

University of Montana

ScholarWorks at University of Montana

Graduate Student Theses, Dissertations, &
Professional Papers

Graduate School

2005

An estimation of distribution algorithm based on linkage discovery and factorization

S. V. Pulavarty

The University of Montana

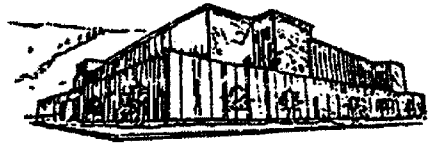
Follow this and additional works at: <https://scholarworks.umt.edu/etd>

Let us know how access to this document benefits you.

Recommended Citation

Pulavarty, S. V., "An estimation of distribution algorithm based on linkage discovery and factorization" (2005). *Graduate Student Theses, Dissertations, & Professional Papers*. 8347.
<https://scholarworks.umt.edu/etd/8347>

This Thesis is brought to you for free and open access by the Graduate School at ScholarWorks at University of Montana. It has been accepted for inclusion in Graduate Student Theses, Dissertations, & Professional Papers by an authorized administrator of ScholarWorks at University of Montana. For more information, please contact scholarworks@mso.umt.edu.



**Maureen and Mike
MANSFIELD LIBRARY**

The University of
Montana

Permission is granted by the author to reproduce this material in its entirety, provided that this material is used for scholarly purposes and is properly cited in published works and reports.

****Please check "Yes" or "No" and provide signature****

Yes, I grant permission _____

No, I do not grant permission _____

Author's Signature: P.S.V.P.M. Sandeep

Date: 11/30/05

Any copying for commercial purposes or financial gain may be undertaken only with the author's explicit consent.

**An Estimation of Distribution Algorithm Based on Linkage Discovery and
Factorization**

by

S.V.P.M.Sandeep Pulavarty

B.Tech, Jawaharlal Nehru Technological University, 2003

presented in partial fulfillment of the requirements

for the degree of

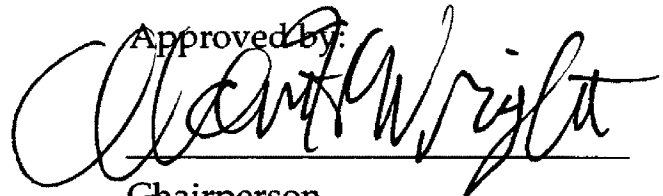
Master of Science

The University of Montana

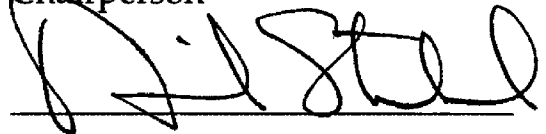
Missoula, Montana

December, 2005

Approved by:



Chairperson



Dean, Graduate School

12-16-05

Date

UMI Number: EP39148

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI EP39148

Published by ProQuest LLC (2013). Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.

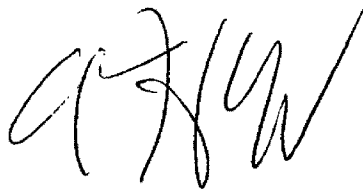
All rights reserved. This work is protected against unauthorized copying under Title 17, United States Code



ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

An Estimation of Distribution Algorithm Based on Linkage Discovery and Factorization

Chairperson: Alden H. Wright



Estimation of Distribution Algorithms (EDA) are a class of algorithms that construct an explicit probabilistic model of distribution based on high fitness individuals in the search space. New individuals are generated by sampling this distribution. The generated individuals guide in constructing the probability distribution for next iteration. For a black box function with k -bounded epistasis that satisfies a property called running intersection property, we show that it is possible to determine the optimum with high probability. This is done by applying the linkage detection algorithm on the black box function, which gives an additively decomposable structure of the black box function. The Boltzmann distribution of a fitness function is the exponential of the fitness normalized to a probability distribution. The factorization of the Boltzmann distribution for the additively decomposable structure is then computed by using the factorization theorem proposed by Mühlenbein et al. The constructed factorization is then sampled to determine the optimum with high probability. As the exponentiation factor in Boltzmann distribution is increased, the probability will be concentrated near optimal points.

•

ACKNOWLEDGMENTS

I would like to take this opportunity to express my thanks to Dr. Alden Wright for guiding me throughout my thesis. I would also thank Dr. Changwon Yoo and Dr. George McRae for offering valuable suggestions in finishing my thesis.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	iii
CHAPTER 1 INTRODUCTION	1
Introduction	1
Thesis Organization	2
CHAPTER 2 OVERVIEW	3
Notation	3
Background	3
Estimation of Distribution Algorithms (EDA's)	6
Walsh Coefficients	7
Factorization of Probability	11
Factorized Distribution Algorithm (FDA)	13
Determining ADF Structure	14
CHAPTER 3 METHODS	16
EDA Based on Linkage Discovery and Factorization	16
Estimating the Number of Optima	21
CHAPTER 4 RESULTS	23
Experiment 1 : Effect of inverse temperature on optimal points	23
Experiment 2 : Complexity of linkage detection algorithm	24

Experiment 3 : Estimating on the number of optima	25
CHAPTER 5 CONCLUSIONS AND FUTURE DIRECTIONS	28
Conclusions	28
Future Directions	28
BIBLIOGRAPHY	29

LIST OF TABLES

3.1	Fitness Values.	18
3.2	Distribution for $p(x_3 x_2)$	19
3.3	Distribution for $p(x_2 x_1)$	19
3.4	Distribution for $p(x_0x_1)$	19

LIST OF FIGURES

4.1	Prob. of maximum fit individual over different string lengths	24
4.2	Number of function evals	26

CHAPTER 1 INTRODUCTION

Introduction

Evolutionary computation is a set of programming techniques based on the concept of evolution to solve complex problems. Evolutionary computation algorithms use a population to guide the search for high quality individuals in the search space. The quality of an individual can be determined by a *fitness function*, which can be defined as a function that takes an individual as input and returns a fitness value proportional to the quality of the individual.

Estimation of Distribution Algorithms (EDA) are a set of evolutionary algorithms that use explicit probability distributions to guide the search for good solutions. The probability distribution is constructed using the population to reflect important characteristics. This distribution is then sampled to generate a new population. This preserves all important characteristics from the previous population and introduces diversity into the new population. This process is repeated until the termination criteria are met.

A fitness function is said to be a *black box fitness function* if we have minimum prior knowledge about it. The Additively Decomposed Function(ADF) structure of a black box fitness function with k bounded epistasis can be determined using the randomized algorithm described by Heckendorn and Wright in [5]. Muhlen-

bein et al [11] proved a factorization theorem that gives a condition called the *Running Intersection Property* (RIP) under which the Boltzmann distribution of an ADF can be factorized. The factorized Boltzmann distribution can then be efficiently sampled to determine high fitness individuals.

Thesis Organization

The rest of this thesis is organized as follows:

- **Chapter 2** provides an overview of related literature, key concepts, and software elements used in this research.
- **Chapter 3** describes a new evolutionary computation algorithm that was developed using linkage detection and factorization.
- **Chapter 4** describes the implementation, test data sets, and results.
- **Chapter 5** contains conclusion remarks, as well as an outline for future work.

CHAPTER 2 OVERVIEW

Notation

In this paper, individuals are represented by strings of length ℓ defined over the binary alphabet $\mathbf{B} = \{0, 1\}$. The set of string positions is denoted by $\mathcal{L} : \{0, 1, \dots, \ell - 1\}$ and the set of all possible bit strings is denoted by \mathbf{X} . Some bit strings are interpreted as masks that specify sets of bit positions. If a and b are interpreted as masks then $a \subseteq b$ iff $a_i = 1$ implies $b_i = 1$. The **bitcount** function $bc(x)$ denotes the number of bits set to 1 in bit string x . For any given string $m \in \mathbf{X}$, \mathbf{X}_m denotes the set $\{x \in \mathbf{X} : x \subseteq m\}$, and \bar{m} is the compliment of m . We use binary operators \wedge and \oplus over bit strings, where \wedge denotes bitwise AND, and \oplus denotes EXCLUSIVE-OR. The *projection* operator $\pi_s : \mathbf{X} \rightarrow \mathbf{X}_s$ projects a string to the bits specified by s . For example, $\pi_{\{1,3\}}x_0x_1x_2x_3 = x_1x_3$.

Background

An *Additively Decomposed Function* (ADF) is a function which can be written as a sum of simpler sub-functions each of which depends on a smaller number of string positions. An ADF is called *k-epistatic* if each sub-function depends on at-most k

string positions. For example, consider a fitness function g which is defined as

$$g(x_0x_1x_2x_3) = g_1(x_0x_1) + g_2(x_2)$$

where $g_1(x_0x_1) = x_0 * x_1$, $g_2(x_2) = x_2$ and x_0, x_1, x_2, x_3 are bits at different string positions. In this function, g takes 4 bits as input, whereas g_1 and g_2 take fewer number of bits as inputs.

If $s_1 = \{0, 1\}$ and $s_2 = \{2\}$, then the above defined ADF can be written using projection as

$$g(x) = g_1(\pi_{s_1}x) + g_2(\pi_{s_2}x)$$

In general, an ADF can be written as a sum of n simpler sub-functions as

$$f(x) = \sum_{i=1}^n f_i(\pi_{s_i}x) \quad (2.1)$$

where x is a bit string of length ℓ , and s_i is a set of bit positions on which sub-function f_i depends on.

A bit position is said to *contribute* to fitness function f if there exists a background string x (i.e. $x_0x_1\dots x_{\ell-1}$) such that flipping that bit in x changes the fitness value. In other words, if

$$f(x_0x_1\dots x_i\dots x_{\ell-1}) \neq f(x_0x_1\dots \bar{x}_i\dots x_{\ell-1})$$

then bit position x_i is said to *contribute* to f .

The *support* of a function is the set of bit positions that contribute to the function. For example, in order to determine whether position 2 contributes to the above defined function g , calculate the difference between $g(zz0z)$ and $g(zz1z)$, where z 's represent a constant background made up of 0's and 1's, that doesn't change from

evaluation to evaluation. If there exists a background such that the difference is non-zero, then position 2 is said to contribute to the fitness function. In the above example, it is easy to see that positions 0, 1 and 2 contribute to g , whereas 3 does not. So the support of g is $\{0, 1, 2\}$.

The contribution of a bit position might depend not only on its bit value, but also on the state of other bits in the domain with which it interacts. This dependency among bits is called *epistasis*. In order to determine whether two bits epistatically interact in a function f , let a and b be masks with $bc(a) = bc(b) = 1$ that specify singleton sets consisting of these bit positions, and c be a constant background such that $bc(c \wedge a) = bc(c \wedge b) = 0$. Then we compute

$$f(c) - f(c \oplus a) - f(c \oplus b) + f(c \oplus a \oplus b)$$

If the result is non-zero for some background string c , then the two bits are epistatically linked. Epistasis for a general set of bits can be determined using a *probe* [5] which is defined as

$$P(f, m, c) = \frac{1}{2^{bc(m)}} \sum_{i \in \mathbf{X}_m} (-1)^{bc(i)} f(i \oplus c) \quad (2.2)$$

where f is the fitness function, $m \in \mathbf{X}$ is a mask specifying the set of bits to be tested for epistasis, \mathbf{X}_m is the set $\{x \in \mathbf{X} : x \subseteq m\}$ and c is a background with the property $bc(c \wedge m) = 0$.

A set of bits are epistatically linked if the contribution of a bit position depends on the state of other bits in the set. The string m represents an *epistatic block* if there exists a background c such that $P(f, m, c) \neq 0$. In other words, the set of bits in an epistatic block are epistatically linked. The order of the probe $P(f, m, c)$ is $bc(m)$.

Using the probe recursion theorem in [5], it can be shown that any subset of an epistatic block is also an epistatic block. A function f has *k-bounded epistasis* if the number of elements in an epistatic block can be at most k .

A *schema* is a string of fixed length which contains special asterisk (*) characters that can denote any string symbol at that position. The *order* of a schema is the number of defined (non-asterisk) positions, and a schema is *contiguous* if all the defined positions are adjacent to each other.

Estimation of Distribution Algorithms (EDA's)

EDA's can be classified into three categories based on the interaction between string positions. The first category of EDA's assume that there are no interdependencies between string positions. Some of the examples in this category are Population Based Incremental Learning (PBIL)[1], Univariate Marginal Distribution Algorithm (UMDA)[7], Compact Genetic Algorithm (cGA)[4] etc. This class of EDA's works well on linear problems and often fails on problems where there are strong interactions between variables. The second category of EDA's allows pairwise interaction between variables. Examples for this category include Mutual Information Maximizing Input Clustering (MIMIC)[2], Bivariate Marginal Distribution Algorithm (BMDA)[13] etc. As this class of EDA's take order two interactions into account, they works well for linear and quadratic problems and often fails for problems with higher order interactions. The third category deals with multivariate dependencies and they construct complex models for the problem to be solved. The probability distributions for this category can be constructed based on different approaches such as Bayesian Optimization Algorithm[12], Factorization of Distribution Algorithms[10] etc.

One approach to consider multivariate dependencies is to use the principle of *maximum entropy*, which states that the distribution should agree with what is known and at the same time it should express maximum uncertainty. This gives the flexibility to adjust the distribution when new data is available. Wright et al [17] constructed the probability distribution based on the maximum entropy principle constrained by schema frequencies of contiguous schemata. This distribution is sampled on each iteration to obtain a new population which undergoes the selection process based on fitness of the individuals. The more highly fit individuals obtained from selection are used to construct a new distribution. This process is repeated until the termination criteria is met.

Another approach to construct the probability distribution is to take the *Boltzmann distribution* of a fitness function, which is the exponential of the fitness normalized to a probability distribution. As the exponentiation factor increases, the probability distribution concentrates near optimal points. Thus the Boltzmann distribution with high exponentiation factor can be sampled to determine optimal points, but the normalization process is not feasible for higher string lengths, as it requires the fitness of all strings to be enumerated.

Walsh Coefficients

To determine the structure of a blackbox fitness function, a change of basis from standard basis to Walsh basis provides an intuitive insight on the interaction of bitwise nonlinearities within the function. The Walsh basis is powerful and functionally complete in that it can represent any real-valued function that can be de-

defined over the space of boolean strings. The fitness function $f : \mathbf{B}^\ell \rightarrow \mathfrak{R}$ can be represented as an ADF using Walsh functions as

$$f(x) = \sum_{j \in \mathbf{X}} w_j \psi_j(x) \quad (2.3)$$

where $\psi_j(x)$ is the j^{th} Walsh function, and w_j is the j^{th} Walsh coefficient.

The j^{th} Walsh function can be defined as:

$$\psi_j(x) = (-1)^{bc(j \wedge x)}$$

So a Walsh function can return either -1 or 1 based on the value of $bc(j \wedge x)$. It is not hard to show that j is the the mask of support set ψ_j . It can be shown that the j^{th} Walsh coefficient can be computed from f by [3]

$$w_j = \frac{1}{2^\ell} \sum_x f(x) \psi_j(x) \quad (2.4)$$

For example, if the bit string is of length $\ell = 4$, then w_{0000} represents Walsh coefficient for the mask 0000, w_{0001} for 0001 and so on. A maximal non-zero walsh coefficient is a Walsh coefficient w_m such that $w_m \neq 0$ and $w_j = 0 \forall j \supset m$.

Equation 2.3 can be represented using vector notation as

$$f = \Psi w$$

where Ψ is the transformation vector with $\Psi_{i,j} = \psi_i(j)$ that maps walsh coefficients vector w onto the function space f . To change from Walsh basis to standard basis back, we perform inverse transformation. As $\psi_i(j) = \psi_j(i)$, Ψ will be symmetric.

The inverse of Ψ can be written as

$$\Psi^{-1} = \frac{1}{2^\ell} \Psi$$

Equation 2.4 can be written in vector notation as

$$w = \frac{1}{2^\ell} \Psi f$$

Lemma 1. *If $m \not\subseteq k$, then $P(\psi_k, m, c) = 0$ for all c with $bc(c \wedge m) = 0$.*

Proof. Let $u = k \wedge m$ and $v = u \wedge m$, so that $m = u \oplus v$. Using probe recursion theorem in [5], we can write

$$\begin{aligned} P(\psi_k, m, c) &= \frac{1}{2^{bc(u)}} \sum_{i \subseteq u} (-1)^{bc(i)} P(\psi_k, m \oplus u, i \oplus c) \\ &= \frac{1}{2^{bc(u)}} \sum_{i \subseteq u} (-1)^{bc(i)} P(\psi_k, v, i \oplus c) \end{aligned}$$

From equation 2.2

$$\begin{aligned} P(\psi_k, v, i \oplus c) &= \frac{1}{2^{bc(v)}} \sum_{j \in X_v} (-1)^{bc(j)} \psi_k(i \oplus j \oplus c) \\ &= \frac{1}{2^{bc(v)}} \psi_k(i \oplus c) \sum_{j \in X_v} (-1)^{bc(j)} = 0 \end{aligned}$$

The last step is justified as ψ_k doesn't depend on any bit positions set to 1 in v , and as $j \in X_v$ the probe is independent of j . In the summation, as the number of positive terms equals the number of negative terms, $P(\psi_k, v, i \oplus c) = 0$. Hence proving the lemma. \square

Lemma 2. *In m is the mask of an epistatic block of a function f , then $\exists a$ such that $m \subseteq a$*

and $w_a \neq 0$.

Proof. This lemma can be proved by contradiction. Lets assume that $m \subseteq a$, implies $w_a = 0$. As the bits specified by m forms an epistatic block, probing on m should return a non-zero value. ie. there exists a background c such that $P(f, m, c) \neq 0$.

From equation 2.3,

$$P(f, m, c) = \sum_{k \in X} P(w_k \psi_k, m, c)$$

and from previous lemma

$$P(w_k \psi_k, m, c) = 0 \quad \text{if } m \not\subseteq k$$

so we are left with only w_a probes, where $m \subseteq a$. We assumed that all $w_a = 0$ implies $P(f, m, c) = 0$, which is a contradiction. \square

Lemma 3. *Given a function f and a mask m such that $w_m \neq 0$, then m is an epistatic block for f .*

Proof. If $w_m \neq 0$ then we know that \exists a maximal walsh coefficient w_a such that $w_a \neq 0$ and $m \subseteq a$. [5] proved that, if w_a is the maximal walsh coefficient, then $\forall c \in X_{\bar{a}}$

$$P(f, a, c) = w_a \neq 0$$

As the probe on mask a is non-zero, the bits specified by a forms an epistatic block. Any subset of an epistatic block is an epistatic block. As $m \subseteq a$, m forms an epistatic block. \square

Factorization of Probability

The following summarizes the material from [11]. The Gibbs or Boltzmann distribution¹ of a function f is defined for $u \geq 1$ by

$$p(x) = \frac{\text{Exp}_u f(x)}{\sum_y \text{Exp}_u f(y)} \quad (2.5)$$

where for notational convenience

$$\text{Exp}_u f(x) = e^{u f(x)} \quad F_u = \sum_y \text{Exp}_u f(y) \quad (2.6)$$

In general, the computation of this distribution requires 2^ℓ parameters, which is not efficient for large values of ℓ .

If f is an ADF, then

$$p(x) = \frac{1}{F_u} \prod_{i=1}^n e_i(x) \quad (2.7)$$

where

$$e_i(x) = \text{Exp}_u f_i(\pi_{s_i} x)$$

Notice that e_i takes a string of length ℓ as argument, whereas f_i is a function only of the bits specified by s_i .

Let $p(\pi_{s_i} x)$ denote marginal probability. In other words,

$$p(\pi_{s_i} x) = \sum_{y \in \mathbf{X}_{\mathcal{L} \setminus s_i}} p(x_{s_i} \oplus y) \quad (2.8)$$

where $x_{s_i} \in \mathbf{X}_{s_i}$ such that $\pi_{s_i} x = \pi_{s_i} x_{s_i}$. Thus, x_{s_i} is a string of length ℓ whereas $\pi_{s_i} x$

¹Remark: The Boltzmann distribution is usually defined as $e^{-\frac{g(x)}{T}} / Z$. The term $g(x)$ is called the energy. Setting $g(x) = -f(x)$ and $1/T = \ln(u)$ gives the above equation.

is a string of length s_i . Under conditions given below, the Boltzmann distribution $p(x)$ of an ADF can be *factorized* so that $p(x)$ can be computed efficiently from sub-functions of the ADF [11]. Given a fitness function f that can be written as an ADF with sub-functions f_i 's whose support masks are s_i 's, then we compute new sets d_i, b_i and c_i as

$$d_i := \bigcup_{j=1}^i s_j \quad (2.9)$$

$$b_i := s_i \setminus d_{i-1} \quad (2.10)$$

$$c_i := s_i \cap d_{i-1} \quad (2.11)$$

we set $d_0 := \emptyset$. s_k is called the *successor* of s_i , if $c_k \subseteq s_i$.

Theorem 1 (Factorization Theorem [11]). *Let $f = \sum_{i=1}^n f_i(\pi_{s_i}x)$ be an ADF and $p(x)$ be its Boltzmann distribution, if*

$$b_i \neq \emptyset \quad \forall i = 1, \dots, n; \quad d_n = \mathcal{L} \quad (2.12)$$

$$\forall i \geq 2 \exists j < i \text{ such that } c_i \subseteq s_j \quad (2.13)$$

Then

$$p(x) = \prod_{i=1}^n p(\pi_{b_i}x | \pi_{c_i}x) \quad (2.14)$$

where $p(\pi_{b_i}x | \pi_{c_i}x)$ is the probability of $\pi_{b_i}x$ given $\pi_{c_i}x$.

The assumption in equations 2.12 and 2.13 is called the *running intersection property*.

It follows from [11] that if f can be written as an ADF with running intersection property and $\mathcal{S}(i)$ is the successor set of s_i , then for any $x \in \mathcal{X}$

$$p(\pi_{b_i}x|\pi_{c_i}x) = \frac{e_i(x_{b_i} \oplus x_{c_i}) \prod_{k \in S(i)} \sum_{z \in X_{b_k}} e_k(z \oplus x_{b_i})}{\sum_{w \in X_{b_i}} e_i(w \oplus x_{c_i}) \prod_{k \in S(i)} \sum_{z \in X_{b_k}} e_k(z \oplus w)} \quad (2.15)$$

Factorized Distribution Algorithm (FDA)

FDA[11] is an EDA which assumes that the function to be optimized is an ADF and uses factorization of the distribution computed using individuals selected based on fitness. It allows nonlinear interaction between variables in the fitness function. A portion of the initial population is generated based on local approximation of the conditional marginal distribution and the rest is generated randomly.

Step 0 : Set $t \leftarrow 0$. Generate $(1 - r) * N \gg 0$ individuals randomly and $r * N$ individuals according to equation 2.15

Step 1 : Select high fitness individuals using a selection method

Step 2 : Compute conditional probabilities $p^s(\pi_{b_i}x|\pi_{c_i}x, t)$ using the individuals selected in Step 2.

Step 3 : Generate a new population based on $p(x, t+1) = \prod_{i=1}^n p^s(\pi_{b_i}x|\pi_{c_i}x, t)$

Step 4 : If termination criteria is met, FINISH

Step 5 : Add the best individual from previous generation the to population generated in Step 3.

Step 6 : Set $t \leftarrow t + 1$. Go to Step 2.

Any popular selection method can be used with FDA and the factorization can be either exact or approximate factorization.

Determining ADF Structure

A *hypergraph* G is defined as a pair (V, E) where V is the set of *vertices* and E is a set of *hyperedges* between vertices. Each hyperedge is a set of vertices in V . Heckendorn and Wright [5] presented two algorithms *Traverse Hypergraph* and *Compute Walsh Coefficients* that can be used in conjunction to determine the ADF structure of the blackbox function f with k bounded epistasis. The *Traverse Hypergraph* algorithm constructs a hypergraph for the function f , in which a vertex represents a string position, and a hyperedge represents a set of positions that form an epistatic block. It detects hyperedges by doing a breadth-first traversal on all possible masks starting with empty mask, then order-1 mask, order-2 mask etc. A mask is tested by probing only if all subsets of that mask have already been determined to be epistatic blocks. It detects all order- j hyperedges, by probing N times (Theorem 9 in [5]) with different backgrounds. N is chosen using

$$N \geq \begin{cases} -2^{k-j} \ln(1 - \delta^{1/J}) & \text{if } j < k \\ 1 & \text{if } j = k \end{cases} \quad (2.16)$$

where J is the number of order- j hyperedges and $\delta < 1$ is the probability with which all order- j hyperedges are to be detected. Note that $\delta < 1$, so this phase is not guaranteed to determine all order- j hyperedges. δ can be made close to 1 (like 0.9999) such that it determines order- j hyperedges with high probability.

The *Compute Walsh Coefficients* algorithm takes a list of hyperedges and their corresponding probe values that are computed by *Traverse Hypergraph* as arguments, and determines the Walsh coefficient for each hyperedge. Each maximal Walsh coefficient represents a sub-function in the ADF structure of function f (Theorem 22 in [5]).

If the determined ADF structure has the running intersection property, then it can then be used to determine the probability of the optimum using *Factorization of Probability* introduced in section 2. By increasing the exponentiation factor in Boltzmann selection, the probability of selecting high fit individuals increases. So if the exponentiation factor is sufficiently large, then the optimal string will have a very high probability when compared to other strings.

Complexity Analysis:

The complexity of algorithm shows the way in which it behaves for larger problems. In this thesis, the complexity is measured in terms of the number of functions evaluations required to execute the algorithm. For example, if the complexity of an algorithm is defined as $O(\ell)$, it means that the number of computations required to execute that algorithm is proportional to ℓ . It can be defined more formally as:

The complexity of a function f is $O(g(\ell))$, if $\exists c_1 \geq 0, \ell_0 \geq 0$ such that $0 \leq f(\ell) \leq c_1 \times g(\ell) \forall \ell \geq \ell_0$. In other words, when $\ell \geq \ell_0$, $c_1 \times g(\ell)$ will be the upper bound of $f(\ell)$. The complexity of a function f is $\theta(g(\ell))$, if $\exists c_1 \geq 0, c_2 \geq 0$ and $\ell_0 \geq 0$ such that $0 \leq c_1 \times g(\ell) \leq f(\ell) \leq c_2 \times g(\ell) \forall \ell \geq \ell_0$. In other words, when $\ell \geq \ell_0$, $c_1 \times g(\ell)$ will be the lower bound of $f(\ell)$ and $c_2 \times g(\ell)$ will be the upper bound of $f(\ell)$.

If the number of maximum Walsh coefficients in *Compute Walsh Coefficients* algorithm is $O(\ell)$, then the expected number of function evaluations of *Traverse Hypergraph* and *Compute Walsh Coefficients* algorithms is $O(\ell^2 \log \ell)$ (corollary 21 in [5]).

CHAPTER 3 METHODS

EDA Based on Linkage Discovery and Factorization

Wright and Pulavarty [18] devised an algorithm with three phases to determine the high fitness individuals for a given black box fitness function with k bounded epistasis and RIP. The first phase determines the ADF structure of the black box function with k -bounded epistasis. The linkage detection algorithm proposed by Heckendorn and Wright [5] is applied on the given black box fitness function. It determines with success probability δ arbitrarily close to 1, the complete epistatic structure in terms of Walsh coefficients of the black box function as outlined in Determining ADF Structure section of chapter 2. As each maximal Walsh coefficient represents an epistatic block (lemma 3 in chapter 2), an additively decomposable structure of the given black box function can be determined easily. The fitness values of sub functions can be determined by performing inverse Walsh transform on the determined Walsh coefficients. As the linkage detection algorithm works with a success probability $\delta < 1$, this phase is not guaranteed to determine the ADF structure. δ can be increased to a value close to 1 (like 0.9999) such that the algorithm works with high probability.

The second phase computes the exact factorization of the Boltzmann distribution for an additively decomposable structure determined in first phase. This phase leverages the factorization theorem outlined in Factorization of Probability

section chapter 2. to determine the exact factorization with less number of computations. It is important to note that the factorization theorem assumes RIP for the additively decomposable structure determined phase one. If the RIP is not satisfied, the factorization of the Boltzmann distribution can be approximated either by choosing only a subset of sub functions of the additively decomposable structure or by merging sub functions or a combination of both. If the RIP is satisfied, this phase doesn't have any point of failure. In this thesis, we assume that the determined ADF satisfies RIP.

The final phase samples the factorized Boltzmann distribution to determine high fitness individuals. As the exponentiation factor increases, the probability will be concentrated near optimal points. During the sampling process, we can guarantee (assuming phase one works) that the individuals with maximum probability are optima if we can find n distinct individuals each with a probability X such that $1 - nX \leq X$. In other words, the total probability of n individuals is nX and the probability of rest of the individuals in the search space is $1 - nX$. If this value is less than X , then the determined points will be the optima.

The following example demonstrates phases 2 and 3 of our algorithm:

As the phase 2 of our algorithm assumes that the ADF structure with RIP will be determined by phase 1 of our algorithm, let's say that the determined ADF structure that satisfies RIP for the given black box function with k -bounded epistasis is

$$f(x) = f_1(\pi_{s_1}x) + f_2(\pi_{s_2}x) + f_3(\pi_{s_3}x) \quad (3.1)$$

Where $s_1 = \{0, 1\}$, $s_2 = \{1, 2\}$ and $s_3 = \{2, 3\}$.

$\pi_{s_i}x$	$f_1(\pi_{s_1}x)$	$f_2(\pi_{s_2}x)$	$f_3(\pi_{s_3}x)$	$Exp_v f_1(\pi_{s_1}x)$	$Exp_v f_2(\pi_{s_2}x)$	$Exp_v f_3(\pi_{s_3}x)$
00	2.36	0.73	1.49	10.5910	2.0751	4.4371
01	0.69	0.14	0.14	1.9937	1.1503	1.1503
10	0.95	0.27	0.94	2.5857	1.3100	2.5600
11	1.64	0.41	1.08	5.1552	1.5068	2.9447

Table 3.1 Fitness Values.

Also, let's assume that the values tabulated in Table 3.1 represents the fitness values of sub functions obtained by performing inverse Walsh transform. The exponent values in Table 3.1 are computed using $v = 1$.

For example, the probability of selecting an individual $x = 0110$ can be computed using equation 2.7 as

$$p(0110) = \frac{Exp_u f_1(01) * Exp_u f_2(11) * Exp_u f_3(10)}{F_u} \quad (3.2)$$

$$= \frac{1.9937 * 1.5068 * 2.56}{347.8374} = 0.02211 \quad (3.3)$$

The probability of an individual can also be computed using factorization theorem, which is used in phase 2 of our algorithm. The first step in the factorization theorem is to construct the required sets as shown below:

We set $d_0 = \emptyset$, and the other d_i 's, b_i 's and c_i 's are computed using equations 2.9, 2.10 and 2.11 as

$$\begin{array}{lll} d_1 = \{0, 1\} & b_1 = \{0, 1\} & c_1 = \{\} \\ d_2 = \{0, 1, 2\} & b_2 = \{2\} & c_2 = \{1\} \\ d_3 = \{0, 1, 2, 3\} & b_3 = \{3\} & c_3 = \{2\} \end{array}$$

x_2	$p(x_3 = 0 x_2)$	$p(x_3 = 1 x_2)$
0	0.7941	0.2059
1	0.4651	0.5349

Table 3.2 Distribution for $p(x_3|x_2)$.

x_1	$p(x_2 = 0 x_1)$	$p(x_2 = 1 x_1)$
0	0.6468	0.3532
1	0.4688	0.5312

Table 3.3 Distribution for $p(x_2|x_1)$.

x_0x_1	$p(x_0x_1)$
00	0.5458
01	0.0895
10	0.1333
11	0.2314

Table 3.4 Distribution for $p(x_0x_1)$.

As the sets b_1, b_2 and b_3 are not empty and $c_2 \subseteq s_1, c_3 \subseteq s_2$, running intersection property is not violated. So the factorization theorem holds. It is easy to see that the successor of s_1 is s_2 , and the successor of s_2 is s_3 , and there are no successors for s_3 . The conditional probability tables can be computed using 2.15.

For example,

$$p(x_3 = 0|x_2 = 1) = \frac{e_3(0000 \oplus 0010)}{e_3(0000 \oplus 0010) + e_3(0001 \oplus 0010)} = 0.4651$$

and

$$\begin{aligned} p(x_2 = 1|x_1 = 1) &= \frac{e_2(0010 \oplus 0100) * \sum_{z \in X_{b_3}} e_3(z \oplus 0010)}{\sum_{w \in X_{b_2}} e_2(w \oplus 0100) * \sum_{z \in X_{b_3}} e_3(z \oplus w)} \\ &= 0.5312 \end{aligned}$$

The probability distributions $p(x_3|x_2), p(x_2|x_1)$ and $p(x_0x_1)$ are tabulated in tables 3.2, 3.3 and 3.4 respectively.

From equation 2.14, the probability of selecting an individual $x = x_0x_1x_2x_3$ is given by

$$p(x) = p(x_0x_1)p(x_2|x_1)p(x_3|x_2) \quad (3.4)$$

For Example,

$$\begin{aligned} p(0110) &= p(x_0 = 0, x_1 = 1)p(x_2 = 1|x_1 = 1)p(x_3 = 0|x_2 = 1) \\ &= 0.0895 * 0.5312 * 0.4651 = 0.02211 \end{aligned}$$

This value matches with the value that is directly computed in equation 3.2, which verifies the factorization theorem.

The third phase of our algorithm samples the Boltzmann distribution based on equation 2.14. In order to sample this Boltzmann distribution, we consider $p(x_0x_1)$ first as x_0x_1 does not depend on any other bit positions. We then select values for x_0x_1 based on the probabilities tabulated in table 3.4. Lets say 00 is selected as it has a probability of 0.55. Now we need to determine the value of x_2 using $p(x_2|x_1)$. As the value of x_1 is already known, we are only left with two choices in Table 3.3. $p(x_2|0)$ can be selected with probability of either 0.65 or 0.35. Lets assume that 0.35 is selected, which gives a value of 1 to x_2 . The next step is to determine x_3 using $p(x_3|x_2)$. As the value of x_2 is 1, x_3 can be assigned a value of 0 or 1 with probabilities 0.47 and 0.53 respectively. Lets assume that x_3 is assigned a value of 1. So the values of x_1, x_2 and x_3 are 0, 1 and 1 respectively, and the value of x that is determined by sampling becomes 0011. This process is repeated until the required number of individuals are sampled.

Estimating the Number of Optima

In an exact Boltzmann distribution, the probability of an optimal point will be the same as any other optimal point. Let f_{opt} and f_{subopt} be the fitnesses of optimum and suboptimum respectively. Probability of an optimum point is

$$p(Opt) = \frac{e^{uf_{opt}}}{F_u}$$

Probability of an subptimum point is

$$p(SubOpt) = \frac{e^{uf_{subopt}}}{F_u}$$

where u is the inverse temperature, and F_u is the summation as described in equation 2.6.

$$\frac{p(Opt)}{p(SubOpt)} = \frac{e^{uf_{opt}}}{e^{uf_{subopt}}}$$

If K is the ratio of the probabilities of optimum and suboptimum, then

$$K = e^{u(f_{opt} - f_{subopt})}$$

$$\ln K = u(f_{opt} - f_{subopt})$$

$$u = \frac{\ln K}{f_{opt} - f_{subopt}}$$

In general, if the ratio of probabilities of optimum and suboptimum is K , then the temperature should be atleast

$$u \geq \frac{\ln K}{f_{opt} - f_{subopt}} \quad (3.5)$$

An optimal point and its corresponding probability can be determined under certain conditions by applying the algorithm using factorization that we described in chapter 2. If an optimal point is known, then f_{opt} can be determined using the fitness function. For some fitness functions, the fitness values are constrained to lie in a discrete set. For example, sometimes fitness values are known to be integers.

In a Boltzmann distribution, as the inverse temperature increases the probability concentrates near optimal points. If the inverse temperature is sufficiently high [18], then the difference between the probabilities of optimum and suboptimum will be high. If α is the probability of the optimal set of points, then the number of optima can be computed by $\alpha/p(Opt)$. The inverse temperature can be made sufficiently high using equation 3.5 such that α tends to 1. So the upper bound on the number of optimal points can be determined using $1/P(Opt)$. It is hard to determine the lower bound on the number of optima, as there can be less number of optima and multiple suboptima, or multiple optima and less number of suboptima. For example, let $P(Opt) = 0.1, P(SubOpt) = 0.001$. In this case, the difference between probabilities of optimum and suboptimum is more. It is possible that there might be only 1 optimum and 900 suboptima, or 9 optima and 1 suboptima.

CHAPTER 4 RESULTS

Pulavarty implemented the proposed EDA and ran a couple of experiments using the code that was developed by Wright to implement linkage detection algorithm. The experiments that were conducted evaluates and verifies some of the core concepts of this thesis.

Experiment 1 : Effect of inverse temperature on optimal points

To experimentally show that in a Boltzmann distribution, as the inverse temperature increases the probability will be concentrated near optimal points.

We used concatenated trap functions as subfunctions in this experiment. A concatenated trap function f of order k is a function which can be written as an ADF of trap subfunctions f_i 's, and each f_i is of order k . A trap function f_i of order- k is defined by

$$f_i(x) = \begin{cases} bc(x) & \text{if } bc(x) \neq 0 \\ k + 1 & \text{if } bc(x) = 0 \end{cases}$$

We used $k = 3$ and $k = 4$. The supports of each of the sub-functions are the ones that are discovered using linkage detection algorithm. The factorization of probability is then applied on these sub-functions to reduce the number of computations.

Figure 4.1 shows the average (over 1000 runs) probability of getting the maximum fit individual for varying exponents in Boltzmann distribution. We experi-

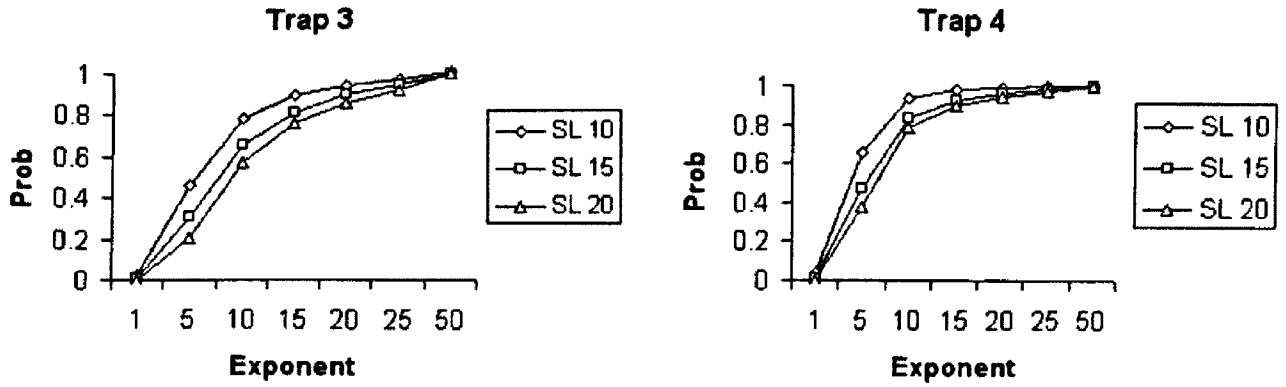


Figure 4.1 Prob. of maximum fit individual over different string lengths

mented on string lengths of 10, 15 and 20 with number of components as 4, 5 and 7 respectively.

It is evident from that graphs that the probability of optimum increases as the exponent increases for a given string length.

Experiment 2 : Complexity of linkage detection algorithm

Hypothesis : The number of function evaluations required by linkage detection algorithm in phase one of our algorithm is $\Theta(\ell^2 \log \ell)$. Note that the complexity of this algorithm has been shown to be $O(\ell^2 \log \ell)$ [5].

If g is a function and c is any non-zero constant such that

$$\lim_{\ell \rightarrow \infty} \frac{g(\ell)}{\ell^2 \log \ell} = c \quad (4.1)$$

is true, then the complexity of g is $\Theta(\ell^2 \log \ell)$. If $c = 0$, then the complexity of g is not $\Theta(\ell^2 \log \ell)$.

In other words, if we can evaluate equation 4.1, then our hypothesis can be

proved or disproved based on the value of c . Note that $g(\ell)$ is the number of function evaluations required to compute linkage detection algorithm in phase one of our algorithm with string length ℓ .

To test this hypothesis we used Needle-in-the-haystack function. A Needle-in-the-haystack function is a function in which its value is 1 for all strings except for one randomly chosen string. The value of that randomly chosen string is either $1+\epsilon$ or $1-\epsilon$ based on whether positive or negative needle is selected. Each subfunction is chosen to have an epistasis of k and there will be $\ell - k + 1$ subfunctions. The support set of i^{th} subfunction f_i is $\{i-1, i, \dots, i+k-2\}$. So there is an overlap of $k-1$ bits between f_i and f_{i+1} . For example, the support sets of f_2 and f_3 are $\{1, 2, 3, 4\}$ and $\{2, 3, 4, 5\}$ respectively, then the overlap is $\{2, 3, 4\}$.

We used order-2 hyperedges with $k = 4$ and $\epsilon = 0.1$ for this experiment. All order-2 hyperedges are detected by probing N times, which is determined by the formula in equation 2.16 with a success probability of 0.9999.

Figure 4.2 plots ratio of $g(\ell)$ and $\ell^2 \log \ell$ for different values of ℓ . It is obvious from the graph that the ratio decreases as the string length increases. If this pattern continues, as ℓ tends to ∞ , the ratio becomes 0. It implies that the complexity of $g(\ell)$ does not appear to be $\Theta(\ell^2 \log \ell)$.

Experiment 3 : Estimating on the number of optima

This experiment demonstrates how to approximate the number of optimum points for a function with multiple optima as outlined in chapter 3.

In this experiment we used Needle-in-the-haystack functions as subfunctions that we described in experiment 2. As there will be $\ell - k + 1$ subfunctions, the optimum for this function lies somewhere between $\ell - k + 1$ and $(\ell - k + 1)(1 + \epsilon)$.

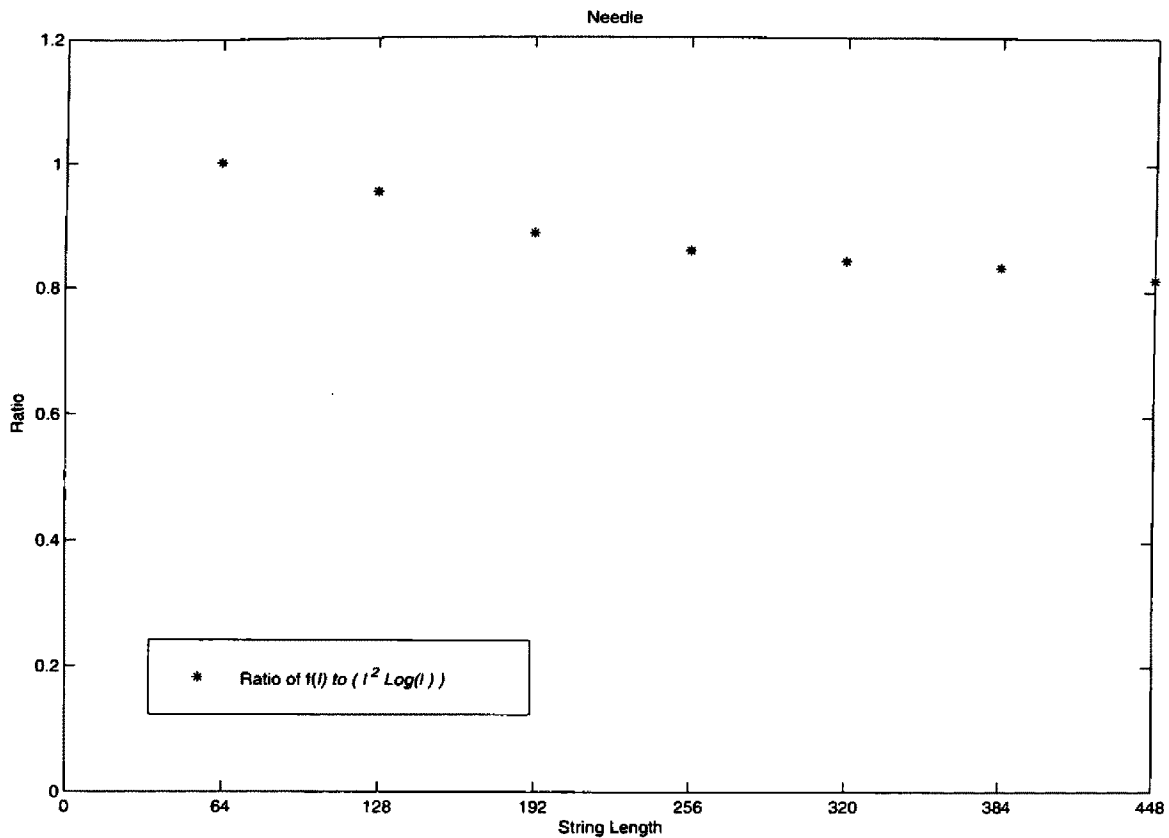


Figure 4.2 Number of function evals

As the subfunctions overlap, it might not be possible for the optimum to have all needles set. This is because of the fact that the overlap might be in such a way that only some needle can be set at a time and others cannot be set. It is most likely that the difference between optimum and suboptimum is ϵ because the difference between needle value $(1 + \epsilon)$ and the rest of the values(1) in a subfunction is ϵ .

We experimented with $\ell = 20$, $k = 4$, $\epsilon = 0.1$ and exponentiation factor of 100. There were 17 subfunctions, among which 9 were positive needles and 8

were negative needles. So, in this experiment optimum lies between 17 and 18.7, and the suboptimum is 0.1 less than the optimum. We calculated the optimum probability using direct method (enumerating all strings) and by using factorization approach. The probability of optimum as determined by factorization was 7.01528×10^{-4} , which suggests that the number of optima would be $1/(7.01528 \times 10^{-4}) = 1425$. The number of optima found when calculated by enumerating all strings was 1424. We increased the exponentiation factor to 1000, which gave the new probability of optimum as $7.02247191009 \times 10^{-4}$ using factorization approach. As $1/(7.02247191009 \times 10^{-4}) = 1424$, increasing the inverse temperature u gave a more accurate result.

CHAPTER 5 CONCLUSIONS AND FUTURE DIRECTIONS

Conclusions

In this thesis, we presented an algorithm with three phases to determine high fitness individuals for a given black box fitness function. Phase one determines the additively decomposable structure of the given black box fitness function by using linkage detection algorithm proposed by Heckendorn and Wright[5]. The factorization of the determined additively decomposable structure is computed in phase 2 of our algorithm by applying factorization theorem[11]. The factorization theorem can be applied only if the determined additively decomposable structure satisfies a property called Running Intersection Property. The final phase samples the factorized distribution. We implemented this algorithm and ran a couple of experiments as shown in the experimental section.

Future Directions

This thesis can be extended by analyzing the effectiveness of the proposed algorithm if the RIP is violated. If it is not satisfied, then an alternative method can be proposed to consider only subfunctions that satisfy RIP and leave other subfunctions.

BIBLIOGRAPHY

- [1] Shumeet Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning,. Technical Report CMU-CS-94-163, Pittsburgh, PA, 1994.
- [2] Jeremy S. de Bonet, Charles L. Isbell, Jr., and Paul Viola. MIMIC: Finding optima by estimating probability densities. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 424. The MIT Press, 1997.
- [3] D. E. Goldberg. Genetic algorithms and walsh functions: Part I: a gentle introduction. *Complex Systems*, 3:129–152, 1989.
- [4] G. R. Harik, F. G. Lobo, and D. E. Goldberg. The compact genetic algorithm. *IEEE-EC*, 3(4):287, November 1999.
- [5] Robert E. Heckendorn and Alden H. Wright. Efficient linkage discovery by limited probing. In Erick Cantú Paz et al., editor, *Genetic and Evolutionary Computation – Gecco 2003*, Lecture Notes in Computer Science LNCS 2724, pages 1003–10014. Springer Verlag, 2003.
- [6] P. Larraaga and J. A. Lozano, editors. *Estimation of distribution algorithms: a new tool for evolutionary computation*. Kluwer Academic Publishers, 2001.

- [7] H. Muhlenbein and T. Mahnig. Evolutionary algorithms: From recombination to search distributions, 2000.
- [8] Heinz Mühlenbein and Robin Höns. The estimation of distributions and the minimum relative entropy principle. 2005.
- [9] Heinz Mühlenbein and Thilo Mahnig. Convergence theory and application of the factorized distribution algorithm. *Journal of Computing and Information Technology*, 7(1):19–32, 1999.
- [10] Heinz Mühlenbein and Thilo Mahnig. The factorized distribution algorithm for additively decomposed functions. In Peter J. Angeline, Zbyszek Michalewicz, Marc Schoenauer, Xin Yao, and Ali Zalzala, editors, *Proceedings of the Congress of Evolutionary Computation*, volume 1, pages 752–759, Mayflower Hotel, Washington D.C., USA, 6-9 July 1999. IEEE Press.
- [11] Heinz Mühlenbein, Thilo Mahnig, and Aberto O. Rodriguez. Schemata, distributions and graphical models in evolutionary optimization. *Journal of Heuristics*, 5:215–247, 1999.
- [12] Martin Pelikan, David E. Goldberg, and Erick Cantú-Paz. BOA: The Bayesian optimization algorithm. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, volume I, pages 525–532, Orlando, FL, 13-17 1999. Morgan Kaufmann Publishers, San Fransisco, CA.
- [13] Martin Pelikan and Heinz Mühlenbein. The bivariate marginal distribution algorithm. In R. Roy, T. Furuhashi, and P. K. Chawdhry, editors, *Advances in*

Soft Computing - Engineering Design and Manufacturing, pages 521–535, London, 1999. Springer-Verlag.

- [14] M. D. Vose and A. H. Wright. *The Walsh Transform and the Theory of the Simple Genetic Algorithm*, chapter 2, pages 25–44. CRC Press, 1996.
- [15] M. D. Vose and A. H. Wright. The simple genetic algorithm and the Walsh transform: Part I, theory. *Evolutionary Computation*, 6(3):253–273, 1998.
- [16] M. D. Vose and A. H. Wright. The simple genetic algorithm and the Walsh transform: Part II, the inverse. *Evolutionary Computation*, 6(3):275–289, 1998.
- [17] A. H. Wright, R. Poli, C. R. Stephens, W. B. Langdon, and Sandeep Pulavarty. An estimation of distribution algorithm based on maximum entropy. In *GECCO 2004: Proceedings of the Genetic and Evolutionary Computation Conference*. Springer Verlag, 2004.
- [18] A. H. Wright and S. V. P. M. Sandeep Pulavarty. On the convergence of an estimation of distribution algorithm based on linkage discovery and factorization. In *GECCO 2005: Proceedings of the Genetic and Evolutionary Computation Conference*, 2005.