Speeding Up Hyper-Heuristics With Markov-Chain Operator Selection and the Only-Worsening Acceptance Operator[*]

Abderrahim Bendahi[1] , **Benjamin Doerr**[2] , **Adrien Fradin**[1] and **Johannes F. Lutzeyer**[2]
[1]École Polytechnique, Institut Polytechnique de Paris
[2]Laboratoire d'Informatique (LIX), CNRS, École Polytechnique, Institut Polytechnique de Paris
{firstname.lastname}@polytechnique.edu

Abstract

The move-acceptance hyper-heuristic was recently shown to be able to leave local optima with astonishing efficiency (Lissovoi et al., Artificial Intelligence (2023)). In this work, we propose two modifications to this algorithm that demonstrate impressive performances on a large class of benchmarks including the classic CLIFF_d and JUMP_m function classes. (i) Instead of randomly choosing between the only-improving and any-move acceptance operator, we take this choice via a simple two-state Markov chain. This modification alone reduces the runtime on JUMP_m functions with gap parameter m from $\Omega(n^{2m-1})$ to $O(n^{m+1})$. (ii) We then replace the all-moves acceptance operator with the operator that only accepts worsenings. Such a, counter-intuitive, operator has not been used before in the literature. However, our proofs show that our only-worsening operator can greatly help in leaving local optima, reducing, e.g., the runtime on Jump functions to $O(n^3 \log n)$ independent of the gap size. In general, we prove a remarkably good runtime of $O(n^{k+1} \log n)$ for our Markov move-acceptance hyper-heuristic on all members of a new benchmark class SEQOPT_k, which contains a large number of functions having k successive local optima, and which contains the commonly studied JUMP_m and CLIFF_d functions for $k = 2$.

1 Introduction

Selection hyper-heuristics are black-box optimization heuristics that function by combining different low-level heuristics. They were first used to solve difficult scheduling problems [Cowling *et al.*, 2000], but then quickly found numerous other applications [Burke *et al.*, 2013; Drake *et al.*, 2020].

The mathematical runtime analysis of hyper-heuristics has started around ten years ago [Lehre and Özcan, 2013], predominantly discussing the impact of selecting different variation operators [Doerr *et al.*, 2018; Lissovoi *et al.*, 2020]. More recently, hyper-heuristics having the choice between different acceptance operators were studied (although a first result can already be found in [Lehre and Özcan, 2013]). In particular, [Lissovoi *et al.*, 2019; Lissovoi *et al.*, 2023] have shown that switching between an elitist selection (the *only-improving operator* OI) and accepting any new solution (the *all-moves operator* AM) can give excellent results. Specifically, they show that the *move-acceptance hyper-heuristic (*MAHH*)* can optimize the CLIFF_d benchmark defined on bit-strings of length n in time $O(n^3)$, whereas comparably simple elitist evolutionary algorithm need time $\Omega(n^d)$; here d is a difficulty parameter of the benchmark that can range from 2 to n. A similar lower bound was shown for the Metropolis algorithm [Doerr *et al.*, 2023b].

However, such performance gains seem to heavily depend on the particular problem to be optimized. For the jump benchmark with difficulty parameter m, simple evolutionary algorithms find the optimum in expected time $O(n^m)$ [Droste *et al.*, 2002], but the MAHH needs $\Omega(n^{2m-1})$ [Doerr *et al.*, 2023a] (for constant m).

In this work, we propose two new ideas that help hyper-heuristics to leave local optima, and greatly improve their performance. We also observe that the proposed modifications resolve the difficulties detected in [Doerr *et al.*, 2023a]. (i) From studying the proofs in [Doerr *et al.*, 2023a], we observe that the use of the random mixing strategy, that is, using the all-moves operator in each iteration independently with some probability p, is problematic. The probability p has to be small to allow for a sufficiently strong drift towards the optimum, but leaving a local optimum with radius m requires $m - 1$ successive uses of the AM operator, which contributes a factor of p^{m-1} to the probability of successfully leaving the local optimum. To mitigate the influence of the required small rate of AM operator uses, we design a simple two-state Markov chain governing the selection of the operators. In other words, for each of the two acceptance operators, we have a switching probability. In each iteration, we use this value to decide whether we should switch to the other operator or continue with the current operator. By taking a value such as $1/2$ for the probability of switching away from the AM operator, longer stretches of using this operator become more likely, which eases the leaving of local optima with larger basins of attraction. We call the resulting algorithm *Markov move-acceptance hyper-heuristic (*MMAHH*)*. As an example of the usefulness of this approach, we show that

[*]An extended version of this manuscript, including appendices, can be accessed on the arXiv repository.

the MMAHH choosing the two operators OI and AM (with the long-term rates of the operators as in previous works) optimizes JUMP functions in time $O(n^{m+1})$, a considerable speed-up from $\Omega(n^{2m-1})$. We note that in general this way of choosing between operators has been used before, but to the best of our knowledge no mathematical runtime result has been shown, and it has not been used in conjunction with acceptance operators until now.

We then propose the only-worsening (OW) acceptance operator. It accepts the new solution only if it is strictly worse than the parent. While this operator contradicts the idea of incremental optimization, in our context it is less counter-intuitive than what appears at first. We recall that the main working principle of the AM operator exploited in previous works is that it allows the algorithm to leave local optima. For this aim, searching for inferior solutions is, in fact, a logical approach. This intuitive consideration is supported by our mathematical runtime analysis, which in particular shows that the MMAHH with the two acceptance operators OI and OW optimizes any JUMP and CLIFF function in expected time $O(n^3 \log n)$. We extend this result to a new benchmark called SEQOPT_k, which consists of a broad class of pseudo-Boolean functions having k local optima that in particular includes the classic ONEMAX, JUMP, CLIFF, and TRAP benchmarks. We show that our MMAHH optimizes any function in SEQOPT_k in expected time $O(n^{k+1} \log n)$.

With this work, we introduce two new ideas for the design of effective move-acceptance hyper-heuristics. We believe that in particular, the only-worsening operator to leave local optima, could give rise to further theoretical study and possibly the design of new benchmarks designed to test its limits. We furthermore hope that future consideration of our SEQOPT_k benchmark enables a joint observation of the effectiveness of hyper-heuristics that subsumes results on the different commonly studied JUMP and CLIFF benchmarks.

2 Preliminaries

In this section, we briefly recall the classic benchmark problems relevant for this work, define our new benchmark SEQOPT_k, define our Markov move-acceptance hyper-heuristic, the only-worsening acceptance operator, and provide the tools to analyze our hyper-heuristics.

2.1 Benchmarks

As standard in the theory of randomized search heuristics [Neumann and Witt, 2010; Auger and Doerr, 2011; Jansen, 2013; Zhou *et al.*, 2019; Doerr and Neumann, 2020], we regard pseudo-Boolean optimization problems, that is, we aim at maximizing functions f that map bit-strings $x \in \{0,1\}^n$ with a fixed positive length $n \in \mathbb{N}_{>0} = \{1,2,3,\ldots\}$ to a numerical value $f(x) \in \mathbb{R}$. When using asymptotic notation, this shall always be for $n \to \infty$.

Well-known examples of such functions in the theory literature include the following ONEMAX, TRAP, CLIFF_d and JUMP_m benchmarks.

For a bit-string $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, let $\|x\|_1 = x_1 + x_2 + \cdots + x_n$ denote the number of ones in x. The following functions are standard benchmarks in the theory of randomized search heuristics.

$$\mathrm{ONEMAX}\colon x \mapsto \|x\|_1;$$

$$\mathrm{CLIFF}_d\colon x \mapsto \begin{cases} \|x\|_1, & \text{if } \|x\|_1 \le n-d; \\ \|x\|_1 - d + \frac{1}{2}, & \text{otherwise;} \end{cases}$$

$$\mathrm{JUMP}_m\colon x \mapsto \begin{cases} m + \|x\|_1, & \text{if } \|x\|_1 \in [0..n-m] \cup \{n\}; \\ n - \|x\|_1, & \text{otherwise.} \end{cases}$$

Here $d \in [1..n-1]$ and $m \in [1..n]$ are difficulty parameters of the CLIFF and JUMP benchmark. The completely deceptive function JUMP_n is also called TRAP. All these functions have $x^* = (1, \ldots, 1)$ as unique global optimum (maximum), and all are *functions of unitation*, that is, the objective value of a solution depends only on the number of ones. This motivates the definition of the *k-th layer* as

$$\mathcal{L}_k := \{x \in \{0,1\}^n \mid \mathrm{H}(x, x^*) = n - \|x\|_1 = k\}$$

for $k \in [0..n]$, where we used $\mathrm{H}(\cdot,\cdot)$ to denote the Hamming distance of two bit-strings. Note that $\mathcal{L}_k$ is the set of all bit-strings at distance k from the global maximum x^*, that is, the numbering starts at the global optimum.

All benchmarks above also have the property that they are composed of intervals of layers in which the function is only increasing or only decreasing; for JUMP_m and CLIFF_d these intervals have lengths 2, $m-1$, and $n-m+1$. In the following two definitions, we extend this property to arbitrary interval numbers and lengths and obtain the very general benchmark SEQOPT_k, having k successive local minima and maxima.

Definition 1 (Monotonicity across layers)**.** *Let $h \in [0..n-1]$ and $f\colon \{0,1\}^n \to \mathbb{R}$. We say that f is increasing (resp. decreasing) between layers $\mathcal{L}_{h+1}$ and $\mathcal{L}_h$ if for any $x \in \mathcal{L}_h$ and $y \in \mathcal{L}_{h+1}$ we have*

$$f(x) > f(y) \textit{ (resp. } f(x) < f(y)\textit{).}$$

We denote this by $\mathcal{L}_h \overset{f}{\succ} \mathcal{L}_{h+1}$ (resp. $\mathcal{L}_h \overset{f}{\prec} \mathcal{L}_{h+1}$).

Definition 2 (The SEQOPT benchmark)**.** *Let $n \ge 2$, $k \in [0..n-2]$ and $d_0 = n > d_1 > d_2 > \cdots > d_k > d_{k+1} = 0$ be integers. We define $\mathrm{SEQOPT}_k(d_1, \ldots, d_k)$ to be the set of all functions $f\colon \{0,1\}^n \to \mathbb{R}$ such that*

(i) *$x^* = (1, \ldots, 1)$ is the unique global maximum of f,*

(ii) *for any $\ell \in [0..k]$, if $k - \ell$ is even then*

$$\mathcal{L}_{d_\ell} \overset{f}{\prec} \cdots \overset{f}{\prec} \mathcal{L}_{d_{\ell+1}},$$

and if $k - \ell$ is odd, f satisfies

$$\mathcal{L}_{d_\ell} \overset{f}{\succ} \cdots \overset{f}{\succ} \mathcal{L}_{d_{\ell+1}}.$$

The union of these classes of functions, for fixed k, will be denoted by

$$\mathrm{SEQOPT}_k = \bigcup_{n > d_1 > \cdots > d_k > 0} \mathrm{SEQOPT}_k(d_1, \ldots, d_k).$$

Note that we have ONEMAX $\in \mathrm{SEQOPT}_0$, TRAP $\in \mathrm{SEQOPT}_1(1)$, $\mathrm{CLIFF}_d \in \mathrm{SEQOPT}_2(d, d-1)$ and, for $m < n$, $\mathrm{JUMP}_m \in \mathrm{SEQOPT}_2(m, 1)$.

$$\begin{tikzcd}
\text{OI} \arrow[loop left, "1-p"] \arrow[r, bend right, "p"'] & \text{OW} \arrow[loop right, "1-q"] \arrow[l, bend right, "q"']
\end{tikzcd}$$

Figure 1: *Transition probabilities between the two operators of the* MMAHH*, here* OI *and* OW.

2.2 The Markov Move-Acceptance Hyper-Heuristic

We now introduce a novel algorithm called the *Markov Move-Acceptance Hyper-Heuristic* algorithm (MMAHH). We recall that the Move-Acceptance Hyper-Heuristic algorithm (MAHH) first proposed in [Lehre and Özcan, 2013] and then intensively studied in [Lissovoi *et al.*, 2023] is a simple randomized local search heuristic which randomly mixes between the OI and AM operators, that is, in each iteration independently is chooses between the AM operator (with some, usually small, probability p) and the OI operator (with probability $1 - p$).

From studying the proof of the unsatisfactory $\Omega(n^{2m-1})$ runtime bound for this algorithm on the JUMP$_m$ benchmark, we learn that the main reason for this negative performance is the low probability of having $m - 1$ consecutive uses of the AM operator, which stems from the independent choice of the operators. To allow for longer phases of using the same operator, our MMAHH leverages a simple 2-state Markov chain to govern the selection of the acceptance operators. In other words, if the current operator is OI, this operator is kept for the next iteration with probability $1 - p$, but changed to AM with probability p. The switching probability from AM to OI is denoted by q. See Algorithm 1 for the pseudocode of the MMAHH, where the MARKOV operator refers to sampling the Markov chain illustrated in Figure 1.

We also introduce a new acceptance operator, only-worsening (OW), to substitute the ALLMOVES (AM) operator. This new operator works in the same fashion as the well-known ONLYIMPROVING (OI) acceptance operator except that OW only accepts worsening moves, i.e., moves decreasing the function value. The idea of this, counter-intuitive, operator is to speed-up leaving local optima. When studying the previous runtime analyses for the MAHH, we see that they profit from the AM operator in that it allows the algorithm to leave local optima. If this is the target, then the OW operator should be even better suited, and this is what we will observe in this work.

2.3 Notation for the Analysis of the MMAHH

Throughout this work, we denote by x_t, y_t and s_t, respectively, the current solution at time t, its Hamming distance to x^* (i.e., $y_t = \mathrm{H}(x_t, x^*)$), and the move-acceptance operator used at time t. We initialize with a random solution $x_0 \sim \mathcal{U}(\{0,1\}^n)$ and $s_0 = \text{OI}$. We always denote by $f\colon \{0,1\}^n \to \mathbb{R}$ the function to be maximized. Moreover, the transition probabilities from OI to OW and OW to OI will be denoted respectively by p and q with $0 < p, q < 1$.

Symbol	Meaning
T	The first hitting time of x^*
$T_s^{(k)}$	The starting time of the k-th phase
p, q	Transition probabilities of the Markov chain
Z_k	The length of the k-th phase
f	A function with a unique global maximum
x^*	The global maximum of f, $x^* = \{1\}^n$
x_t	The bit-string during the t-th iteration
y_t	The Hamming distance to x^* at time t
s_t	The move-acceptance operator at time t
x_0	The initial bit-string, $x_0 \sim \mathcal{U}(\{0,1\}^n)$
s_0	The initial move-acceptance operator, $s_0 = \text{OI}$

Table 1: Table of Notation.

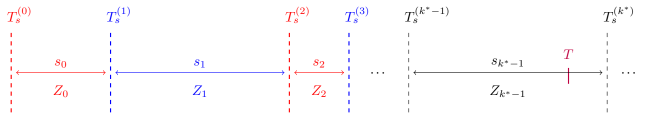

Figure 2: *A generic setting, depicting the phases (red or blue), their length $(Z_k)_{k\in\mathbb{N}}$, the switching times $(T_s^{(k)})_{k\in\mathbb{N}}$ and the stopping time T occurring during the phase k^*.*

We call $T := \inf\{t \geq 0 \mid x_t = x^*\}$ the runtime, that is, the first time we reach the global maximum. Let $T_s^{(k)}$ denotes the k-th switching time between the operators. Hence $T_s^{(0)} = 0$ and for any $k \geq 0$, we have $T_s^{(k+1)} = \inf\{t \geq T_s^{(k)} \mid s_t \neq s_{T_s^{(k)}}\}$. We also define $Z_k = T_s^{(k+1)} - T_s^{(k)}$ to be the length of the k-th *phase*, where a *phase* is a (maximum) time interval in which the same operator is used.

This notation is summarized in Table 1 and illustrated in Figure 2.

Algorithm 1: The Markov move-acceptance hyper-heuristic with the acceptance operators OI and OW.

```
1  Initialization:
2      $t \leftarrow 0$
3      $x_0 \sim \mathcal{U}(\{0,1\}^n)$, a uniformly sampled bit-string
4      $s_0 \leftarrow$ OI
5  while true do
6      $x' \leftarrow$ RAMDOMONEBITFLIP$(x_t)$
7      if $s_t =$ OW and $f(x') < f(x_t)$ then
8          $x_{t+1} \leftarrow x'$
9      else if $s_t =$ OI and $f(x_t) < f(x')$ then
10         $x_{t+1} \leftarrow x'$
11     $s_{t+1} \leftarrow$ MARKOV$(s_t)$
12     $t \leftarrow t + 1$
```

2.4 Tools for the Analysis of the MMAHH

We now develop the tools we need to analyze the MMAHH. Unfortunately, the generality of the Markov chain of our setup disallows the use of the methods previously employed in the analysis of the MAHH, which were mostly drift arguments based on the fitness of the current solution. Nonetheless, with some mathematical effort, we manage to obtain very precise estimates of the quantities of interest.

Probability of improvement in one phase. We start by computing the probabilities that a single phase of OI usages leads to a certain fitness improvement. From the way we constructed the Markov chain, the length Z_k of each phase is independent from the solution $x_{T_s^{(k)}}$ it starts with (and hence its fitness $y_{T_s^{(k)}}$), and follows a geometric law.

For $0 \leq h, k \leq n$, let $p_k^h = \Pr[y_{T_s^{(1)}} \leq h \mid y_0 = k, s_0 = \mathrm{OI}]$ denote the probability to reach $\mathcal{L}_h$ in one OI phase when starting in state OI in $\mathcal{L}_k$, assuming that we optimize the OneMax benchmark. The following computation of these probabilities is a cornerstone of our analysis.

Lemma 3. *For all $0 < p < 1$ and $0 \leq h, k \leq n$, we have*

$$p_k^h = \begin{cases} 1, & \text{if } k \leq h; \\ \frac{1}{1-p} \frac{\Gamma(k+1)\Gamma\left(\frac{np}{1-p}+h+1\right)}{\Gamma(h+1)\Gamma\left(\frac{np}{1-p}+k+1\right)}, & \text{otherwise.} \end{cases}$$

The next key lemma shows that $p = \Theta(\frac{1}{n \log(n)})$, that is, a quasi-linear length of the OI phase, suffices to have a constant probability p_n^0 to optimize OneMax in one OI phase even when starting in the all-zero string.

Lemma 4. *Let $c > 0$ be a constant and let $p = \frac{1}{cn \log(n)}$. Then $p_n^0 = (1 + o(1))e^{-1/c}$.*

From a simple domination argument, exploiting that our hyper-heuristics have to visit all intermediate levels, we immediately obtain the following minimality statement.

Lemma 5 (Minimality of p_n^0)**.** *For all $0 \leq h, k \leq n$, we have $p_k^h \geq p_n^0$.*

As OI and OW operators work in a symmetric fashion, similar results hold on OW and decreasing moves. More specifically, given $0 \leq h, k \leq n$, we have

$$\Pr[y_{T_s^{(1)}} \geq n - h \mid y_0 = n - k, s_0 = \mathrm{OW}] = \begin{cases} 1, & \text{if } k \leq h; \\ \frac{1}{1-q} \frac{\Gamma(k+1)\Gamma\left(\frac{nq}{1-q}+h+1\right)}{\Gamma(h+1)\Gamma\left(\frac{nq}{1-q}+k+1\right)}, & \text{otherwise.} \end{cases}$$

In particular, when $q = \Theta(\frac{1}{n \log(n)})$, the probability to climb down OneMax entirely in one phase of OW is $\Omega(1)$.

We can further extend Lemma 3 to the SEQOPT benchmark. Note that for any k and $f \in \mathrm{SEQOPT}_k$, for each ℓ the transition probability between layer $\mathcal{L}_\ell$ and $\mathcal{L}_{\ell-1}$ (resp. $\mathcal{L}_\ell$ and $\mathcal{L}_{\ell+1}$, when defined), are the same as for OneMax, namely, $\frac{\ell}{n}$ (resp. $\frac{n-\ell}{n}$). Thus, the computations done in Lemma 3 still hold for f in regions where f is monotonic across the layers, as defined in Definition 1.

We further note that, for OneMax again, the average number of pairs of phases of OI followed by OW needed to reach the optimum is bounded from above by $\frac{1}{p_n^0}$, since in the worst scenario, every phase of OW bring us back to the all-zero string. Extending this insight again to SEQOPT, we see that in the quasi-linear regime, reaching a neighboring local optimum starting from a local optimum of any SEQOPT function only takes $O(1)$ pairs of phases. This particular fact will be referred as to the *one-phase approximation*, stated more formally in the next lemma.

Lemma 6 (One-phase approximation)**.** *Let $p = \Theta(\frac{1}{n \log(n)})$ and $q = \Theta(\frac{1}{n \log(n)})$. Let $f \in \mathrm{SEQOPT}_k(d_1, \ldots, d_k)$, $\ell \in [0..k]$ and $x_0 \in \mathcal{L}_{d_\ell}$. Then the MMAHH algorithm reaches some $x \in \mathcal{L}_{d_{\ell-1}} \cup \mathcal{L}_{d_{\ell+1}}$ in $O(1)$ phases on average.*

3 MMAHH With OI+AM on Jump

The core objective of this section is to prove that the sole use of the Markov chain improves significantly the performance of the MAHH on Jump_m, reducing its average runtime from $\Omega\left(\frac{n^{2m-1}}{(2m-1)!}\right)$ to $O(n^{m+1})$ as outlined in the next theorem.

To clearly point out the contribution of the Markov chain, we consider the MMAHH on Jump_m where $1 < m < \frac{n}{2}$ using the OI and AM operators, with probability p and q to choose them respectively (we just replace OW in Figure 1 by AM).

Theorem 7 (Runtime analysis of the MMAHH on Jump_m with OI-AM)**.** *The time T taken by the MMAHH on Jump_m with $1 < m < \frac{n}{2}$ to reach x^*, using OI and AM, satisfies*

$$\mathrm{E}[T] = O\left((1 + pn)\left(\frac{1}{p} + \frac{1}{q}\right)N_{n,m,q}\right),$$

where $N_{n,m,q} = n + \frac{n^m}{(m-1)!(1-q)^{m-2}}$, $n \geq 3$ and p, q are such that $\frac{m}{2(n-2m)}\left(q + \frac{4}{n}\right) \geq p$.

Notably, it suffices to have $\frac{m}{2n}q \geq p$ from where we have:

(i) When $p = \frac{m}{cn}q$ for some $c \geq 2$ and $q = \frac{1}{2}$ then

$$\mathrm{E}[T] = O\left(n^2 + \frac{2^{m-2}n^{m+1}}{(m-1)!}\right) = O(n^{m+1}).$$

(ii) When $p = \frac{m}{cn}q$ and $q = \frac{1}{dm}$ where $c \geq 2$ and $d \geq 1$ then

$$\mathrm{E}[T] = O\left(n^2 + \frac{e^{1/d}\, n^{m+1}}{(m-1)!}\right) = O\left(\frac{n^{m+1}}{(m-1)!}\right).$$

The proof of the theorem relies mainly on drift theorems. Hence we start by studying the drift in the following lemmas. Throughout this part, the drift over a phase of AM (resp. OI) starting in $i \in [0..n]$ (denoting the number of zero bits) will be Δ_i^{AM} (resp. Δ_i^{OI}) and defined as

$$\Delta_i^{\mathrm{AM}} = \mathrm{E}\left[y_{T_s^{(2k)}} - y_{T_s^{(2k+1)}} \mid y_{T_s^{(2k)}} = i, s_{T_s^{(2k)}} = \mathrm{AM}\right].$$

Lemmas 8, 9 and 10 give the drift over a phase of AM, a phase of OI and a pair AM+OI of phases.

Lemma 8 (Drift over a phase of AM)**.** *Let $i \in [0..n]$ and an integer $k \geq 0$, the drift over a phase of AM on OneMax is*

$$\mathrm{E}[y_{T_s^{(2k)}} - y_{T_s^{(2k+1)}} \mid y_{T_s^{(2k)}} = i, s_{T_s^{(2k)}} = \mathrm{AM}] = \frac{2i - n}{2 + q(n-2)}.$$

Lemma 9 (Drift over a phase of OI)**.** *Let $i \in [0..n]$ and an integer $k \geq 0$, the drift over a phase of* OI *on* ONEMAX *is*

$$\mathrm{E}[y_{T_s^{(2k)}} - y_{T_s^{(2k+1)}} \mid y_{T_s^{(2k)}} = i, s_{T_s^{(2k)}} = \mathrm{OI}] = \frac{i}{1+p(n-1)}.$$

Lemma 10 (Drift over a pair AM+OI of phases)**.** *For any $i \in [0..n]$ and $k \geq 0$ an integer, the drift over a phase of* AM *followed by a phase of* OI *on* ONEMAX *is*

$$\mathrm{E}[y_{T_s^{(2k)}} - y_{T_s^{(2k+2)}} \mid y_{T_s^{(2k)}} = i, s_{T_s^{(2k)}} = \mathrm{AM}] = \Delta_i^{\mathrm{AM}} + \Delta_{\left(i-\Delta_i^{\mathrm{AM}}\right)}^{\mathrm{OI}},$$

i.e., we can split the drift across two phases and plug the average position after a phase of AM *directly in the drift of a phase of* OI*.*

Now, the following lemma provides an upper bound on the average number of phases of AM needed to reach the global maximum at $x^* = \{1\}^n$ from a local maximum in layer $\mathcal{L}_m$.

Lemma 11. *The probability to reach the global maximum x^* during a single phase of* AM *starting from a local maximum in layer $\mathcal{L}_m$ is*

$$\Pr[x_{T_s^{(1)}} = x^* \mid x_0 \in \mathcal{L}_m, s_0 = \mathrm{AM}] \geq (1-q)^{m-2}\frac{m!}{n^m}.$$

Let

$$X^* = \{x \in \{0,1\}^n \mid \|x\|_1 \in \{n-m, n\}\},$$

the set of (local and global) maxima of $\textsc{Jump}_m$ and we consider the potential

$$d\colon x \mapsto \begin{cases} |n-m-\|x\|_1|, & \text{if } x \neq x^*; \\ 0, & \text{otherwise.} \end{cases}$$

In the next two lemmas, we work out lower bounds on the drift of the bit-string sequence, distinguishing two cases based on the landscape of the $\textsc{Jump}_m$ function. In Lemma 12, we lower bound the drift in the gap region, i.e., at positions $x \in \{0,1\}^n$ such that $n-m < \|x\|_1 < n$, directly on the sequence $(x_t)_{t\geq 0}$ using the potential d defined earlier. This result will provide an upper bound on the average time needed to climb towards x^* from a point x_0 in the gap region. On the other hand, Lemma 13 deals with the drift in the region with the ONEMAX-like slope, i.e., bit-strings $x \in \{0,1\}^n$ for which $0 \leq \|x\|_1 < n-m$. This time, we work at the scale of a pair of phases AM+OI since locally, depending on the current operator OI or AM, the drift might have opposite signs, notably in the region $[\frac{n}{2}..(n-m)]$. To this aim, we will use Lemma 10 along with the additive drift theorem with overshooting [Kötzing and Krejca, 2019] to upper bound the expected time to climb this left slope from an initial point x_0.

Lemma 12 (Drift in the gap region)**.** *Let $x \in \{0,1\}^n$ with $n-m < \|x\|_1 < n$. Then*

$$\mathrm{E}[d(x_t) - d(x_{t+1}) \mid x_t = x] \geq \frac{2d(x)}{n}.$$

Lemma 13 (Average time spent in the slope towards a local maximum)**.** *Let $x_0 \in \{0,1\}^n$ such that $\|x_0\|_1 < n-m$ and $T_0 = \inf\{t \geq 0 \mid \|x_t\|_1 = n-m\}$ be the time taken by the* MMAHH *to reach a local maximum of* $\textsc{Jump}_m$*, starting in (x_0, s) with $s \in \{\mathrm{OI}, \mathrm{AM}\}$. Then*

$$\mathrm{E}[T_0] = O\left((n-\|x_0\|_1)(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right),$$

provided that $\frac{m}{2(n-2m)}\left(q+\frac{4}{n}\right) \geq p$ and $n \geq 3$.

In particular, if $\|x_0\|_1 = n-m-1$, then

$$\mathrm{E}[T_0] = O\left(m(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right).$$

Lemma 14 (Average time to climb towards a maximum)**.** *Let $T_1 = \inf\{t \geq 0 \mid x_t \in X^*\}$ be the time taken by the* MMAHH *starting with (x_0, OI) to reach x^* or a local maximum of* $\textsc{Jump}_m$ *then*

$$\mathrm{E}[T_1] = O\left(n(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right),$$

provided that $\frac{m}{2(n-2m)}\left(q+\frac{4}{n}\right) \geq p$ and $n \geq 3$.

We can now prove the main theorem of this section.

Proof of Theorem 7. The overall runtime of the MMAHH can be split in two times T_1 and T_2 such that $T = T_1 + T_2$,

$$T_1 = \inf\{t \geq 0 \mid x_t \in X^*\},$$

and T_2 is the time, starting in a local maximum of $\textsc{Jump}_m$ (if it happens), to reach x^*. First, by Lemma 14 we have

$$\mathrm{E}[T_1] = O\left(n(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right),$$

and it remains to upper bound $\mathrm{E}[T_2]$. Of course, $T_2 = 0$ if we already reached x^* during the first phase. Now, assume we do not, and hence, we are in some local maximum of $\textsc{Jump}_m$, i.e., some $x \in \{0,1\}^n$ such that $\|x\|_1 = n-m$. We then define excursions that start upon leaving layer $\mathcal{L}_m$ (the set of local maxima of $\textsc{Jump}_m$) and end either when we come back to this set of local maxima in $\mathcal{L}_m$ (in case of a failure) or when we reach x^*. As every excursion starts in state AM, the number N of such excursions can be upper bounded by the number N^* of phases of AM needed to reach x^* from layer $\mathcal{L}_m$. By Lemma 11, this can be bounded by

$$\mathrm{E}[N] \leq \mathrm{E}[N^*] \leq \frac{n^m}{m!(1-q)^{m-2}}.$$

Then, if we denote by e_i the i-th excursion and by $\lambda(e_i)$ its length, , i.e., the time of the excursion e_i, we can write

$$T_2 = T_0^w + \sum_{i=1}^{N}(\lambda(e_i) + T_i^w),$$

where T_i^w for $0 \leq i \leq N$ is waiting time between each excursion ($T_N^w = 0$ and T_0^w is the time before the first excursion starts). These waiting times can all be bounded from above in expectation by $\frac{1}{p}$ since only the OI operator can be stuck in this set of local maxima of JUMP_m. Now, we can split the excursions into two families: the left-excursions (starting with the move from $n-m$ to $n-m-1$) and the right-excursions (beginning with the move from $n-m$ to $n-m+1$). Given an excursion e_i, where $1 \le i \le N$, then as all the left-excursions (resp. right-excursions) follow the same distribution, we know that if e_i is a left-excursion, by Lemma 13 we have

$$\begin{aligned}\mathrm{E}[\lambda(e_i)] &\le 1 + O\left(m(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right)\\ &= O\left(m(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right),\end{aligned}$$

and if e_i is a right-excursion, from the proof of Lemma 14 and Lemma 12 we obtain

$$\mathrm{E}[\lambda(e_i)] \le 1 + \frac{n}{2} = O(n).$$

Thus, as $O(n) = O(mn) = O\left(m(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right)$ we conclude that

$$\mathrm{E}[\lambda(e_i)] = O\left(m(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right).$$

Hence, by Wald's theorem we obtain

$$\begin{aligned}\mathrm{E}[T_2] &= \mathrm{E}[T_0^w] + \mathrm{E}\left[\sum_{i=1}^{N}[\lambda(e_i) + T_i^w]\right]\\ &\le \frac{1}{p} + \mathrm{E}[N]\left(O\left(m(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right) + \frac{1}{p}\right)\\ &= O\left(\mathrm{E}[N]m(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right)\\ &= O\left(\frac{n^m}{(m-1)!(1-q)^{m-2}}(1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)\right).\end{aligned}$$

Finally, building on what we computed, we have

$$\begin{aligned}\mathrm{E}[T] &= \mathrm{E}[T_1] + \mathrm{E}[T_2]\\ &= O\left((1+pn)\left(\frac{1}{p}+\frac{1}{q}\right)N_{n,m,q}\right),\end{aligned}$$

where $N_{n,m,q} = n + \frac{n^m}{(m-1)!(1-q)^{m-2}}$. □

4 Runtime Analysis of the MMAHH with OnlyWorsening Acceptance on SEQOPT

We now state and prove the runtime of our MMAHH, which uses a Markov chain to select either the OI or OW acceptance operator, on the SEQOPT benchmark.

Theorem 15 (Runtime analysis of the MMAHH). *Assume $p = \Theta(\frac{1}{n\log(n)})$, $q = \Theta(\frac{1}{n\log(n)})$. Let $k \in [0..(n-2)]$ and $f \in \text{SEQOPT}_k(d_1,\dots,d_k)$ where $n > d_1 > \dots > d_k > 0$. If $k = O(1)$ then, the* MMAHH *with* OI *and* OW *reaches the global maximum at $x^* = (1,\dots,1)$ of f in runtime T with the expectation*

$$\mathrm{E}[T] = O\left(\frac{n^{k+1}}{d_1\cdots d_k}\log(n)\right).$$

Proof. We will prove the desired result by induction on the local optima d_ℓ for $\ell \in [1..k]$. To do so we denote the first hitting time of a local optimum at d_ℓ, by T_ℓ. The inductive hypothesis is then stated as follows,

$$\mathrm{E}[T_\ell] = O\left(\frac{n^\ell}{d_1\cdots d_{\ell-1}}\log(n)\right). \qquad (1)$$

We first demonstrate (1) for the base case with $\ell = 1$. We denote by k_1^* the number of phases needed to first reach a local optimum at d_1, i.e., $k_1^* = \inf\left\{k \in \mathbb{N} \mid y_{T_s^{(k)}} = d_1\right\}$ and T_1 the time taken to reach said local optimum. Here we upper bound $\mathrm{E}[T_1]$ by assuming $(y_0, s_0) = (n, \text{OW})$. Since for times $t \in [0..T_1]$ x_t is confined to the slope, for which $y_t \in [d_1, n]$, with monotonic fitness value, our *one-phase approximation* in Lemma 6 applies and gives

$$\mathrm{E}[k_1^*] \le \frac{1}{p_n^0} = O(1),$$

because $p, q = \Theta(1/n\log(n))$. Now, since

$$T_1 \le \sum_{k=0}^{k_1^*-1} Z_k,$$

and, by Lemma 22, $\mathrm{E}[Z_k] \le \max\left\{\frac{1}{p}, \frac{1}{q}\right\} = O(n\log(n))$, we can use Wald's Theorem (see Theorem 17) to obtain

$$\mathrm{E}[T_1] \le \frac{1}{p_n^0}\max\left\{\frac{1}{p}, \frac{1}{q}\right\} = O(n\log(n)),$$

as desired. This establishes (1) in the case $\ell = 1$.

Now, for the induction step suppose that (1) is true for some $\ell \in [1..k]$. Without loss of generality, we assume that at d_ℓ, the layer $\mathcal{L}_{d_\ell}$ is a set of local maxima of f (our argument equally holds for local minima by exchanging OI and OW). From these maxima, we define an excursion as a walk that starts by leaving $\mathcal{L}_{d_\ell}$ and, either comes back to $\mathcal{L}_{d_\ell}$ without having hit $\mathcal{L}_{d_{\ell+1}}$ (in case of a failure) or reach the new (unvisited) set of minima at layer $\mathcal{L}_{d_{\ell+1}}$ (in case of a success). We illustrate this situation in Figure 3, where failing excursions are depicted in red and a successful excursion is shown in green.

Since we assume layer $\mathcal{L}_{d_\ell}$ to be a set of local maxima of f, the average waiting time T_i^w between the $(i-1)$-th and i-th excursion is $\mathrm{E}[T_i^w] = O(\frac{1}{p}) = O(n\log(n))$, where the constant does not depend on i. Further, let k^* be the total number of excursions before layer $\mathcal{L}_{d_{\ell+1}}$ is reached for the first time and let e_i be the i-th excursion of length $\lambda(e_i)$. Then, we have

$$T_{\ell+1} = T_\ell + \sum_{i=1}^{k^*}\left(T_{i-1}^w + \lambda(e_i)\right),$$

where T_0^w is the waiting time before the first excursion starts.

Now, among the excursions, there are those starting by accepting the flip of a one-bit (which we call the left-excursions) and the others, starting by accepting a zero-bit flip (the right-excursions). First, the length of any left-excursion can be upper-bounded by $\mathrm{E}[T_\ell]$. Second, as any right-excursion is

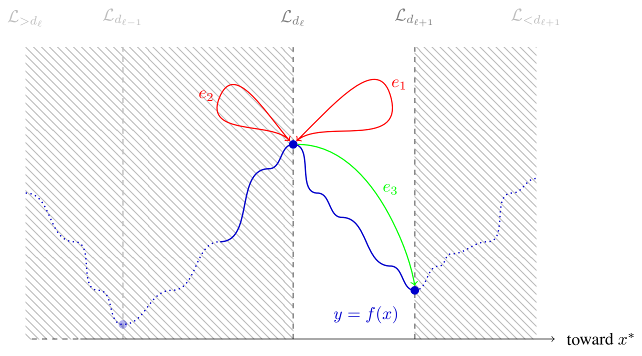

Figure 3: *Illustration of a function* $f\colon \{0,1\}^n \to \mathbb{R}$ *with a set of local maxima in layer* $\mathcal{L}_{d_\ell}$ *where three kinds of excursions can occur: a failing left excursion in red, the right excursion that fails to reach* $\mathcal{L}_{d_{\ell+1}}$ *also in red and the successful one in green.*

confined in the slope $[d_{\ell+1}, d_\ell]$ where f is decreasing across the layers $\mathcal{L}_{d_\ell}, \ldots, \mathcal{L}_{d_{\ell+1}}$, our *one-phase approximation* in Lemma 6 applies once more and gives, here for OW, that an average number of $O(1)$ right-excursions are needed in order to reach layer $\mathcal{L}_{d_{\ell+1}}$ and the total length of all these right-excursions is $O(n\log(n))$.

Moreover, upon leaving $\mathcal{L}_{d_\ell}$, there is a probability $\frac{d_\ell}{n}$ (resp. $\frac{n-d_\ell}{n}$) that the excursion will be a right-excursion (resp. left-excursion) hence, the average number of left-excursions performed before a right-excursion occurs is $\frac{n}{d_\ell}$ thus the average number of excursions needed is given by

$$\mathrm{E}[k^*] = O\left(\frac{n}{d_\ell}\right),$$

using Wald's theorem [Wald, 1944] in the simplified version of [Doerr and Künnemann, 2015].

Finally, with Wald's theorem again and using the induction hypothesis on $\mathrm{E}[T_\ell]$, we obtain

$$\begin{aligned}\mathrm{E}[T_{\ell+1}] &\le \mathrm{E}[T_\ell] \\ &\quad + \mathrm{E}[k^*]\left(O(n\log(n)) + O\left(\frac{n^\ell}{d_1\cdots d_{\ell-1}}\log(n)\right)\right) \\ &= O\left(\frac{n^{\ell+1}}{d_1\cdots d_\ell}\log(n)\right),\end{aligned}$$

since, for any $i \in [1..k^*]$, we have

$$\mathrm{E}[T^w_{i-1}+\lambda(e_i)] = \mathrm{E}[T^w_{i-1}]+\mathrm{E}[\lambda(e_i)] \le O(n\log(n))+\mathrm{E}[T_\ell],$$

where the left-hand side does not depend on i. The stated result hence follows by considering the case $\ell = k+1$ in (1), where assumption $k = O(1)$ implies that that the hidden constant in (1), which is a $O(1)^k$, is still a $O(1)$.

□

The factor $O(n\log(n))$ in the complexity derived in Theorem 15 represents the expected time the MMAHH requires to transition from one local optimum to a neighboring local optimum. The other multiplicative factors, i.e., $O({}^{n^k}\!/\!{}_{(d_1\cdots d_k)})$, can be interpreted as the time needed for a biased random walk on the set of local optima, starting from $(0,\ldots,0)$, to reach the global maximum.

The generality of our SEQOPT benchmark allows us to immediately extend our result in Theorem 15 to a variety of known benchmark functions, for which the MMAHH exhibits remarkable performance.

Corollary 16. *Let* $p = \Theta({}^1\!/\!{}_{n\log(n)})$ *and* $q = \Theta({}^1\!/\!{}_{n\log(n)})$. *Then, the runtime* T *of the* MMAHH

(i) on $\textsc{Jump}_m$ *with* $m \in [2..\frac{n}{2}]$ *satisfies*

$$\mathrm{E}(T) = O\left(\frac{n^3}{m}\log(n)\right),$$

(ii) on $\textsc{Cliff}_d$ *with* $d \in [2..\frac{n}{2}]$ *satisfies*

$$\mathrm{E}(T) = O\left(\frac{n^3}{d^2}\log(n)\right),$$

(iii) on $\textsc{CliffJump}_{d,r,s}$ *with* $s \in \mathbb{N}^+$, $d, r \in [1..\frac{n}{2}]$ *with* $r < d$ *satisfies*

$$\mathrm{E}(T) = O\left(\frac{n^3}{d(d-r)}\log(n)\right).$$

These results highlight the power of MMAHH: on $\textsc{Cliff}_d$ we incur only an extra $O(\log n)$ factor compared to the MAHH's $O(n\log n + \frac{n^3}{d^2})$ runtime (using probability $p = 1/((1+\varepsilon)n)$) as stated in [Lissovoi *et al.*, 2023], while achieving drastic speed-ups on $\textsc{Jump}_m$, reducing the $O\big(n\log n + \frac{n^{2m}(1+\varepsilon)^{m-1}}{(m^2\, m!)}\big)$ bound of the MAHH, and similarly outperforming it on $\textsc{CliffJump}_{d,r,s}$ beyond the previously established $O\big((d-r)\frac{(1+\varepsilon)^r n^{2r+2}}{(r+1)^2(r+1)!}\big)$ result.

5 Conclusion

In this work, we proposed two new building-blocks for the design of efficient move-acceptance hyper-heuristics. The Markov chain-based selection of the acceptance operator instead of the random mixing strategy used in previous theoretical works already with non-elaborate switching probabilities like $q = \frac{1}{2}$ greatly increases the rate of longer phases of the non-elitist selection operator (AM or OW), which led to provable significant speed-ups. The new OnlyWorsening (OW) acceptance operator was designed to aid the heuristic leave local optima. Used with the classic MAHH it gives comparable results as AM but, together with the Markov selection strategy it can lead to drastic speed-ups, e.g., polynomial runtimes on all $\textsc{SeqOpt}_k$ functions, k a constant. Both from the runtime guarantees proven in this work and from the working principles visible in our proofs we are very optimistic that our new building-blocks have a true potential for improving the hyper-heuristics used today.

The presented MMAHH algorithm mixing between OI and OW cannot traverse plateaus in the fitness landscape. The extension of these acceptance operators to accept solutions of equal fitness is an interesting direction for follow-up work. Furthermore, our theoretical analysis suggests that the empirical exploration of our proposed MMAHH is a promising direction for future research.

Acknowledgements

The research of B. Doerr is supported by the FMJH Program PGMO. J. Lutzeyer is supported by the French National Research Agency (ANR) via the "GraspGNNs" JCJC grant (ANR-24-CE23-3888).

References

[Auger and Doerr, 2011] Anne Auger and Benjamin Doerr, editors. *Theory of Randomized Search Heuristics*. World Scientific Publishing, 2011.

[Burke *et al.*, 2013] Edmund K. Burke, Michel Gendreau, Matthew R. Hyde, Graham Kendall, Gabriela Ochoa, Ender Özcan, and Rong Qu. Hyper-heuristics: a survey of the state of the art. *Journal of the Operational Research Society*, 64:1695–1724, 2013.

[Cowling *et al.*, 2000] Peter I. Cowling, Graham Kendall, and Eric Soubeiga. A hyperheuristic approach to scheduling a sales summit. In *Practice and Theory of Automated Timetabling, PATAT 2000*, pages 176–190. Springer, 2000.

[Doerr and Künnemann, 2015] Benjamin Doerr and Marvin Künnemann. Optimizing linear functions with the $(1+\lambda)$ evolutionary algorithm—different asymptotic runtimes for different instances. *Theoretical Computer Science*, 561:3–23, 2015.

[Doerr and Neumann, 2020] Benjamin Doerr and Frank Neumann, editors. *Theory of Evolutionary Computation—Recent Developments in Discrete Optimization*. Springer, 2020. Also available at http://www.lix.polytechnique.fr/Labo/Benjamin.Doerr/doerr_neumann_book.html.

[Doerr *et al.*, 2018] Benjamin Doerr, Andrei Lissovoi, Pietro S. Oliveto, and John Alasdair Warwicker. On the runtime analysis of selection hyper-heuristics with adaptive learning periods. In *Genetic and Evolutionary Computation Conference, GECCO 2018*, pages 1015–1022. ACM, 2018.

[Doerr *et al.*, 2023a] Benjamin Doerr, Arthur Dremaux, Johannes F. Lutzeyer, and Aurélien Stumpf. How the move acceptance hyper-heuristic copes with local optima: drastic differences between jumps and cliffs. In *Genetic and Evolutionary Computation Conference, GECCO 2023*, pages 990–999. ACM, 2023.

[Doerr *et al.*, 2023b] Benjamin Doerr, Taha El Ghazi El Houssaini, Amirhossein Rajabi, and Carsten Witt. How well does the Metropolis algorithm cope with local optima? In *Genetic and Evolutionary Computation Conference, GECCO 2023*, pages 1000–1008. ACM, 2023.

[Drake *et al.*, 2020] John Drake, Ahmed Kheiri, Ender Özcan, and Edmund Burke. Recent advances in selection hyper-heuristics. *European Journal of Operational Research*, 285:405–428, 09 2020.

[Droste *et al.*, 2002] Stefan Droste, Thomas Jansen, and Ingo Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 276:51–81, 2002.

[Jansen, 2013] Thomas Jansen. *Analyzing Evolutionary Algorithms – The Computer Science Perspective*. Springer, 2013.

[Kötzing and Krejca, 2019] Timo Kötzing and Martin S. Krejca. First-hitting times under drift. *Theoretical Computer Science*, 796:51–69, 2019.

[Lehre and Özcan, 2013] Per Kristian Lehre and Ender Özcan. A runtime analysis of simple hyper-heuristics: to mix or not to mix operators. In *Foundations of Genetic Algorithms, FOGA 2013*, pages 97–104. ACM, 2013.

[Lissovoi *et al.*, 2019] Andrei Lissovoi, Pietro S. Oliveto, and John Alasdair Warwicker. On the time complexity of algorithm selection hyper-heuristics for multimodal optimisation. In *Conference on Artificial Intelligence, AAAI 2019*, pages 2322–2329. AAAI Press, 2019.

[Lissovoi *et al.*, 2020] Andrei Lissovoi, Pietro S. Oliveto, and John Alasdair Warwicker. How the duration of the learning period affects the performance of random gradient selection hyper-heuristics. In *Conference on Artificial Intelligence, AAAI 2020*, pages 2376–2383. AAAI Press, 2020.

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

[Neumann and Witt, 2010] Frank Neumann and Carsten Witt. *Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity*. Springer, 2010.

[Wald, 1944] Abraham Wald. On cumulative sums of random variables. *Annals of Mathematical Statistics*, 15:283–296, 1944.

[Zhou *et al.*, 2019] Zhi-Hua Zhou, Yang Yu, and Chao Qian. *Evolutionary Learning: Advances in Theories and Algorithms*. Springer, 2019.