Theoretical Computer Science 545 (2014) 76–93

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Exact computation of the expectation surfaces for uniform crossover along with bit-flip mutation

Francisco Chicano [a,*], Darrell Whitley [b], Enrique Alba [a]

[a] Dept. of Lenguajes y Ciencias de la Computación, University of Málaga, 29071 Málaga, Spain
[b] Computer Science Dept., Colorado State University, Fort Collins, CO 80523, USA

ARTICLE INFO

Keywords:
Uniform crossover
Bit-flip mutation
Walsh decomposition
Landscape theory
Fitness landscapes

ABSTRACT

Uniform crossover and bit-flip mutation are two popular operators used in genetic algorithms to generate new solutions in an iteration of the algorithm when the solutions are represented by binary strings. We use the Walsh decomposition of pseudo-Boolean functions and properties of Krawtchouk matrices to exactly compute the expected value for the fitness of a child generated by uniform crossover followed by bit-flip mutation from two parent solutions. We prove that this expectation is a polynomial in ρ, the probability of selecting the best-parent bit in the crossover, and μ, the probability of flipping a bit in the mutation. We provide efficient algorithms to compute this polynomial for Onemax and MAX-SAT problems, but the results also hold for other problems such as NK-Landscapes. We also analyze the features of the expectation surfaces.

1. Introduction

In evolutionary computation, the classical variation operators are crossover and mutation. Crossover takes two solutions as input and generates one or more solutions combining the original ones. Mutation takes one solution and introduces a small change on it. These operators are usually run in sequence: crossover comes first and then mutation is applied to the output solution of crossover. The operators act on the genotype, that is, on the representation of the solutions. Thus, they are linked to the solution representation. Many different crossover and mutation operators have been defined for different representations in the literature. In the case of the binary strings some examples are 1-point crossover, 2-point crossover, uniform crossover, bit-flip mutation and 1-bit-flip mutation [1].

In particular, Uniform Crossover (UX) builds a new solution by randomly selecting each "allele" from one of the parent solutions. The "allele" in the best of the two parents is selected with probability ρ, which is called the *bias*. A common value for this bias is $\rho = 0.5$, where each parent has the same probability of providing its "allele" to the offspring. The so-called bit-flip mutation (BF) flips each individual bit of the binary string with a probability μ. In the literature it is common to use $\mu = 1/n$, which flips one bit per solution on average. From the point of view of runtime analysis, it has been proven that the value $\mu = c/n$ where c is a constant is optimal if we want to minimize the number of iterations to reach the global optimum on a linear pseudo-Boolean function using a $(1+1)$ Evolutionary Algorithm (EA) [2].

In this work we extend the results in [3] by providing a closed-form expression for the expected value of the objective function evaluated in a solution that is the result of applying UX and BF to two parent solutions. This expected value is a function of ρ and μ and, for this reason, we also call it *expectation surface*. The previous work [3] only considered UX. With the results presented in this paper we combine UX and BF and also prove in a different form the results in [4,5] for

* Corresponding author.
E-mail addresses: chicano@lcc.uma.es (F. Chicano), whitley@cs.colostate.edu (D. Whitley), eat@lcc.uma.es (E. Alba).

0304-3975/$ – see front matter
http://dx.doi.org/10.1016/j.tcs.2014.01.002

the expectation after BF. We also extend the formula to compute higher order moments, and not only the expected fitness value.

From a theoretical point of view, the closed-form formulas could help to understand the behavior of the two operators acting together. From a practical point of view, they could be used to develop new search strategies or operators. In particular, using for example the expected value and the standard deviation we can compute some bounds for the fitness values that the algorithm can find with a high probability after applying the operators [6]. Using this approach it would be possible to try different values for ρ and μ before applying the operators in order to improve the chances of finding good solutions.

The remainder of the paper is organized as follows. In the next section the mathematical tools required to understand the rest of the paper are presented. In Section 3 we present our main contribution of this work: the expected fitness value of the solution generated by UX and BF. We discuss in that section how the formula can be used to compute higher order moments (other than the expectation). Section 4 provides closed-form formulas for the expression of the expected fitness value in the case of the Onemax and MAX-SAT problems. In Section 5 we analyze the features of the expectation surface and provide a procedure to build an instance of weighted MAX-SAT having the desired surface. Finally, Section 6 presents the conclusions and future work.

2. Background

Let us first clarify the notation used for binary strings. We write here $\mathbb{B}^n$ as a synonym of $\mathbb{Z}_2^n$: $\mathbb{Z}_2^n = \mathbb{B}^n$. This set forms an Abelian group with the component-wise sum in $\mathbb{Z}_2$ (exclusive OR), denoted with $\oplus$. Given an element $z \in \mathbb{B}^n$, we will denote with $|z|$ the number of ones of z and with $\bar{z}$ the complement of the string (all bits inverted). Given a set of binary strings W and a binary string u we denote with $W \wedge u$ the set of binary strings that can be computed as the bitwise AND of a string in W and u, that is, $W \wedge u = \{w \wedge u \mid w \in W\}$. For example, $\mathbb{B}^4 \wedge 0101 = \{0000, 0001, 0100, 0101\}$. Given a string $w \in \mathbb{B}^n$, the set $\mathbb{B}^n \wedge w$ defines a hyperplane in $\mathbb{B}^n$ [7]. We will denote with $\underline{i}$ the binary string with position i set to 1 and the rest set to 0. We omit the length of the string n in the notation, but it will be clear from the context. For example, if we are considering binary strings in $\mathbb{B}^4$ we have $\underline{1} = 1000$ and $\underline{3} = 0010$. We will denote with $\vee$ the bitwise OR operator between two binary strings. We will assume that the $\wedge$ operator has precedence over $\vee$. However, we will use parentheses to clarify the precedence.

Definition 1. We define a *pseudo-Boolean* function f as a map between $\mathbb{B}^n$, the set of binary strings of length n, and $\mathbb{R}$, the set of real numbers.

Let us consider the set of all the pseudo-Boolean functions defined over $\mathbb{B}^n$, $\mathbb{R}^{\mathbb{B}^n}$. We can think of a pseudo-Boolean function as an array of 2^n real numbers, each one being the function evaluation of a particular binary string of $\mathbb{B}^n$. Each pseudo-Boolean function is, thus, a particular vector in a vector space with 2^n dimensions. Let us define the dot-product between two pseudo-Boolean functions as:

$$\langle f, g \rangle = \sum_{x \in \mathbb{B}^n} f(x) g(x).$$

Now we introduce a set of functions that will be relevant for our purposes in the next sections: the *Walsh functions* [8].

Definition 2. The (non-normalized) Walsh function with parameter $w \in \mathbb{B}^n$ is a pseudo-Boolean function defined over $\mathbb{B}^n$ as:

$$\psi_w(x) = \prod_{i=1}^{n} (-1)^{w_i x_i} = (-1)^{|w \wedge x|} = (-1)^{\sum_{i=1}^{n} w_i x_i}, \tag{1}$$

where the subindex in w_i and x_i denotes the i-th component of the binary strings w and x, respectively.

We can observe that the Walsh functions map $\mathbb{B}^n$ to the set $\{-1, 1\}$. We define the *order* of a Walsh function ψ_w as the value $|w|$. Some properties of the Walsh functions are given in the following proposition. The proof can be found in Vose [9].

Proposition 1. *Let us consider the Walsh functions defined over $\mathbb{B}^n$. The following identities hold*:

$$\psi_0 = 1, \tag{2}$$

$$\psi_{w \oplus t} = \psi_w \psi_t, \tag{3}$$

$$\psi_w(x \oplus y) = \psi_w(x) \psi_w(y), \tag{4}$$

$$\psi_w(x) = \psi_x(w), \tag{5}$$

$$\psi_w^2 = 1, \tag{6}$$

$$\sum_{x\in\mathbb{B}^n} \psi_w(x) = 2^n \delta_0^{|w|} = \begin{cases} 2^n & \text{if } w = 0, \\ 0 & \text{if } w \neq 0, \end{cases} \tag{7}$$

$$\psi_{\underline{i}}(x) = (-1)^{x_i} = 1 - 2x_i, \tag{8}$$

$$\langle \psi_w, \psi_t \rangle = 2^n \delta_w^t, \tag{9}$$

where δ denotes the Kronecker delta.

There exist 2^n Walsh functions in $\mathbb{B}^n$ and, according to (9), they are orthogonal, so they must form a basis of the set of pseudo-Boolean functions. Any arbitrary pseudo-Boolean function f can be expressed as a weighted sum of Walsh functions. We can represent f in the Walsh basis as follows:

$$f(x) = \sum_{w\in\mathbb{B}^n} a_w \psi_w(x),$$

where the *Walsh coefficients* a_w are defined by:

$$a_w = \frac{1}{2^n} \langle \psi_w, f \rangle. \tag{10}$$

The previous expression is called *Walsh expansion* (or decomposition) of f. The interested reader can refer to Terras [10] to find more information on Walsh functions and their properties.

Definition 3. Let $f : \mathbb{B}^n \to \mathbb{R}$ be a pseudo-Boolean function with Walsh expansion $f = \sum_{w\in\mathbb{B}^n} a_w \psi_w$. We define the order-$j$ elementary component of f in the one-flip neighborhood as:

$$f_{[j]} = \sum_{\substack{w\in\mathbb{B}^n \\ |w|=j}} a_w \psi_w,$$

for $0 \leqslant j \leqslant n$. As a consequence of the Walsh expansion of f we can write:

$$f = \sum_{j=0}^{n} f_{[j]}.$$

The reason why $f_{[j]}$ is called elementary component is because $f_{[j]}$ together with the one-flip neighborhood in the binary space forms what is called an *elementary landscape*. The reader interested in elementary landscapes in the binary hypercube can read [11].

Krawtchouk matrices play a relevant role in the mathematical development of the next sections. For this reason we present here some of their properties. The reader interested in these matrices can read [12]. The n-th order Krawtchouk matrix, $\mathcal{K}^{(n)}$ is an $(n+1)\times(n+1)$ integer matrix with indices between 0 and n and whose elements are defined as:

$$\mathcal{K}_{r,j}^{(n)} = \sum_{l=0}^{n} (-1)^l \binom{n-j}{r-l}\binom{j}{l}, \tag{11}$$

for $0 \leqslant r, j \leqslant n$. We assume in the previous expression that $\binom{a}{b} = 0$ if $b > a$ or $b < 0$. The elements of the Krawtchouk matrices can also be implicitly defined with the help of the following generating function:

$$(1+x)^{n-j}(1-x)^j = \sum_{r=0}^{n} x^r \mathcal{K}_{r,j}^{(n)}. \tag{12}$$

From (12) we deduce that $\mathcal{K}_{0,j}^{(n)} = 1$. Observe that $\mathcal{K}_{0,j}^{(n)}$ is the constant coefficient in the polynomial. We can also directly deduce from the definition in (11) that $\mathcal{K}_{r,0}^{(n)} = \binom{n}{r}$. The square of a Krawtchouk matrix is proportional to the identity matrix [12], that is: $(\mathcal{K}^{(n)})^2 = 2^n I$. A direct consequence of this is that the determinant of $\mathcal{K}^{(n)}$ is nonzero and, thus, Krawtchouk matrices are invertible. Krawtchouk matrices also appear when we sum Walsh functions. The following proposition provides an important result in this line [3].

Proposition 2. *Let $t \in \mathbb{B}^n$ be a binary string and $0 \leqslant r \leqslant n$. Then the following three identities hold for the sum of Walsh functions*:

$$\sum_{\substack{w\in\mathbb{B}^n \\ |w|=r}} \psi_w(x) = \mathcal{K}^{(n)}_{r,|x|}, \tag{13}$$

$$\sum_{\substack{w\in\mathbb{B}^n\wedge t \\ |w|=r}} \psi_w(x) = \mathcal{K}^{(|t|)}_{r,|x\wedge t|}, \tag{14}$$

$$\sum_{w\in\mathbb{B}^n\wedge t} \psi_w(x) = 2^{|t|}\delta_0^{|x\wedge t|}. \tag{15}$$

Proof. Using the definition of Walsh functions (1) we can write:

$$\sum_{\substack{w\in\mathbb{B}^n \\ |w|=r}} \psi_w(x) = \sum_{\substack{w\in\mathbb{B}^n \\ |w|=r}} (-1)^{|w\wedge x|},$$

we can change the index of the sum from w to $l = |w \wedge x|$. Once written with the new index l we only need to count for each l how many binary strings w with $|w| = r$ have the property $|w \wedge x| = l$, that is:

$$\sum_{\substack{w\in\mathbb{B}^n \\ |w|=r}} (-1)^{|w\wedge x|} = \sum_{l=0}^{n} (-1)^l \left|\left\{w \in \mathbb{B}^n \;\middle|\; |w| = r \text{ and } |w \wedge x| = l\right\}\right|. \tag{16}$$

We can compute the cardinality of the inner set in (16) using counting arguments. We need to count in how many ways we can distribute the r 1's in the string w such that they coincide with the 1's of x in exactly l positions. In order to do this, first let us put l 1's in the positions where x has 1. We can do this in $\binom{|x|}{l}$ different ways. Now, let us put the remaining $r - l$ 1's in the positions where x has 0. We can do this in $\binom{n-|x|}{r-l}$ ways. Multiplying both numbers we have the desired cardinality:

$$\left|\left\{w \in \mathbb{B}^n \;\middle|\; |w| = r \text{ and } |w \wedge x| = l\right\}\right| = \binom{|x|}{l}\binom{n-|x|}{r-l}. \tag{17}$$

We should notice here that the cardinality is zero in some cases. This happens when $l > |x|$, $l > r$ or $r - l > n - |x|$. However, in these cases we defined the binomial coefficient to be zero and we can keep the previous expression. If we use (17) in (16) and take into account the definition (11) then we get (13).

Let us now proof (14) and (15). Given two binary strings $x, t \in \mathbb{B}^n$, let us denote with $x|_t$ the binary string of length $|t|$ composed of all the bits of x in the positions i where $t_i = 1$. The string t acts as a mask for x. This notation allows us to simplify the sums in (14) and (15):

$$\sum_{\substack{w\in\mathbb{B}^n\wedge t \\ |w|=r}} \psi_w(x) = \sum_{\substack{w\in\mathbb{B}^n\wedge t \\ |w|=r}} \psi_{w|_t}(x|_t) = \sum_{\substack{u\in\mathbb{B}^{|t|} \\ |u|=r}} \psi_u(x|_t) = \mathcal{K}^{(|t|)}_{r,|x\wedge t|} \quad \text{by (13)},$$

$$\sum_{w\in\mathbb{B}^n\wedge t} \psi_w(x) = \sum_{w\in\mathbb{B}^n\wedge t} \psi_{w|_t}(x|_t) = \sum_{u\in\mathbb{B}^{|t|}} \psi_u(x|_t) = \sum_{u\in\mathbb{B}^{|t|}} \psi_{x|t}(u) = 2^{|t|}\delta_0^{|x\wedge t|} \quad \text{by (7)}. \qquad \square$$

3. Analysis of the operators

We are interested in this section in providing expressions for the expected value of the objective function evaluated in the offspring, the output of the crossover and mutation operators. In the past, Chicano et al. [3] provided expressions for the expectation of the objective function after applying the uniform crossover and, previously, Sutton et al. [5] and Chicano and Alba [4] provided similar expressions for the expected value after the bit-flip mutation. The result in this section extends the previous work by considering the combined effect of both the uniform crossover and the bit-flip mutation. We will see how the formulas derived in the mentioned works are particular cases of the one presented here. We will also see how the expression for the expectation can also be used to compute higher order statistical moments.

Let $x, y \in \mathbb{B}^n$ be the input solutions for the crossover and $C(x, y) \in \mathbb{B}^n$ the random variable giving the output of the crossover operator. If the crossover computes two children, we can imagine that it deterministically returns one. We will use the notation $M(z) \in \mathbb{B}^n$ to represent the random variable giving the output of a mutation operator applied to solution $z \in \mathbb{B}^n$. The combined effect of crossover and mutation will be given by the random variable $M(C(x, y))$ and $f(M(C(x, y))) \in \mathbb{R}$ is the fitness value of the output solution (a real random variable). At this moment we don't need to fix any operator. The following proposition provides a formula for $\mathbb{E}\{f(M(C(x, y)))\}$.

Proposition 3. *The expected value of the objective function f after applying crossover and mutation can be written in terms of the Walsh coefficients of the objective function and the two operators in the following way*:

$$\mathbb{E}\{f(M(C(x,y)))\} = 2^{2n} \sum_{w,w' \in \mathbb{B}^n} a_w m_{w,w'} b_{w'}(x,y), \tag{18}$$

where

$$a_w = \frac{1}{2^n}\langle \psi_w, f\rangle = \frac{1}{2^n} \sum_{x \in \mathbb{B}^n} f(x)\psi_w(x), \tag{19}$$

$$m_{w,w'} = \frac{1}{2^{2n}} \sum_{z,t \in \mathbb{B}^n} \Pr\{M(t) = z\}\psi_w(z)\psi_{w'}(t), \tag{20}$$

$$b_{w'}(x,y) = \frac{1}{2^n} \sum_{z \in \mathbb{B}^n} \Pr\{C(x,y) = z\}\psi_{w'}(z). \tag{21}$$

Proof.

$$\begin{aligned}\mathbb{E}\{f(M(C(x,y)))\} &= \sum_{z \in \mathbb{B}^n} f(z) \Pr\{M(C(x,y)) = z\} \\ &= \sum_{z,t \in \mathbb{B}^n} f(z) \Pr\{M(t) = z\} \Pr\{C(x,y) = t\},\end{aligned}$$

and using the Walsh decomposition of f and the probability mass functions we can rewrite the previous expression as

$$\begin{aligned}&= \sum_{z,t \in \mathbb{B}^n} \left(\sum_{w \in \mathbb{B}^n} a_w \psi_w(z) \right) \left(\sum_{w' \in \mathbb{B}^n} m_{w'}(t)\psi_{w'}(z) \right) \left(\sum_{w'' \in \mathbb{B}^n} b_{w''}(x,y)\psi_{w''}(t) \right) \\ &= \sum_{z,t \in \mathbb{B}^n} \sum_{w,w',w'' \in \mathbb{B}^n} a_w \psi_w(z) m_{w'}(t)\psi_{w'}(z) b_{w''}(x,y)\psi_{w''}(t) \\ &= \sum_{t \in \mathbb{B}^n} \sum_{w,w',w'' \in \mathbb{B}^n} a_w \left(\sum_{z \in \mathbb{B}^n} \psi_w(z)\psi_{w'}(z) \right) m_{w'}(t) b_{w''}(x,y)\psi_{w''}(t) \\ &= \sum_{t \in \mathbb{B}^n} \sum_{w,w',w'' \in \mathbb{B}^n} a_w 2^n \delta_w^{w'} m_{w'}(t) b_{w''}(x,y)\psi_{w''}(t) \\ &= 2^n \sum_{t \in \mathbb{B}^n} \sum_{w,w'' \in \mathbb{B}^n} a_w m_w(t) b_{w''}(x,y)\psi_{w''}(t) \\ &= 2^n \sum_{w,w'' \in \mathbb{B}^n} a_w b_{w''}(x,y) \left(\sum_{t \in \mathbb{B}^n} \psi_{w''}(t) m_w(t) \right) \\ &= 2^{2n} \sum_{w,w'' \in \mathbb{B}^n} a_w m_{w,w''} b_{w''}(x,y),\end{aligned}$$

where we introduced $m_w(t) = \frac{1}{2^n} \sum_{z \in \mathbb{B}^n} \Pr\{M(t) = z\}\psi_w(z)$ for the sake of brevity. □

The previous proposition changes a sum in the space of the solutions to a sum in terms of Walsh coefficients. From a computational point of view there is no advantage in the general case. The advantage comes when we focus on particular crossover and mutation operators and functions with a known Walsh decomposition. In particular, we will see in the following lemmas that the Walsh coefficients for the uniform crossover and the bit-flip mutation have simple expressions.

Let $x, y \in \mathbb{B}^n$ be the parent solutions for UX. For each position (bit) of the child binary string z, UX selects the bit in x with probability ρ and the bit in y with probability $1 - \rho$, where $\rho \in [0, 1]$ is called the *bias*. In most cases the bias is $\rho = 0.5$. We will replace the notation $C(x, y)$ used to represent a generic random variable representing the child of a crossover by a new notation including the parameter ρ of UX: $U_\rho(x, y)$.

In UX each position of the binary string is treated independently. Thus, the probability distribution of $U_\rho(x, y)$ can be written as a product of simpler probability distributions related to each bit. Let us denote with $B_\rho(x_i, y_i)$ the random variable with range in $\mathbb{B}$ that represents the bit selected to be at position i of the child if the parent bits at this position are x_i and y_i in UX with bias ρ. The probability distribution of $B_\rho(x_i, y_i)$ is:

$$\Pr\{B_\rho(x_i, y_i) = z_i\} = \rho\delta_{x_i}^{z_i} + (1 - \rho)\delta_{y_i}^{z_i}, \tag{22}$$

where δ denotes the Kronecker delta.

The probability distribution of UX is:

$$\Pr\{U_\rho(x,y)=z\}=\prod_{i=1}^{n}\Pr\{B_\rho(x_i,y_i)=z_i\}. \tag{23}$$

The following lemma provides the Walsh decomposition of $\Pr\{U_\rho(x,y)=z\}$. We decorate the Walsh coefficients with ρ to highlight the dependence of the coefficient on ρ.

Lemma 1. *Let $x, y, w \in \mathbb{B}^n$ and $\rho \in [0,1]$. The following identity holds for the Walsh coefficient $b_{w,\rho}(x,y)$ of the probability mass function $\Pr\{U_\rho(x,y)=z\}$:*

$$b_{w,\rho}(x,y)=\frac{1}{2^n}\psi_w(y)(1-2\rho)^{|(x\oplus y)\wedge w|}. \tag{24}$$

Proof. From (10) the Walsh coefficient $b_{w,\rho}(x,y)$ is:

$$\begin{aligned}
b_{w,\rho}(x,y) &= \frac{1}{2^n}\sum_{z\in\mathbb{B}^n}\psi_w(z)\Pr\{U_\rho(x,y)=z\}\\
&= \frac{1}{2^n}\sum_{z\in\mathbb{B}^n}\psi_w(z)\prod_{i=1}^{n}\Pr\{B_\rho(x_i,y_i)=z_i\} \quad \text{by (23)}\\
&= \frac{1}{2^n}\sum_{z\in\mathbb{B}^n}\left(\prod_{i=1}^{n}(-1)^{w_i z_i}\right)\prod_{i=1}^{n}\Pr\{B_\rho(x_i,y_i)=z_i\} \quad \text{by (1)}\\
&= \frac{1}{2^n}\sum_{z\in\mathbb{B}^n}\prod_{i=1}^{n}(-1)^{w_i z_i}\Pr\{B_\rho(x_i,y_i)=z_i\}\\
&= \frac{1}{2^n}\prod_{i=1}^{n}\sum_{z\in\mathbb{B}}(-1)^{w_i z}\Pr\{B_\rho(x_i,y_i)=z\}.
\end{aligned} \tag{25}$$

For the inner sum we can write

$$\begin{aligned}
\sum_{z\in\mathbb{B}}(-1)^{w_i z}\Pr\{B_\rho(x_i,y_i)=z\} &= \Pr\{B_\rho(x_i,y_i)=0\}+(-1)^{w_i}\Pr\{B_\rho(x_i,y_i)=1\}\\
&= 1-2\delta_1^{w_i}\Pr\{B_\rho(x_i,y_i)=1\},
\end{aligned}$$

where we exploit the fact that we must get 0 or 1 in a bit after the crossover, that is:

$$\Pr\{B_\rho(x_i,y_i)=0\}+\Pr\{B_\rho(x_i,y_i)=1\}=1.$$

Including this result in (25) we have

$$\begin{aligned}
b_{w,\rho}(x,y) &= \frac{1}{2^n}\prod_{i=1}^{n}\left(1-2\delta_1^{w_i}\Pr\{B_\rho(x_i,y_i)=1\}\right)\\
&= \frac{1}{2^n}\prod_{\substack{i=1\\ w_i=1}}^{n}\left(1-2\Pr\{B_\rho(x_i,y_i)=1\}\right).
\end{aligned} \tag{26}$$

Using the definition of $\Pr\{B_\rho(x_i,y_i)=z_i\}$ in (22):

$$\Pr\{B_\rho(x_i,y_i)=1\}=\rho\delta_{x_i}^1+(1-\rho)\delta_{y_i}^1,$$

and the factor inside (26) is

$$\left(1-2\Pr\{B_\rho(x_i,y_i)=1\}\right)=(-1)^{y_i}\left(1-2\rho+2\rho\delta_{x_i}^{y_i}\right).$$

We can develop (26) in the following way:

$$
\begin{aligned}
b_{w,\rho}(x,y) &= \frac{1}{2^n} \prod_{\substack{i=1\\ w_i=1}}^{n} (-1)^{y_i}\left(1-2\rho+2\rho\delta_{x_i}^{y_i}\right)\\
&= \frac{1}{2^n}\left(\prod_{\substack{i=1\\ w_i=1}}^{n} (-1)^{y_i}\right) \prod_{\substack{i=1\\ w_i=1}}^{n} \left(1-2\rho+2\rho\delta_{x_i}^{y_i}\right)\\
&= \frac{1}{2^n}\psi_w(y) \prod_{\substack{i=1\\ w_i=1}}^{n} \left(1-2\rho+2\rho\delta_{x_i}^{y_i}\right). \qquad (27)
\end{aligned}
$$

The expression $(1-2\rho+2\rho\delta_{x_i}^{y_i})$ takes only two values: 1 if $y_i = x_i$ and $1-2\rho$ when $x_i \neq y_i$. A factor $1-2\rho$ is included in the product for all the positions i in which $x_i \neq y_i$ and $w_i = 1$. Then the product in (27) becomes $(1-2\rho)^{|(x\oplus y)\wedge w|}$ and we obtain (24). □

Now we present the Walsh coefficients for the bit-flip mutation with probability μ, denoted as M_μ. We decorate the coefficients with the parameter μ.

Lemma 2. *The Walsh coefficients of the bit-flip mutation are*

$$m_{w,w',\mu} = \frac{1}{2^n}\delta_w^{w'}(1-2\mu)^{|w|}. \qquad (28)$$

Proof. This result has been proven in previous work [9,13,14]. We keep here the proof for completeness and because the notation is slightly different. Let us start writing the definition of $m_{w,w',\mu}$ in (20)

$$m_{w,w',\mu} = \frac{1}{2^{2n}} \sum_{z,t\in\mathbb{B}^n} \Pr\{M_\mu(t)=z\}\psi_{w'}(t)\psi_w(z),$$

and now we take into account that the probability can be decomposed as a product of factors to write

$$
\begin{aligned}
m_{w,w',\mu} &= \frac{1}{2^{2n}} \sum_{z,t\in\mathbb{B}^n} \psi_{w'}(t)\psi_w(z) \prod_{i=1}^{n} \Pr\{M_\mu(t_i)=z_i\}\\
&= \frac{1}{2^{2n}} \sum_{z,t\in\mathbb{B}^n} \psi_{w'}(t) \prod_{i=1}^{n} (-1)^{w_i z_i} \Pr\{M_\mu(t_i)=z_i\}\\
&= \frac{1}{2^{2n}} \sum_{t\in\mathbb{B}^n} \psi_{w'}(t) \prod_{i=1}^{n} \sum_{z\in\mathbb{B}} (-1)^{w_i z} \Pr\{M_\mu(t_i)=z\}\\
&= \frac{1}{2^{2n}} \sum_{t\in\mathbb{B}^n} \psi_{w'}(t) \prod_{i=1}^{n} \left((-1)^{w_i(1-t_i)}\mu + (-1)^{w_i t_i}(1-\mu)\right)\\
&= \frac{1}{2^{2n}} \sum_{t\in\mathbb{B}^n} \psi_{w'}(t) \prod_{i=1}^{n} \left((-1)^{w_i}(-1)^{w_i t_i}\mu + (-1)^{w_i t_i}(1-\mu)\right)\\
&= \frac{1}{2^{2n}} \sum_{t\in\mathbb{B}^n} \psi_{w'}(t) \prod_{i=1}^{n} (-1)^{w_i t_i}\left(1+\left((-1)^{w_i}-1\right)\mu\right)\\
&= \frac{1}{2^{2n}} \left(\sum_{t\in\mathbb{B}^n} \psi_{w'}(t)\psi_w(t)\right) \prod_{i=1}^{n} \left(1+\left((-1)^{w_i}-1\right)\mu\right).
\end{aligned}
$$

The factors in this product take two possible values: $(1-2\mu)$ when $w_i = 1$ and 1 otherwise. Using (3) and (9) we can write:

$$m_{w,w',\mu} = \frac{1}{2^n}\delta_w^{w'}(1-2\mu)^{|w|}. \qquad \square$$

Now we are ready to present the main result of this work combining the results of Proposition 3 and Lemmas 1 and 2.

Theorem 1. *Let f be a pseudo-Boolean function defined over $\mathbb{B}^n$ and a_w with $w \in \mathbb{B}^n$ its Walsh coefficients. The following identity holds for $\mathbb{E}\{f(M_\mu(U_\rho(x,y)))\}$:*

$$\mathbb{E}\{f(M_\mu(U_\rho(x,y)))\} = \sum_{r,l=0}^{n} A_{x,y}^{(r,l)}(1-2\rho)^r(1-2\mu)^l, \tag{29}$$

where the coefficients $A_{x,y}^{(r,l)}$ are defined by:

$$A_{x,y}^{(r,l)} = \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |(x\oplus y)\wedge w|=r}} a_w \psi_w(y), \tag{30}$$

and $A_{x,y}^{(r,l)} = 0$ for $r > l$.

Proof. According to (18), (24) and (28) we can write

$$\begin{aligned}
\mathbb{E}\{f(M_\mu(U_\rho(x,y)))\} &= 2^{2n} \sum_{w,w'\in\mathbb{B}^n} a_w m_{w,w',\mu} b_{w',\rho}(x,y) \\
&= 2^{2n} \sum_{w,w'\in\mathbb{B}^n} a_w \frac{1}{2^n}\delta_w^{w'}(1-2\mu)^{|w|}\frac{1}{2^n}\psi_{w'}(y)(1-2\rho)^{|(x\oplus y)\wedge w'|} \\
&= \sum_{w,w'\in\mathbb{B}^n} a_w \delta_w^{w'}(1-2\mu)^{|w|}\psi_{w'}(y)(1-2\rho)^{|(x\oplus y)\wedge w'|} \\
&= \sum_{w\in\mathbb{B}^n} a_w (1-2\mu)^{|w|}\psi_w(y)(1-2\rho)^{|(x\oplus y)\wedge w|} \\
&= \sum_{r,l=0}^{n} \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |(x\oplus y)\wedge w|=r}} a_w (1-2\mu)^{|w|}\psi_w(y)(1-2\rho)^{|(x\oplus y)\wedge w|} \\
&= \sum_{r,l=0}^{n} (1-2\mu)^l(1-2\rho)^r \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l| \\ |(x\oplus y)\wedge w|=r}} a_w\psi_w(y) \\
&= \sum_{r,l=0}^{n} A_{x,y}^{(r,l)}(1-2\mu)^l(1-2\rho)^r
\end{aligned}$$

and we get (29). The identity $A_{x,y}^{(r,l)} = 0$ when $r > l$ can be concluded from the definition of $A_{x,y}^{(r,l)}$, since $r = |(x\oplus y)\wedge w| \leqslant |w| = l$. □

Note that the expression for the expected fitness after applying the two operators is a polynomial in $(1-2\rho)$ and $(1-2\mu)$. The complexity of computing the expected value depends on the complexity of computing the coefficients $A_{x,y}^{(r,l)}$ and the expression (29). The computation of the coefficients depends on the problem at hand and little can we say without fixing a problem. However, we can claim that the difficulty of the problem has nothing to do with the complexity of computing the coefficients. We will see in Section 4.2 that this complexity is $O(mk^3)$ for MAX-kSAT, where m is the number of clauses. Once we have the coefficients, (29) can be computed by iterating over all the (r,l) pairs. There are $O(n^2)$ pairs in the worst case, and this is the worst case complexity for computing (29).

We say that a pseudo-Boolean function is k-bounded epistatic if it can be written as a sum of terms that depend at most on k different variables. One example of k-bounded epistatic pseudo-Boolean function is the objective function of the MAX-kSAT problem. If our function is a k-bounded epistatic pseudo-Boolean function, the values of r and l will be k at most. This means that the complexity is reduced to $O(k^2)$ in the worst case. This is the complexity of computing (29) in MAX-kSAT or NK-landscapes for $K = k-1$, for example.

The expected fitness value after a bit-flip mutation or a uniform crossover are particular cases of (29) and are given in the next corollary in Eqs. (31) and (32). Chicano and Alba [4] derived Eq. (31) and Sutton et al. [5] also discovered that the expectation is a polynomial in μ. In a recent paper, Sutton et al. generalized that expression for q-ary strings [15]. On the other hand, the expected fitness value after uniform crossover (32) was presented by Chicano, Whitley and Alba in [3]. The difference between the proofs presented here for (31) and (32) and the ones in the previous work is that here we deduce both expressions as particular cases of Theorem 1.

Corollary 1. *The expected fitness value after bit-flip mutation with probability μ on solution x is given by*:

$$\mathbb{E}\{f(M_\mu(x))\} = \sum_{l=0}^{n} f_{[l]}(x)(1-2\mu)^l, \tag{31}$$

and the expected fitness value after applying uniform crossover to solutions x and y with bias ρ is given by:

$$\mathbb{E}\{f(U_\rho(x,y))\} = \sum_{r=0}^{n} A_{x,y}^{(r)}(1-2\rho)^r, \tag{32}$$

where we define $A_{x,y}^{(r)}$ as:

$$A_{x,y}^{(r)} = \sum_{l=0}^{n} A_{x,y}^{(r,l)}. \tag{33}$$

Proof. Let us start with the mutation. The expected fitness value after the mutation of solution x is the same as the expected fitness value after applying uniform crossover and bit-flip mutation to solutions x and x, because the output individual of the crossover will be x. Thus, we can use (29) with the coefficients $A_{x,x}^{(r,l)}$. These coefficients are:

$$A_{x,x}^{(r,l)} = \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |(x\oplus x)\wedge w|=r}} a_w\psi_w(x) = \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |0\wedge w|=r}} a_w\psi_w(x) = \delta_0^r \sum_{\substack{w\in\mathbb{B}^n \\ |w|=l}} a_w\psi_w(x) = \delta_0^r f_{[l]}(x).$$

Including the coefficients in (29) we obtain:

$$\begin{aligned}\mathbb{E}\{f(M_\mu(x))\} = \mathbb{E}\{f(M_\mu(U_\rho(x,x)))\} &= \sum_{r,l=0}^{n} A_{x,x}^{(r,l)}(1-2\mu)^l(1-2\rho)^r \\ &= \sum_{r,l=0}^{n} \delta_0^r f_{[l]}(x)(1-2\mu)^l(1-2\rho)^r = \sum_{l=0}^{n} f_{[l]}(x)(1-2\mu)^l.\end{aligned}$$

Regarding the uniform crossover, we can again use (29) with $\mu = 0$ because the bit-flip mutation keeps the solution unchanged in this case:

$$\begin{aligned}\mathbb{E}\{f(U_\rho(x,y))\} = \mathbb{E}\{f(M_0(U_\rho(x,y)))\} &= \sum_{r,l=0}^{n} A_{x,y}^{(r,l)}(1-2\cdot 0)^l(1-2\rho)^r \\ &= \sum_{r,l=0}^{n} A_{x,y}^{(r,l)}(1-2\rho)^r = \sum_{r=0}^{n}\left(\sum_{l=0}^{n} A_{x,y}^{(r,l)}\right)(1-2\rho)^r,\end{aligned}$$

and using the definition of $A_{x,y}^{(r)}$ we obtain (32). □

When UX is used in the literature a common value for ρ is $1/2$. In this case, the expression for the expected fitness value is simpler, as the following corollary proves.

Corollary 2. *Let f be a pseudo-Boolean function defined over $\mathbb{B}^n$ and a_w with $w \in \mathbb{B}^n$ its Walsh coefficients. The expected value of the fitness function after applying UX and bit-flip mutation to solutions x and y with bias $\rho = 1/2$ and bit-flip probability μ is*:

$$\mathbb{E}\{f(M_\mu(U_{1/2}(x,y)))\} = \sum_{l=0}^{n} A_{x,y}^{(0,l)}(1-2\mu)^l.$$

Proof. If we set $\rho = 1/2$ in the polynomial (29) all the terms $(1-2\rho)^r$ with $r > 0$ vanish and the expected fitness value includes only the terms with coefficients $A_{x,y}^{(0,l)}$. □

Theorem 1 provides an expression for the expectation of the fitness. But expectation is just one statistical parameter of a real random variable. We can take more information on the random variable by computing higher order moments. Fortunately, Theorem 1 allows us to compute also higher order moments. We only need to consider that the c-th moment is the expectation of f^c.

Corollary 3. *The c-th moment of f after UX with bias ρ and BF with probability μ applied to solutions x and y is*:

$$\mathbb{E}\{f^c(M_\mu(U_\rho(x,y)))\} = \sum_{r,l=0}^{n} A_{x,y}^{(r,l,c)}(1-2\rho)^r(1-2\mu)^l, \tag{34}$$

where the coefficients $A_{x,y}^{(r,l,c)}$ are defined as:

$$A_{x,y}^{(r,l,c)} = \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |(x\oplus y)\wedge w|=r}} a_w^{(c)}\psi_w(y) \tag{35}$$

and $a_w^{(c)}$ is the Walsh coefficient of f^c for string w: $a_w^{(c)} = \frac{1}{2^n}\sum_{x\in\mathbb{B}^n}\psi_w(x)f^c(x)$.

Proof. It is a direct consequence of Theorem 1 and the definition of the c-th moment. □

In the previous corollary the computational costly part is the computation of the coefficients. Sutton et al. [11] provide a polynomial time algorithm for computing the Walsh coefficients of the powers of f in the case of k-bounded epistasis. As a consequence, for k-bounded epistatic pseudo-Boolean functions it is possible to compute the coefficients $A_{x,y}^{(r,l,c)}$ in polynomial time if c is fixed.

The last result of this section proves that the coefficients $A_{x,y}^{(r,l)}$ of a sum of functions are the sum of the corresponding coefficients of the individual functions. This result simplifies several proofs of the next section.

Proposition 4. *Let $A_{x,y}^{(r,l)}$ be the polynomial coefficients for f and $B_{x,y}^{(r,l)}$ the polynomial coefficients for g. Then, the polynomial coefficients for $h = f + g$ are*

$$C_{x,y}^{(r,l)} = A_{x,y}^{(r,l)} + B_{x,y}^{(r,l)}.$$

Proof. Let a_w with $w\in\mathbb{B}^n$ be the Walsh coefficients of f and b_w the Walsh coefficients of g. Then, the Walsh coefficients of $h = f + g$ are $c_w = a_w + b_w$. Therefore:

$$\begin{aligned} C_{x,y}^{(r,l)} &= \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |(x\oplus y)\wedge w|=r}} c_w\psi_w(y) = \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |(x\oplus y)\wedge w|=r}} (a_w+b_w)\psi_w(y) \\ &= \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |(x\oplus y)\wedge w|=r}} a_w\psi_w(y) + \sum_{\substack{w\in\mathbb{B}^n,\ |w|=l \\ |(x\oplus y)\wedge w|=r}} b_w\psi_w(y) \\ &= A_{x,y}^{(r,l)} + B_{x,y}^{(r,l)}. \qquad \square \end{aligned}$$

4. Onemax and MAX-SAT

If the Walsh decomposition of the objective function f is known, the result of Theorem 1 allows one to compute the expected fitness after UX and BF. One can argue that the computation of the coefficients of the polynomial (29) can be costly. However, we can restrict the cost to be polynomial by considering k-bounded epistatic pseudo-Boolean functions. This class of problems includes MAX-SAT and NK-landscapes, as well as all the linear pseudo-Boolean functions such as Onemax. In order to illustrate that this computation can be efficient for these problems, we provide expressions for the coefficients $A_{x,y}^{(r,l)}$ in the case of two well-known problems in combinatorial optimization: Onemax and MAX-SAT.

4.1. Onemax

This is a toy combinatorial optimization problem defined over binary strings which is commonly studied due to its simplicity. The objective function for Onemax is:

$$f(x) = |x| = \sum_{i=1}^{n} x_i.$$

Using (8) we can write:

$$f(x) = \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} \frac{1-\psi_{\underline{i}}(x)}{2} = \frac{n}{2} - \frac{1}{2}\sum_{i=1}^{n}\psi_{\underline{i}}(x),$$

and we deduce that the Walsh coefficients for Onemax are:

$$a_w = \begin{cases} n/2 & \text{if } |w| = 0, \\ -1/2 & \text{if } |w| = 1, \\ 0 & \text{if } |w| > 1. \end{cases}$$

Since all the nonzero Walsh coefficients have order 0 or 1, only the coefficients $A_{x,y}^{(r,0)}$ and $A_{x,y}^{(r,1)}$ can be nonzero. Furthermore, since $A_{x,y}^{(r,l)} = 0$ if $r > l$, we conclude that the only nonzero coefficients are $A_{x,y}^{(0,0)}$, $A_{x,y}^{(0,1)}$ and $A_{x,y}^{(1,1)}$. The next proposition provides the value for these coefficients.

Proposition 5. *Let $x, y \in \mathbb{B}^n$ be two binary strings, the nonzero polynomial coefficients $A_{x,y}^{(r,l)}$ for the Onemax problem are*:

$$A_{x,y}^{(0,0)} = \frac{n}{2}, \tag{36}$$

$$A_{x,y}^{(0,1)} = -\frac{1}{2}|\overline{x \oplus y}| + |x \wedge y|, \tag{37}$$

$$A_{x,y}^{(1,1)} = -\frac{1}{2}|x \oplus y| + |\bar{x} \wedge y|. \tag{38}$$

Proof. The first coefficient $A_{x,y}^{(0,0)} = n/2$ is exactly $a_0 = n/2$. The expression of the $A_{x,y}^{(0,1)}$ coefficient is:

$$\begin{aligned} A_{x,y}^{(0,1)} &= \sum_{\substack{w \in \mathbb{B}^n,\ |w|=1 \\ |(x \oplus y) \wedge w| = 0}} a_w \psi_w(y) = -\frac{1}{2} \sum_{\substack{w \in \mathbb{B}^n,\ |w|=1 \\ |(x \oplus y) \wedge w| = 0}} \psi_w(y) = -\frac{1}{2} \sum_{\substack{i=1 \\ x_i = y_i}}^{n} (1 - 2y_i) \\ &= -\frac{1}{2}\big(n - |x \oplus y|\big) + \sum_{\substack{i=1 \\ x_i = y_i}}^{n} y_i = -\frac{1}{2}|\overline{x \oplus y}| + |x \wedge y|, \end{aligned}$$

where we used (8) in the third step and we took into account that the binary string $x \oplus y$ has 1 in the positions in which $x_i \neq y_i$ and $\overline{x \oplus y}$ has 1 in the positions in which $x_i = y_i$.

The expression of the $A_{x,y}^{(1,1)}$ coefficient is:

$$\begin{aligned} A_{x,y}^{(1,1)} &= \sum_{\substack{w \in \mathbb{B}^n,\ |w|=1 \\ |(x \oplus y) \wedge w| = 1}} a_w \psi_w(y) = -\frac{1}{2} \sum_{\substack{i=1 \\ x_i \neq y_i}}^{n} (1 - 2y_i) = -\frac{1}{2}|x \oplus y| + \sum_{\substack{i=1 \\ x_i \neq y_i}}^{n} y_i \\ &= -\frac{1}{2}|x \oplus y| + |\bar{x} \wedge y|, \end{aligned}$$

and we have expressions (36) to (38). □

4.2. MAX-SAT

MAX-SAT is an NP-hard combinatorial optimization problem that amounts to maximizing the number of satisfied clauses of a Boolean formula in conjunctive normal form. It is related to the SAT decision problem, which has received a lot of attention due to its importance for computer science.

Let us assume that n Boolean decision variables exist in the Boolean formula and let C be a set of clauses. In the MAX-SAT problem each clause $c \in C$ is composed of literals, each one being a decision variable x_i or a negated decision variable $\overline{x_i}$. For each clause $c \in C$ we define the vectors $v(c) \in \mathbb{B}^n$ and $u(c) \in \mathbb{B}^n$ as follows [16]:

$$v_i(c) = \begin{cases} 1 & \text{if } x_i \text{ appears (negated or not) in } c, \\ 0 & \text{otherwise,} \end{cases} \tag{39}$$

$$u_i(c) = \begin{cases} 1 & \text{if } x_i \text{ appears negated in } c, \\ 0 & \text{otherwise.} \end{cases} \tag{40}$$

According to this definition $u(c) \wedge v(c) = u(c)$. The objective function of this problem is defined as

$$f(x) = \sum_{c \in C} f_c(x); \quad \text{where}$$

$$f_c(x) = \begin{cases} 1 & \text{if } c \text{ is satisfied with assignment } x, \\ 0 & \text{otherwise.} \end{cases} \tag{41}$$

A clause c is satisfied with x if at least one of the literals is true. Using the vectors $v(c)$ and $u(c)$ we can say that c is satisfied by x if $(\bar{x} \wedge u(c)) \vee (x \wedge v(c) \wedge \overline{u(c)}) \neq 0$. If $|v(c)| = k$ is the same for all the clauses the problem is called MAX-kSAT.

Sutton et al. [16] provide the Walsh decomposition for the MAX-SAT problem. Let the function f_c evaluate one clause $c \in C$. The Walsh coefficients for f_c are:

$$a_w = \begin{cases} 0 & \text{if } w \wedge \overline{v(c)} \neq 0, \\ 1 - \frac{1}{2^{|v(c)|}} & \text{if } w = 0, \\ \frac{-1}{2^{|v(c)|}} \psi_w(u(c)) & \text{otherwise.} \end{cases}$$

The Walsh coefficients for MAX-SAT simplify if we use $g_c = 1 - f_c$ instead of f_c. Using the g_c functions the objective function for MAX-SAT is $f = m - \sum_{c \in C} g_c$. The Walsh coefficients of g_c are:

$$a_w = \begin{cases} 0 & \text{if } w \wedge \overline{v(c)} \neq 0, \\ \frac{1}{2^{|v(c)|}} \psi_w(u(c)) & \text{otherwise.} \end{cases} \tag{42}$$

The following lemma provides the polynomial coefficients $A^{(r,l)}_{x,y}(c)$ for the function g_c, where we include the clause in the coefficient to distinguish the value of one clause from another.

Proposition 6. *Let $x, y \in \mathbb{B}^n$ be two binary strings and $0 \leqslant r \leqslant l \leqslant n$. Then, the following identity holds for the polynomial coefficients $A^{(r,l)}_{x,y}(c)$ in the case of g_c:*

$$A^{(r,l)}_{x,y}(c) = \frac{1}{2^{|v(c)|}} \mathcal{K}^{(\beta)}_{r,\alpha} \mathcal{K}^{(|v(c)|-\beta)}_{l-r,\gamma}, \tag{43}$$

where $\alpha = |v(c) \wedge (x \oplus y) \wedge (u(c) \oplus y)|$, $\beta = |v(c) \wedge (x \oplus y)|$ and $\gamma = |v(c) \wedge \overline{(x \oplus y)} \wedge (u(c) \oplus y)|$.

Proof. In the following we will remove the argument c in the vectors $v(c)$ and $u(c)$ to alleviate the notation. The nonzero Walsh coefficients a_w are the ones for which $w \wedge \bar{v} = 0$, which are exactly $w \in \mathbb{B}^n \wedge v$. Thus, we can restrict the sum of (30) to these binary strings. Thus, all the coefficients with $l > |v|$ will be zero. We can write:

$$\begin{aligned} A^{(r,l)}_{x,y} &= \sum_{\substack{w \in \mathbb{B}^n,\ |w|=l \\ |(x \oplus y) \wedge w| = r}} a_w \psi_w(y) \\ &= \sum_{\substack{w \in \mathbb{B}^n \wedge v,\ |w|=l \\ |(x \oplus y) \wedge w| = r}} \frac{1}{2^{|v|}} \psi_w(u) \psi_w(y) \quad \text{by (42)} \\ &= \frac{1}{2^{|v|}} \sum_{\substack{w \in \mathbb{B}^n \wedge v,\ |w|=l \\ |(x \oplus y) \wedge w| = r}} \psi_w(u \oplus y) \quad \text{by (4)}. \end{aligned}$$

We can now write each w as the sum of two strings w' and w'' where $w' \in \mathbb{B}^n \wedge (v \wedge (x \oplus y))$ and $w'' \in \mathbb{B}^n \wedge (v \wedge (\overline{x \oplus y}))$. By definition $w' \wedge w'' = 0$ and $w' \oplus w'' = w$, thus, $|w| = |w'| + |w''|$.

$$\begin{aligned} A^{(r,l)}_{x,y} &= \frac{1}{2^{|v|}} \sum_{\substack{w' \in \mathbb{B}^n \wedge (v \wedge (x \oplus y)) \\ |w'| = r}} \sum_{\substack{w'' \in \mathbb{B}^n \wedge (v \wedge \overline{(x \oplus y)}) \\ |w''| = l-r}} \psi_{w' \oplus w''}(u \oplus y) \\ &= \frac{1}{2^{|v|}} \left(\sum_{\substack{w' \in \mathbb{B}^n \wedge (v \wedge (x \oplus y)) \\ |w'| = r}} \psi_{w'}(u \oplus y) \right) \cdot \left(\sum_{\substack{w'' \in \mathbb{B}^n \wedge (v \wedge \overline{(x \oplus y)}) \\ |w''| = l-r}} \psi_{w''}(u \oplus y) \right). \end{aligned}$$

Let us now define $\alpha = |v(c) \wedge (x \oplus y) \wedge (u(c) \oplus y)|$, $\beta = |v(c) \wedge (x \oplus y)|$ and $\gamma = |v(c) \wedge \overline{(x \oplus y)} \wedge (u(c) \oplus y)|$. Then, using the results of Proposition 2 we can write:

$$A^{(r,l)}_{x,y} = \frac{1}{2^{|v|}} \mathcal{K}^{(\beta)}_{r,\alpha} \mathcal{K}^{(|v \wedge \overline{(x \oplus y)}|)}_{l-r,\gamma}.$$

Taking into account that $|v \wedge \overline{(x \oplus y)}| + \beta = |v|$ we get (43). □

All the $A^{(r,l)}_{x,y}(c)$ coefficients for each particular clause can be efficiently computed in $O(|v(c)|)$ time. The bitwise operations required to compute α, β and γ only need to explore the bits set to one in $v(c)$ (that is, $|v(c)|$ bits). With the values

of α, β and γ each of the $(|v(c)|+1)(|v(c)|+2)/2$ coefficients can be computed in $O(1)$. Thus, all the $A^{(r,l)}_{x,y}(c)$ coefficients can be computed in $O(|v(c)|^2)$. The coefficients for the MAX-SAT objective function f are given in the following proposition.

Proposition 7. *Let $x, y \in \mathbb{B}^n$ be two binary strings and $0 \leqslant r \leqslant l \leqslant n$. Then, the following identity holds for the polynomial coefficients $A^{(r,l)}_{x,y}$ of the MAX-SAT problem*:

$$A^{(r,l)}_{x,y} = m\delta^{r,l}_{0,0} - \sum_{c\in C} A^{(r,l)}_{x,y}(c),$$

where $A^{(r,l)}_{x,y}(c)$ is given by (43), $\delta^{r,l}_{0,0} = \delta^r_0\delta^l_0$ and m is the number of clauses. Each one of these coefficients can be computed in $O(mk)$ where k is the maximum number of literals in a clause. All the coefficients can be computed in order $O(mk^3)$.

Proof. It is a direct consequence of the definition of the objective function and Propositions 4 and 6. □

From these results, we conclude that the expectation surface of the fitness value after applying UX and BF to two solutions in the MAX-SAT problem is a polynomial in ρ and μ with degree at most $k = \max_{c\in C}\{|v(c)|\}$. Interestingly, the complexity of the computation of this polynomial does not depend on the number of variables n, only on the number of clauses m (linearly) and the maximum number of literals in a clause k (cubic dependence). This means that it can be efficiently computed if the value of k is low.

5. Further analysis of the expectation surface

From (29) we know that the expected fitness after UX and BF is a polynomial in ρ and μ. For the sake of simplicity, we will make the variable change $\xi = 1-2\rho$ and $\eta = 1-2\mu$. The new variables take values in the interval $[-1, 1]$. A zero value in any of the new variables corresponds to a $1/2$ value in the old variable. Using the new variables we can write (29) as:

$$\mathbb{E}_f(\xi,\eta) = \sum_{l=0}^{n}\sum_{r=0}^{l} A^{(r,l)}_{x,y}\xi^r\eta^l, \tag{44}$$

where we limit the sum to the nonzero values ($r \leqslant l$) and we used $\mathbb{E}_f(\xi,\eta)$ instead of $\mathbb{E}\{f(M_\mu(U_\rho(x,y)))\}$ to highlight the dependency of the expectation on the parameters (we omit the solutions x and y). If we plot the expectation against ξ and η we get a surface like the one in Fig. 1.

Since $\xi, \eta \in [-1, 1]$ (the domain is compact) and the expectation is a continuous function of ξ and η there exists a point (ξ^*, η^*) in which the expectation is maximal. For different problems the features of the $\mathbb{E}_f(\xi,\eta)$ functions are different. As an example, in Onemax the maximum expected value is obtained in a point (ξ^*, η^*) which is in the corner of the domain. That is, we obtain the maximum expected fitness when there is no crossover (one of the parents is taken) and the mutation flips all the bits or none. However, this configuration prevents an EA from evolving the population. This example highlights that the parameters of the algorithm maximizing the expected value in one step of the algorithm are not always useful for the evolution of the EA. A more interesting value to optimize would be the probability of improving the solutions in the population. This probability is much harder to compute, but the result of Corollary 3 could be used to compute it.

The features of the expectation surfaces for each problem heavily depend on the Walsh decomposition of the objective function. However, in this section we are interested in the features of the $\mathbb{E}_f(\xi,\eta)$ functions when all the objective functions f are considered. In particular, we wonder if (ξ^*, η^*) can be anywhere in $[-1, 1]^2$ and, more generally, if $\mathbb{E}_f(\xi,\eta)$ can be as similar as we want to any arbitrary continuous and differentiable function of two variables. That is, given an arbitrary smooth function $h(\xi,\eta)$ and $\varepsilon > 0$, can we find an objective function f and solutions x and y such that $|\mathbb{E}_f(\xi,\eta) - h(\xi,\eta)| < \varepsilon$ for $\xi, \eta \in [-1, 1]^2$?

The answer to this question is trivially negative. It is not possible for the expectation surface to be similar to a function having a term in $\xi^r\eta^l$ with $r > l$ because $A^{(r,l)}_{x,y} = 0$ in this case. Although $\mathbb{E}_f(\xi,\eta)$ can only be a polynomial, it can be arbitrarily similar to any function $h(\xi,\eta)$ provided that the Taylor expansion of h converges in $[-1, 1]^2$. However, the condition $A^{(r,l)}_{x,y} = 0$ for $r > l$, implies a similar condition in h, namely:

$$\frac{\partial^{r+l}h}{\partial^r\xi\,\partial^l\eta} = 0 \quad \text{for } r > l. \tag{45}$$

We can then claim that the functions to which $\mathbb{E}_f(\xi,\eta)$ can be arbitrarily similar must have a convergent Taylor expansion in $[-1, 1]^2$ and fulfill (45). But, can $\mathbb{E}_f(\xi,\eta)$ be arbitrarily similar to all the functions with these properties? The answer is positive in this case. Furthermore, we prove in Theorem 2 that we only need to consider instances of the weighted MAX-SAT problem. Let us first define the problem.

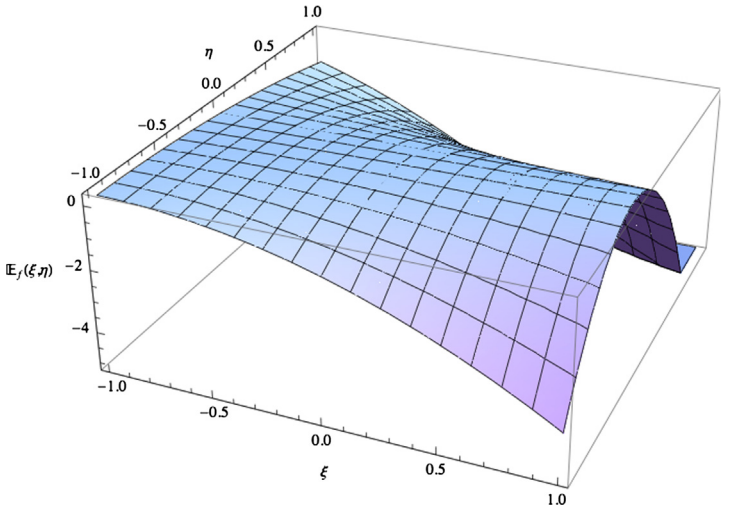

Fig. 1. Expectation surface for a weighted MAX-SAT instance. $\mathbb{E}_f(\xi,\eta) = -(\frac{381}{250}+\xi)\eta - (\frac{23}{10}+\frac{106}{35}\xi+\xi^2)\eta^2$. The maximum expectation is reached in $(\xi^*,\eta^*) = (-0.8,-0.7)$ or $(\rho^*,\mu^*) = (0.9, 0.85)$.

Definition 4. Let C be a set of clauses where each clause $c \in C$ has an associated real weight $w_c \in \mathbb{R}$. The *weighted MAX-SAT* problem consists in finding an assignment of Boolean variables that maximizes the weighted sum of satisfied clauses. The objective function is as follows:

$$f(x) = \sum_{c\in C} w_c f_c(x), \tag{46}$$

where $f_c(x)$ is defined as in (41).

After defining the weighted MAX-SAT problem we can present the next theorem.

Theorem 2. *Let $h(\xi,\eta)$ be an analytic function satisfying (45) whose Taylor expansion centered in $(0,0)$ converges in $[-1,1]^2$. Then, for any $\varepsilon > 0$ we can find an instance of the weighted MAX-SAT problem with objective function f and two solutions x and y such that the expectation surface of the objective function fulfills $|\mathbb{E}_f(\xi,\eta) - h(\xi,\eta)| < \varepsilon$ for $\xi,\eta \in [-1,1]^2$.*

Proof. Since the Taylor expansion of h converges in $[-1,1]^2$, we only need to prove that, for an arbitrary polynomial in the form of (44), we can find an instance of weighted MAX-SAT and two solutions x and y such that the expectation surface of the instance is arbitrarily near to the polynomial. We will prove this by classifying the clauses according to their α, β and γ values and considering the effect of each one in the polynomial coefficients.

If the maximum degree of η is k, then we will consider clauses with k literals. Let us remember here that according to (44) the degree of η must be always higher than or equal to the degree of ξ in an expectation surface. Now, we will impose $x = 0$ and we will restrict the attention to clauses in which $\alpha = 0$. We call these clauses (β,γ)-clauses. Let us analyze what $\alpha = 0$ means. Taking into account the definition of α given in Proposition 6 and considering $x = 0$ we have $\alpha = |v(c) \wedge y \wedge (u(c) \oplus y)|$. The XOR bitwise operator can be written using AND and OR operators in the following way: $u(c) \oplus y = (u(c) \wedge \overline{y}) \vee (\overline{u(c)} \wedge y)$. After a simple manipulation we obtain $\alpha = |v(c) \wedge \overline{u(c)} \wedge y|$. According to Eqs. (39) and (40) the expression $v(c) \wedge \overline{u(c)}$ is a bit mask having 1 in those positions i for which variable x_i is a positive variable appearing in the clause c. Thus, imposing $\alpha = 0$ means that the assignment y does not make true any of the positive literals of the clause c. If we fix y (in addition to $x = 0$) this is the property that characterizes the (β,γ)-clauses. Considering that $x = 0$ and $\alpha = 0$ for the clauses, the value $\beta = |v(c) \wedge y|$ of a clause is the number of negative literals of clause c that are false in the assignment y (because the corresponding variables are true). The meaning of $\gamma = |v(c) \wedge u(c) \wedge \overline{y}| = |u(c) \wedge \overline{y}|$ is the number of negative literals of clause c that are true in the assignment y (because the corresponding variables are false).

Now we use (43) to write the contribution of a (β,γ)-clause with k literals to the coefficient $A^{(r,l)}_{0,y}(c)$ of the g_c function of the clause:

$$A^{(r,l)}_{0,y}(c) = \frac{1}{2^k}\mathcal{K}^{(\beta)}_{r,0}\mathcal{K}^{(k-\beta)}_{l-r,\gamma} = 2^{-k}\binom{\beta}{r}\mathcal{K}^{(k-\beta)}_{l-r,\gamma}, \tag{47}$$

where we should recall here that $\mathcal{K}^{(\beta)}_{r,0} = \binom{\beta}{r}$. A first consequence of (47) is that a clause does not contribute to any coefficient in which $r > \beta$, since the binomial coefficient in (47) is zero in this case. We can build a square matrix, $\mathcal{W}$, in which we write in the columns the contribution of a (β,γ)-clause to all the polynomial coefficients. We have $0 \leqslant \gamma \leqslant k - \beta$, so

if we fix β, γ ranges from 0 to $k-\beta$. We will sort the clauses according to the lexicographical order of (β,γ), and their position in the sorted list will be the column of the matrix $\mathcal{W}$ they have assigned. That is, in column 1 of matrix $\mathcal{W}$ we write the contribution of a $(0,0)$-clause, in column 2 the contribution of a $(0,1)$-clause, and so on. Column $k+1$ has the contribution of a $(0,k)$-clause and from columns $k+2$ to $2k+1$ we write the contributions of the $(1,\gamma)$-clauses, where $0\leqslant\gamma\leqslant k-1$. Next, we write the contributions of the $(2,\gamma)$-clauses, $(3,\gamma)$-clauses, etc. The last column of the matrix contains the contribution of the $(k,0)$-clauses. The number of columns of $\mathcal{W}$ is $(k+1)(k+2)/2$. Regarding the rows, we sort the coefficients $A_{0,y}^{(r,l)}(c)$ according to the lexicographical order of the (r,l) pair. We should recall here that $0\leqslant r\leqslant l\leqslant k$. The number of rows will also be $(k+1)(k+2)/2$, so $\mathcal{W}$ is a square matrix.

All the columns associated to (β,γ)-clauses with the same value of β are contiguous in the matrix and all the rows associated to $A_{0,y}^{(r,l)}(c)$ coefficients with the same value of r are also contiguous. Thus, all the elements associated to the same values of r and β form a block in $\mathcal{W}$. We have already said that the elements for $r>\beta$ are all zero, thus, the matrix is an upper triangular block matrix of the form:

$$\mathcal{W}=\left[\begin{array}{c|c|c|c} \begin{array}{c}r=0,\beta=0\\ block\end{array} & \begin{array}{c}r=0,\beta=1\\ block\end{array} & \cdots & \begin{array}{c}r=0,\beta=k\\ block\end{array}\\ \hline \mathbf{0} & \begin{array}{c}r=1,\beta=1\\ block\end{array} & \cdots & \begin{array}{c}r=1,\beta=k\\ block\end{array}\\ \hline \vdots & \vdots & \ddots & \vdots\\ \hline \mathbf{0} & \mathbf{0} & \cdots & \begin{array}{c}r=k,\beta=k\\ block\end{array}\end{array}\right],$$

where each nonzero block is a $(k-r+1)\times(k-\beta+1)$ matrix with the rows indexed by $l-r$ (from 0 to $k-r$) and the columns indexed by γ (from 0 to $k-\beta$). The content of the (r,β)-block is:

$$\left[\begin{array}{c}r,\beta\\ block\end{array}\right]=\frac{1}{2^k}\binom{\beta}{r}\begin{bmatrix}\mathcal{K}^{(k-\beta)}_{(l-r=)0,(\gamma=)0} & \mathcal{K}^{(k-\beta)}_{0,1} & \cdots & \mathcal{K}^{(k-\beta)}_{0,k-\beta}\\ \mathcal{K}^{(k-\beta)}_{1,0} & \mathcal{K}^{(k-\beta)}_{1,1} & \cdots & \mathcal{K}^{(k-\beta)}_{1,k-\beta}\\ \vdots & \vdots & \ddots & \vdots\\ \mathcal{K}^{(k-\beta)}_{k-\beta,0} & \mathcal{K}^{(k-\beta)}_{k-\beta,1} & \cdots & \mathcal{K}^{(k-\beta)}_{k-\beta,k-\beta}\\ \mathbf{0}^{(\beta-r)} & \mathbf{0}^{(\beta-r)} & \ldots & \mathbf{0}^{(\beta-r)}\end{bmatrix}$$
$$=\frac{1}{2^k}\binom{\beta}{r}\begin{bmatrix}\mathcal{K}^{(k-\beta)}\\ \mathbf{0}^{(\beta-r)\times(k-\beta)}\end{bmatrix},$$

and $\mathbf{0}^{(\beta-r)}$ denotes a column vector with $\beta-r$ zeroes and $\mathbf{0}^{(\beta-r)\times(k-\beta)}$ denotes a matrix of size $(\beta-r)\times(k-\beta)$ with zeroes. Observe that if $r=\beta$, the case of the blocks in the diagonal of $\mathcal{W}$, $\mathbf{0}^{(\beta-r)\times(k-\beta)}$ has no elements and the block does not have rows with zeroes.

An upper triangular block matrix like $\mathcal{W}$ is invertible if the blocks in the diagonal are all invertible. The blocks in the diagonal are proportional to the Krawtchouk matrices, which are invertible, as we argued in Section 2. Thus, matrix $\mathcal{W}$ is invertible.

Let us assume that in our instance of weighted MAX-SAT we have exactly one (β,γ)-clause for all the possible values of β and γ, with $0\leqslant\beta\leqslant k$ and $0\leqslant\gamma\leqslant k-\beta$. We denote with χ the column vector containing the weights w_c of each clause in lexicographical order of (β,γ). Then, the vector $\mathcal{W}\cdot\chi$ contains the coefficients $A_{x,y}^{(r,l)}$ for the function $g=\sum_{c\in C}w_c g_c$. This function is not the objective function of weighted MAX-SAT (46), the relationship between both functions is given by:

$$f(x)=\sum_{c\in C}w_c f_c(x)=\sum_{c\in C}w_c-\sum_{c\in C}w_c g_c(x)=\sum_{c\in C}w_c-g(x),$$

and the relationship between the polynomial coefficients $A_{x,y}^{(r,l)}$ of the two functions is:

$$A_{x,y}^{(r,l)}(f)=\delta_{0,0}^{r,l}\sum_{c\in C}w_c-A_{x,y}^{(r,l)}(g). \tag{48}$$

Let ω be the desired vector of coefficients $A_{x,y}^{(r,l)}(f)$ for the weighted MAX-SAT objective function f, where the coefficients appear in lexicographical order of (r,l). If we ignore for moment the constant $\sum_{c\in C}w_c$ in (48), assuming that it is 0, we can obtain the weights of the weighted MAX-SAT instance solving the linear equation $-\omega=\mathcal{W}\cdot\chi$, that is: $\chi=-\mathcal{W}^{-1}\omega$. Since $\mathcal{W}$ is invertible, the solution always exists. The corresponding weighted MAX-SAT instance will have an expectation surface that is shifted by a constant value with respect to the desired surface $h(\xi,\eta)$. In order to get exactly the desired expectation surface we have to add the constant $\sum_{c\in C}w_c=\mathbf{1}^t\cdot\chi$ to the objective function, where $\mathbf{1}^t$ is a row vector with

ones. We can add this constant by adding two new clauses to the instance, both with weight $\mathbf{1}^t \cdot \chi$ and one literal each: x_i and $\overline{x_i}$, for an arbitrary variable x_i. Exactly one of these two clauses is true for each solution and since both have the same weight the net effect on the instance is to add a constant to the objective function, the constant we need to shift the expectation surface to the desired position.

There is one last issue to solve that has to do with the possibility of generating all the (β, γ)-clauses. We exploit the fact that the number of variables in the problem does not appear in the definition of α, β or γ. Thus, we can always build a new clause of the required type just adding more variables to the instance when necessary and assigning them the appropriate truth value in solution y to build a new clause with the correct values for β and γ (see Example 1 below). □

Now we have a satisfactory answer to the question of how the expectation surface looks like. With the result of the previous theorem we can answer the other question we are interested: can the maximal parameters (ξ^*, η^*) be anywhere in $[-1, 1]^2$?

Corollary 4. *Given a point $(a, b) \in [-1, 1]^2$ where $b \neq 0$, it is always possible to find an instance of weighted MAX-SAT and two solutions x and y such that the expectation surface takes the maximum value in $(\xi^*, \eta^*) = (a, b)$.*

Proof. Let us consider the function:

$$h(\xi, \eta) = 2b\eta + 2ab\xi\eta - \eta^2\left(\xi^2 + 1\right) + d,$$

for an arbitrary constant d. If $b \neq 0$, this function has a maximum in (a, b). In order to check this, we first compute the gradient of $h(\xi, \eta)$:

$$\nabla h(\xi, \eta) = \left(2ab\eta - 2\eta^2\xi, 2ab\xi + 2b - 2\eta\left(\xi^2 + 1\right)\right).$$

The extrema can be found only in the points in which $\nabla h = (0, 0)$. Let us solve the equations:

$$2ab\eta - 2\eta^2\xi = 0,$$
$$2ab\xi + 2b - 2\eta\left(\xi^2 + 1\right) = 0.$$

The first equation of the system has two possible solutions: $\eta = 0$ and $\eta\xi = ab$. If we consider these two possibilities in the second equation we obtain two possible solutions:

$$(\xi, \eta) = (-1/a, 0),$$
$$(\xi, \eta) = (a, b).$$

The value of h in the two points is:

$$h(-1/a, 0) = 0,$$
$$h(a, b) = b^2\left(a^2 + 1\right),$$

and, clearly, $h(a, b) > h(-1/a, 0)$ since $b \neq 0$. The Hessian evaluated in (a, b) is:

$$H(h)_{(a,b)} = \left.\begin{pmatrix} -2\eta^2 & 2ab - 4\eta\xi \\ 2ab - 4\eta\xi & -2(\xi^2 + 1) \end{pmatrix}\right|_{(a,b)} = \begin{pmatrix} -2b^2 & -2ab \\ -2ab & -2(a^2 + 1) \end{pmatrix},$$

where we can observe that it is negative definite, since $-2b^2 < 0$ and the determinant, $4b^2 > 0$. Thus, the point (a, b) is maximal. Due to the negative quartic term $-\eta^2(\xi^2 + 1)$ the value of $h(\xi, \eta)$ has no lower bound, and the global maximum must be that of (a, b).

According to Theorem 2 we can build an instance of weighted MAX-SAT having as expectation surface $h(\xi, \eta)$. □

Example 1. Let us illustrate the results of Theorem 2 and Corollary 4 with an example. Let us suppose that we want an instance of weighted MAX-SAT and two solutions such that the expectation surface has the maximal value in $(\rho^*, \mu^*) = (0.3, 0.6)$, or in ξ, η coordinates, $(\xi^*, \eta^*) = (0.4, -0.2)$. According to Corollary 4 we just need to find an instance and two solutions x and y such that the expectation surface is:

$$h(\xi, \eta) = 2 \cdot (-0.2)\eta + 2 \cdot 0.4 \cdot (-0.2)\xi\eta - \eta^2\left(\xi^2 + 1\right) + d = -0.4\eta - 0.16\xi\eta - \eta^2\left(\xi^2 + 1\right) + d, \tag{49}$$

where d is an arbitrary constant. Let us assume $d = 0$ at this moment.

Let $x = 0$, as in the proof of Theorem 2. The highest degree of η in the polynomial is $k = 2$, and this is the number of literals of the weighted MAX-SAT instance. That is, we are considering a weighted MAX-2SAT instance. The vector of $A^{(r,l)}_{0,y}$ coefficients for this function is:

$$\omega = \begin{pmatrix} A_{0,y}^{(0,0)} \\ A_{0,y}^{(0,1)} \\ A_{0,y}^{(0,2)} \\ A_{0,y}^{(1,1)} \\ A_{0,y}^{(1,2)} \\ A_{0,y}^{(2,2)} \end{pmatrix} = \begin{pmatrix} 0.00 \\ -0.40 \\ -1.00 \\ -0.16 \\ 0.00 \\ -1.00 \end{pmatrix}.$$

The matrix $\mathcal{W}$ in this case is:

$$\mathcal{W} = \frac{1}{4}\left[\begin{array}{ccc|cc|c} 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 0 & -2 & 1 & -1 & 0 \\ 1 & -1 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right].$$

The solution to the linear system $\mathcal{W} \cdot \chi = -\omega$ is:

$$\chi = \begin{pmatrix} 2.24 \\ -0.32 \\ 1.44 \\ -3.68 \\ -3.68 \\ 4.00 \end{pmatrix},$$

which are the weights for the (β, γ)-clauses. Let us now build the clauses.

Let us start with an instance of just two variables and we will add new variables as required. In the following we will denote the boolean variables in the clauses with x_i, but this does not mean that it refers to the solution x. On the other hand, we will use y_i to denote the values of the variables in the assignment y. A $(0,0)$-clause could be $x_1 \vee x_2$ when $y_1 = y_2 = 0$. A $(0,1)$-clause for the same assignment could be $\overline{x_1} \vee x_2$. A $(0,2)$-clause could be $\overline{x_1} \vee \overline{x_2}$. In order to add a $(1,0)$-clause we need a new variable x_3, since we need to make false a negative literal. This can only happen if the variable is true, but variables x_1 and x_2 are false in our assignment. The new clause could be $\overline{x_3} \vee x_1$ with $y_3 = 1$. An example of $(1,1)$-clause could be $\overline{x_3} \vee \overline{x_1}$. Finally, in order to add the $(2,0)$-clause we need a new variable x_4. The clause could be $\overline{x_3} \vee \overline{x_4}$ and $y_4 = 1$. According to the proof of Theorem 2 we should now add two single-variable clauses to shift the objective function of the instance in a constant. However, according to Corollary 4 the expectation surface that takes the only maximum value in (ρ^*, μ^*) can be shifted in an arbitrary constant. Thus, we don't need to adjust the objective function. The only consequence is that the constant d, which was assumed to be 0 at the beginning, will be, in general $d \neq 0$, but the maximum of the expectation surface will still be at $(\rho^*, \mu^*) = (0.3, 0.6)$. In Table 1 we summarize the instance and the two values x and y.

It is not possible to find an instance of a problem for which the only optimal parameter combination is found when $\eta^* = 0$ ($\mu = 1/2$). The reason is that in this case, the bit-flip mutation is equivalent to a random selection of a solution in the search space and the expected fitness after the operators is just $\bar{f}$ (average in the search space) regardless the value of ρ. It is possible, however, to find instances in which the optimal parameters are found in the line defined by $\eta^* = 0$.

Corollary 4 formalizes and generalizes a result that was illustrated in [3]. In that work it was presented a MAX-3SAT instance for which the optimal bias ρ was less than $1/2$. This means that the maximum expected fitness is obtained if the probability of selecting the bits from the worst parent is higher, what is counterintuitive. Now we have proven that it is

Table 1
Instance of weighted MAX-2SAT having the expectation surface given by (49) with solutions $x = 0000$ and $y = 0011$.

Clause	Weight
$x_1 \vee x_2$	2.24
$\overline{x_1} \vee x_2$	−0.32
$\overline{x_1} \vee \overline{x_2}$	1.44
$\overline{x_3} \vee x_1$	−3.68
$\overline{x_3} \vee \overline{x_1}$	−3.68
$\overline{x_3} \vee \overline{x_4}$	4.00

possible to build instances of weighted MAX-SAT for which not only the bias ρ, but also the bit-flip probability μ provides the maximum expected fitness in any arbitrary value.

6. Conclusions and future work

We have derived an expression for computing the expected fitness value of a solution which is the result of applying the uniform crossover and the bit-flip mutation to two solutions x and y. This expression is a function of the bias ρ of the uniform crossover and the probability μ of flipping a bit in the bit-flip mutation. We prove that this function is a polynomial in ρ and μ and the degree of the polynomial is bounded by the number of bits in which x and y differ and the maximum order of the nonzero coefficients in the Walsh decomposition of the objective function. We also provide an expression for computing higher order moments of the fitness distribution after the operators are applied.

We have developed the expression for the expectation as a closed-form formula for two optimization problems, Onemax and MAX-SAT, although we can extend the results also to NK-landscapes and other pseudo-Boolean functions with a known Walsh decomposition. We have also studied the features of the expectation surfaces and derived a way of building an instance of weighted MAX-SAT having the desired surface.

In the future we plan to investigate the utility of these theoretical results in practice. In particular, it is possible to use the first computed moments of the fitness distribution after applying the operators in order to evaluate if the resulting solution is promising or not for the search. The first moments can provide us some bounds on the quality of the solutions obtained. Whether or not this information can be used to create a new search strategy is unknown but we think it deserves more research.

The results of this paper are applicable to the binary hypercube. In the future we should extend them to the generalized hypercube. It would also be interesting to have similar results for the symmetric space (permutations).

Acknowledgements

This research has been partially funded by the Spanish Ministry of Science and Innovation and FEDER under contract TIN2011-28194 (the roadME project). It was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number FA9550-11-1-0088. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

[1] T. Baeck, D.B. Fogel, Z. Michalewicz (Eds.), Evolutionary Computation 1: Basic Algorithms and Operators, Taylor & Francis, 2000.
[2] C. Witt, Optimizing linear functions with randomized search heuristics – the robustness of mutation, in: C. Dürr, T. Wilke (Eds.), 29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012), in: Leibniz International Proceedings in Informatics (LIPIcs), vol. 14, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2012, pp. 420–431.
[3] F. Chicano, D. Whitley, E. Alba, Exact computation of the expectation curves for uniform crossover, in: Proceedings of GECCO, 2012, pp. 1301–1308.
[4] F. Chicano, E. Alba, Exact computation of the expectation curves of the bit-flip mutation using landscapes theory, in: Proceedings of GECCO, 2011, pp. 2027–2034.
[5] A.M. Sutton, D. Whitley, A.E. Howe, Mutation rates of the $(1+1)$-EA on pseudo-boolean functions of bounded epistasis, in: Proceedings of GECCO, 2011, pp. 973–980.
[6] F. Chicano, J. Ferrer, E. Alba, Elementary landscape decomposition of the test suite minimization problem, in: M.B. Cohen, M. Ó Cinnéide (Eds.), Search Based Software Engineering, in: Lect. Notes Comput. Sci., vol. 6956, Springer, 2011, pp. 48–63.
[7] R.B. Heckendorn, Embedded landscapes, Evol. Comput. 10 (4) (2002) 345–369.
[8] J.L. Walsh, A closed set of normal orthogonal functions, Am. J. Math. 45 (1) (1923) 5–24.
[9] M.D. Vose, The Simple Genetic Algorithm: Foundations and Theory, MIT Press, 1999.
[10] A. Terras, Fourier Analysis on Finite Groups and Applications, Cambridge University Press, 1999.
[11] A.M. Sutton, L.D. Whitley, A.E. Howe, Computing the moments of k-bounded pseudo-boolean functions over hamming spheres of arbitrary radius in polynomial time, Theor. Comput. Sci. 425 (2012) 58–74.
[12] P. Feinsilver, J. Kocik, Krawtchouk polynomials and Krawtchouk matrices, in: R. Baeza-Yates, J. Glaz, H. Gzyl, J. Hüsler, J. Palacios (Eds.), Recent Advances in Applied Probability, Springer US, 2005, pp. 115–141.
[13] M.D. Vose, A.H. Wright, The simple genetic algorithm and the Walsh transform: Part i, theory, Evol. Comput. 6 (3) (1998) 253–273.
[14] C. Chryssomalakos, C.R. Stephens, What basis for genetic dynamics?, in: K. Deb (Ed.), Proceedings of GECCO, in: Lect. Notes Comput. Sci., vol. 3102, Springer, 2004, pp. 1018–1029.
[15] A.M. Sutton, F. Chicano, L.D. Whitley, Fitness function distributions over generalized search neighborhoods in the q-ary hypercube, Evol. Comput. 21 (4) (2013) 561–590.
[16] A.M. Sutton, L.D. Whitley, A.E. Howe, A polynomial time computation of the exact correlation structure of k-satisfiability landscapes, in: Proceedings of GECCO, 2009, pp. 365–372.