distance to the best found local optimum, the algorithm may encounter difficulties at identifying new promising optima. In particular, if the algorithm identifies some slope that leads away from the previous local optimum, then the mutation rate will increase as the new optimum is approached. Firstly, this makes the new optimum hard to identify. Secondly, the high mutation rates in its proximity lead to high wastage of fitness function evaluations defeating our main motivation of reducing such wastage compared to static mutation potentials. We rigorously prove this effect for the well-studied TWOMAX bimodal benchmark function where the expected runtime does not improve compared to that of the static hypermutations to optimise each slope of the function (i.e., $\Theta(n^2 \log n)$). On the other hand we use the CLIFF benchmark function to show that the IPH can escape local optima when coupled with ageing. We remark that static hypermutations cannot optimise the function efficiently because the operator only stops if a constructive mutation is identified, thus will not return inferior solutions that may survive the ageing operator and allow the algorithm to escape the local optima of the function class.

To fix the issue we define a *Symmetric* IPH operator that decreases the potential with respect to the distance to the best local optimum and uses the same rate of decrease in all other directions. We prove the effectiveness of our strategy, and subsequent speed-ups over static hypermutations, for all the unimodal and multimodal function classes considered in

[2]The well-known Opt-IA AIS uses both operators in combination [26].

TABLE I: Expected runtimes of the different mutation operators when embedded in a (1+1) algorithmic framework for the benchmark functions considered in this paper. '*' indicates that the results are obtained assuming the optimum is known. For all the other results, the optimum in unknown to the algorithm.

(1+1) Algorithms	OneMax	LeadingOnes	TwoMax	Cliff$_d$
EA (SBM)	$\Theta(n \log n)$ [35]	$\Theta(n^2)$ [35]	$O(n^n)$ [36],	$O(n^d)$ [37]
IA$_{\text{static}}$	$\Theta(n^2 \log n)$ [38]	$\Theta(n^3)$ [38]	$\Theta(n^2 \log n)$	$O(\frac{n^{d+1} \cdot e^d}{d^d})$ [38]
IA$^*_{\text{M}_{\text{linHD}}}$	$\Theta(n^2)$	$\Theta(n^3)$	-	-
IA$^*_{\text{M}_{\text{expoF(x)}}}$	$O(n^{\frac{3}{2}} \log n)$, $\Omega(n^{\frac{3}{2}-\epsilon})$	$O(\frac{n^3}{\log n})$, $\Omega(n^{\frac{5}{2}+\epsilon})$	-	-
IA$^*_{\text{M}_{\text{expoHD}}}$	$O(n^{\frac{3}{2}} \log n)$, $\Omega(n^{\frac{3}{2}-\epsilon})$	$O(n^{\frac{5/2+\epsilon}{\ln n}})$, $\Omega(n^{\frac{9}{4}-\epsilon})$	-	-
Opt-IA$_{\text{M}_{\text{expoHD}}}$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$O(n^{\frac{3}{2}} \log n + \tau n^{\frac{1}{2}} + \frac{n^{\frac{7}{2}}}{d^2})$
Opt-IA$_{\text{Sym M}_{\text{expoHD}}}$	$\Theta(n \log n)$	$\Theta(n^2)$	$O(n^{\frac{3}{2}} \log n)$	$O(n^{\frac{3}{2}} \log n + \tau n^{\frac{1}{2}} + \frac{n^{\frac{7}{2}}}{d^2})$

this paper. A summary of the results obtained in the paper is provided in Table II highlighting the speed-ups achieved by the IPH operator compared to the standard bit mutations traditionally used by evolutionary algorithms and to the static hypermutation operator with FCM.

Compared to the original extended abstract [34], this paper has been substantially reorganised, apart for the introduction of two new theorems (Theorems 6 and 7) and the inclusion of all the proofs including those missing in the previous abstract.

II. Inversely Proportional Mutation Potentials

Hypermutations with mutation potential using FCM flip at most a linear amount of bits (i.e., the mutation potential $M = cn$) chosen uniformly at random without replacement. After each of the M bit flips, the fitness of the produced solution is evaluated and if a *constructive mutation* is identified, then the operator is stopped and the solution returned. Otherwise the algorithm continues flipping the bits without setting them back until the potential M of bit flips is reached. If no constructive solution has been identified earlier, then the Mth solution is returned. The original inversely proportional hypermutation operator (IPH) proposed for Opt-IA uses mutation potential $M = \lceil(1 - f_{\text{OPT}}/v)\rceil cn$ for minimisation problems, where f_{OPT} is the best known fitness and v is the fitness of the individual [26], [31]. In an analysis for OneMax, such a mutation potential was shown not to decrease below $cn/2$, with cn being the highest possible mutation potential [31]. As a result no speed-ups at hillclimbing are achieved compared to static hypermutations as the mutation potential is always at least linear. In this section, we introduce three different IPH operators that will be analysed in this paper and that provably speed-up the hillclimbing performance for OneMax. Two have been already considered in the literature while the third one is newly proposed by us based on analysed drawbacks of the first two.

Hamming Distance Based Linear Decrease

Zarges analysed an inversely proportional hypermutation operator where the probability of flipping each bit increases linearly with the Hamming distance to the optimum or its best available estimate [27]. Precisely, this operator flips each bit with probability $H(x, \text{best})/n$ where n is the size of the problem and $H(x, \text{best})$ is the Hamming distance of the current point to the best individual (or an estimate if not available). As the expected number of bit-flips is $H(x, \text{best})$ during each execution of a mutation operator, a mutation potential inspired by this inversely proportional hypermutation is:

$$M_{\text{linHD}}(x) := H(x, \text{best}).$$

Fitness Difference Based Exponential Decrease

In Clonalg's IPH, the mutation probability decreases as an exponential function of the fitness of the current solution [25]. Precisely, each bit flips with probability $e^{-\rho \cdot v}$ where v is the normalised fitness value and ρ is a decay parameter that regulates the speed at which the mutation rate decreases. Since we consider maximisation problems, we use $v = \frac{f(x)}{f(\text{best})}$ as suggested by [27] where $f(\text{best})$ is the best known fitness value. Using the expected number of bits flipped by this mutation operator as a mutation potential gives $M = n \cdot e^{-\rho v}$. According to both experimental and theoretical analyses [27], a reasonable value for ρ is $\ln n$. We call this mutation potential $\text{M}_{\text{expoF(x)}}$ and define it as:

$$M_{\text{expoF(x)}}(x) := \left\lfloor n^{1-\frac{f(x)}{f(\text{best})}} \right\rfloor.$$

Hamming Distance Based Exponential Decrease

It is well understood that fitness-proportional selection is sensitive to the difference between the fitness values of candidate solutions, i.e., it struggles to distinguish between solutions that have similar, yet different, fitness values [39]–[42]. For this reason, nowadays selection operators that rank individuals by fitness (e.g., tournament, linear ranking, comma selection) are generally preferred. We also consider a measure that avoids using the fitness values directly because, apart from the above consideration, the ideal amount of bits to be flipped clearly depends on the genotypic distance of the candidate solution to the optimum rather than on their difference in fitness values. To this end, we suggest a mutation potential which is similar to $M_{\text{expoF(x)}}$ with the exception that it uses the normalised Hamming distance to the best estimate rather than the normalised fitness. We call this mutation potential M_{expoHD} and define it as $M_{\text{expoHD}} = n \cdot e^{-\rho \frac{n - H(x,\text{best})}{n}}$ where n is the problem size (the maximum Hamming distance between any two points), $H(x, \text{best})$ is the Hamming distance to the best known solution and ρ is the decay of the mutation potential. For the choice of $\rho = \ln n$, we get:

$$M_{\text{expoHD}}(x) := \left\lfloor n^{\frac{H(x,\text{best})}{n}} \right\rfloor .$$

III. Performance Gains in Ideal Conditions

In this section, we evaluate the performance of the three different IPH operators, assuming the optimum is known (i.e., best = opt). Under this assumption, the operators exhibit their ideal behaviour (i.e., the mutation potential decreases with the desired rate as the optimum is approached). Our aim is to evaluate what speed-ups can be achieved in ideal conditions compared to the well-studied static mutation potentials. To achieve such comparisons we perform runtime analyses of the (1+1) IA using the IPH on the standard OneMax and LeadingOnes unimodal benchmark functions for which the performance of the same algorithm using static mutation potentials is known [28]. The simple to define OneMax function counts the number of 1-bits in the bit string and is normally used to evaluate the hill climbing ability of algorithms [5], [43]–[45]. On the other hand, LeadingOnes is a slightly harder unimodal problem which counts the consecutive number of 1-bits at the beginning of the bit string before the first 0-bit. Evolutionary algorithms using standard bit mutation (or k-bit flip local search mutation) optimise the functions respectively in $\Theta(n \log n)$ and $\Theta(n^2)$ expected fitness function evaluations [10], [11], [35], [46], [47], while static hypermutations are a linear factor slower [38].

While in practice both evolutionary algorithms and artificial immune systems use a population of solutions, in this paper we consider single trajectory algorithms to better highlight the performance of all the considered hypermutation operators. The pseudo-code of a simple $(1+1)$ IA is given in Algorithm 1. It simply uses one candidate solution in each iteration to which inversely proportional hypermutations are applied. For the function Hypermutate(x), IPH performs two steps: 1) it calculates the mutation potential M, and 2) it creates y by flipping at most M distinct bits of x selected uniformly at random without replacement one after another until a constructive mutation happens. Otherwise it returns the last constructed solution. A constructive mutation is a mutation step where the evaluated bit string is at least as fit as its parent. The results proven in this section and comparisons with static hypermutations and the IPH operators from the literature are summarised in Table II.

Before stating the main results, we first introduce the following lemma (the Ballot theorem) which is used throughout this paper to derive lower bounds on the expected runtime of the algorithms [31], [38]. This theorem is essentially used to show the probability of picking more 1-bits than 0-bits during one hypermutation operation given that there are more 1-bits in the initial bit-string (or vice-versa).

TABLE II: Comparison of the expected runtimes obtained by different hypermutation schemes for OneMax and LeadingOnes. The original IPH operator of Opt-IA [26] has the same expected runtimes as the $(1+1)$ IA using static hypermutations [31].
*: the result can be easily drawn from the analysis for OneMax [27].

$(1+1)$ IA	OneMax	LeadingOnes
Using static hypermutation	$\Theta(n^2 \log n)$ [38]	$\Theta(n^3)$ [38]
Using IPH of Clonalg	$e^{\Omega(n)}$ [27]	$e^{\Omega(n)}$*
Using IPH of Opt-IA	$\Theta(n^2 \log n)$ [31]	$\Theta(n^3)$ [31]
Using IPH with M_{linHD}	$\Theta(n^2)$	$\Theta(n^3)$
Using IPH with $M_{\text{expoF(x)}}$	$O(n^{(3/2)} \log n)$, $\Omega(n^{3/2-2\epsilon})$	$O(n^3/\log n)$, $\Omega(n^{5/2+\epsilon})$
Using IPH with M_{expoHD}	$O(n^{(3/2)} \log n)$, $\Omega(n^{3/2-2\epsilon})$	$O(n^{\frac{5/2+\epsilon}{\ln n}})$, $\Omega(n^{9/4-\epsilon})$

Algorithm 1 $(1+1)$ IA

1: Initialise $x \in \{0,1\}^n$ uniformly at random;
2: evaluate $f(x)$;
3: **while** termination condition is not satisfied **do**
4: $\quad y :=$ Hypermutate(x);
5: $\quad$ **if** $f(y) \geq f(x)$ **then**
6: $\quad\quad x := y$.
7: $\quad$ **end if**
8: **end while**

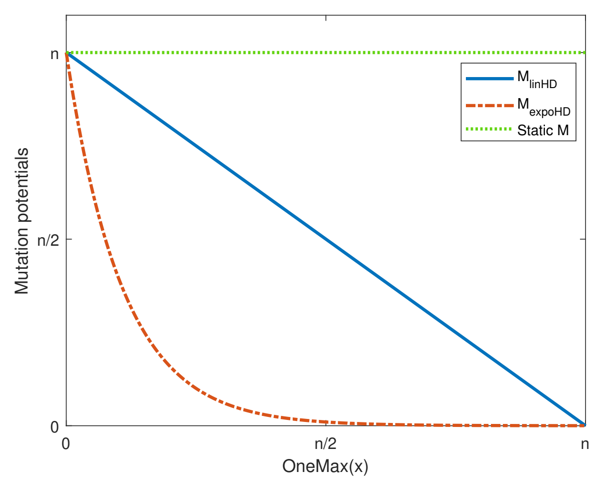

Fig. 1: Static and inversely proportional mutation potentials for OneMax.

Lemma 1 (Ballot Theorem [48]). *In a ballot, suppose that candidate P receives p votes and candidate Q receives q votes such that $p > q$. The probability that throughout the counting P is always ahead of Q is $(p-q)/(p+q)$.*

A. Performance for OneMax in Ideal Conditions

The following theorems derive the expected runtimes using each of the three IPH operators for the function

ONEMAX$(x) := \sum_{i=1}^{n} x_i$. Figure 1 shows how the studied mutation potentials ideally decrease during the run of the algorithm when optimising the function. Notice that for ONEMAX the behaviours of $\mathrm{M}_{\mathrm{expoF(x)}}$ and $\mathrm{M}_{\mathrm{expoHD}}$ are the same. By decreasing the potential linearly with the decrease of the Hamming distance, a logarithmic factor may be shaved off from the expected runtime of the $(1+1)$ IA compared to the expected runtime achieved by the same algorithm using static hypermutation potentials.

Theorem 1. *The $(1+1)$ IA using IPH with M_{linHD} optimises* ONEMAX *in $\Theta(n^2)$ expected fitness function evaluations.*

Proof. Considering i as the number of 0-bits in the candidate solution, the probability of improvement in the first step is i/n. Knowing that at most n improvements are needed to find the optimum and in case of failure, $H(x, \mathrm{opt}) = i$ fitness function evaluations would be wasted, the total expected time to optimise ONEMAX is at most $\sum_{i=1}^{n} \frac{n}{i} \cdot i = O(n^2)$.

For the lower bound, we use the Ballot theorem (Lemma 1). By Chernoff bounds, the number of 0-bits in the initialised solution is at least $n/3$ w.o.p. Considering the number of 0-bits as $i = q$ and the number of 1-bits as $n - i = p$, the probability of an improvement is at most $1-(p-q)/(p+q) = 1-(n-2i)/n = 2i/n$ by the Ballot theorem, where $i = H(x, opt)$. This means that we need to wait at least $n/(2i)$ iterations to see an improvement and each time the mutation operator fails to improve the fitness, i fitness function evaluations will be wasted. Considering that at least $n/3$ improvements are needed, the expected time to optimise ONEMAX is larger than $\sum_{i=1}^{n/3} \frac{n}{2i} \cdot i = \Omega(n^2)$. □

The following theorem shows that a greater speed up may be achieved if the potential decreases exponentially rather than linearly. Note that for ONEMAX the Hamming distance of a solution to the optimum and its difference in fitness are the same. Figure 1 highlights the reason why these operators waste fewer fitness evaluations.

Theorem 2. *The $(1+1)$ IA using IPH with either $M_{expoF(x)}$ or M_{expoHD} optimises* ONEMAX *in $O(n^{3/2} \log n)$ and $\Omega(n^{3/2-\epsilon})$ expected fitness function evaluations for any arbitrarily small constant $\epsilon > 0$.*

Proof. We prove the results for $\mathrm{M}_{\mathrm{expoF(x)}}$ which applies to $\mathrm{M}_{\mathrm{expoHD}}$ as well. To prove the upper bound, we use that by Chernoff bounds the initialised solution has at most $n/2+n^{2/3}$ 0-bits w.o.p.. The probability of improvement in the first mutation step is at least i/n with i being the number of 0-bits. As we need at most $n/2 + n^{2/3}$ improvements and each time the mutation fails to make an improvement at most $n^{1-\frac{n-i}{n}}$ fitness function evaluations are wasted, the total expected time to find the optimum will be $E(T) \leq \sum_{i=1}^{n/2+n^{2/3}} \frac{n}{i} \cdot n^{\frac{i}{n}} = O(n^{3/2} \log n)$ by pessimistically assuming $i = n/2 + n^{2/3}$ in $n^{i/n}$.

To prove the lower bound, we again consider i as the number of 0-bits. By Chernoff bounds, i is at least $n/2 - \epsilon n$ in the initialised bit string for some arbitrarily small constant $\epsilon > 0$. The probability of improvement is at most $2i/n$ by the Ballot theorem, and the number of wasted fitness function evaluations at each failure is $n^{\frac{i}{n}}$. If we consider the time spent between levels $n(1/2 + \epsilon/2)$ and $n(1/2 + \epsilon)$, we get the expected time of $n^{\frac{n/2-\epsilon n}{n}} \cdot \sum_{i=n/2+n\epsilon/2}^{n/2+\epsilon n} \Omega(1) = n^{1/2-\epsilon} \cdot \epsilon n/2 \cdot \Omega(1) = \Omega(n^{3/2-\epsilon})$. □

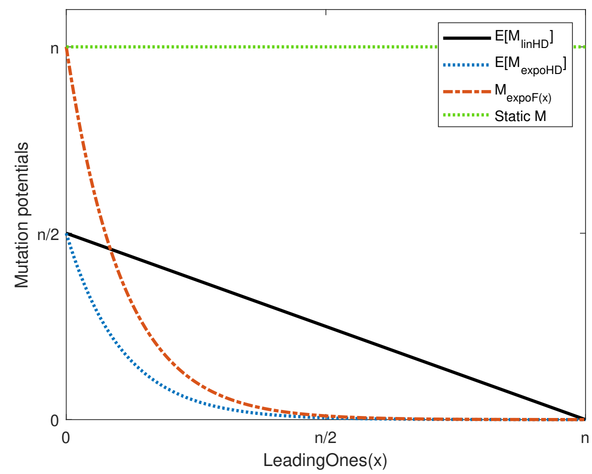

Fig. 2: Static and inversely proportional mutation potentials for LEADINGONES. Expected values are shown for those using Hamming distance.

B. *Performance for* LEADINGONES *in Ideal Conditions*

In the previous subsection, it was shown that decreasing the mutation rate linearly with the Hamming distance to the optimum allows for a logarithmic factor speed-up for ONEMAX compared to using static hypermutations. The following theorem shows that no asymptotic improvement over static mutation potentials is achieved for LEADINGONES:= $\sum_{i=1}^{n} \prod_{j=1}^{i} x_i$. The main reason for the lack of improvement is that a linear mutation potential is sustained until at least a linear number of improvements are achieved.

Theorem 3. *The $(1+1)$ IA using IPH with M_{linHD} optimises* LEADINGONES *in $\Theta(n^3)$ expected fitness function evaluations.*

Proof. The proof for the lower bound follows that of Theorem 3.6 in [28] for the expected runtime of the $(1+1)$ IA using static hypermutation with potential $M := cn$ with the exception that the amount of wasted fitness function evaluations in case of failure at finding an improvement within a hypermutation is now $H(x, \mathrm{opt})$ instead of cn.

Considering i as the number of leading 1-bits, we denote the expected number of fitness function evaluations until an improvement happens by $E(f_i)$. Any such candidate solution has i leading 1-bits with a 0-bit following, and then $n - i - 1$ *trailing bits*. The trailing bits stay uniformly distributed throughout the run since the hypermutation operation can only terminate with an accepted solution by immediately flipping the leftmost 0-bit, which does not alter the trailing bits, or by immediately flipping a random trailing bit, which does not affect the probability of the flipped position being 1 or 0 after it is flipped [35]. We take into account three possible events E_1, E_2 and E_3 that can happen in the first bit flip: E_1 is the event of flipping a leading 1-bit which happens with probability i/n, E_2 is the event of flipping the first 0-bit which happens with probability $1/n$, and E_3 is the event of flipping any other bit which happens with probability $(n-i-1)/n$. So we get $E(f_i|E_1) = H(x, opt) + E(f_i)$, $E(f_i|E_2) = 1$, and $E(f_i|E_3) = 1 + E(f_i)$. Hence, by the law of total expectation, the expected number of fitness function evaluations for every improvement is $E(f_i) = \frac{i}{n}(H(x, opt) + E(f_i)) + \frac{1}{n} \cdot 1 + \frac{n-i-1}{n}(1 + E(f_i))$. Solving the equation for $E(f_i)$ gives us $E(f_i) = i \cdot H(x, opt) + n - i \geq i \cdot H(x, opt)$. Since the bits after the leftmost 0-bit are distributed uniformly at random [35], the value of $H(x, opt)$ is not less than $1 + \frac{(n-i-1)}{2} - \frac{\epsilon}{2}n$ w.o.p. by Chernoff bounds. Thus, we conclude $E(f_i) \geq \frac{(1-\epsilon)in - i^2 + i}{2}$. The right hand side of the inequality has a second derivative of -2 with respect to i, and therefore it is concave down. For the interval $i \in \{n/2 - \epsilon^* n, \ldots, n/2 + \epsilon^* n\}$ its value is in the order of $\Omega(n^2)$ for some arbitrarily small $\epsilon^* > 0$.

As we are computing the lower bound, we have to take into account the probability of skipping any level i. To do this, we need to calculate the expected number of consecutive 1-bits that follow the leftmost 0-bit (called free riders [35]). The expected number of free riders is $\sum_{i=1}^{n-i-1} i \cdot 1/2^{i+1} < \frac{1}{2} \cdot \sum_{i=0}^{\infty} i \cdot (1/2)^i = \frac{1}{2} \cdot \frac{1/2}{(1-1/2)^2} = 1$. This means that the probability of not skipping a level i is $\Omega(1)$. On the other hand, with probability at least $1 - 2^{-n/3}$, the initial solution does not have more than $n/3$ leading 1-bits. Thus, we obtain a lower bound of $(1 - 2^{-n/3}) \sum_{i=n/2-\epsilon^* n}^{n/2+\epsilon^* n} \Omega(1) \cdot E(f_i) = \Omega(1) \sum_{i=n/2-\epsilon^* n}^{n/2+\epsilon^* n} \frac{in - i^2 + i - i\epsilon n}{2} = \Omega(n^3)$ on the expected number of fitness function evaluations.

Now we prove the upper bound. The probability of improvement in each step is at least $1/n$ which is the probability of flipping the leftmost 0-bit as first bit-flip. Since at most n improvements are needed and each failure at improving yields $\frac{n-i-1}{2} + 1 + \epsilon n$ wasted fitness function evaluations in expectation (i.e., $E[H(x, \text{opt})]$) where i is the number of the leading 1-bits, the expected time to find the optimum is at most $\sum_{i=1}^{n} n \cdot \left(\frac{n-i-1}{2} + 1 + \epsilon n\right) = O(n^3)$. □

Theorem 4 shows that the exponential fitness-based mutation potential prioivides at least a logarithmic and at most a $\sqrt{n}$ factor speed-up compared to static mutation potentials. Before proving the main results, we introduce the following lemma that will be used in the proof of the upcoming theorems.

Lemma 2. *For large enough n and any arbitrarily small constant ϵ, $n^{1/n^\epsilon} = (1 + \frac{\ln n}{n^\epsilon})(1 \pm o(1))$.*

Proof. By raising $\left(1 + \frac{\ln n}{n^\epsilon}\right)$ to the power of $\frac{n^\epsilon}{\ln n} \cdot \frac{\ln n}{n^\epsilon}$ we have,

$$\left(1 + \frac{\ln n}{n^\epsilon}\right)^{\frac{n^\epsilon}{\ln n} \cdot \frac{\ln n}{n^\epsilon}} = (1 \pm o(1))\, e^{\frac{\ln n}{n^\epsilon}} = (1 \pm o(1)) n^{1/n^\epsilon}.$$

□

Theorem 4. *The $(1+1)$ IA using IPH with $M_{expoF(x)}$ optimises* LEADINGONES *in* $O(n^3/\log n)$ *and* $\Omega(n^{(5/2)+\epsilon})$ *expected fitness function evaluations for any arbitrarily small constant $\epsilon > 0$.*

Proof. As already shown in the proof of Theorem 3, the expected number of leading 1-bits is smaller than 1 in the initialised bit string. The probability of improvement in the first step is $1/n$. In the case of failure at improving in the first step, at most $n^{\frac{n-i}{n}}$ fitness function evaluations are wasted where i is the number of leading 1-bits. Therefore, the total expected time to find the optimum is $E(T) \leq \sum_{i=1}^{n-1} n \cdot n^{(n-i)/n} = O(n^3/\log n)$ considering that $\sum_{i=1}^{n-1} n^{i/n} = \sum_{i=2}^{n} (n^{1/n})^{i-1} < \sum_{i=1}^{n} (n^{1/n})^{i-1} = \frac{1-n}{1-n^{1/n}}$ (using the general partial power series sum: $\sum_{i=0}^{m} a^i = \frac{1-a^{m+1}}{1-a}$ for $a \neq 1$) that gets in turn bounded from above by $n^2/(\log n)$ using Lemma 2.

The proof of the lower bound is similar to the proof of Theorem 3, except for the calculation of $E(f_i)$ when we want to consider the amount of wasted fitness function evaluations in case of E_1 happening. Here we have $E(f_i) = \frac{i}{n}\left(n^{(n-i)/n} + E(f_i)\right) + \frac{1}{n} \cdot 1 + \frac{n-i-1}{n}(1 + E(f_i))$. Solving it for $E(f_i)$ gives us $E(f_i) = i \cdot n^{(n-i)/n} + n - i$. Hence, the expected time to optimise LEADINGONES is $(1 - 2^{-n/2}) \sum_{i=n/2}^{n} E(f_i) = \Omega(1) \sum_{i=n/2}^{n} \left(i \cdot n^{(n-i)/n} + n - i\right) = \Omega(1) \left(\sum_{i=n/2}^{n} (n-i) + \sum_{i=n/2}^{n} \left(i \cdot n^{(n-i)/n}\right)\right)$. Evaluating the second sum in the interval $i \in [n/2 + \frac{\epsilon n}{2}, n/2 + \epsilon n]$, we get $\epsilon n/2 \cdot (n/2 + \epsilon n) \cdot n^{1/2 - \epsilon/2} = \Omega(n^{5/2-\epsilon})$. □

An advantage of Hamming distance-based exponential decays of the mutation potential compared to fitness-based ones is provided by the following theorem for LEADINGONES. The reason can be seen in Figure 2. While the initial fitness is very low, the potential is very high, but the actual number of bits that have to be flipped to reach the optimum (i.e., the Hamming distance), is much smaller. More precisely, fitness-based potentials suggest to flip n bits when only $n/2$ bits have to be flipped in expectation to reach the optimum in one step. M_{expoHD} exploits this property, thus wastes fewer fitness evaluations than $M_{\text{expoF(x)}}$.

Theorem 5. *The $(1+1)$ IA using IPH with M_{expoHD} optimises* LEADINGONES *in $O(\frac{n^{5/2+\epsilon}}{\ln n})$ and $\Omega(n^{9/4-\epsilon})$ expected fitness function evaluations for any arbitrarily small constant $\epsilon > 0$.*

Proof. The proof for the upper bound is similar to the proof of Theorem 3, however, each failure in improvement yields $n^{H(x, \text{opt})/n}$ wasted fitness function evaluations. Since the bits after the leading ones are uniformly distributed, by Chernoff bounds the number of 0-bits (Hamming distance to the optimum) is smaller than $1 + ((n-i-1)/2) + \epsilon n$ w.o.p. Hence, the expected time to optimise LEADINGONES is $\sum_{i=1}^{n} n \cdot n^{\frac{1+((n-i-1)/2)+\epsilon n}{n}} = n \sum_{i=1}^{n} n^{\frac{1}{2} - \frac{i}{2n} + \epsilon + \frac{1}{2n}} = n \cdot n^{1/2+\epsilon+\frac{1}{2n}} \sum_{i=1}^{n} n^{-\frac{i}{2n}}$. Knowing that $\sum_{i=0}^{\infty} n^{-\frac{i}{2n}} = \frac{1}{1-n^{(-1/(2n))}}$ and $n^{-1/(2n)} \leq \left(1 - \frac{\ln n}{2n}\right)(1 - o(1))$ for all $n > 1$ by Lemma 2, we get $1 - n^{(-1/(2n))} < \frac{\ln n}{2n}(1 + o(1))$. Hence, the expected time is $E(T) < n^{(3/2)+\epsilon} \cdot n^{1/(2n)} \cdot O\left(\frac{n}{\ln n}\right) = O\left(\frac{n^{(5/2)+\epsilon}}{\ln n}\right)$.

The proof of the lower bound is also similar to the proof of Theorem 3. By taking the same steps and solving the equation for $E(f_i)$,we get $E(f_i) = i \cdot n^{1/n+\frac{n-i-1}{2n}-\epsilon} + n - i$. Then, by taking into account the probabilities of starting with fewer than $n/2$ leading 1-bits and skipping a level i, we get the expected runtime of

$$\begin{aligned}&(1-2^{-n/2})\sum_{i=n/2}^{n} E(f_i)\\ &\geq \Omega(1)\sum_{i=n/2+\epsilon n/2}^{n/2+\epsilon n}\left(in^{1/n+((n-i-1)/2n)-\epsilon}+n-i\right)\\ &\geq \Omega(1)\sum_{i=n/2+\epsilon n/2}^{n/2+\epsilon n} in^{1/2-\epsilon+(1/(2n))-(i/(2n))} = \Omega(n^{(9/4)-\epsilon}).\end{aligned}$$

□

Given that M_{expoHD} provides larger hill-climbing speed-ups compared to the other mutation potentials and is stable to the scaling of fitness functions, we will use it in the remainder of the paper.

IV. Inefficient Behaviour of IPH in Practice

In this section we consider the usage of M_{expoHD} in realistic applications where the optimum is unknown. To this end, the best seen solution will be used by the operator rather than the unknown optimum. We combine IPH using M_{expoHD} with hybrid ageing, as in the Opt-IA AIS [26], and embed it in a $(1+1)$ Opt-IA shown in Algorithm 2. In this algorithm, $\alpha(x)$ returns the age of individual x. After the Hypermutate(x) operator an ageing operator is applied. Ageing has been shown to enable algorithms to escape from local optima either by identifying a gradient leading away from it or by restarting the whole optimisation process. Among the different variants of ageing, *hybrid* ageing has been shown to be very efficient at escaping local optima [28], [29], [32]. Using this operator, individuals are assigned with initial age zero. During each iteration of the algorithm, the age increases by 1 and is passed to the offspring if it does not improve over its parent's fitness. If the offspring is fitter than the parent, then its age is set to 0. At the end of each iteration, any individual with age larger than a threshold τ is removed with probability $1/2$ and in case there is no other individual left, a new individual is initialised uniformly at random.

Our aim behind the algorithm design is that by escaping local optima via ageing, the number of local optima that are identified by the algorithm increases over time. Thus as time goes by M_{expoHD} should approximate its ideal behaviour better and better. However, we will show that this is not the case. For the purpose we consider the well-studied bimodal benchmark function TwoMax illustrated in Figure 3 [36], [49]–[52]:

$$\text{TwoMax}(x) := \max\left\{\sum_{i=1}^{n} x_i,\ n - \sum_{i=1}^{n} x_i\right\}. \quad (1)$$

TwoMax is often used to evaluate the global exploration capabilities of evolutionary algorithms, i.e., whether the population can identify both optima of the function. The standard $(\mu+1)$ EA fails to identify both optima of TwoMax efficiently, hence, the function has been often used to evaluate the effectiveness of diversity mechanisms for improving the global exploration capabilities of populations-based EAs [36], [49]–[51]. Recently it has been shown that diversity mechanisms are not necessary for the $(\mu+1)$ EA to efficiently identify both optima of the function. It suffices to decrease the selection pressure of the algorithm e.g., by employing inverse tournament selection. In particular, the reason why the traditional $(\mu+1)$ EA fails is that the uniform parent selection usually employed leads to premature convergence of the population [52]. With sufficiently low selective pressure, or with an appropriate diversity mechanism, and a linear population size the $(\mu+1)$ EA can optimise TwoMax in approximately $\Theta(n^2 \log n)$ w.o.p. In the next section we will present an AIS using IPH that is considerably faster.

In this section, our analysis shows that once the $(1+1)$ Opt-IA using IPH with M_{expoHD} escapes from one local optimum, the mutation rate will increase as the algorithm climbs up the other branch. As a result, the algorithm struggles to identify the other optimum and wastes more and more fitness function evaluations as it approaches it. Thus, the whole purpose behind IPH is defeated.

On the bright side, we will show that M_{expoHD} combined with ageing can efficiently escape from local optima. We will use the well known Cliff_d function for this purpose where static hypermutations are inefficient:

$$\text{Cliff}_d(x) := \begin{cases}\sum_{i=1}^{n} x_i & \text{if } \sum_{i=1}^{n} x_i \leq n-d,\\ \sum_{i=1}^{n} x_i - d + 1/2 & \text{otherwise.}\end{cases} \quad (2)$$

The class of Cliff functions, illustrated in Figure 4, is usually used to assess the performance of non-elitist algorithms. It was originally designed as a problem instance where non-elitist EAs outperform elitist EAs [37]. This function has a OneMax slope with length $n-d$ which leads the algorithms towards the local optima. The local optima are followed by a second OneMax slope (of length d) of lower fitness which leads to the global optimum. While elitist algorithms need to make a jump of size d to find the global optimum, non-elitist algorithms can easily find the second slope by accepting inferior solutions, and then hillclimb up to the global optimum [3], [15], [38], [53]. Due to the high mutation rates of static hypermutations, even if they were to escape the local optima of Cliff (which is unlikely due to FCM and linear mutation potentials), they would still jump back with high probability.

We start by showing that without knowing the location of the optimum in advance the algorithm becomes very efficient for hill-climbing on OneMax and LeadingOnes.

Theorem 6. *The expected runtime of the* $(1+1)$ *Opt-IA using IPH with* M_{expoHD} *for optimising* OneMax *is* $\Theta(n \log n)$ *with* $\tau = \Omega(n^{1+\epsilon})$ *for some constant* $\epsilon > 0$.

Proof. Let x_t be the current solution at the beginning of the iteration t. Immediately after the initialisation, the best seen solution is the current individual x_1 itself and the mutation potential is $M = \text{M}_{\text{expoHD}} = n^{H(x_1,x_1)/n} = n^0 = 1$. Note that $x_t \neq x_{t+1}$ if and only if either $f(x_{t+1}) \geq f(x_t)$ or x_{t+1}

Algorithm 2 $(1+1)$ Opt-IA

```
Initialise x ∈ {0,1}^n uniformly at random;
set α(x) := 0 and best := x;
evaluate f(x);
while termination condition is not satisfied do
    α(x) := α(x) + 1;
    y := Hypermutate(x)
    if f(y) > f(x) then
        α(y) := 0;
    else
        α(y) := α(x);
    end if
    if f(y) ≥ f(best) then
        then set best := y;
    end if
    for w ∈ {x, y} do
        if α(w) ≥ τ then
            with probability p_die = 1/2, reinitialise w uniformly at random with α(w) = 0;
        end if
    end for
    Set x = arg max_{z∈{x,y}} f(z);
end while
```

is reinitialised after x_t is removed from the population due to ageing. Thus, the mutation operator flips a single bit at every iteration until ageing is triggered for the first time. The improvement probability will be at least $1/n$ until either 1^n or 0^n is sampled. Given that the ageing threshold τ is at least $n^{1+\epsilon}$ for some constant $\epsilon > 0$, the probability that the current solution will not improve τ times consecutively is at most $(1-\frac{1}{n})^{n^{1+\epsilon}} = e^{-\Omega(n^{\epsilon})}$. Hence, the first optimum will be found before ageing is triggered w.o.p. Given that the ageing operator is not triggered, the expected time to find the first optimum can be shown with a fitness level argument. Let k be the number of 0-bits in x_t. The probability that a mutation improves x_t is thus k/n and the expected time for such an improvement is n/k. If we sum over all possible k, we obtain the expected runtime $\sum_{k=0}^{n} n/k \leq n \ln n$, which is conditional on ageing not being triggered. If ageing triggers the mutation potential can be at most n and due to FCM the runtime would be at most $O(n^2 \log n)$ following the same line of argument while considering n wasted fitness function evaluations for each mutation. Since the ageing does not trigger with overwhelming probability, the expected runtime is $O(n \log n)$. To prove the lower bound we make a case distinction with respect to whether the ageing is triggered before the optimum is found. If the ageing mechanism is triggered, then by our assumption $\tau = \Omega(n^{1+\epsilon})$ the runtime is $\Omega(n \log n)$. For the second case, the operator flips a single bit per mutation. Since the algorithm is initialised with a uniformly random solution, the number of 0-bits in the initial solution is at least $n/3$ with overwhelming probability due to Chernoff bound on the binomial distribution. Similar to our argument for the upper bound, the expected time until the number of 0-bits in the solution decreases from k to $k-1$ is k/n. Therefore, the expected runtime is at least $(1-2^{-\Omega(n)}) \cdot \sum_{k=1}^{n/3} k/n = \Omega(n \log n)$. □

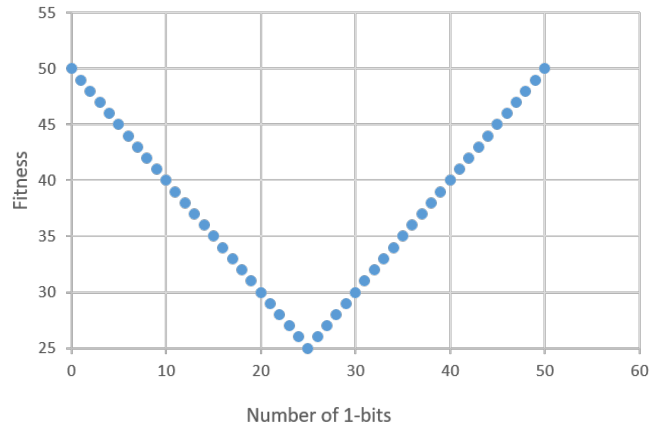

Fig. 3: TWOMAX test function.

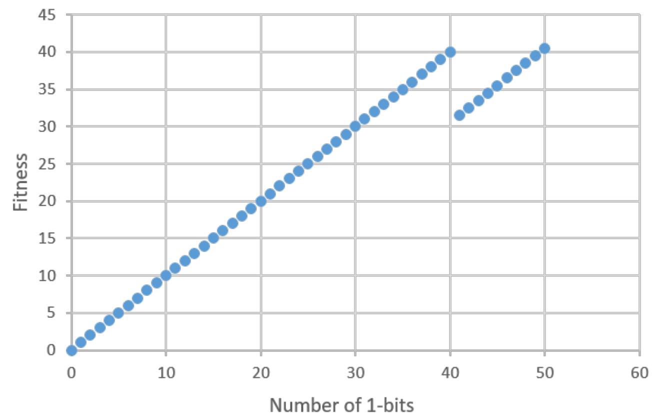

Fig. 4: CLIFF test function.

Theorem 7. *The expected runtime of the* $(1+1)$ *Opt-IA using IPH with* M_{expoHD} *for optimising* LEADINGONES *is* $\Theta(n^2)$ *with* $\tau = \Omega(n^{1+\epsilon})$ *for some constant* $\epsilon > 0$.

Proof. For any suboptimal $x \in \{0,1\}^n$ it is sufficient to flip the leftmost 0-bit of x to improve its LEADINGONES value. Thus, the same argument in Theorem 6 which establishes that the ageing will not trigger with overwhelming probability also holds. We can similarly continue with a fitness level argument, however the improvement probability for LEADINGONES is always $1/n$ regardless of the current function value and thus the sum yields $\sum_{i=0}^{n} n = n^2$. For the lower bound, it is sufficient to prove that the conditional expectation given that ageing does not trigger is $\Omega(n^2)$ since ageing will not trigger with overwhelming probability. In that case we have to consider how much the objective function value will increase in expectation when the leftmost 0-bit is flipped by the operator. The bits on the right of the leftmost 0-bit does not contribute to the fitness value and thus each can be a 0-bit or 1-bit with equal probability. Therefore, the expected number of "free-riders", i.e., the sequence of consecutive 1-bits that follow the leftmost 0-bit is at most $\sum_{i=0}^{n} 2^{-i} \leq 2$. Since the expected increase in after every mutation is at most 2, the expected number of necessary successful mutations where the leftmost 0-bit is flipped is at least $n/2$ to find the optimal solution. The expected time for each such successful mutation is $(1/n)^{-1} = n$ iterarions and consequently the expected time until the optimum is found given that the ageing operator does not trigger is $\Omega(n^2)$. □

We now move on to TWOMAX. The following theorem shows that after the algorithm escapes from the local optimum, the mutation rate increases as the algorithm climbs up the opposite branch. This behaviour causes a large waste of fitness evaluations defying the objectives of IPH.

Theorem 8. *The expected runtime of the* $(1+1)$ *Opt-IA using IPH with* M_{expoHD} *for optimising* TWOMAX *is* $\Theta(n^2 \log n)$ *with* $\tau = \Omega(n^{1+\epsilon})$ *for some constant* $\epsilon > 0$*.*

Proof. We follow the same argument in Theorem 6 to conclude that in expected $O(n \log n)$ time, the algorithm will find one of the optima. After finding the first optimum, no single-bit flip can yield an equally fit solution. The individual then reaches age τ and the ageing reinitialises the current solution. Thus, with $(1 - 2^{-n}) \cdot \left(1 - e^{-\Omega(n^\epsilon)}\right)$ probability a new solution will be initialised within $\tau + n$ steps. Now, the current *best seen* is the first discovered optimum.

Let the first and second branch denote the subsets of the solution space which consist of solutions with less than and more than Hamming distance $n/2$ to the first discovered optimum respectively. We first consider the case when the new solution is initialised on the first branch. Given that the current solution has fitness i, the distance to the best seen is $n - i$ and the mutation potential is $M = n^{\frac{n-i}{n}}$. Since the first constructive mutation ensures that the probability of improvement is always at least $1/n$, w.o.p., ageing will not be triggered until either optimum is discovered. Thus, given that the current solution is always on the first branch, the proof of Theorem 2 carries over and the expected time to find the first discovered optimum once again is at most $O(n^{3/2} \log n)$ in expectation. When the first discovered optimum is sampled again, w.o.p. the current solution is reinitialised with the first discovered optimum as the best seen solution in at most $\tau + n$ iterations.

Now, we consider the case where the current solution is on the second branch. A lower bound on the probability that the current solution will reach the optimum of the second branch before sampling an improving solution on the first branch will conclude the proof since the expected time to do so is $O(n^{3/2} \log n)$ given that the current solution does not switch branches before.

We will start by bounding the mutation potential for a solution on the second branch with fitness value $n - k$. Since the M_{expoHD} is always smaller than M_{linHD}, we can assume that no more than $n - k$ bits will be flipped.

Without losing generality, let the second branch be the branch with more 0-bits. For a hypermutation operation with mutation potential N, the Nth solution that will be sampled has the highest probability of finding a solution with at least $n - k$ 1-bits (i.e., switching to the first branch). The current solution has $n - k$ 0-bits and k 1-bits. If more than $k/2$ 1-bits are flipped, then the number of 1-bits in the final solution after $n - k$ mutation steps is less than $n - k$ since the number of 0-bits flipped to 1-bits is less than $n - k - k/2$ and the number of remaining 1-bits is less than $k/2$. The event that at most $k/2$ 1-bits are flipped is equivalent to the event that in a uniformly random permutation of n bit positions at least $k/2$ 1-bits are ranked in the last k positions of the random permutation. We will now bound the probability that exactly $k/2$ 1-bits are in the last k positions since having more 1-bits has a smaller probability. Each particular outcome of the last k positions has the equal probability of $\prod_{i=0}^{k-1}(n-i)^{-1}$. There are $\binom{k}{k/2}$ different equally likely ways to choose $k/2$ 1-bits and $k!$ different permutations of the last k positions, thus the probability of having exactly $k/2$ 1-bits in the last k positions is $\prod_{i=0}^{k-1} \frac{1}{n-i} \cdot \binom{k}{k/2} \cdot k! = \prod_{i=0}^{k-1} \frac{1}{n-i} \cdot \left(\frac{k!}{(k/2)!}\right)^2 < \left(\frac{k}{n-k}\right)^k$. For any $k \in [4, \frac{n}{2} - \Omega(\sqrt{n})]$ this probability is in the order of $\Omega(1/n^4)$. Using a union bound over the probabilities of having more than $k/2$ 1-bits and the probabilities of improving before the final step, we get $\frac{k}{2} \cdot (n-k) \cdot \Omega(1/n^4) = \Omega(1/n^2)$. Since the new solutions are uniformly sampled, the number of bits in the solutions are initially distributed binomially with parameters n and $1/2$ which has a variance in the order of $\Theta(\sqrt{n})$ and implies that with constant probability k is less than $\frac{n}{2} - \Omega(\sqrt{n})$ in the initial solution. Given that the expected time in terms of generations is in the order of $O(n \log n)$, the total probability of switching branches while $k \in [k, \frac{n}{2} - \Omega(\sqrt{n})]$ is at most $\left(1 - 1/n^2\right)^{O(n \log n)} = 1 - \Omega(1)$. Finally, we will consider the cases of $k \in \{1, 2, 3\}$ separately. When $k = 1$, the probability of flipping less than $k/2$ 1-bits is equivalent to the probability of not flipping the single 1-bit which happens with probability $1/n$. For $k = 2$, similarly we have to flip at most one 1-bit with probability $O(1/n)$. For $k = 3$, it is necessary that at least two 1-bits are not flipped which happens with probability at most $O(1/n^2)$. Thus, the probability of switching branches when $k < 4$ is at most $O(1/n)$, which gives a lower bound on the total probability of switching branches in the order of $1 - \Omega(1)$ given that the initial solution has at least $k > \frac{n}{2} - \Omega(\sqrt{n})$ 1-bits (which also occurs with at least $\Omega(1)$ probability). Given that no switch occurs, the expected time to find the second optimum is $\sum_{i=1}^{n/2+\epsilon n} \frac{n}{i} \cdot n^{\frac{n-i}{n}} = O(n^2 \log n)$.

For the lower bound, assume that the first discovered solution is 1^n and consider the last $\frac{n}{\log n}$ fitness-levels before the second optimum 0^n is discovered. Due to FCM, the current solution has to visit each of these levels and the minimum mutation potential in these fitness levels is $n^{\frac{n-n/\log n}{n}} = \frac{n}{e}$. Thus, the expected time is at least $\sum_{i=1}^{n/\log n} \frac{n}{i} \cdot \frac{n}{e} = \Omega(n^2 \log n)$. □

Now we show that differently from static hypermutations, M_{expoHD} combined with ageing can escape from the local optima of CLIFF_d, hence optimises the function efficiently. The condition that d is not prohibitively large is necessary to avoid that the reinitialised solutions due to ageing have larger fitness than a solution with $n - d + 1$ 1-bits. We believe that this assumption is realistic for practical applications.

Theorem 9. *The* $(1+1)$ *Opt-IA using IPH with* M_{expoHD} *and* $\tau = \Omega(n^{1+\epsilon})$ *for an arbitrarily small constant* ϵ*, optimises* CLIFF_d *with* $d < n(\frac{1}{4}-\epsilon)$ *in* $O(n^{3/2}\log n+\tau n^{1/2}+\frac{n^{7/2}\log n}{d^2})$ *expected fitness function evaluations.*

Proof. The analysis will follow a similar idea to the proof of Theorem 8. After initialisation, the initial mutation potential is $M = 1$ since the current solution is the best seen solution. With single bit-flips it takes in expectation at most $O(n \log n)$ to find a local optimum of the cliff (i.e., a search point with $n-d$ 1-bits). Since the local optima cannot be improved with single bit flips, in τ generations after it was first discovered the ageing will be triggered and in the following n steps the current solution will be removed from the population due to ageing with probability at least $1-2^{-n}$. The Hamming distance of the reinitialised solution will be distributed binomially with parameters n and $1/2$ and w.o.p. will be smaller than $n/2+n^{2/3}$, yielding an initial mutation potential of $M = O(n^{1/2})$. We pessimistically assume that the mutation potential will not decrease until a local optima is found again, which implies that the expected time will be at most $O(n^{3/2}\log n+\tau n^{1/2})$ since each iteration will waste an extra $O(n^{1/2})$ fitness function evaluations. After finding a local optima again, the mutation potential will be $M = 1$ since it will replace the previously observed local optima as the best seen. The process of reinitialisation and reaching the local optima will repeat until a solution at the bottom of the Cliff is created and accepted.

If the local optima produces an offspring with $n-d+1$ bits, which happens with probability d/n and if this solution survives the ageing operator (with probability $(1-p_{\text{die}})$), then the reinitialised solution will be rejected w.o.p. since its fitness value will be smaller than $n-d$ due to our assumption $d < n(\frac{1}{4}-\epsilon)$. The Hamming distance of this new solution to the best seen will be exactly one since it is created via a single bit-flip, thus its mutation potential will be $M = 1$. Moreover, if the surviving offspring improves again (with probability $(d-1)/n$) in the next iteration, it will reset its age to zero and will have Hamming distance of at least two to any local optima. In expected $O(n/\log n)$ generations, this solution will reach the global optimum unless a solution with less or equal $n-d$ 1-bits is sampled before. Initially, this is impossible since for at least $\omega(1)$ steps we have $M = 1$, and later $M < 3$ holds as long as the distance to the last seen local optima is at most $n/\ln n$ since $n^{\frac{n/\ln n}{n}} = e$. Note that the number of 1-bits does not always reflect the actual Hamming distance since more than one bit can be flipped in an accepted offspring. We will pessimistically assume that all improvements have increased the Hamming distance by three until the total Hamming distance reaches $n/\ln n$, which implies that there have been $n/(3\ln n)$ accepted solutions. The Ballot theorem implies that sampling a solution that is at least as good as the parent (which are the only solutions that are accepted) has probability at most $2i/n$ where i is the number of 0-bits in the solution. Since the probability of improving in the first step is at least i/n, we can conclude that the conditional probability that an accepted offspring is a strict improvement is at least $1/2$. Thus, when the Hamming distance to the local optima reaches $n/\ln n$, in expectation, the current solution will have at least $n/(6\ln n)$ extra 1-bits compared to the local optima and at least $n^{3/5}$ extra 1-bits w.o.p. by Chernoff bounds. The Hamming distance to the local optima can be at most $2d$ since both the local optima and the current solution have less than d 0-bits. Since $d < n/4$, the mutation potential is at most $n^{\frac{2d}{n}} \leq \sqrt{n}$. Thus, no hypermutation can yield a solution with less than $n-d+2$ 1-bits. Therefore, once a solution with $n-d+2$ bits is added to the population, the algorithm finds the optimum w.o.p. in $O(n \log n)$ iterations and in at most $O(n^{3/2}\log n)$ fitness function evaluations since the mutation potential is at most $\sqrt{n}$. The probability of obtaining a solution with $n-d+2$ 1-bits at the end of each cycle of reinitialisation and removal of the local optima due to ageing is $(1-p_{\text{die}})^2\cdot(d/n)\cdot((d-1)/n)$. Since each such cycle takes $O(n^{3/2}\log n)$ fitness evaluations, our claim follows. □

V. An Efficient IPH with Mutation Potentials for Opt-IA

In the previous section, we observed that towards the end of the optimisation process the mutation potential may increase as the current solution approaches an undiscovered, potentially promising optimum. This behaviour is against the design intentions of the inversely proportional hypermutation operator since in the final part of the optimisation process it gets harder to find improvements and high mutation potentials lead to many wasted fitness function evaluations. The underlying reason of this behaviour in both the CLIFF and TWOMAX landscapes is the necessity to follow a gradient that leads away from the local optimum to find the global one. Considering that this necessity would be ubiquitous in optimisation problems, we propose a new method to control mutation potentials in this section. The newly proposed mutation potential is called *Symmetric* M_{expoHD} and is defined as:

$$\text{Symmetric } \mathrm{M}_{\text{expoHD}} := \max\left\{\left\lfloor n^{\frac{H(\text{best},\text{org}(x))-H(x,\text{org}(x))}{n}}\right\rfloor, 1\right\}. \quad (3)$$

Symmetric $\mathrm{M}_{\text{expoHD}}$ uses a mutation potential that is inversely proportional to the current solution's Hamming distance to its origin, where the origin (returned by $\text{org}(x)$ in (3)) is defined as the ancestor of the current bit string after the last removal of a solution due to ageing (Figure 5) i.e., the re-initialised individual or the one that survived ageing. At initialisation, the origin of each solution is set to itself and the newly created offspring inherit the origin of their parents.

This mutation potential reliably decreases (at the same rate it would decrease if it was approaching the currently best seen local optimum) as the current solution improves and moves away from its origin up until it starts doing local search and finds a local optimum. Every time a local optimum is found, ageing is triggered after approximately τ steps and then both surviving and reinitialised individuals reset their origin to their own bit string. We embed IPH with Symmetric $\mathrm{M}_{\text{expoHD}}$ in Algorithm 3.

Since our results for ONEMAX and LEADINGONES in the previous section rely on ageing not being triggered, they carry over to the symmetric variant. The following theorem shows

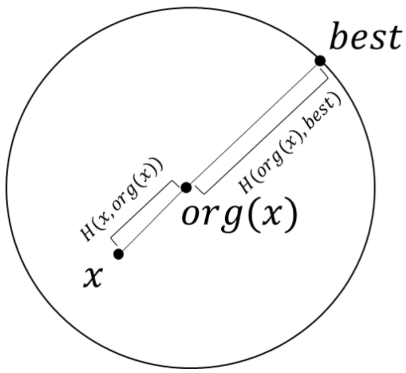

Fig. 5: The geometric representation of Symmetric $\mathrm{M}_{\mathrm{expoHD}}$. The mutation potential is determined according to the ratio of the Hamming distance between the current solution (x) and its origin (org(x)) and the Hamming distance between its origin and the best seen solution (best).

Algorithm 3 (1+1) Opt-IA with symmetric IPH

Initialise $x \in \{0,1\}^n$ uniformly at random;
set org$(x) := x$, $\alpha(x) := 0$, and best$:= x$;
evaluate $f(x)$;
while termination condition not satisfied **do**
    $\alpha(x) := \alpha(x) + 1$;
    $y :=$ Hypermutate(x) with Symmetric $\mathrm{M}_{\mathrm{expoHD}}$;
    org$(y) :=$ org(x);
    if $f(y) > f(x)$ **then**
        $\alpha(y) := 0$;
        if $f(y) \geq best$ **then**
            best$:= y$;
        end if
    else
        $\alpha(y) := \alpha(x)$;
    end if
    for $w \in \{x, y\}$ **do**
        if $\alpha(w) \geq \tau$ **then**
            with probability $1/2$, reinitialise w uniformly at random with $\alpha(w) = 0$;
            set org$(w) = w$
        end if
    end for
    Set $x = \arg\max_{z \in \{x,y\}} f(z)$;
end while

that once one optimum of TWOMAX has been identified, the value of the Symmetric $\mathrm{M}_{\mathrm{expoHD}}$ potential decreases as both optima are approached as desired and the wished for speed-up in the runtime is achieved.

Theorem 10. *The* $(1+1)$ *Opt-IA using IPH with Symmetric* M_{expoHD} *with* $\tau = \Omega(n^{1+\epsilon})$ *for any arbitrarily small constant* $\epsilon > 0$, *needs* $O(n^{3/2} \log n)$ *fitness function evaluations in expectation to optimise* TWOMAX.

Proof. The expected time until the first branch is optimised is $O(n \log n)$ since the best seen search point is the current best individual and consequently the mutation potential is $M = 1$. Since the improvement probability is at least $1/n$ and $\tau = \Omega(n^{1+\epsilon})$, the ageing operator does not trigger before finding one of the optima w.o.p. Once one of the optima is found, ageing reinitialises the individual while the first discovered optima stays as the current best seen search point. For the randomly reinitialised solution, the Hamming distance to the best seen is binomially distributed with parameters n and $1/2$. Using Chernoff bounds, we can bound the distance to the previously seen optima by at most $n/2 + n^{2/3}$ w.o.p. This Hamming distance implies an initial mutation potential of $M < n^{\frac{\frac{n}{2}+n^{2/3}}{n}} = n^{\frac{1}{2}+\frac{1}{n^{1/3}}}$ which decreases as the individual increases its distance to the origin and can never go above its initial value where the distance is zero. Pessimistically assuming that the mutation potential will be $n^{\frac{1}{2}+\frac{1}{n^{1/3}}} = O(n^{1/2})$ throughout the run, we can obtain the above upper bound by summing over all levels and using coupon collector's argument [54]. $\square$

Since the Opt-IA using static hypermutations requires $\Theta(n^2 \log n)$ expected function evaluations to climb up either branch of TWOMAX [38], this result represents a $\sqrt{n}$ speed-up over static mutation potentials and over the known upper bounds for population based EAs to find both optima w.o.p.

The following theorem shows that Symmetric $\mathrm{M}_{\mathrm{expoHD}}$ is also efficient for CLIFF_d.

Theorem 11. *The* $(1+1)$ *Opt-IA using IPH with Symmetric* M_{expoHD} *and* $\tau = \Omega(n^{1+\epsilon})$ *optimises* CLIFF_d *with* $d < n(\frac{1}{4}-\epsilon)$ *and* $d = \Theta(n)$ *in* $O(n^{3/2} \log n + \tau n^{1/2} + \frac{n^{7/2}}{d^2})$ *expected fitness function evaluations.*

Proof. The proof is almost identical to the proof of Theorem 9. The most important distinction is that once a solution with $n - d + 2$ 1-bits is created, its mutation potential remains as $M = 1$ until the global optimum is found. The reason is that when ageing is triggered, the surviving solutions all reset their origin to themselves, i.e., start doing randomised local search. $\square$

VI. CONCLUSION

We have presented a thourough analysis of Inversely Proportional Hypermutations (IPH). Previous theoretical studies have shown disappointing results concerning the IPH operators from the literature. In this paper we have proposed a new IPH based on Hamming distance and exponential decay. We have shown its effectiveness in isolation for unimodal functions compared to static hypermutations in the ideal conditions when the optimum is known. Furthermore, we have provided a symmetric version of the operator for the complete Opt-IA AIS to be used in practical applications where the optimum is usually unknown. We have proved its efficiency for two well-studied multimodal functions with considerably different characteristics. Although our analysis used standard benchmark functions with significant structures to practical optimisation, we point out that the Opt-IA AIS with our proposed IPH can be shown to efficiently find arbitrarily good approximations to the NP-Hard Number Partitioning problem by following the same proof strategy of [30] that uses that local optima are escaped from thanks to the ageing operator. Future work should evaluate the performance of the proposed algorithm for other combinatorial optimisation problems with practical applications. A desirable further improvement to the immune system would be to also provide it with the capability to escape from local optima via large mutations, something that is currently not possible since the mutation rate is lowest on the local optima.

REFERENCES

[1] T. Friedrich, F. Quinzan, and M. Wagner, “Escaping large deceptive basins of attraction with heavy-tailed mutation operators,” in *Proc. of GECCO 2018*, 2018, pp. 293–300.

[2] T. Friedrich, A. Göbel, F. Quinzan, and M. Wagner, “Heavy-tailed mutation operators in single-objective combinatorial optimization,” in *Proc. of PPSN XV*, 2018, pp. 134–145.

[3] D. Corus, P. S. Oliveto, and D. Yazdani, “Fast immune system inspired hypermutation operators for combinatorial optimisation,” *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 5, pp. 956–970, 2021.

[4] B. Doerr, H. P. Le, R. Makhmara, and T. D. Nguyen, “Fast genetic algorithms,” in *Proc. of GECCO 2017*, 2017, pp. 777–784.

[5] D. Corus and P. S. Oliveto, “Standard steady state genetic algorithms can hillclimb faster than mutation-only evolutionary algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 5, pp. 720–732, 2017.

[6] D.-C. Dang, T. Friedrich, T. Kötzing, M. S. Krejca, P. K. Lehre, P. S. Oliveto, D. Sudholt, and A. M. Sutton, “Escaping local optima using crossover with emergent diversity,” *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 484–497, 2018.

[7] D. Corus and P. S. Oliveto, “On the benefits of populations on the exploitation speed of standard steady-state genetic algorithms,” in *Proc. of GECCO 2019*, 2019, pp. 1452–1460.

[8] P. S. Oliveto, D. Sudholt, and C. Witt, “A tight lower bound on the expected runtime of standard steady state genetic algorithms,” in *Proc. of GECCO 2020*, 2015, pp. 1323–1331.

[9] B. Doerr and C. Doerr, “Theory of parameter control for discrete black-box optimization: Provable performance gains through dynamic parameter choices,” in *Theory of Evolutionary Computation - Recent Developments in Discrete Optimization*, B. Doerr and F. Neumann, Eds. Springer, 2020, pp. 271–321.

[10] B. Doerr, A. Lissovoi, P. S. Oliveto, and J. A. Warwicker, “On the runtime analysis of selection hyper-heuristics with adaptive learning periods,” in *Proc. of GECCO 2018*, 2018, pp. 1015–1022.

[11] B. Doerr, C. Doerr, and J. Yang, “k-bit mutation with self-adjusting k outperforms standard bit mutation,” in *Proceedings of the International Conference on Parallel Problem Solving from Nature*, ser. PPSN ’16. Springer, 2016, pp. 824–834.

[12] A. Lissovoi, P. S. Oliveto, and J. A. Warwicker, “Simple Hyper-Heuristics Control the Neighbourhood Size of Randomised Local Search Optimally for LeadingOnes,” *Evolutionary Computation*, vol. 28, no. 3, pp. 437–461, 09 2020.

[13] ——, “How the duration of the learning period affects the performance of random gradient selection hyper-heuristics.” in *Proceedings of the AAAI Conference on Artificial Intelligence*, ser. AAAI ‘20, 2020, pp. 2376–2383.

[14] P. S. Oliveto, T. Paixão, J. Pérez Heredia, D. Sudholt, and B. Trubenová, “How to escape local optima in black box optimisation: when non-elitism outperforms elitism,” *Algorithmica*, vol. 80, pp. 1604—-1633, 2018.

[15] A. Lissovoi, P. S. Oliveto, and J. A. Warwicker, “On the time complexity of algorithm selection hyper-heuristics for multimodal optimisation,” in *Proc. of AAAI 2019*, 2019, pp. 2322–2329.

[16] D.-C. Dang, A. V. Eremeev, and P. K. Lehre, “Escaping local optima with non-elitist evolutionary algorithms,” in *Proc. of AAAI 2021*, 2021.

[17] B. Doerr, “Does comma selection help to cope with local optima?” in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO ’20. ACM, 2020, p. 1304–1313.

[18] B. Case and P. K. Lehre, “Self-adaptation in non-elitist evolutionary algorithms on discrete problems with unknown structure,” *IEEE Transactions on Evolutionary Computation (To appear - arXiv:2004.00327)*, vol. 24, no. 4, pp. 650–663, 2020.

[19] D.-C. Dang, T. Jansen, and P. K. Lehre, “Populations can be essential in dynamic optimisation,” in *Proc. of GECCO 2015*, 2015, pp. 1407–1414.

[20] D.-C. Dang and P. K. Lehre, “Efficient optimisation of noisy fitness functions with population-based evolutionary algorithms,” in *Proc. of FOGA 2015*, 2015, pp. 62–68.

[21] L. R. de Castro and J. Timmis, *Artificial Immune Systems: A New Computational Intelligence Paradigm*. Springer-Verlag, 2002.

[22] Z. Huang and Y. Zhou, “Runtime analysis of somatic contiguous hypermutation operators in MOEA/D framework,” in *Proc. of AAAI*, 2020, pp. 2359–2366.

[23] F. M. Burnet, *The Clonal Selection Theory of Acquired Immunity*. Cambridge University Press, 1959.

[24] J. Brownlee, “Clonal selection algorithms,” Complex Intelligent Systems Laboratory (CIS), Swinburne University of Technology, Tech. Rep., 2007.

[25] L. N. de Castro and F. J. V. Zuben, “Learning and optimization using the clonal selection principle,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 239–251, 2002.

[26] V. Cutello, G. Nicosia, M. Pavone, and J. Timmis, “An immune algorithm for protein structure prediction on lattice models,” *IEEE Transactions on Evolutionary Computation*, vol. 11, pp. 101–117, 2007.

[27] C. Zarges, “Rigorous runtime analysis of inversely fitness proportional mutation rates,” in *Proc. of PPSN*, 2008, pp. 112–122.

[28] D. Corus, P. S. Oliveto, and D. Yazdani, “On the runtime analysis of the Opt-IA artificial immune system,” in *Proc. of GECCO 2017*, 2017, pp. 83–90.

[29] ——, “Artificial immune systems can find arbitrarily good approximations for the NP-hard partition problem,” in *Proc. of PPSN 2018*, 2018, pp. 16–28.

[30] ——, “Artificial immune systems can find arbitrarily good approximations for the NP-hard number partitioning problem,” *Artificial Intelligence*, vol. 247, pp. 180–196, 2019.

[31] T. Jansen and C. Zarges, “Variation in artificial immune systems: hypermutations with mutation potential,” in *Proc. of ICARIS 2011*, 2011, pp. 132–145.

[32] P. S. Oliveto and D. Sudholt, “On the runtime analysis of stochastic ageing mechanisms,” in *Proc. of GECCO 2014*, 2014, pp. 113–120.

[33] C. Horoba, T. Jansen, and C. Zarges, “Maximal age in randomized search heuristics with aging,” in *Proc. of GECCO 2009*, 2009, pp. 803–810.

[34] D. Corus, P. S. Oliveto, and D. Yazdani, “On inversely proportional hypermutations with mutation potential,” in *Proc. of GECCO 2019*, 2019, p. 215–223.

[35] S. Droste, T. Jansen, and I. Wegener, “On the analysis of the $(1+1)$ evolutionary algorithm,” *Theoretical Computer Science*, vol. 276, no. 1-2, pp. 51–81, 2002.

[36] T. Friedrich, P. S. Oliveto, D. Sudholt, and C. Witt, “Analysis of diversity-preserving mechanisms for global exploration,” *Evolutionary Computation*, vol. 17, no. 4, pp. 455–476, 2009.

[37] J. Jägersküpper and T. Storch, “When the plus strategy outperforms the comma strategy and when not,” in *Proc. of FOCI 2007*, 2007, pp. 25–32.

[38] D. Corus, P. S. Oliveto, and D. Yazdani, “When hypermutations and ageing enable artificial immune systems to outperform evolutionary algorithms,” *Theoretical Computer Science*, vol. 832, pp. 166–185, 2019.

[39] P. S. Oliveto and C. Witt, “On the runtime analysis of the simple genetic algorithm,” *Theoretical Computer Science*, vol. 545, pp. 2–19, 2014.

[40] ——, “Improved time complexity analysis of the simple genetic algorithm,” *Theoretical Computer Science*, vol. 605, pp. 21–41, 2015.

[41] D. Whitley, “The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best,” in *Proc. of the Third International Conference on Genetic Algorithms*, 1989, pp. 116–121.

[42] F. Neumann, P. S. Oliveto, and C. Witt, “Theoretical analysis of fitness-proportional selection: landscapes and efficiency,” in *Proc. of GECCO 2009*, 2009, pp. 835–842.

[43] D. Corus, D.-C. Dang, A. V. Eremeev, and P. K. Lehre, “Level-based analysis of genetic algorithms and other search processes,” *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 5, pp. 707–719, 2017.

[44] D. Corus and P. S. Oliveto, “On the benefits of populations for the exploitation speed of standard steady-state genetic algorithms,” *Algorithmica*, vol. 82, pp. 3676–3706, 2020.

[45] B. Doerr, C. Doerr, and F. Ebel, “From black-box complexity to designing new genetic algorithms,” *Theoretical Computer Science*, vol. 567, pp. 87–104, 2015.

[46] S. Böttcher, B. Doerr, and F. Neumann, “Optimal fixed and adaptive mutation rates for the leadingones problem,” in *Parallel Problem Solving from Nature*, ser. PPSN ‘10. Springer, 2010, pp. 1–10.

[47] A. Lissovoi, P. S. Oliveto, and J. A. Warwicker, “How the duration of the learning period affects the performance of random gradient selection hyper-heuristics,” in *Proc. of AAAI 2020*, 2020, pp. 2376–2383.

[48] W. Feller, *An Introduction to Probability Theory and Its Applications*. John Wiley & Sons, 1968.

[49] D. Sudholt, "The benefits of population diversity in evolutionary algorithms: a survey of rigorous runtime analyses," in *Theory of Randomized Search Heuristics in Discrete Search Spaces*, B. Doerr and F. Neumann, Eds. Spinger, 2019, ch. 8, pp. 359–400.

[50] P. S. Oliveto, D. Sudholt, and C. Zarges, "On the benefits and risks of using fitness sharing for multimodal optimisation," *Theoretical Computer Science*, vol. 773, pp. 53–70, 2019.

[51] E. C. Osuna and D. Sudholt, "Analysis of the clearing diversity-preserving mechanism," in *Proc. of FOGA 2017*, 2017, pp. 55–63.

[52] D. Corus, A. Lissovoi, P. S. Oliveto, and C. Witt, "On steady-state evolutionary algorithms and selective pressure: Why inverse rank-based allocation of reproductive trials is best," *ACM Transactions on Evolutionary Learning and Optimization*, vol. 1, no. 1, 2021.

[53] M. A. Hevia-Fajardo and D. Sudholt, "Self-adjusting offspring population sizes outperform fixed parameters on the cliff function," in *Proc. of FOGA 2021*, 2021, pp. 1–15.

[54] P. S. Oliveto and X. Yao, "Runtime analysis of evolutionary algorithms for discrete optimisation," in *Theory of Randomized Search Heuristics: Foundations and Recent Developments*. World Scientific, 2011, ch. 2, pp. 21–52.

VII. Biography Section

If you have an EPS/PDF photo (graphicx package needed), extra braces are needed around the contents of the optional argument to biography to prevent the LaTeX parser from getting confused when it sees the complicated `\includegraphics` command within an optional argument. (You can create your own custom macro containing the `\includegraphics` command to make things simpler here.)

If you include a photo:

Michael Shell Use `\begin{IEEEbiography}` and then for the 1st argument use `\includegraphics` to declare and link the author photo. Use the author name as the 3rd argument followed by the biography text.

If you will not include a photo:

John Doe Use `\begin{IEEEbiographynophoto}` and the author name as the argument followed by the biography text.