On Inversely Proportional Hypermutations with Mutation Potential

Dogan Corus
Department of Computer Science
University of Sheffield
Sheffield
d.corus@sheffield.ac.uk

Pietro S. Oliveto
Department of Computer Science
University of Sheffield
Sheffield
p.oliveto@sheffield.ac.uk

Donya Yazdani
Department of Computer Science
University of Sheffield
Sheffield
dyazdani1@sheffield.ac.uk

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 mutation potentials (IPM) and argue that the potential should decrease inversely to the distance to the optimum rather than to the difference in fitness. Afterwards, we define a simple (1+1) Opt-IA that uses IPM hypermutations and ageing for realistic applications where optimal solutions are unknown. The aim of this AIS is to approximate the ideal behaviour of the inversely proportional hypermutations better and better as the search space is explored. We prove that such desired behaviour and related speed-ups occur for a well-studied bimodal benchmark function called TwoMax. Furthermore, we prove that the (1+1) Opt-IA with IPM efficiently optimises a second multimodal function, Cliff, by escaping its local optima while Opt-IA with static mutation potential cannot, thus requires exponential expected runtime in the distance between the local and global optima.

CCS CONCEPTS

• **Computing methodologies** → *Bio-inspired approaches.*

KEYWORDS

Hypermutations, Artificial Immune Systems, Runtime Analysis.

ACM Reference Format:
Dogan Corus, Pietro S. Oliveto, and Donya Yazdani. 2019. On Inversely Proportional Hypermutations with Mutation Potential. In *Genetic and Evolutionary Computation Conference (GECCO '19), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3321707.3321780

GECCO '19, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6111-8/19/07…$15.00
https://doi.org/10.1145/3321707.3321780

1 INTRODUCTION

Artificial Immune Systems (AIS) for optimisation are generally inspired by the clonal selection principle [2]. For this reason they are also often referred to as clonal selection algorithms [1]. In the literature, two key features of clonal selection algorithms have been identified [9]:

(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 [10] and Opt-IA [8].

The ideal behaviour of the IPH operator is that the mutation rate is minimal in proximity of the global optimum and should increase 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 some 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 (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 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 [26]. 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 OneMax(x)/*opt* where *opt* is the optimum value) and an exponential decay (i.e.,

[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.

each bit flips with probability $e^{-\rho \frac{\text{OneMax(x)}}{opt}}$ 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 mutation operator will require exponential time to optimise OneMax with overwhelming probability[2] (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 of the global optimum (i.e., fitness values of $n/2$ and n, respectively), 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 $1/2$ for every possible fitness value of OneMax). Hence, the desired behaviour of the mutation operator, achieved by assuming the optimum is known, leads to very inefficient algorithms even for the simple OneMax function.

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). This 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 [3, 7] and at finding arbitrarily good approximations for the Number Partitioning NP-Hard problem [4, 5] at the expense of being slower at hillclimbing during exploitation phases (i.e., it is up to a linear factor slower than standard bit mutation for easy functions such as OneMax and LeadingOnes). Hence, differently from other clonal selection algorithms (such as Clonalg), hypermutations with mutation potential coupled with FCM can cope with the desired behaviour of inversely proportional mutation rates.

In this paper, we consider whether Opt-IA may become faster during exploitation phases if inversely proportional mutation potentials are applied rather than static ones. The reason to believe this is that 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. 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 of the inversely proportional hypermutations with mutation potential used by Opt-IA for OneMax has shown that the mutation rate is always in the range of $[(c/2)n, cn]$ where $M = cn$ is the highest mutation potential. Hence, it does not decrease inversely proportional to the optimum as desired [16].

In this paper we first show that considerable speed-ups in exploitation phases may be achieved compared to static hypermutations, if 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 [3]. Through the analysis, we show 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 IPMs since it provides the largest speed-ups. Furthermore, using the Hamming distance rather than fitness as measure to quantify proximity to the optimum makes the operator robust to fitness function scaling. Hence, we consider this operator which is called M_{expoHD} in the rest of the paper.

Afterwards, we propose an AIS called (1+1) Opt-IA which uses IPH and ageing to be applied in practical applications where the optimum is not known[3]. The algorithm uses the best solution it has seen to estimate the mutation rates. In the literature it has been shown that ageing allows algorithms to escape from local optima by identifying a new slope of increasing fitness or to completely restart the optimisation process if it cannot escape [3, 4, 14, 15]. The idea behind the algorithm is that the more 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.

We use the Cliff benchmark function to show that the IPH can escape local optima when coupled with ageing. Static hypermutations cannot optimise the function efficiently because once the local optimum is escaped, the high mutation rates force the algorithm to jump back to the local optima with high probability. Our analysis though reveals that in general such a strategy does not produce the desired effect. 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 can 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 the runtime of static hypermutations to optimise each slope of the function (i.e., $\Theta(n^2 \log n)$). To this end, we define a *Symmetric* IPH operator that decreases the potential with respect to the distance to the best seen 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 the TwoMax benchmark function while showing that the local optima of Cliff can still be escaped by the Symmetric IPH operator.

Due to space limitations, some proofs are omitted[4].

[2] We say events occur "with overwhelming probability" (w.o.p.) when they occur with probability at least $1 - o(1/poly(n))$.

[3] The well-known Opt-IA AIS uses both operators in combination [8].

[4] A complete version of the paper with all the proofs is available in [6].

2 PRELIMINARIES

The original IPH potential proposed for Opt-IA was $M = \lceil(1 - f_{OPT}/v)\rceil cn$ for minimisation problems, where f_{OPT} is the best known fitness and v is the fitness of the individual. 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 [16]. In this section, we introduce three different IPH operators that will be analysed in this paper. Two have been already considered in the literature while the third one is newly proposed by us based on the performance of the first two.

2.1 Hamming Distance Based Linear Decrease

Zarges analysed an IPH operator where the probability of flipping each bit increases linearly with the Hamming distance to the optimum (or its best available estimate) [26]. Precisely, this operator flips each bit with probability $\min\{H(x, best)\}/n$ where n is the size of the problem and $\min\{H(x, best)\}$ is the minimum Hamming distance of the current point to a best individual. Here, we consider the mutation potential version of this operator. As the expected number of bit-flips is $\min\{H(x, best)\}$ during each execution of the mutation operator, the mutation potential would be

$$M_{\text{linHD}}(x) := \min\{H(x, best)\}. \tag{1}$$

2.2 Fitness Difference Based Exponential Decrease

In Clonalg's IPH operator, the mutation rate decreases as an exponential function of the fitness of the current solution [10]. 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 only maximisation problems, we use $v = \frac{f(x)}{f(best)}$ as suggested by [26] where $best$ is the best known fitness value. Using this mutation operator as a mutation potential gives $M = n \cdot e^{-\rho \cdot v}$. According to both practical and theoretical results [26], a reasonable value for ρ is $\ln n$. We call this mutation potential $\text{M}_{\text{expoF(x)}}$ and define it as

$$M_{\text{expoF(x)}}(x) := \left\lfloor n^{1-\frac{f(x)}{f(best)}} \right\rfloor. \tag{2}$$

2.3 Hamming Distance Based Exponential Decrease

Since it is well understood that using differences in fitness values makes randomised search heuristics unstable towards the scaling of fitness functions [17, 20, 21, 25], we also consider a mutation potential which is similar to $\text{M}_{\text{expoF(x)}}$ 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 $\text{M}_{\text{expoHD}} = n \cdot e^{-\rho \frac{n - H(x, best)}{n}}$ where n is the maximum Hamming distance from any search point to the optimum (which is always at most n), $H(x, best)$ 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

$$M_{\text{expoHD}}(x) := \left\lfloor n^{\frac{H(x, opt)}{n}} \right\rfloor. \tag{3}$$

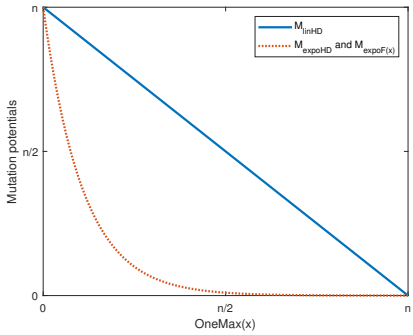

Figure 1: Different mutation potentials for OneMax.

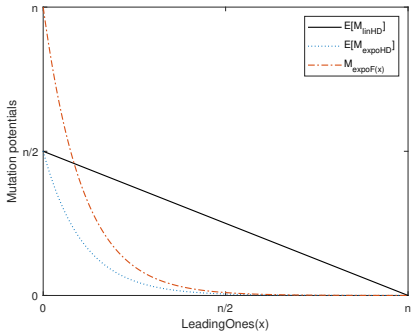

Figure 2: Mutation potentials for LeadingOnes. Expected values are shown for those using Hamming distance.

3 IDEAL BEHAVIOUR ANALYSIS

In this section, we evaluate the performance of the different inversely proportional mutation potentials, 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^{hyp} using the IPHs on the OneMax and LeadingOnes unimodal benchmark functions for which the performance of the same algorithm using static mutation potentials is known [3]. 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. On the other hand, LeadingOnes is a more complicated unimodal problem which counts the consecutive number of 1-bits at the beginning of the bit string before the first 0-bit.

The pseudo-code of the (1 + 1) IA^{hyp} is given in Algorithm 1. It simply uses one candidate solution in each iteration to which inversely proportional hypermutations are applied. The results proven in this section and comparisons with static hypermutations are summarised in Table 1. As previously mentioned, a *constructive mutation* in Algorithm 1 is a mutation step where the evaluated bit string is at least as fit as its parent.

Table 1: Comparison of the expected runtimes obtained by different hypermutation schemes for OneMax and LeadingOnes. The original IPH operator of Opt-IA [8] has the same expected runtimes as the $(1+1)$ IA^{hyp} using static hypermutations [16].

Algorithms	OneMax	LeadingOnes
$(1+1)$ IA^{hyp} using M_{linHD}	$\Theta(n^2)$	$\Theta(n^3)$
$(1+1)$ IA^{hyp} using $M_{\text{expoF(x)}}$	$O(n^{(3/2)+\epsilon}\log n)$, $\Omega(n^{3/2-2\epsilon})$	$O(n^3/\log n)$, $\Omega(n^{5/2+\epsilon})$
$(1+1)$ IA^{hyp} using M_{expoHD}	$O(n^{(3/2)+\epsilon}\log n)$, $\Omega(n^{3/2-2\epsilon})$	$O(n^{\frac{5/2+\epsilon}{\ln n}})$, $\Omega(n^{9/4-\epsilon})$
$(1+1)$ IA^{hyp} with $M=n$ [3]	$\Theta(n^2\log n)$ [3]	$\Theta(n^3)$ [3]

Algorithm 1 $(1+1)$ IA^{hyp}

1: Initialise $x \in \{0,1\}^n$ uniformly at random;
2: **while** termination condition is not satisfied **do**
3: create y by flipping at most $M := IPM(x)$ distinct bits of x selected uniformly at random one after another until a *constructive mutation* happens;
4: **if** $f(y) \geq f(x)$ **then**
5: $x := y$.

Figures 1 and 2 show how the studied mutation potentials ideally decrease during the run of the algorithm when optimising OneMax and LeadingOnes, respectively.

3.1 OneMax

The following theorems derive the expected runtimes of the three variants of IPM for OneMax$(x) := \sum_{i=1}^{n} x_i$. 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^{hyp} compared to the expected runtime achieved by the same algorithm using static hypermutation potentials.

Theorem 3.1. *The $(1+1)$ IA^{hyp} using 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, $H(x,opt) = i$ fitness function evaluations would be wasted, the total expected time to optimise OneMax is at most $\sum_{i=1}^{n} \frac{n}{i} \cdot i = O(n^2)$.

In order to prove a lower bound, we use the Ballot theorem [7, 12]. 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. Its proof deviates from the proof of Theorem 3.2 only in the amount of wasted evaluations. Note that for OneMax the Hamming distance of a solution to the optimum and its difference in fitness are the same. The proof is omitted due to lack of space.

Theorem 3.2. *The $(1+1)$ IA^{hyp} using either $M_{expoF(x)}$ or M_{expoHD} optimises OneMax in $O(n^{3/2+\epsilon}\log n)$ and $\Omega(n^{3/2-\epsilon})$ expected fitness function evaluations for any arbitrarily small constant $\epsilon > 0$.*

3.2 LeadingOnes

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

Theorem 3.3. *The $(1+1)$ IA^{hyp} using M_{linHD} optimises LeadingOnes in $\Theta(n^3)$ expected fitness function evaluations.*

Proof. The probability of improvement in each step is at least $1/n$ which is the probability of flipping the leftmost 0-bit. As at most n improvements are needed and each failure in improvement yields $\frac{n-i-1}{2} + 1 + \epsilon n$ wasted fitness function evaluations in expectation (i.e., the expected $H(x,opt)$ value), the expected time to find the optimum is at most $\sum_{i=1}^{n} n \cdot \left(\frac{n-i-1}{2} + 1 + \epsilon n\right) = O(n^3)$.

The proof for the lower bound is the same as the proof of Theorem 3.6 in [3] for the expected runtime of the $(1+1)$ IA$^{hyp}_{\geq}$ with the exception that the amount of wasted fitness function evaluations in case of failure is $H(x,opt)$ instead of n now[5].

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 $n-i-1$ other bits which are distributed uniformly at random [11]. 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 $E(f_i|E_3) = 1 + E(f_i)$. Knowing that the bits

[5] $(1+1)$ IA$^{hyp}_{\geq}$ is a simple (1+1) EA algorithm using static hypermutation operator (i.e., $M = n$) instead of standard bit mutation. Its selection mechanism accepts equally fit solutions.

after the left most 0-bit are distributed uniformly at random [11], the value of $H(x, opt)$ is not less than $1+\frac{(n-i-1)}{2}-\epsilon n$ w.o.p. by Chernoff bounds. Hence, by the law of total expectation, the expected number of fitness function evaluations for every improvement is $E(f_i) = \frac{i}{n}\left(1+\frac{n-i-1}{2}-\epsilon n+E(f_i)\right)+\frac{1}{n}\cdot 1+\frac{n-i-1}{n}(1+E(f_i))$. Solving the equation for $E(f_i)$ gives us $E(f_i) = \frac{in-i^2-i-i\epsilon n}{2}$.

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 [11]). 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 $1-2^{-n/2}$, the initial solution does not have more than $n/2$ leading 1-bits. Thus, we obtain a lower bound of $(1-2^{-n/2})\sum_{i=n/2}^{n} E(f_i) = \Omega(1)\sum_{i=n/2}^{n}\frac{in-i^2-i-i\epsilon n}{2} = \Omega(n^3)$ on the expected number of fitness function evaluations. □

Theorem 3.5 shows that exponential fitness-based mutation potential gives us 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 upcoming theorems.

Lemma 3.4. *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 3.5. *The $(1+1)$ IA^{hyp} using $M_{expoF(x)}$ 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 3.3, the expected number of leading 1-bits is less than 1 in the initialised bit string. The probability of improvement in the first step is $1/n$. In case of failure in improving at the first step, at most $n^{\frac{n-i}{n}}$ fitness function evaluations would get wasted where i is the number of leading 1-bits. Therefore, the total expected time to find the optimum is $E(T) \le \sum_{i=1}^{n-1} n\cdot n^{(n-i)/n} = O(n^3/\log n)$ considering that $\sum_{i=1}^{n-1} n^{i/n} = \sum_{i=2}^{n}(n^{1/n})^{i-1}$ which is less than $\sum_{i=1}^{n}(n^{1/n})^{i-1} = \frac{1-n}{1-n^{1/n}}$ that gets in turn lower bounded by $n^2/(\log n)$ using Lemma 3.4.

The proof for the lower bound is similar to the proof of Theorem 3.3, except in 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 are provided by the following theorem for LeadingOnes. The reason can be seen in Fig. 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 less fitness evaluations than $M_{expoF(x)}$. The proof of the following theorem is similar to that of the two previous theorems and is omitted due to lack of space.

Theorem 3.6. *The $(1+1)$ IA^{hyp} using 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 $\epsilon > 0$.*

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

4 INEFFICIENT BEHAVIOUR 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 found solution will be used by the operator rather than the unknown optimum. We combine M_{expoHD} with hybrid ageing, as in the Opt-IA AIS [8], and call it (1 + 1) Opt-IA . Its pseudo-code is provided in Algorithm 2. 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.

Our aim is that the the greater the number of local optima that are identified by the algorithm, the better M_{expoHD} approximates its ideal behaviour. However, we will show that this is not the case, e.g., for the well-studied bimodal benchmark function TwoMax (Eq. **??**) [13, 19, 23, 24]: TwoMax := $\max\left\{\sum_{i=1}^{n} x_i,\ n-\sum_{i=1}^{n} x_i\right\}$. TwoMax is usually used to evaluate the global exploration capabilities of evolutionary algorithms, i.e., whether the population identifies both optima of the function. Our analysis shows that once the (1 + 1) Opt-IA 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 escape from local optima. We will use the well known Cliff function (Eq. 4) for this purpose where static hypermutations are inefficient. Due to the high mutation rates of static hypermutations, even if they escape the local optima of Cliff, they jump back with high probability.

$$\textsc{Cliff}_k(x) := \begin{cases} \sum_{i=1}^{n} x_i & \text{if } \sum_{i=1}^{n} x_i \le n-k, \\ \sum_{i=1}^{n} x_i - k + 1/2 & \text{otherwise.} \end{cases} \quad (4)$$

Both functions are illustrated in Fig 3.

Among the different variants of ageing, *hybrid* ageing has been shown to be very efficient at escaping local optima [3, 4, 18]. This operator assigns an initial *age* = 0 to individuals. During each iteration, the age increases by 1 and is passed to the offspring

Algorithm 2 (1 + 1) Opt-IA with $\mathrm{M}_{\mathrm{expoHD}}$

Initialise $x \in \{0,1\}^n$ uniformly at random;
set $x.age := 0$ and $best := x$;
while termination condition is not satisfied **do**
  $x.age := x.age + 1$;
  create y by flipping at most $M := \mathrm{M}_{\mathrm{expoHD}}(x)$ distinct bits of x selected uniformly at random one after another until a *constructive mutation* happens;
  if $f(y) > f(x)$ **then**
    $y.age := 0$;
  else
    $y.age := x.age$;
  if $f(y) \geq f(best)$ **then**
    then set $best := y$;
  for $w \in \{x, y\}$ **do**
    if $w.age \geq \tau$ **then**
      with probability $1/2$, reinitialise w uniformly at random with $w.age = 0$;
  Set $x = \arg\max_{z \in \{x,y\}} f(z)$;

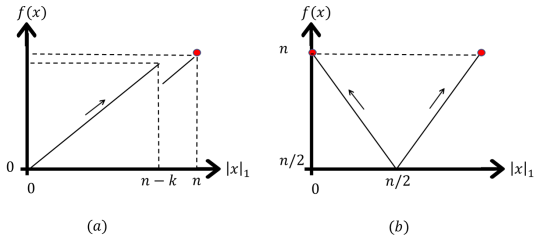

Figure 3: (a) Cliff and (b) TwoMax test functions. $|x|_1$ shows the number of 1-bits in the bit string.

if the offspring 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.

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 4.1. *The expected runtime of* (1+1) *Opt-IA with* M_{expoHD} *for optimising TwoMax is* $O(n^2 \log n)$ *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 $M = \mathrm{M}_{\mathrm{expoHD}} = n^{H(x_1,x_1)/n} = n^0 = 1$. 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} 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 is at most $O(n \log n)$ with similar proof to that of the standard (1+1) RLS_1[6]. 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 in $\tau + n$ steps. Now, the current *best* 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. the 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 3.2 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 in 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 in the second branch with fitness value $n - k$. Since the $\mathrm{M}_{\mathrm{expoHD}}$ is always smaller than $\mathrm{M}_{\mathrm{linHD}}$, 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 position 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

[6] (1 + 1) RLS_1 or random local search flips exactly one bit at each iteration.

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 $\sum_{i=1}^{n/2+\epsilon n} \frac{n}{i} \cdot n^{\frac{n-i}{n}} = O(n^2 \log n)$. □

Now we show that differently from static hypermutations, M_{expoHD} combined with ageing can escape from the local optima of Cliff, hence optimises the function efficiently.

Theorem 4.2. *The $(1+1)$ Opt-IA with M_{expoHD} and $\tau = \Omega(n^{1+\epsilon})$ for an arbitrarily small constant ϵ, optimises Cliff_k with $k < n(\frac{1}{4} - \epsilon)$ and $k = \Theta(n)$ in $O(n^{3/2} \log n + \tau n^{1/2} + \frac{n^{7/2}}{k^2})$ expected fitness function evaluations.*

Proof. The analysis will follow a similar idea to the proof of Theorem 4.1. 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 $O(n)$ to find a local optimum of the cliff (i.e., a search point with $n - k$ 1-bits) as the improvement probability is always at least $(n-k)/n = \Omega(1)$. 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 itself until the following event happens.

If the local optima produces an offspring with $n - k + 1$ bits with probability k/n and if this solution survives the ageing operator (with probability $(1 - p_{die})$), then the reinitialised solution will be rejected since its fitness value is less than $n - k$ w.o.p. 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 $(k-1)/n$) 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 $n - k$ 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. As shown in the proof of Theorem 3.1, the Ballot theorem [7, 12]] 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 $2k$ since both the local optima and the current solution have less than k 0-bits. Since $k < n/4$, the mutation potential is at most $\sqrt{n}$. Thus, no hypermutation can yield a solution with less than $n - k + 2$ 1-bits. Therefore, once a solution with $n - k + 2$ 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 $n - k + 2$ 1-bits at the end of each cycle of reinitialisation and removal of the local optima due to ageing is $(1 - p_{die})^2 \cdot (k/n) \cdot ((k-1)/n)$. Since each such cycle takes $O(n^{3/2})$ fitness evaluations, our claim follows. □

5 AN EFFICIENT OPT-IA WITH IPH

In the previous section, we observed in the analyses of both Cliff and TwoMax 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 mutation potential 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.

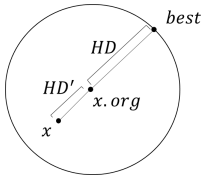

Figure 4: The geometric representation of Symmetric M_{expoHD}. The mutation potential is determined according to the ratio of the Hamming distance between the current solution (x) and its origin ($x.org$) and the Hamming distance between its origin and the best seen solution.

Algorithm 3 (1+1) Opt-IA with Symmetric M_{expoHD}

Initialise $x \in \{0,1\}^n$ uniformly at random;
set $x.origin := x$, $x.age := 0$, and $best := x$;
while termination condition not satisfied **do**
  $x.age := x.age + 1$;
  Create y by flipping at most $M :=$ Symmetric $M_{expoHD}(x)$ distinct bits of x selected uniformly at random one after another until a *constructive mutation* happens;
  $y.origin := x.origin$;
  if $f(y) > f(x)$ **then**
    $y.age := 0$;
    if $f(y) \geq best$ **then**
      $best := y$;
  else
    $y.age := x.age$;
  for $w \in \{x, y\}$ **do**
    if $w.age \geq \tau$ **then**
      with probability $1/2$, reinitialise w uniformly at random with $w.age = 0$;
      set $w.origin = w$;
  Set $x = \arg\max_{z \in \{x,y\}} f(z)$;

Algorithm 3 uses a mutation potential inversely proportional to the current solution's Hamming distance to its origin, where the origin is defined as the ancestor of the current bitstring after the last removal of a solution due to ageing (Fig. 4). We call the newly proposed mutation potential *Symmetric M_{expoHD}* and define it as

$$\text{Symmetric } M_{expoHD} := \max\left\{\left\lfloor n^{\frac{H(best,x.org)-H(x,x.org)}{n}}\right\rfloor, 1\right\}. \quad (5)$$

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 bitstring.

The following theorem shows that once one optimum of TwoMax has been identified, the value of the Symmetric M_{expoHD} potential decreases as both optima are approached as desired and the wished for speed-up in the runtime is achieved.

Theorem 5.1. *The* (1 + 1) *Opt-IA using 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 [22]. □

The following theorem shows that Symmetric M_{expoHD} is also efficient for Cliff.

Theorem 5.2. *The* (1 + 1) *Opt-IA using Symmetric M_{expoHD} with $\tau = \Omega(n^{1+\epsilon})$ optimises Cliff_k with $k < n(\frac{1}{4} - \epsilon)$ and $k = \Theta(n)$ in $O(n^{3/2}\log n + \tau n^{1/2} + \frac{n^{7/2}}{k^2})$ expected fitness function evaluations.*

Proof. The proof is almost identical to the proof of Theorem 4.2. The most important distinction is that once a solution with $n-k+2$ 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. □

6 CONCLUSION

We have presented an analysis of inversely proportional hypermutations (IPH). Previous theoretical studies have shown disappointing results concerning the IPH operators from the literature. In this paper we proposed a new IPH operator 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. Future work should evaluate the performance of the proposed algorithm for combinatorial optimisation problems with practical applications.

Acknowledgments: This work has received funding from the EPSRC under grant agreement number EP/M004252/1.

REFERENCES

[1] Jason Brownlee. 2007. *Clonal Selection Algorithms.* Technical Report. Complex Intelligent Systems Laboratory (CIS), Swinburne University of Technology.

[2] Frank M. Burnet. 1959. *The Clonal Selection Theory of Acquired Immunity.* Cambridge University Press.

[3] Dogan Corus, Pietro S. Oliveto, and Donya Yazdani. 2017. On the Runtime Analysis of the Opt-IA Artificial Immune System. In *Proc. of GECCO 2017.* 83–90.

[4] Dogan Corus, Pietro S. Oliveto, and Donya Yazdani. 2018. Artificial Immune Systems Can Find Arbitrarily Good Approximations for the NP-hard Partition Problem. In *Proc. of PPSN 2018.* 16–28.

[5] Dogan Corus, Pietro S. Oliveto, and Donya Yazdani. 2019. Artificial Immune Systems Can Find Arbitrarily Good Approximations for the NP-Hard Number Partitioning Problem. *To appear in Artificial Intelligence* (2019). https://doi.org/10.1016/j.artint.2019.03.001

[6] Dogan Corus, Pietro S. Oliveto, and Donya Yazdani. 2019. On Inversely Proportional Hypermutations with Mutation Potential. In *arXiv.* –. https://arxiv.org/abs/1903.11674.

[7] Dogan Corus, Pietro S. Oliveto, and Donya Yazdani. 2019. When Hypermutations and Ageing Enable Artificial Immune Systems to Outperform Evolutionary Algorithms. *To appear in Theor. Comp. Sci.* (2019). https://doi.org/10.1016/j.tcs.2019.03.002

[8] Vincenzo Cutello, Giuseppe Nicosia, Mario Pavone, and Jonathan Timmis. 2007. An Immune Algorithm for Protein Structure Prediction on Lattice Models. *IEEE Trans. Evol. Comp.* 11 (2007), 101–117.

[9] Leandro N. de Castro and Jonathan Timmis. 2002. *Artificial Immune Systems: A New Computational Intelligence Approach.* Springer Science and Business Media.

[10] Leonardo N. de Castro and Fernando J. Von Zuben. 2002. Learning and Optimization Using the Clonal Selection Principle. *IEEE Trans. Evol. Comp.* 6, 3 (2002), 239–251.

[11] Stefan Droste, Thomas Jansen, and Ingo Wegener. 2002. On the Analysis of the (1+1) Evolutionary Algorithm. *Theor. Comp. Sci.* 276, 1-2 (2002), 51–81.

[12] William Feller. 1968. *An Introduction to Probability Theory and Its Applications.* John Wiley & Sons.

[13] Tobias Friedrich, Pietro S. Oliveto, Dirk Sudholt, and Carsten Witt. 2009. Analysis of Diversity-Preserving Mechanisms for Global Exploration. *Evolutionary Computation* 17, 4 (2009), 455–476.

[14] Christian Horoba, Thomas Jansen, and Christine Zarges. 2009. Maximal Age in Randomized Search Heuristics with Aging. In *Proc. of GECCO 2009.* 803–810.

[15] Thomas Jansen and Christine Zarges. 2011. On the Role of Age Diversity for Effective Aging Operators. *Evolutionary Intelligence* 4, 2 (2011), 99–125.

[16] Thomas Jansen and Christine Zarges. 2011. Variation in Artificial Immune Systems: Hypermutations with Mutation Potential. In *Proc. of ICARIS 2011.* 132–145.

[17] Frank Neumann, Pietro S. Oliveto, and Carsten Witt. 2009. Theoretical Analysis of Fitness-proportional Selection: Landscapes and Efficiency. In *Proc. of GECCO 2009.* 835–842.

[18] Pietro S. Oliveto and Dirk Sudholt. 2014. On the Runtime Analysis of Stochastic Ageing Mechanisms. In *Proc. of GECCO 2014.* 113–120.

[19] Pietro S. Oliveto, Dirk Sudholt, and Christine Zarges. 2018. On the Benefits and Risks of Using Fitness Sharing for Multimodal Optimisation. *Theor. Comp. Sci.* (2018), –. https://doi.org/doi.org/10.1016/j.tcs.2018.07.007

[20] Pietro S Oliveto and Carsten Witt. 2014. On the Runtime Analysis of the Simple Genetic Algorithm. *Theor. Comp. Sci.* 545 (2014), 2–19.

[21] Pietro S. Oliveto and Carsten Witt. 2015. Improved Time Complexity Analysis of the Simple Genetic Algorithm. *Theor. Comp. Sci.* 605 (2015), 21–41.

[22] Pietro S. Oliveto and Xin Yao. 2011. Runtime Analysis of Evolutionary Algorithms for Discrete Optimization. In *Theory of Randomized Search Heuristics*, Anne Auger and Benjamin Doerr (Eds.). World Scientific, 21–52.

[23] Edgar Covantes Osuna and Dirk Sudholt. 2017. Analysis of the Clearing Diversity-preserving Mechanism. In *Proc. of FOGA 2017.* 55–63.

[24] Dirk Sudholt. 2019. The Benefits of Population Diversity in Evolutionary Algorithms: A Survey of Rigorous Runtime Analyses. In *Theory of Randomized Search Heuristics in Discrete Search Spaces*, Benjamin Doerr and Frank Neumann (Eds.). Spinger, –. https://arxiv.org/abs/1801.10087.

[25] Darrell Whitley. 1989. 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.* 116–121.

[26] Christine Zarges. 2008. Rigorous Runtime Analysis of Inversely Fitness Proportional Mutation Rates. In *Proc. of PPSN X.* 112–122.