Novel ABC- and BBO-Based Evolutionary Algorithms and Their Illustrations to Wireless Communications

by

Saeed Ashrafinia

B.Sc., Sharif University of Technology, 2007

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

in the
School of Engineering Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY

Spring 2013

Approval

Name: **Saeed Ashrafinia**

Degree: **Master of Science (Electrical Engineering)**

Title of Thesis: ***Novel ABC- and BBO-Based Evolutionary Algorithms and Their Illustrations to Wireless Communications***

Examining Committee:

Chair: **John Jones**
Professor

Daniel C. Lee
Senior Supervisor
Professor

Rodney Vaughan
Supervisor
Professor

Sami Muhaidat
Internal Examiner
Assistant Professor
School of Engineering Science

Date Defended/Approved: December 19th, 2012

Partial Copyright License

Abstract

In this thesis we discuss some new Evolutionary Algorithms (EAs) as potential low-complex solvers to some optimization problems in wireless communications. Delivering high performance results while maintaining low computational complexity is extremely important in solving complex optimization problems or problems with a large search space. We propose our enhancements to Biogeography-Based Optimization (BBO) and Artificial Bee Colony (ABC) algorithms. We further present a novel high performance low-complex EA for optimization problems in both continuous and discrete domains, that combines the advantages of both BBO and ABC algorithms, which is referred to as the Hybrid ABC/BBO algorithm. This algorithm has shown higher performance in comparison to other EAs when applied to some optimization problems. We applied these algorithms to a single-objective unconstrained optimization problem (Multi Device STBC-MIMO), a single-objective constrained optimization problem (relay assignment in cognitive radio systems), and a multi-objective constrained optimization problem (Green Resource Allocation in cognitive radio systems). We provide the formulation of these problems and compared the hybrid algorithm results with exhaustive search (where applicable), and further demonstrate the superiority of the hybrid algorithm in terms of complexity and performance over ABC, BBO, other mainstream EAs and optimization solvers through simulations.

Keywords: Evolutionary algorithms; optimization; cognitive radio; mimo detection; biogeography-based optimization; artificial bee colony

To My Beloved Parents

Acknowledgments

First and foremost, all praises are due to the almighty Allah, who has blessed me with gift, strength, knowledge and patience to undertake my research.

I would like to thank my senior supervisor Professor Daniel C. Lee for his trust, guidance and support over the past years. I have learnt a lot from his expertise, understanding, and patience during my graduate studies. Through his invaluable advices, he helped me to form a personality of an independent researcher and thinker.

I would also like to thank my supervisor, Professor Rodney Vaughan, for his support and trust. His kind support was always there for anyone who was in need. He would definitely be a role model of an exceptional faculty member with impressive characteristics in my future life in academia.

Moreover, I would like to thank Professor Sami Muhaidat for his consistent support during my studies. Professor Muhaidat is a respected knowledgeable faculty member, and a true friend; and my deep appreciation is for his kind support and patience.

I am grateful for the support and help of faculty, staff and graduate students of the mobile communication lab. I give my special thanks to Dr. Muhammad Naeem, Udit Pareek, Mehdi Seyfi, Maryam Dehghani, Miladamir Tootonchian, Moein Shayegannia, Ali Zarei, Ehsan Seyedin, Seyed Amin Hejazi, Hafiz Munsub Ali, and others, who provided me with great help and support during the last few years.

My deepest appreciations go out to my family members to whom I owe so much. I would like to thank my parents for the sacrifices they have made, and for the inspiration and support they have provided throughout my life.

Table of Contents

List of Tables

List of Figures

List of Acronyms and Abbreviations

ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
AF	Amplify and Forward
AFS	Artificial Bee Swarm
AIS	Artificial Immune System
BACO	Binary Ant Colony Optimization
BBO	Biogeography-Based Optimization
BER	Bit Error Rate
BPSO	Binary Particle Swarm Optimization
CCPA	Constraint Check with Power Allocation
CPU	Central Processing Unit
CRS	Cognitive Radio System
DABC	Discrete Artificial Bee Colony
DE	Differential Evolution
EA	Evolutionary Algorithm
ECG	Electro Cardio Gram
EDA	Estimation of Distributions Algorithm
ES	Evolutionary Strategies
ESA	Exhaustive Search Algorithm
GA	Genetic Algorithm
GASP	GA with Single-Point crossover
GCCRN	Green Cooperative Cognitive Radio Network
HSI	Habitat Suitability Index
i.i.d.	independent and identically distributed
IABC	Interactive Artificial Bee Colony
ICT	Information and Communication Technology
IDS	Intrusion Detection System
LTI	Linear Time-Invariant
MAP	Maximum A-Posteriori
MD	Multi-Device

MIMO	Multi Input Multi Output
ML	Maximum Likelihood
MMSE	Minimum Mean Square Error
MO	Multi Objective
MOO	Multi Objective Optimization
OBL	Oppositional-Based Learning
PC	Personal Computer
PSK	Phase Shift Keying
PSO	Particle Swarm Optimization
PU	Primary User
QAM	Quadrature Amplitude Modulation
ROC	Receiver Operating Characteristic
SD	Sphere-Decoding
SDR	Semi-Definite Relaxation
SGA	Stud Genetic Algorithm
SIV	Suitability Index Variable
SNR	Signal to Noise Ratio
SNR	Signal to Noise Ratio
SO	Single Objective
STBC	Space-Time Block Code
TSP	Travelling Salesman Problem
UCAV	Uninhabited Combat Air Vehicle
VBA	Virtual Bee Algorithm
VEABC	Vector Evaluated Artificial Bee Colony
VEGA	Vector Evaluated Genetic Algorithm
WSM	Weighted Sum Method
ZF	Zero Forcing

List of Symbols

λ	BBO immigration rate
λ^i	The immigration rate of the i^{th} BBO habitat (i^{th} individual)
μ	BBO emigration rate
μ^i	The emigration rate of the i^{th} BBO habitat (i^{th} individual)
I	Maximum immigration rate
E	Maximum emigration rate
N	Population size (number of individuals in an EA iteration, number of species in BBO, number of food sources in ABC and hybrid)
G	Maximum number of EA iterations
H	A BBO habitat (an individual of the BBO algorithm)
H^i	The i^{th} habitat of the BBO ecosystem (The i^{th} individual of the BBO population)
$F(\ )$	The fitness function of the optimization problem
s_j	The j^{th} SIV of a BBO habitat (j^{th} component of a BBO individual)
D	Dimension of an individual (number of components in an individual vector)
$\mathbb{Y}$	A constraint set of real numbers
$\mathbb{R}$	The set of real numbers
$\mathbb{Z}$	The set of integer numbers
w	Number of pieces in the emigration curve of BBO's Piece-wise Constant Emigration curve
$\mathcal{X}$	A group of N ABC food source positions (the ABC's population set)
ϕ	A random real number $\in [-1,1]$
x	An ABC food source position (an ABC individual)
x_j	The j^{th} coordinate of an ABC food source position (j^{th} component of an ABC individual)
x^i	The i^{th} food source position of ABC (i^{th} individual of the ABC population)
x_j^i	The j^{th} coordinate of the i^{th} ABC food source position (j^{th} component of the i^{th} ABC individual)
v	A new explored ABC food source position (a new explored ABC individual)
v_j	The j^{th} coordinate of a new explored ABC food source position (j^{th} component of a new explored ABC individual)
v^i	The i^{th} new explored food source position of ABC (i^{th} new explored individual of the ABC population)

v_j^i	The j^{th} coordinate of the new explored i^{th} ABC food source position (j^{th} component of the i^{th} new explored ABC individual)
$\hat{x}$	A randomly generated ABC food source position (ABC individual)
$\hat{x}_j$	The j^{th} coordinate of a randomly generated ABC food source position (j^{th} component of a randomly generated ABC individual)
$\hat{x}^i$	The i^{th} randomly generated food source position of ABC (i^{th} randomly generated individual of the ABC population)
$\hat{x}_j^i$	The j^{th} coordinate of the i^{th} randomly generated ABC food source position (j^{th} component of the i^{th} randomly generated ABC individual)
$trial$	Number of times that the nectar of a food source position (an individual) is evaluated
$trial^i$	Number of times that the nectar of the i^{th} food source position (the fitness value of the i^{th} individual in the ABC population) is evaluated
p_i	The selection probability of the i^{th} food source position (the i^{th} individual of the ABC population)
F_i	Fitness of the i^{th} food source position (the i^{th} individual of the ABC population)
x_{min}	Lower bound of a feasible ABC food source position coordinate (ABC individual component)
x_{max}	Upper bound of a feasible ABC food source position coordinate (ABC individual component)
t	Maximum number of trials before an ABC individual is banned and its associated employed bee becomes a scout bee
$\mathcal{F}$	Computational complexity of the objective function F
n	Number of feasible discrete numbers between x_{min} and x_{max}
$H^i(s_j)$	The j^{th} SIV of the i^{th} BBO habitat (j^{th} component of the i^{th} BBO individual)
m	BBO mutation factor
N_T	Number of transmit antennas
N_R	Number of receiving antennas
T	Number of time slots in the space-time code block
K	Number of user devices (mobile devices in MD-STBC-MIMO, secondary user devices in cognitive radio systems)
$\boldsymbol{H}$	MIMO channel matrix
$\boldsymbol{H}_k$	The channel vector from the k^{th} device to the receiver
$\tilde{Y}$	STBC complex output matrix
$\boldsymbol{S}$	STBC input signal matrix

$\boldsymbol{S}_k$	The input signal vector from mobile device *k*
$\tilde{Z}$	Additive White Gaussian Noise
Q	The number of symbols conveyed in a space time code block
α_q	The q^{th} symbol real part
β_q	The q^{th} symbol imaginary part
Ω	A re-expressed version of the channel matrix $\boldsymbol{H}$
Z	A real-valued vector representing noise
y	A rearranged output matrix
χ	A rearranged input matrix
N_S	The total number of symbols (from all mobile devices) transmitted in a space-time coded block
$\hat{l}$	The shortest Euclidean distance
$\mathbb{x}$	An individual of the MD-STBC-MIMO problem
$\mathbb{x}_i$	The i^{th} component of an individual of the MD-STBC-MIMO problem
$\mathcal{M}$	The size of the symbol constellation
L	Number of relays in a cognitive radio system
M	Number of Primary Users in a cognitive radio system
$h_{s,k}$	The channel gain from the source to the k^{th} secondary user
$h_{s,l}$	The channel gain from the source to the l^{th} relay
$h_{s,m}$	The channel gain from the source to the m^{th} Primary User
$h_{l,m}$	The channel gain from the l^{th} relay to the m^{th} Primary User
$h_{l,k}$	The channel gain from the l^{th} relay to the k^{th} secondary user
p_l	The transmit power of the l^{th} relay
p_l^{max}	The maximum transmit power of the l^{th} relay
$I_{l,m}$	The interference power from the l^{th} relay to the m^{th} Primary User
s	A normalized complex-valued transmitted symbol
P_s^k	The transmission power of the source to the k^{th} secondary user
Z_l	The complex white Gaussian noise received at the l^{th} relay
W	The bandwidth of each user band
$\varepsilon_{l,k}$	A binary assignment indicating that weather the l^{th} relay is connected to the k^{th} secondary user
C_s^k	The capacity between the source to the k^{th} secondary user
P_L	The set of relay power levels

δ	The denominator of relay power levels (also $\delta + 1$ is the number of possible relay power values in the set P_L)
$I_{m,k}^{max}$	The maximum interference allowed to the m^{th} Primary User over the k^{th} user band
$\bar{\varepsilon}$	A LK-dimensional vector of assignment variables $\varepsilon_{l,k}$
f_i	The i^{th} fitness function in a multi objective optimization problem
w_i	The weight of the i^{th} fitness function in the Weighted Sum Method
$g_j(\ )$	The j^{th} inequality function in the constraint set of a multi objective optimization problem
$h_j(\ )$	The j^{th} function in the constraint set of a multi objective optimization problem
U	Number of inequality constraints of a multi objective optimization problem
V	Number of equality constraints of a multi objective optimization problem
$g_{l,m}$	The channel gain from the l^{th} relay to the m^{th} Primary User
C_k	The channel capacity of the k^{th} secondary user
$\bar{f_c}$	The objective function of the sum-rate capacity
f_c	$1 - \bar{f_c}$
f_{CO_2}	The objective function of CO_2 emissions
P	Transmission power
X	A constant representing the CO_2 emissions in grams/hour
$\boldsymbol{p}$	An L-dimensional vector comprising L relays' power levels
$\boldsymbol{p}^j$	The j^{th} individual of the EA comprising an L-dimensional vector of L relays' power levels
p_l	The power of the l^{th} relay
p_l^j	The power of the l^{th} relay of the j^{th} individual
$\boldsymbol{\varepsilon}$	An $L \times K$ binary relay assignment matrix indicating relay to secondary users connectivity
$E_l^{CO_2}$	The CO_2 emissions due to the l^{th} relay
$E_{l^{max}}^{CO_2}$	The maximum CO_2 emissions due to a relay
P_{Low}	The lower limit of a relay power
P_{High}	The upper limit of a relay power
P_{Upper}	The feasible upper limit of a relay power defined in (7.5)
$S(\ )$	An L-dimensional binary vector of temporary relay assignment values
$\mathcal{L}_k$	The set of relays that is temporarily assigned to the k^{th} secondary user

Part I:
Preliminary of Evolutionary Algorithms

1. Introduction

Recent advances in optimization facilitate progress in many areas of communications. In wireless and mobile communications, this progress provides opportunities for improving existing services and introducing new standards. Supporting data traffic over multi-hop wireless networks, efficient symbol detection, resource allocation and subcarrier assignment in MIMO communication, wireless sensor networks cognitive radio systems and mobile ad hoc networks, and network planning in wireless mesh networks etc., are challenging technical problems. This challenge is due to various factors and constraints, including computational complexity, limited bandwidth and battery power, channel variability and user mobility, protocol and standard compatibility, higher data rates, system robustness, etc. Optimization methods have been recognized as useful techniques that contribute to these challenges.

1.1. Evolutionary Algorithms

Evolutionary Algorithms (EAs) have been recognized as global optimization problem solving techniques. EAs are inspired by natural evolution and survival of the fittest [29]. Evolutionary Algorithms have been shown to be effective optimization methods for many problems. They can efficiently be applied to problems with a large search space or problems in which the objective function is complex, not differentiable, or not clearly specified (e.g., black box approaches) [29]. The success of EAs in many difficult optimization problems can be attributed to the large number of available techniques and adjustable parameters that can be tailored to particular cases [1]. Some evolutionary algorithms, such as Genetic Algorithm (GA) [2], Evolution Strategies (ES) [3], Particle Swarm Optimization (PSO) [4], and Ant Colony Optimization (ACO) [5], have been long studied. Some other popular techniques more recently introduced include Differential Evolution (DE) [6], Biogeography-Based Optimization (BBO) [7], Estimations of Distribution Algorithm (EDA) [8], Artificial Bee Colony (ABC) [9], etc.

Despite their differences, different EAs follow similar high-level approaches [10], which can be summarized as the following steps:

1. Generate initial population randomly.
2. Evaluate the fitness value of the individuals in the initial population.
3. Repeat the following steps until termination condition satisfied:
    a. Select higher quality individuals (parents) with a better fitness value.
    b. Generate new individuals from parents through variation operators (crossover, mutation, etc.)
    c. Evaluate the new population's fitness value.

The variation operators contribute to explore the search space by running specific procedures for generating new population [1] [11]. By applying these operators, the algorithm explores new solutions that potentially return better results. The variation operator is more discussed in the next subsection.

The remainder of this section is organized as follows: Section 1.1.1 describes EAs' general components. Section 1.2 explains the reasons behind choosing the two particular EAs (ABC and BBO) in this thesis. The organization of this thesis is presented in Section 1.3 and a summary of author's contributions is presented in Section 1.4.

1.1.1. General Components

Two EA components: initialization and variation operators, do not depend directly on the problem but on the representation [1]. These components are elaborated more in detail in the rest of this subsection:

1.1.1.1. Initialization

The initialization step specifies how to choose initial population (the first set of individuals). In most applications it is relatively simple: the initial population consists of individuals generated at random from some probability distribution [1] [11]. Some other approaches employ more complex or intelligent procedures to generate feasible individuals that satisfy the constraints of a constrained optimization problem. While in principle such procedures can be used for any problem, their relative cost and benefits need to be considered prior to implementation [28].

1.1.1.2. Population

Two important features of a population are its size and feasibility. The population size is simply the number of individuals within a population, which is usually kept constant from one generation to another. The importance of choosing the right population size is because it may affect the search time, complexity and the algorithm's robustness in noisy fitness settings [11] [12] [13] [14]. Feasibility, on the other hand, matters in problems defined within a certain domain or constraints. EAs can operate in discrete or continuous domain, or both. Further, if the optimization problem contains more constraints, the algorithm may require specific procedures in order to ensure the population is feasible when it is modified during each algorithm iteration.

1.1.1.3. Parent Selection

Parent selection is the process of selecting parent solutions for mutation and recombination. The goal is to select parents whose offspring have a high chance of improving their predecessor's fitness, which is usually accomplished through some variant of fitness-proportionate selection; i.e. candidates with higher fitness values have a higher chance of being selected for reproduction. Consequently, low-quality candidates are rarely selected; although in many applications they are selected in order to prevent the search to be stocked in local optima [11]. Parent selection can be randomized to reduce the complexity, or it can be smarter to improve the algorithm's performance.

1.1.1.4. Variation Operators

The role of variation operators is to explore and exploit the search space by generating new individuals from the existing population (parents) [1] [11]. A typical EA such as GA has three variation operators: crossover, selection and mutation. In the selection procedure, the parents of the next generation are chosen based on higher fitness (greedy selection). The parents reproduce one or two individuals by copying some of their components, with two possible changes: cross over and mutation.

Crossover recombines the parental genes (components) and mutation replaces some of the components of an individual with randomly generated values. Mutation is applied to some or all of the components of one candidate solution. The modifications

are usually stochastic [11]. The role of mutation varies in EA types. Nonetheless, the general role of mutation is to explore new points within the search space to ensure that the algorithm is not stocked in local optima. More discussion is presented in Sections 2.2.2 and 3.3.1.3.

1.1.1.5. Elitism

If an EA contains elitism, it prevents a few best individuals to be altered via the variation operators. Thus, the evolutionary algorithm ensures that the best individuals are safely transferred to the next iteration. If the algorithm does not incorporate elitism, there may be a chance of losing high quality individuals and it depends on the algorithm's variation operator(s) procedure. Some algorithms like ABC, BBO and EDA may retain their best individuals, while some others like GA have the tendency to lose their best individual through the crossover operator.

1.1.1.6. Termination Condition

Most EAs have no guarantee about finding the optimum solution in some reasonable bounded time. As such, the algorithm requires specifying one or more termination conditions. Some common ones include [11]:

- **Maximum iterations**: the algorithm iterates for a predefined number of iterations.
- **Within optimum**: the maximum theoretical value of the fitness function is known, and the search is terminated when it comes to within $\pm\varepsilon$ of that optimum.
- **Time limit**: user-defined maximum running time has elapsed. Other related measures, such as CPU time, the number of generations or the number of fitness evaluations can be used as well.
- **Convergence**: the search has converged, i.e. fitness improvement in the last few generations stayed below some small threshold.

Note that some of these conditions should be employed carefully in algorithms that have the ability to get out of local optima.

In some cases, a combination of the above termination conditions is used. For example, an algorithm might be terminated either when it comes to within $\pm\varepsilon$ of the optimum, or a time limit has passed, whichever comes first [11].

1.2. Algorithms Discussed in This Thesis

The author's first experience with EAs was applying Dan Simon's Biogeography-Based Optimization (BBO) algorithm [7] Simon's paper showed good performance compared with seven well-known EAs applied to 14 standard benchmark functions and a real-world sensor problem [7]. After observing superior performance of BBO implemented for computationally efficient joint symbol detection to a Multi-Device STBC-MIMO system (Chapter 5) [17], with its BER results very close to Sphere Decoding with much less complexity, the algorithm was implemented for other wireless communication problems, while improving the algorithm and proposing some techniques to reduce its complexity (Section 2.4).

Furthermore, we studied another novel EA, the Artificial Bee Colony (ABC) Algorithm presented by Karaboga [9], which has shown good performance results especially for continuous optimization problems. Karaboga's comparison of ABC with four well-known EAs for 50 benchmark functions demonstrates good performance of this EA. However, the algorithm required adjustments to be applied to discrete optimization problems. We modify the algorithm (Section 3.4), improve its performance and reduce its complexity (Section 3.5). The results of applying the enhanced ABC to some wireless communication applications (Chapter 5 and 6) were very promising and even better than BBO or any other EA we included in our comparison.

These two algorithms (ABC and BBO) are the two with the highest efficiencies in their class being applied to some optimization problems [7] [9] [16] [17] [18], and they have usually returned the best solutions to different real world optimization problems, including problems in wireless communications.

Subsequently, after extensively studying BBO and ABC along with other EAs, and implementing them for wireless communication problems, we recognized the strong and weak aspects of each EA (Section 4.2). The interesting point about ABC and BBO algorithms was that the disadvantage of one is the advantage of the other (Section 4.3). Taking the advantage of each of these algorithms' strong features, we develop a novel EA by hybridizing these algorithms, which is presented in Chapter four.

In 1995, Wolpert and Macready presented an important result in their "*No Free Lunch*" theory for optimization [15]. Their work illustrated that all algorithms that search for an extremum of a cost function perform exactly the same, when averaged over all possible cost functions. In other words, "any two algorithms are equivalent when their performance is averaged across all possible problems" [29]. Hence, from a problem solving perspective it is difficult to formulate a universal optimization algorithm that could solve all the problems. Droste et al have proved the theorem for more practical case in [30]. Therefore, the algorithms discussed in this thesis may not unanimously outperform other EAs. Nonetheless, for the problems discussed in Part II of this thesis, in addition to some other problems studied but not included in this thesis cited in Section 1.4, the implementation of hybrid algorithm returns better result than other EAs including BBO and ABC.

1.3. Organization of Thesis

This thesis focuses on implementing EAs for various wireless communication problems. The thesis consists of two parts. **Part I** contains the preliminary of EAs. We start **Chapter 2** and **Chapter 3** with some sections for introducing BBO and ABC, and we present some improvements to each of these algorithms in the subsequent sections. We also provide a review of some of the interesting BBO and ABC research reported in literature. The novel hybrid ABC/BBO algorithm is proposed in **Chapter 4**, with a brief introduction of hybrid EAs, and a discussion on strong and weak aspects of ABC and BBO, and finally presenting the algorithm's procedure.

Part II of the thesis consists of implementations of EAs discussed in Part I to three wireless communication problems and their performances comparison with other EAs and deterministic solvers. **Chapter 5** addresses a computationally efficient symbol detection optimization problem in multi-user STBC-MIMO systems. **Chapter 6** contains a joint relay assignment and power allocation in cognitive radio systems using EAs. Finally, **Chapter 7** discussed a green resource allocation in cognitive radio systems. Each of these three chapters contains a background of the problem, followed by defining the system model and formulating the problem. The rest of the chapters contain discussions on the methods of implementing EAs to that specific problem, and a

discussion on complexity where possible. Finally the simulation results are presented and a summary is given in the conclusion section.

1.4. Summary of Contributions

This thesis discusses two recent EAs in the next two chapters in Part I. Although these algorithms are already discussed in the literature and implemented to various problems, we confronted some novel, important and interesting issues while studying and implementing these algorithms. Our contributions to the BBO algorithm are presented in:

- Section 2.3: BBO Migration Models
- Section 3.3: Comparison between Scout bees phase and mutation
- Section 3.4: Discrete Artificial Bee Colony Algorithm
- Section 3.5: Improvements to the DABC algorithm

The BBO algorithm is originally presented for the optimization problems in a discrete or integer domain, and the ABC algorithm is primarily introduced for continuous optimization problems. The proposed hybrid algorithm in Chapter 4 can be applied to both types of problems. Moreover, this algorithm is implemented for optimization problem types, and in all of these problems, this algorithm beats its predecessors BBO and ABC in terms of BER performance (Chapter 5) and sum rate (Chapters 6 and 7), as well as other mainstream EAs such as GA, EDA, ACO, in addition to other deterministic algorithms. A collection of optimization problems that the author has implemented these EAs for them are presented in Table 1.1. These algorithms are applied to some other optimization problems not included in this thesis that are published in various journals and conference proceedings. The list of these papers is presented at the end of the Reference Section of this chapter.

In Part II of this thesis, the author has made the following contributions:

- Section 5.4: Evolutionary Algorithms for solving MD-STBC-MIMO problem
- Section 5.5: Computational Complexity
- Section 5.6: Simulation Results

- Section 6.3: Evolutionary Algorithms-Based Relay Assignment with Greedy Power Allocation
- Section 6.4: Simulation Results
- Section 7.4: Hybrid Solver for GCCRN MOO Problem

The research on EAs has contributed to the community by publishing a number of papers in reputed conferences and journals. These publications, along with few others in submission and reports, can be found at the Reference Section of this chapter, which are: [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]. Some of these publications have not been published yet, and are mentioned in the Reference Section. The author of the thesis is the first author for most of these papers. The author has also presented some of the conference papers in IEEE conference himself [17] [18] [22].

Table 1.1: List of Contributed Literature

Optimization Problem Type	Problem	Reference
Single objective, unconstraint	Computationally Efficient Symbol Detection in Multi-User STBC-MIMO Systems	[17] [19] [20] Chapter 5
Single objective, constraint	Relay Selection in Relay Assisted Cognitive Radio Systems	[21]
Single objective, constraint	Joint Relay Assignment and Power Allocation in Cognitive Radio Systems	[18] [22] Chapter 6
Single objective, constraint	Critical Node Detection Problem	[24]
Multi objective, constraint (converted to single objective using Weighted Sum Method)	Wireless Mesh Network Planning	[23]
Multi objective, constraint (converted to single objective using Weighted Sum Method)	Green Resource Allocation in Cognitive Radio Systems	[25] [27] Chapter 7
Multi objective, constraint (converted to single objective using Weighted Sum Method)	Base Station and Relay Station Broadband Network Planning	[26]

References

[1] A. Fialho. *Adaptive Operator Selection for Optimization*. PhD thesis, Universite Paris-Sud XI, Orsay, France, Dec. 2010.

[2] D. Goldberg, "*Genetic algorithms in search, optimization, and machine learning, Artificial Intelligence*, Addison-Wesley Pub. Co., 1989.

[3] H.-G. Beyer and H.-P. Schwefel. "Evolution Strategies: A Comprehensive Introduction," *Journal Natural Computing*, vol.1, no.1, pp. 3–52, 2002.

[4] Kennedy, J.; Eberhart, R. (1995). "Particle Swarm Optimization". *Proceedings of IEEE International Conference on Neural Networks*, pp. 1942–1948.\

[5] M. Dorigo & L. M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem," *IEEE Transactions on Evolutionary Computation,* vol.1, no.1, pp. 53–66. 1997.

[6] Storn, R.; Price, K. (1997). "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces". *Journal of Global Optimization* vol.11, pp. 341–359.

[7] D. Simon, "Biogeography-based optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 12, pp. 702 – 713, December 2008.

[8] P. Larrañaga and J. A Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Kluwer Academic Publishers, 2001.

[9] D. Karaboga B. Basturk, "An artificial bee colony (ABC) algorithm for numeric function optimization," in *IEEE Swarm Intelligence Symposium 2006*, Indianapolis, IN, May 12-14, 2006.

[10] M. Pacula, "Evolutionary Algorithms for Compiler-Enabled Program Autotuning," *PhD thesis*, Massachusetts Institute of Technology, June 2011.

[11] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*, Springer-Verlag, 2003.

[12] J. Ansel, M. Pacula, S. Amarasinghe, and U.-M. O'Reilly, "An ancient evolutionary algorithm for solving bottom up problems," In *Annual Conference on Genetic and Evolutionary Computation*, Dublin, Ireland, July 2011.

[13] D. V. Arnold and H.-G. Beyer. "On the benefits of populations for noisy optimization," *Evolutionary Computation*, vol. 11, no. 2, pp. 111-127, 2003.

[14] J. Branke. “Creating robust solutions by means of evolutionary algorithms,” In *Parallel Problem Solving from Nature, PPSN V*, vol. 1498 of *Lecture Notes in Computer Science*, pp. 119-. Springer Berlin / Heidelberg, 1998.

[15] D.H. Wolpert, W.G. Macready, "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation* vol. 1, no. 1, pp. 67-83, 1997.

[16] B. Akay D. Karaboga, "A comparative study of Artificial Bee Colony algorithm," *Applied Mathematics and Computation*, no. 214, pp. 108-132, 2009.

[17] S. Ashrafinia; M. Naeem; D. Lee; “A low complexity evolutionary algorithm for multi-user MIMO detection”, Computational Intelligence in Multicriteria Decision-Making (MDCM), 2011 IEEE Symposium on, 11-15 April 2011, pp. 8 – 13, Paris, France.

[18] S. Ashrafinia; U. Pareek; M. Naeem; D. Lee; “Biogeography-based optimization for joint relay assignment and power allocation in cognitive radio systems”, Swarm Intelligence (SIS), 2011 IEEE Symp. on, 11-15 April 2011, pp 1 – 8, Paris, France.

[19] S. Ashrafinia; M.Naeem; D. Lee; “Discrete Artificial Bee Colony for Computationally Efficient Symbol Detection in Multi-Device STBC MIMO Systems”, accepted in the Journal of Advances in Artificial Intelligence, Nov. 2012.

[20] S. Ashrafinia; Naeem, M.; Lee, D.; “Heuristic Joint Symbol Detection in MIMO Systems”, Preliminary accepted in *Journal of Algorithms in Cognition, Informatics and Logic, special issue on “Bio-inspired Computing: Theory and Applications.* The paper got accepted for publication in June 2009, but the journal’s special issue was canceled by the publisher.

[21] S. Ashrafinia; U. Pareek; M. Naeem; D. Lee ; “Source and Relay Power Selection Using Biogeography-Based Optimization for Cognitive Radio Systems”, *IEEE VTC 2011 – fall*, 5-8 Sep. 2011, San Francisco, CA.

[22] S. Ashrafinia; U. Pareek; M. Naeem; D. Lee; “Binary Artificial Bee Colony for Cooperative Relay Communication in Cognitive Radio Systems”,*IEEE International Conf. on Comm., ICC 2012*, 10-15 Jun. 2012, Ottawa, Canada

[23] Ali H. M., Ashrafinia, S.; Liu J.C.; Lee, D.; “Wireless Mesh Network Planning Using Quantum Inspired Evolutionary Algorithm”, IEEE VTC 2011 – fall, 5-8 Sep. 2011, San Francisco, CA.

[24] S. Ashrafinia, *Critical Node Detection Problem,* Directed Studies course report, Simon Fraser University, Aug. 2011.

[25] S. Ashrafinia; Naeem, M.; Lee, D.; “Hybrid Evolutionary Algorithm for Resource Allocation in Cognitive Radio and Green Communication Systems”, Submitting to IEEE Transactions on Systems, Man, and Cybernetics.

[26] H. M. Ali; S. Ashrafinia; J. C. Liu; and Daniel C. Lee; "A Study of Evolutionary Algorithms for Base Station and Relay Station Broadband Network Planning", Submitting to the journal of Ad Hoc Networks.

[27] M. Naeem; S. Ashrafinia; D. Lee; “Green Resource Allocation in Cognitive Radio Systems”, accepted on Sep. 2012 for proceedings of IEEE ICSPCS 2012, Dec 2012, Sydney, Australia.

[28] A. Bhattacharya; Chattopadhyay,"Hybrid Differential Evolution With Biogeography-Based Optimization for Solution of Economic Load Dispatch," *Power Systems, IEEE Transactions on* , vol.25, no.4, pp.1955-1964, Nov. 2010.

[29] D. Wolpert; H.Macready, "Coevolutionary free lunches," *Evolutionary Computation, IEEE Transactions on* , vol.9, no.6, pp. 721- 735, Dec. 2005.

[30] S. Droste, T. Jansen, I. Wegener, "Optimization with randomized search heuristics: the (A)NFL theorem, realistic scenarios, and difficult functions," *Theoretical Computer Science,* vol. 287, no. 1, pp. 131–144, 2002.

2. Biogeography-Based Optimization Algorithm

This chapter discusses the Biogeography-Based Optimization (BBO) algorithm. BBO is a new global optimization population-based EA based on the biogeography theory, which is the study of the geographical distribution of biological species. An introduction to biogeography is given in Section 2.1 Section 2.2 contains the BBO terminologies, definitions and algorithm. Our contribution to the BBO algorithm, as low-complex migration models, is discussed in Section 2.3.

2.1. Introduction to Biogeography

The science of biogeography is the study of the geographical distribution of biological species. The early study of biogeography was performed by Wallace [1] and Charles Darwin [2] during the 19^{th} century. These studies until the mid-twentieth century were mostly descriptive and historical. Later in 1960s, Robert MacArthur and Edward Wilson studied the mathematical models of biogeography, and wrote The Theory of Island Biogeography [3]. Their work mainly was focused on the distribution of species among neighboring islands, and their interest was in mathematical models for the extinction and migration of species. During the recent few years, the implementations of biogeography have been a breakthrough to the engineering applications of the Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), as well as neural networks, fuzzy logic, and other areas of computer intelligence. More elaboration on of biogeography and its mathematical models is provided in [5].

2.2. Biogeography-Based Optimization (BBO)

Biogeography-based optimization, introduced by Simon [5], is a **population-based, stochastic global optimizer Evolutionary Algorithm**, based on the

mathematics of biogeography theory describes in the two sections above. Consider an optimization problem:

$$\max_{H} \quad F(H) \qquad \text{subject to } H \in \mathbb{Y} \tag{2.1}$$

where $H \equiv (s_1, s_2, \dots, s_D)^T$ is a vector and $\mathbb{Y}$ is a constraint set. In the original BBO, each candidate solution is represented by a vector variable of the optimization problem. In the context of evolutionary algorithms, a candidate solution is also referred to as an "individual," and a group of individuals is referred to as a "population" of individuals. In BBO, each individual (candidate solution to an optimization problem) is analogically considered as a habitat (island) in Biogeography. The fitness value $F(H)$ of each individual H corresponds to the Habitat Suitability Index (HSI) of an island in Biogeography. In Biogeography, features that affect HSI include vegetation, rainfall, topographic diversity, temperature, etc., and these features are characterized by variables that are called Suitability Index Variable (SIV). As mentioned earlier, a candidate solution H in optimization problem 2.1 analogically corresponds to a habitat (an island) in Biogeography. Then, the components of H (i.e.$s_j, j \in \{1, \dots, D\}$) correspond to an island's SIVs, and $F(H)$ correspond to the HSI of habitat H. We will often use these terminologies to refer to an individual H, its components $s_1, s_2, \dots, s_D$ and its fitness value.

The advantage of biogeography, as the nature's way of distributing species, is analogous to general problem solutions. Suppose there's a problem presented with some candidate solutions, which can be in any area of life, such as engineering, economics, medicine, business, operations research, etc., as long as there is a quantifiable measure of the suitability of a given solution (be able to calculate the fitness value). A good solution is analogous to a habitat with a high HSI, while a poor one represents a habitat with a low HSI. In an EA, high HSI habitats tend to share their features (copy their SIVs) with low HSI solutions' features. By sharing features, habitats with high HSI are not going to lose their current features – they will remain intact and low HSI solutions import features from high HSI habitats. Poor solutions accept many features from good solutions, which may improve their HSI, thus their quality. This new approach to problem solving has been presented by Dan Simon is called Biogeography-Based Optimization (BBO) [5].

2.2.1. BBO Migration

The main consequence of BBO migration is using the immigration and emigration rates of each solution to share information between habitats probabilistically. One approach to the migration can be defined as following: with a random probability we modify each solutions based on other solutions. Thus if a given solution is selected to be modified, then we use its immigration rate λ to decide probabilistically whether to modify each SIV in that solution. If a given SIV in a given solution is selected to be modified (i.e. it is ready to accept the shared information from other solutions), then we use the emigration rates of the other solutions to probabilistically decide which of the solutions should emigrate (share information of) a randomly selected SIV to solution. One has to remember that the associated immigration or emigration curves of a solution vector (habitat) represents the corresponding rates of each SIV inside that solution [5].

BBO Migration and Other EAs

The BBO migration procedure is analogous to the Global Recombination approach in the breeder GA [6] and Evolutionary Strategies (ES) [7] algorithms, where many parents contribute to a single offspring. However, it is not quite the same; in the ES algorithm Global Recombination produces new solutions, while BBO's migration substitutes (copies) existing solutions. Global recombination in GA is a reproductive process, whereas migration in BBO is an adaptive process – it modifies existing habitats.

2.2.2. BBO Mutation

Sometimes unusual natural incidents may happen to a habitat, such as large flotsam arriving from a neighboring habitat, disease, hat drastically change the HSI of a natural habitat, and may cause a species count to differ from its equilibrium value. As a result, a habitat's HSI can change suddenly due to such random events. This natural behavior in biogeography is interpreted as *mutation* in the BBO algorithm [5].

Mutation in BBO, just like other EAs like GA, prevents the algorithm to be stocked in local optima, and it can occur for any SIV in any habitat in each generation. In the BBO algorithm we employ a simple mutation procedure, in which all SIVs are randomly compared with a relatively small constant called the *mutation factor,* denoted

by m, and they will be mutated into some new randomly generated SIV that is eligible according to the problem domain.

2.2.3. BBO Elitism

Similar to other population-based optimization algorithms, elitism may be incorporated into BBO; so the algorithm prevents them from being modified during the migration process, and the best solutions retain in the population.

2.2.4. BBO Definitions and Algorithm

Simon defines BBO terminologies in [5]. In this thesis, we present slightly modified definitions according to the notations used as the first step towards formalizing the BBO algorithm. In these definitions, $\mathbb{R}$ is referred to the set of real numbers, and $\mathbb{Z}$ is referred to the set of integers.

- ***Definition 2.1*** A habitat $H \in SIV^D$ is a vector of D integers, and represents a feasible solution to some problem.
- ***Definition 2.2*** A Suitability Index Variable SIV $\in \mathbb{Z}$ is an integer that is allowed in a habitat (a component of an individual).
- ***Definition 2.3*** A Habitat Suitability Index HSI $: \mathbb{Y} \to \mathbb{R}$ is a measure of the goodness of the solution that is represented by the habitat (HSI is referred to as "fitness" in most population-based optimization algorithms), where $\mathbb{Y} \subseteq \mathbb{Z}^D$ is the domain of all feasible solutions to the problem.
- ***Definition 2.4*** An ecosystem $H^{(N)}$ is a group of N habitats, where N is a constant representing the size of an ecosystem (population size).
- ***Definition 2.5*** Immigration rate $\lambda(HSI): \mathbb{R} \to \mathbb{R}$ is a monotonically non-increasing function of HSI. λ^i is proportional to the likelihood that SIVs from neighboring habitats migrate into habitat H^i.
- ***Definition 2.6*** Emigration rate $\mu(HSI): \mathbb{R} \to \mathbb{R}$ is a monotonically non-decreasing function of HSI. μ^i is proportional to the likelihood that SIVs from habitat H^i migrate to neighboring habitats.

Generally, population-based Evolutionary Algorithms have an iterative nature – i.e., they are run for a number of generations (trials, iterations). The number of these generations usually is determined by a variety of models described in Section 1.1.1.6. The BBO algorithm uses a fixed number of generations.

Given the above definitions of BBO terminology, we provide the Biogeography-Based Optimization algorithm pseudo-code in table 2.1.

Table 2.1. Pseudo code of the BBO algorithm

For each iteration G

Migration Procedure

for each island H^i, $i \in 1, \ldots, N$;
for each SIV s_j, $j \in 1, \ldots, D$
with probability λ^i, $H^i(s_j)$ accepts immigration;
If $H^i(s_j)$ is decided to accept immigration, **then**,
probabilistically select an island H^k that emigrates to $H^i, i \neq k$, based on μ^k;
$H^i(s_j) = H^k(s_j)$ *{*$\mathrm{H^k(s_j)}$ *migrates into* $\mathrm{H^i(s_j)}$*}*;
end if
next SIV
next island

Mutation Procedure

for each island H^i
for each SIV s_j
if $rand < m$, **then** *{Use mutation factor* m *to decide whether to mutate* $\mathrm{H^i(s_j)}$*;}*
replace $H^i(s_j)$with a randomly generated SIV;
end if
next SIV
next island

Calculates HSI

for each island H^i
calculate HIS
next island

Save beset results

sort population
save the best island

next iteration

2.3. BBO Migration Models

There are various types of migration curves because of different mathematical models of biogeography [3]. Simon has initially presented BBO with linear emigration and immigration rate curves [5]. Thereupon, most of the literature utilized the linear emigration and immigration curves [8-13].

Recently, several more complex curves are presented in [14], including the comparison of various non-linear migration models, such as quadratic migration curve, sinusoidal migration curve and generalized sinusoidal migration curve. Ma and Simon concluded in [14] that a sinusoidal model for emigration and immigration curves has a better performance than other models, including the linear model. In fact, they only compared the result for best performance and closest result to the optimal value. However, they have included neither computational complexity nor time comparisons in their research, so one could have chosen the preferred migration model based on the performance – computational complexity trade-off.

Complexity of the employed algorithms for optimization problems is a major concern in communications applications. In real-time applications, such as real-time symbol detection in wireless communication, it is crucial to employ an algorithm that not only runs with a reasonable performance, but also returns the result quickly. The computational power is an issue in some other applications, such as wireless sensors, or mobile hand-held devices, that minimizing the consuming power is vital in order to prolong the battery life. In addition, more complex algorithms demand more complicated circuitry, which further increase the production cost. Therefore, it would be essential to employ an algorithm that returns the best performance along with low complexity.

We have run simulations using the Simon's code in Matlab© language [15] that he used to generate the results of his first paper [5], in which he employs linear emigration and immigration curves. During our first implementation of BBO to joint symbol detection in a multi-device STBC MIMO system [16], we observed although BBO returns better results that some other EAs like GA and EDA, Simon's implementation of linear curves takes more time than other EAs, including our contributed low-complex migration curves.

This section continues with Simon's linear migration model [5]. The rest of the section contains two other migration models with lower complexity proposed by the author, which are implemented for the optimization problems discussed in Part II of the thesis.

2.3.1. Linear Immigration – Linear Emigration Model

The immigration and emigration curves are functions of the number of species in a single habitat. A typical leaner model [5] for λ and μ curves is illustrated in Figure 2-1.

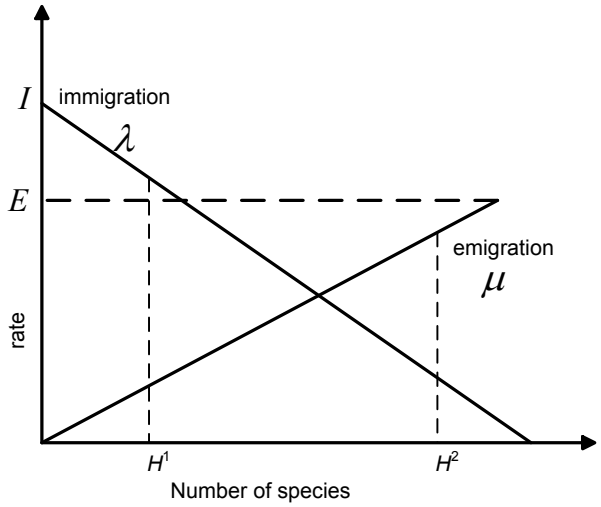


Figure 2.1. Linear immigration rate and constant emigration curves

In the immigration curve λ, the maximum possible immigration rate to the habitat is I, which happens when there is no species in the habitat. As the number of species inside the habitat increases, the immigration rate decreases; because the habitat has become more crowded and there would be less resources for new species to successfully survive. When the habitat has its largest possible number of species N that it can support, the immigration rate becomes zero.

In the emigration curve μ, while there are no species inside the habitat, the emigration rate has to be zero. As the number of species increases, the habitat becomes more crowded and some species tend to leave the habitat and explore other possible habitats, thus the emigration curve increases. The maximum emigration rate E happens when the habitat has the largest possible number of species it can support.

The expressions for emigration rate and immigration rates can be expressed as:

$$\lambda^k = I\left(1 - \frac{k}{n}\right), \qquad \mu^k = \frac{k}{n}E \tag{1.9}$$

2.3.2. Linear Immigration – Constant Emigration Model

A low-complexity migration curve may consist of a linear immigration rate and a constant emigration rate curves as depicted in Figure 2.2.

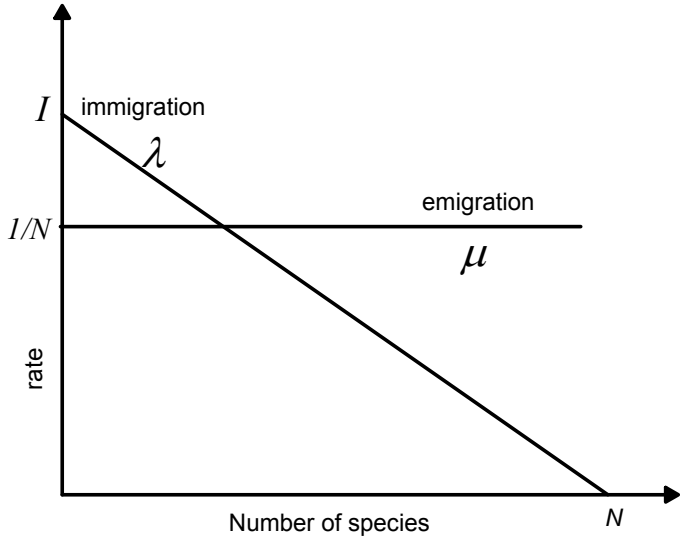


Figure 2.2. Linear Immigration – Constant Emigration curve

In this model, the emigration rate μ is constant. Emigration rate is the likelihood of species to emigrate from a habitat; that is how likely a habitat wants to share its SIVs with other individuals. A constant emigration curve implies that all habitats have the same probability to share their SIVs. This statement induces μ to be constantly uniformly distributed. For a maximum number of N habitats, emigration rate is constantly uniformly distributed if $\mu = 1/N$. Therefore, the two curves can be expressed as:

$$\lambda^k = I\left(1 - \frac{k}{N}\right), \qquad \mu^k = \mu = \frac{1}{N} \tag{2.7}$$

where μ denotes the constant emigration rate, and k represents the species count. As the number of species increases, the immigration rate linearly decreases. The maximum immigration rate I occurs when there are zero species in the habitat. The immigration rate becomes zero when the habitat accommodates the largest possible number of species.

2.3.3. Linear Immigration – Piece-wise Constant Emigration Model

In this model, the emigration rate is again constant, while we use a piece-wise constant immigration rate as depicted in Figure 2.3.

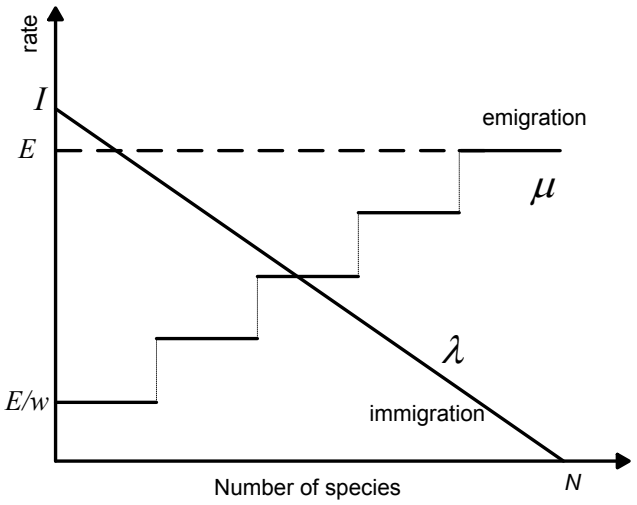


Figure 2.3. Linear Immigration – Piece-wise Constant Emigration curve

The immigration rate in this model is also linear with the maximum of I. But the emigration rate is piece-wise constant, composed of w constant segments defined over w intervals of equal size, with a maximum of E. This model is closer to the linear immigration rate – linear emigration rate. The two curves can be expressed as follows:

$$\lambda^k = I\left(1 - \frac{k}{N}\right),$$

$$\mu^k = \begin{cases} \frac{E}{w} & 0 < k \leq \left\lceil \frac{N}{w} \right\rceil \\ \frac{2E}{w} & \left\lceil \frac{N}{w} \right\rceil < k \leq \left\lceil \frac{2N}{w} \right\rceil \\ \vdots & \vdots \\ E & \left\lceil \frac{(w-1)N}{w} \right\rceil < k \leq N \end{cases} \tag{2.8}$$

Once again, as the number of species increases, the immigration rate decreases linearly. The behavior of the emigration rate curve (size of the intervals) is varies with w and the maximum number of species N. The more number of the species in a habitat results in the higher emigration rate (higher k $\rightarrow$ higher μ), while this emigration rate is equal for habitats with closer number of species. The above emigration expression also shows that more segments (larger w) will result in a closer curve to the linear model. A comparison between these first two models in presented in section 6.4 in Figure 6.11.

References

[1] A. Wallace, *The Geographical Distribution of Animals*. 1876.

[2] C. Darwin, *Origin of species*. Everyman's library, Dent, 1947.

[3] R. MacArthur and E. Wilson, *The theory of island biogeography*, Princeton landmarks in biology, Princeton University Press, 1967.

[4] T. Wesche, C. Goertler, W. Hubert, U. of Wyoming, and W. W. R. Center, *Modified habitat suitability index model for brown trout in southeastern Wyoming. Water resources center publication*, Wyoming Water Research Center, 1987.

[5] D. Simon, "Biogeography-based optimization," *Evolutionary Computation, IEEE Transactions on,* vol. 12, pp. 702–713, December 2008.

[6] H.Mhlenbein and D. Schlierkamp-voosen, "Predictive models for the breeder genetic algorithm i. continuous parameter optimization," *Evolutionary Computation, IEEE Transactions on,* vol. 1, pp. 25–49, 1993.

[7] T. B¨ack, "Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms," *Oxford University Press*, 1996.

[8] A. Bhattacharya and P. Chattopadhyay, "Biogeography-based optimization for solution of optimal power flow problem," in Electrical Engineering/ Electronics Computer Telecommunications and Information Technology (ECTI-CON), 2010 International Conference on, pp. 435 –439, May 2010.

[9] U. Singh, H. Kumar, and T. Kamal, "Design of yagi-uda antenna using biogeography based optimization," *Antennas and Propagation, IEEE Transactions on,* vol. 58, pp. 3375–3379, Oct. 2010.

[10] A. Bhattacharya and P. Chattopadhyay, "Oppositional biogeography-based optimization for multi-objective economic emission load dispatch," *in India Conference (INDICON)*, 2010 Annual IEEE, pp. 1–6, Dec. 2010.

[11] A. Bhattacharya and P. Chattopadhyay, "Biogeography-based optimization for different economic load dispatch problems," *Power Systems, IEEE Transactions on*, vol. 25, pp. 1064–1077, May 2010.

[12] A. Bhattacharya and P. K. Chattopadhyay, "Biogeography-based optimization and its application to nonconvex economic emission load dispatch problems," in *Advances in Power System Control, Operation and Management (APSCOM 2009),* 8th International Conference on, pp. 1 –6, Nov. 2009.

[13] M. Lohokare, S. Pattnaik, S. Devi, K. Bakwad, and J. Joshi, "Parameter calculation of rectangular microstrip antenna using biogeography-based optimization," in *Applied Electromagnetics Conference (AEMC),* pp. 1 –4, Dec. 2009.

[14] D. Simon, R. Rarick, M. Ergezer, and D. Du, “Analytical and numerical comparisons of biogeography-based optimization and genetic algorithms,” *Information Sciences,* vol. 181, no. 7, pp. 1224-1248, April 2011.

[15] D. Simon, A probabilistic analysis of a simplified biogeography-based optimization algorithm, *Evolutionary Computation,* vol. 19, no. 2, pp. 167-188, Summer 2011.

[16] S. Ashrafinia; M. Naeem; D. Lee ; “Computationally Efficient Symbol Detection Using Biogeography-Based Optimization in Multi-User STBC-MIMO Systems”, accepted in the *Journal of Advances in Artificial Intelligence*, Nov. 2012.

[17] D. Simon, “A probabilistic analysis of a simplified biogeography-based optimization algorithm,” *Evolutionary Computation, IEEE Transactions on,* vol. 19, pp. 167–188, June 2011.

[18] J. Kennedy, J. Kennedy, R. Eberhart, and Y. Shi, “Swarm intelligence*,” The Morgan Kaufmann series in evolutionary computation,* Morgan Kaufmann Publishers, 2001.

[19] R. Storn and K. Price, “Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces,” *J. of Global Optimization*, vol. 11, pp. 341–359, Dec. 1997.

[20] M. Dorigo and T. Stutzle, *Ant colony optimization*, Bradford Books, MIT Press, 2004.

[21] D. Goldberg, *Genetic algorithms in search, optimization, and machine learning, Artificial Intelligence*, Addison-Wesley Pub. Co., 1989.

[22] H.-G. Beyer and H.-P. Schwefel, “Evolution strategies – a comprehensive introduction,” *Natural Computing: an international journal,* vol. 1, pp. 3–52, May 2002.

[23] T. Back, U. Hammel, and H.-P. Schwefel, “Evolutionary computation: comments on the history and current state,” *Evolutionary Computation, IEEE Transactions on*, vol. 1, pp. 3 –17, Apr. 1997.

[24] H. Pattee and U. S. O. of Naval Research, *Natural automata and useful simulation: proceedings.* Edited by H.H. Pattee [and others]. Spartan Books, 1966.

[25] A. E. Eiben, “Multi-parent recombination,” in *Handbook of Evolutionary Computation*, pp. 3–3, IOP Publishing Ltd. and Oxford University Press, 1997.

[26] D. Fogel, T. Back, and Z. Michalewicz, “Evolutionary Computation: Basic algorithms and operators,” *Evolutionary Computation,* Institute of Physics Publishing, 2000.

[27] A. E. Eiben and C. Schippers, “Multi-parent’s niche: N-ary crossovers on nk-landscapes,” in Proceedings of the 4th Conference on Parallel Problem Solving from Nature, number 1141 in Lecture Notes in Computer Science, pp. 319–328, Springer, 1996.

[28] D. Ackley, “A connectionist machine for genetic hillclimbing,” *Kluwer international series in engineering and computer science,* Kluwer Academic Publishers, 1987.

[29] D. Simon, R. Rarick, M. Ergezer, and D. Du, “Analytical and numerical comparisons of biogeography-based optimization and genetic algorithms,” *Inf. Sci.,* vol. 181, pp. 1224–1248, April 2011.

3. Artificial Bee Colony Algorithm

This chapter introduces the Artificial Bee Colony (ABC) algorithm, and continues with presenting some improvements to the existing ABC algorithm and introducing a modified ABC customized for discrete optimization. ABC is a recently presented EA for real (continuous) parameter optimization in unconstrained optimization problems. An introduction to the swarm intelligence is presented in the next section. Section 3.2 explains the behavior of real bees, from which ABC has been inspired. The ABC definitions and algorithm is presented in Section 3.3. Our improvements to the ABC algorithm are Sections 3.4 and 3.5. In Section 3.4, we present a modified ABC algorithm specifically developed for discrete optimization problems. In Section 3.5 we discuss some further improvements to the original ABC as well as the novel discrete ABC algorithm to enhance the algorithm performance and complexity.

3.1. Introduction to swarm intelligence

In recent years, swarm intelligence has become a research interest to many research scientists of related fields. Bonabeau et al. define the swarm intelligence as “any attempt to design algorithms or distributed problem solving devices inspired by the collective behavior of social insect colonies and other animal societies…” [2]. The term “swarm” is generally referred to any restrained collection of interacting agents or individuals. In fact, the classical example of a swarm colony is bees swarming around their hive. The metaphor can easily be extended to other species with a similar behavior. Some examples of swarms in different fields include [1]:

- an ant colony whose individual agents are ants,
- a flock of birds is a swarm of birds,
- an immune system is a swarm of cells
- a crowd is a swarm of people.

A few models have been presented to model the intelligent behavior of honeybee swarms, and have been applied to some combinatorial-type problems [2, 3, 4, 5, 6, 7, 8, 9]. There are only a few research articles on numerical optimization problems based on intelligent behavior of honeybee swarms. Yang has developed a Virtual Bee Algorithm (VBA) to solve the numerical optimization problems [10]. VBA is only capable of optimizing the functions with two parameters. Karaboga has presented a bee swarm intelligence algorithm called the *Artificial Bee Colony* (ABC) algorithm [11] for multivariable numerical functions, that has shown good performance compared to other mainstream EAs such as GA, PSO, ACO, DE and others [12, 13].

3.2. Real bees behavior

A model of forage selection that leads to the emergence of collective intelligence of honeybee swarms comprises three components: food sources, employed bees and unemployed bees. We explain them from [14] and [18] for better comprehension of the algorithm's behavior.

1. **Food sources**: a food source quality depends on several factors including its proximity to the nest, richness or the quality of the nectar, and the effortlessness of extracting this nectar. For simplicity, the profitability of a food source can be represented with a single quantity called **food nectar** [18].
2. **Employed bees**: Every time a bee returns to the hive from a patch of flowers, she brings home not only food stored in her pollen baskets and honey stomach, but also the information about her food source stored in her brain. She can share this knowledge with her nest mates by means of the waggle dance communication behavior [18].
3. **Unemployed bees**: Unemployed bees need to locate a food source, either because they are just beginning their forage careers or because they have recently abandoned a depleted patch of flowers [18]. Most such unemployed bees follow the recruitment dances of their nest mates to find a food source [18]. These two types can be summarized as:

3.1. **Onlooker bees**: onlooker bees check the recruitment dances of their nest mates to receive the information of their food source. The exchange of information about food sources that her nest mates present on the dance floor at the hive. Once an employed bee shared its food source position and profitability information with an onlooker bee, the onlooker becomes an employed bee that flies to the food source [18].

3.2. **Scout bees**: these are explorer bees without any guidance while looking for food. As a result of such behavior, the quality of their food source is characterized as low, while occasionally one can accidentally discover a food source rich in quality [18].

3.3. The Artificial Bee Colony (ABC) algorithm

The original Artificial Bee Colony (ABC) algorithm is presented by Karaboga [12] for real (continuous) parameter optimization in unconstrained optimization problems. ABC is a population-based, stochastic global optimizer Evolutionary Algorithm, based on the theory of foraging bees described in the above section. This algorithm demonstrates good accuracy and efficiency, compared with other EAs such as Differential Evolution (DE) [15], Ant Colony Optimization (ACO) [16], PSO and GA, for numeric problems with multi-dimensions [13].

Considering an optimization problem:

$$\max_x F(x) \qquad (3.1)$$

subject to: $x \in \mathbb{Y}$

where $x \equiv (x_1, x_2, \dots, x_D)^T$ is a vector of D real numbers, and $\mathbb{Y}$ is a constraint set. In the original ABC, each candidate solution is represented by a vector variable of the optimization problem. In the context of evolutionary algorithms, a candidate solution is also referred to as an “individual”, and a group of candidate solutions is referred to as a “population” of individuals. In ABC, each individual (candidate solution to an optimization

problem) is analogically considered as a food source location. The fitness value $F(x)$ of each individual (food source) x corresponds to the nectar quality of the food source.

3.3.1. The Artificial Bee Colony Definitions and Algorithm

In this section the definitions of some terms in the ABC algorithm is provided as a first step towards formalizing the ABC algorithm, following by the algorithm's pseudo-code. An exact definition of the algorithm's terminologies is required before presenting the algorithm itself. In the definitions below, $\mathbb{R}$ and $\mathbb{Z}$ are referred to the set of real numbers and integer numbers, respectively.

- ***Definition 3.1*:** A food source position $x \equiv (x_1, x_2, \dots, x_D) \in \mathbb{R}^D$ is a vector of D real numbers, and represents a solution to some problem. x is referred to as an individual.
- ***Definition 3.2*:** A food source position parameter $x_j \in \mathbb{R}$ is a real number representing one coordinate of a food source, where $\mathbb{Y} \in \mathbb{R}$ is the set of all real numbers that are allowed in the problem domain.
- ***Definition 3.3*:** A food source's profitability, or the food source's nectar quality, $F: \mathbb{Y} \to \mathbb{R}$ is a measure of the goodness of the solution that is measured by the bees. (Nectar quality is referred to as "fitness function" in most population-based optimization algorithms.) where $\mathbb{Y} \subseteq \mathbb{R}^D$ is the domain of all feasible solutions to the problem.
- ***Definition 3.4*:** A local environment $\mathcal{X} = \{x^1, x^2, \dots, x^N\}$ is a group of N food source positions, where N is a constant representing the number of food source positions handled by the algorithm during each generation. $\mathcal{X}$ can be referred to as the algorithm population set; while N would be the population size, and x^i **represents the i^{th} individual in the population**.
- ***Definition 3.4:*** Similar to the BBO algorithm, the ABC algorithm is a population-based EA with an iterative nature – i.e. it runs for a number of generations (trials, iterations). See Section 1.1.1.6.

The ABC algorithm is similar to the real behaviour of bees in finding food source position and sharing information to other bees. As mentioned in the previous section, the colony of bees is classified into three types:

- Employed bees
- Onlooker bees
- Scout bees

Each employed bee maintains one location, which analogically is the location of its food source, and there are N employed bees in each generation of this evolutionary algorithm. The N locations maintained by the employed bees are N candidate solutions (individuals) to (3.1) and are referred to as individuals in the population. A pseudo code of the ABC algorithm is given in Table 3.1.

The main steps of the algorithms are presented in the pseudo code in table 3.1.

Table 3.1. The ABC algorithm general pseudo code

```
1. Send scouts (generate initial population)
2. Repeat
3.      Employed bees phase
4.      Onlooker bees phase
5.      Scout bee phase
6.      Memorize the best food source found so far
7. Until termination condition satisfied
```

At the beginning, the algorithm generates the initial set of food source positions (individuals) by sending scouts; i.e. it randomly generates a set of N random vectors, where N denotes the size of employed bees. There are N vectors $x^i,\ i \in \{1,2,\dots,N\}$ in the local environment $\mathcal{X}$ (N individuals in the population), each consists of D real parameters $x^i \equiv \left(x_1^i, x_2^i, \dots, x_D^i\right)^T$, where D denotes the number of optimization parameters.

As soon as a scout founds a new food source (a new vector of parameters generated), she is assigned to the source and she will become an employed bee. An

employed bee then saves the food source position in her memory depending on the local information (visual information) to determine the nectar quality (fitness value) of the food source and keeps that in her memory. As a result, at the end of this step there will be N employed bees, each assigned to N food sources and know the fitness value for all N vector solutions. Next, the algorithm will proceed to the generation loop. This loop will continue until the algorithm meets one of the conditions mentioned in Definition 3.4. A detailed explanation for each of the three phases is as follows:

3.3.1.1. The Employed Bees Phase

In the employed bees phase, an employed bee associated with the i^{th} food source position x^i and has saved its nectar quality in its memory, searches for a new food source in the neighborhood in accordance with the following expression:

$$v_j^i = x_j^i + \phi_j^i\left(x_j^i - x_j^k\right) \qquad (3.2)$$

where x_j^i denotes the j^{th} component of the current food source position (individual), v_j^i denotes the j^{th} component of the new food source position (location of a food source in a neighborhood), $j \in \{1,2,\dots,D\}$ is a randomly selected component of the food source position vector, and $k \in \{1,2,\dots,N\}$ is a randomly chosen food source index such that $i \neq k$. x^k is referred to as a neighbor of x^i. ϕ_j^i is a random number between $[-1,1]$ that controls the production of neighbor food sources around x_j^i and represents the comparison of two food positions visually by a bee. If the nectar quality of the new food source is better than the one she already has in her memory, she remembers this new location and its nectar quality; otherwise she still keeps the location and the nectar quality of the previous source (this is called the *greedy selection*).

There are several issues that have to be considered about the employed bees:

- Note that during this phase, the algorithm modifies only one parameter (the j^{th} component) of the solution vector x^i using (3.2), and copies the rest of the components from x^i to v^i. The expression (3.2) is referred to as the "***ABC's Explorer***".
- The algorithm has a control over the solutions' domain such that if a parameter value generated by (3.2) exceeds its predetermined lower bound and upper bond,

it is set to an acceptable value. For instance, the value of the parameter exceeding the upper bound can be set to its limit value.

- Each employed bee (or its corresponding food source) has a variable associated to it representing the number of trials, denoted by $trial$. It is initially set to zero, and increases by one for each fitness function evaluation. If the new fitness value is better than the prior, the algorithm resets $trial$ to zero. The algorithm uses the value of $trial$ in the scout bees' phase to decide to change an employed bee into a scout.

As described in the section 3.2, after N employed bees have found their new food source positions and tested their nectars, they choose the best food source via greedy selection and return to the hive. Onlooker bees are waiting for them in the dance level to receive the information of the food sources from employed bees.

3.3.1.2. Onlooker Bees Phase

During this phase, first employed bees share their information about the nectar quality of food sources with onlooker bees. An onlooker chooses a food source with a probability related to nectar amount. Better nectar quality of a food source results in its higher probability to be selected by onlookers. An onlooker bee can select an employed bee to follow based on different selection methods. A typical selection method, also presented in the original ABC paper, is the *roulette wheel* selection method [15]. Through this method, the selection probability can be calculated by the following expression:

$$p_i = \frac{F(x^i)}{\sum_{k=1}^{N} F(x^k)} \tag{3.3}$$

where $F_i,\ i \in \{1,2,\dots,N\}$ is the fitness value of the i^{th} solution, which is proportional to the nectar amount of the i^{th} food source. We will present more selection methods in section 3.5.2.

As soon as the onlooker receives the information from an employed bee, she becomes an employed bee has the information about her associated food source position and its nectar quality, and flies to the food source. Now she has become an employed bee associated with that food source. Since then, the new employed bee

(former onlooker) performs the same act as the employed bee in the previous phase; i.e., she searches for a new food source in the neighbor of her associated food source using (3.2) for higher nectar quality, and saves the best food source and its nectar to her memory. Then she tests the nectar of this new food source and selects the better food source via the greedy selection, and keeps that information in her memory.

3.3.1.3. Scout Bees Phase

Scout bees are free bees responsible for finding new food sources and evaluate their nectar. As soon as a scout bee finds a food source, she turns into an employed bee. If there is no improvement in the nectar quality of a particular food source, the algorithm abandons that source, and its associated employed bee turns into a scout bee that randomly searches for a new food source. A scout bee is a former employed bee, and becomes an employed bee again once it has been associated to a food source. She tests the nectar and saves it in her memory, and returns to the hive to dance in front of onlookers. The number of bees in the hive remains intact N during the algorithm

During each cycle in the original ABC algorithm, maximum of one employed bee is selected and classified as the scout bee [17, 12]. If there is more employed bee to become a scout, the algorithm selects only one employed bee randomly. The classification is controlled by a control parameter "$trial$" that has to be less than a predetermined upper-bound t. If a food source is not improved after t number of trials, the algorithm removes it from the population in the scout bees' phase, and the employed bee associated to the food source turns into a scout that searches for a new food source. In other words, scout bees are sent for those food sources that $\forall x^i \mid \{trail^i > t\}$.

The food source of which the nectar is abandoned is replaced with a new food source, which is simulated by producing a position randomly and replacing it with the abandoned one. The scout randomly generates a new food source $\hat{x}^i \equiv (\hat{x}_1^i, \hat{x}_2^i, \dots, \hat{x}_D^i)$ to be replaced with the abandoned source x^i, and each component of the new food source $\hat{x}^i$ – i.e. $\hat{x}_j^i$, $j \in \{1,2, \dots, D\}$ – is randomly generated using the following expression:

$$\hat{x}_j^i = x_{min} + rand[0,1](x_{max} - x_{min}), \quad \forall j \qquad (3.4)$$

where x_{max} and x_{min} are the upper bound and the lower bound of a feasible parameter variable. If the constraint set $\mathbb{Y}$ in (3.1) has more complexity than a box, then the algorithm should incorporate a constraint checking procedure to make sure that each individual is feasible. As soon as a scout finds a new food source position, she turns into an employed bee. This employed bee then measures the nectar quality of the food source, keeps it in her memory, and returns to the hive.

At the end of each cycle, the best solution is chosen via sorting the nectar qualities of all N employed bees. If the new value is better than that of the previous generation, the algorithm saves the fitness value and the corresponding food source position (individual).

Comparison between the Scout Bees Phase and Mutation

The role of the scout bees in the algorithm is similar to the "mutation" procedure in other EAs like Genetic Algorithm [15]. The GA applies mutation to a predetermined portion of its population during every algorithm generation. Each individual in the portion would be selected for mutation with respect to a mutation probability, and its parameters will be replaced with randomly generated values.

The scout bees' phase in the ABC algorithm has an advantage and a disadvantage compared with the GA's mutation process. The advantage of the scout bee procedure is in using the $trial$ variable. ABC selects only those individuals that have not improved after a predetermined number of trials indicated by the variable "$trial$"; while the mutation procedure do not have such an intelligent choice of individuals. There is a chance of losing a potential good solution in the mutation procedure, because individuals are being selected completely random.

On the other hand, the drawback of the scout bees' phase in ABC algorithm is that it decreases exploration by enforcing maximum of one food source replacement during each generation. Even though this disadvantage is a part of the original ABC algorithm and has been employed in various applications, we have removed this limitation, and improved the algorithm's performance as explained in subsection 3.5.3.

In order to employ the ABC algorithm in some wireless communication problems, we have to modify the algorithm to operate in the discrete domain. As a result, we present the Discrete Artificial Bee Algorithm (DABC) in the next section, and we further present other developments to the original ABC algorithm in subsequent sections.

3.4. Discrete Artificial Bee Colony Algorithm

The original ABC algorithm was presented for continuous optimization problems [12]. Almost all of the applications and optimization problems solved by this algorithm available in literature have encompassed a continuous (real) domain, while there are less than a few publications on implementing ABC to discrete problems (the only two published approaches are explained in Sections 3.5.1.2. and 3.5.1.3). However, some applications in wireless communications, e.g. problems in symbol detection and resource allocation, have a discrete or binary nature. The ABC algorithm has a high performance compared to other EAs such as GA, DE, PSO, etc. in the continuous domain [13]. With a motivation to apply the ABC algorithm's idea to solving discrete optimization problems, we developed a discrete version of the algorithm and assessed its performance on problems in the discrete domain.

The definition of the terms employed in the ABC algorithm is given in section 3.3.1. Here we restate some of those definitions incorporated in the DABC (Discrete Artificial Bee Colony) algorithm. We have the following definitions for the optimization problem below:

$$\max_x F(x)$$

$$\text{subject to: } x \in \mathbb{Y}$$

Definition 3.5: A food source position $x \equiv (x_1, x_2, \dots, x_D) \in \mathbb{Z}^D$ is a vector of D integer numbers, and represents a candidate solution to the problem. Vector x is referred to as an individual as in most population-based EAs.

The principles of the DABC algorithm would be the same as the ABC algorithm in some steps. Employed bees, onlooker bees and scouts retain their responsibilities.

However, some of the expressions comprising continuous solution vectors and their parameter values have to be adjusted for discrete computation.

During the employed bees and onlooker bees phases, the ABC uses (3.2) to produce a candidate food position from the old one in memory. DABC finds a new food source v^i in the neighborhood of the current food source position x^i. Each components of the new food source location v_j^i is derived using the following expression:

$$v_j^i = randint\left[\min\{x_j^k\ ,\ 2x_j^i - x_j^k\}\ ,\ \max\{x_j^k\ ,\ 2x_j^i - x_j^k\}\right] \tag{3.5}$$

where "randint (u, v)"' returns a random integer number between u and v, $j \in \{1,2,\dots,D\}$ is a random component of a food source position vector and $i, k \in \{1,2,\dots,N\}$ are randomly chosen indexes, where $i \neq k$. This expression is held in both employed and onlooker bees phases. The evaluation of (3.5) is expressed in Appendix of this chapter.

Similarly, scout bees have to explore new food source positions in a discrete set. In DABC, a scout discovers a new food source $x^i \equiv \left(\hat{x}_1^i, \hat{x}_2^i, \dots, \hat{x}_D^i\right)^t$ to replace it with the abandoned source x^i. The discrete parameter variables $\hat{x}_j^i$, $j \in \{1,2,\dots,D\}$ of the new food source position can be randomly generated using the following expression:

$$\hat{x}_j^i = randint\ [x_{min}\ , x_{max}] \tag{3.6}$$

where x_{min} and x_{max} are the minimum and the maximum possible values for a feasible parameter variable, and $randint(u, v)$ is a function that generates random integer numbers between u and v. Expression (3.6) is used by scouts at the initial step where population is generated, and during each interval in the scout bees' phase. A pseudo code of the DABC algorithm is given in Table 3.2.

Table 3.2. The DABC Algorithm Pseudo Code

1. Initialize the population of solutions $x^i,\ i = 1,2,\dots,N$,
2. Evaluate $F(x^i), \forall i$,
3. **for** g = 1 to G %(maximum algorithm iterations)%
4. run the modified employed bee phase, (Table 3.3)
5. run the modified onlooker bee phase, (Table 3.4)
6. run the modified scout bee phase, (Table 3.5)
7. save the best results,
8. end **for**,

Table 3.3. The DABC Employed Bee Phase Pseudo Code

1. $trial^i = 0,\ \forall i$
2. **for** each food source $x^i,\ i = 1,2,\dots,N$,
3. Select a random food source $k,\ k \neq i \in \{1,2,\dots,D\}$,
4. Select a random component $j,\ j \in \{1,2,\dots,D\}$,
5. $v_j^i = randint\left[\min\{x_j^k\ ,\ 2x_j^i - x_j^k\}\ ,\ \max\{x_j^k\ ,\ 2x_j^i - x_j^k\}\right]$
6. $trial^i = trial^i + 1$,
7. **if** $v^i \neq x^i$ **then**,
8. Evaluate $F(v^i)$,
9. $x^i \leftarrow v^i$,
10. $trial^i = 0$,
11. end **if**,
12. end **for**,

Table 3.4. The DABC Onlooker Bee Phase Pseudo Code

1. Calculate probability values p_i for x^i, $\forall i$ using (3.9~3.12),
2. $t = 1; i = 1;$
3. **for** $t = 1, \dots, N$, %(t corresponds to the t^{th} onlooker bee)%
4. **if** $rand > p_i$ **then**,%(select the i^{th} employed bee to follow)%
5. Select a random food source k, $k \neq i \in \{1,2, \dots, D\}$,
6. Select a random component j, $j \in \{1,2, \dots, D\}$,
7. $v_j^i = randint\left[\min\{x_j^k , 2x_j^i - x_j^k\} , \max\{x_j^k , 2x_j^i - x_j^k\}\right]$
8. $trial^i = trial^i + 1$,
9. **if** $v^i \neq x^i$ **then**,
10. Evaluate $F(v^i)$,
11. $x^i \leftarrow v^i$,
12. $trial^i = 0$,
13. end **if**,
14. $t = t + 1$,
15. end **if**
16. $i = i + 1;$
17. **if** $i > N$ **then** $i = 1;$ %(reset i)%
18. end **for**,

Table 3.5. The DABC Scout Bee Phase Pseudo Code

1. **for** $\forall x^i \mid \{trial^i > t\}$,
2. **for** each component j, $j \in \{1,2, \dots, D\}$,
3. $\hat{x}_j^i = randint\,[x_{min} , x_{max}]$,
4. end **for**,
5. Evaluate $F\left(\hat{x}^i\right)$,
6. $x^i \leftarrow \hat{x}^i$,
7. $trial^i = 0$,
8. end **for**,

3.5. Improvements to the DABC algorithm

In addition to our contribution to the original ABC algorithm and developing the DABC algorithm, we have further enhanced the DABC algorithm by optimizing different steps of ABC to increase its performance. The following subsections discuss these developments to the DABC algorithm, which some of them are applicable to the ABC as well, and some are further implemented to the hybrid algorithm discussed in Chapter 4.

3.5.1. Selecting a Neighbor Food Source

In every algorithm generation, employed bees select a neighbor food source twice. The first attempt is during the employed bees phase, when they are exploiting their current food source position. The second time is during the onlooker bees phase, while they are exploiting the area around the source they have received their information from employed bees. In this subsection, we study the selection methods applicable to the integer and binary domain.

3.5.1.1. Random Bounded Integer Selection for Integer Problems

The first neighborhood selection method for the DABC algorithm is the one we presented in (3.5), we employ a random integer generator. The advantage of this method is that since it is applicable to the problems in the integer domain, it works with the problems in the binary domain as well. Our simulation results also prove that this method outperform the other two neighborhood selection methods for integer problems.

3.5.1.2. Using Sigmoid Function for Binary Problems

Wang et al. have presented a discrete selection method to the ABC algorithm in [18]. Their binary encoding method employs a sigmoid function of velocity as a logical choice for binary selection. However, this procedure only applies to the binary optimization problems, whereas the explorer procedure proposed in this thesis is applicable to larger class of problems in the integer or discrete domain.

3.5.1.3. Using Logical Selection Expression for Binary Problems

Salim et al. present another neighbor selection method for binary problems in [19]. They replace the operation signs with (binary) logical operators such as “OR” ($\vee$),

"AND" ($\wedge$), and "XOR" ($\oplus$) functions. Similar to the last method, this expression is only applicable to binary problems.

3.5.2. Employed Bee Selection Probability

As mentioned before, the preference of a food source by an onlooker bee depends on the nectar amount $F(x^i)$ of that food source. As the nectar amount of the food source increases, the probability with the preferred source by an onlooker bee increases proportionally. Clearly, in such a scheme good food sources should have more chance (higher probability) to be selected by onlooker. In the original ABC algorithm, this phenomenon is applied using the Roulette Wheel Selection method stated in (3.3). We have employed some other selection probability expressions to our simulations, including the following:

$$p_i = \frac{a\,.\,F(x^i)}{\sum_{k=1}^{N} F(x^k)} + b,\ i \in \{1,2,\dots,N\} \tag{3.9}$$

$$p_i = \frac{F(x^i)}{\left(\sum_{k=1}^{n} F(x^k)\right)^2},\ i \in \{1,2,\dots,N\} \tag{3.10}$$

$$p_i = \frac{F(x^i)}{\max_k\{F(x^k)\}},\ i,k \in \{1,2,\dots,N\} \tag{3.11}$$

$$p_i = \sqrt{\frac{F(x^i)}{\max_k\{F(x^k)\}}},\ i,k \in \{1,2,\dots,N\} \tag{3.12}$$

Our simulations demonstrate that the last expression (3.12) will yield to the better results. Analysis of these probabilities and algorithm's performance for different classes of problems based on them is left for future work.

3.5.3. Improvements to Scout Bees Phase

Another improvement we present to the ABC algorithm is simple, yet effective that boosts its performance. The original ABC algorithm selects only one food source to abandon during each algorithm generation, and replaces its associated employed bee with a scout. We have removed this limit of "maximum one" food source, and allow the

algorithm to choose all existing food sources $x^i,\ i \in \{1,2,\dots,N\}$ of which $trial^i > t$. The removal of this limit boosts the exploration attribute of the algorithm, and increases the algorithm's chance to get out of local optima.

References

[1] Wolpert, D.H.; Macready, W.G.; , "No free lunch theorems for optimization," *Evolutionary Computation, IEEE Transactions on* , vol.1, no.1, pp.67-82, Apr 1997

[2] E. Bonabeau, M. Dorigo, *G. Theraulaz, Swarm Intelligence: From Natural to Artificial Intelligence,* Oxford University Press, New York, NY,1999.

[3] L. a. De Castro, J. Timmis, *Artificial Immune Systems: A New Computational Approach*, Springer-Verlag*,* New York, NY, 2002.

[4] P. Miller; *The Smart Swarm: How Understanding Flocks, Schools, and Colonies Can Make UsBetter at Communicating, Decision Making, and Getting Things Done,* Avery, New York, NY, 2010.

[5] V. Tereshko, "Reaction-diffusion model of a honeybee colony's foraging behaviour, Springer- Berlin," *Lecture Notes in Computer Science, vol. 1917, no. M. Schoenauer, et al. (Eds.), Parallel Problem Solving from Nature VI*, p. 807–816, 2000.

[6] T. Lee, V. Tereshko, "How information mapping patterns determine foraging behaviour of a honey bee colony," *Open Syst. Inf. Dyn*, vol. 9, pp. 181-193, 2002.

[7] A. Loeengarov. V. Tereshko, "Collective decision-making in honey bee foraging dynamics," Computing Information System Journal, vol. 9, no. 3, pp. 1-7, 2005.

[8] D. Teodorovic, "Transport modeling by multi-agent systems: a swarm intellgence approach," *Transport. Plann. Technology*, vol. 26, no. 4, pp. 289-312, 2003.

[9] D. Lucic, "Transportation Modeling: An Artificial Life Approach," in *ICTAI*, 216-223, 2002.

[10] D. Teodorovic, M. Dell'orco; "Bee colony optimisation—a cooperative learning approach to complex transportation problems," in *10th EWGT Meeting,* Poznan, 2005.

[11] K. Benatchba, L. Admane, and M. Koudil, "Using bees to solve data-mining problem expressed as a max-sat one, artificial intelligence and knowledge engineering applications: a bioinspired approach," in *First International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2005,* Las Palmas, Canary Islands, Spain, 2005.

[12] H. Wedde, M. Farooq, Y. Zhang, "BeeHive: an efficient fault-tolerant routing algorithm inspired by honey bee behavior, ant colony, optimization and swarm intelligence," in *4th International Workshop, ANTS 2004*, Brussels, Belgium, September 5-8, 2004.

[13] X. Yang, "Engineering Optimizations via Nature-Inspired Virtual Bee Algorithms,," *Lecture Notes in Computer Science, Springer-Verlag GmbH*, no. 3562, p. 317, 2005.

[14] D. Karaboga, "An Idea Based On Honey Bee Swarm For Numerical Optimization," *Technical Report-TR06,* Erciyes University, 2005.

[15] D. Karaboga. B. Basturk, "An artificial bee colony (ABC) algorithm for numeric function optimization," in *IEEE Swarm Intelligence Symposium 2006,* Indianapolis, IN, May 12-14, 2006.

[16] B. Akay. D. Karaboga, "A comparative study of Artificial Bee Colony algorithm," *Applied Mathematics and Computation,* no. 214, pp. 108-132, 2009.

[17] R. Jeanne, "The evolution of the organization of work in social insects," *Monit. Zool. Ital.,* vol. 20, pp. 267-287, 1986.

[18] T. Seeley, *The wisdom of the hive: the social physiology of honey bee colonies,* Harvard University Press,Cambridge, MA, 1995.

[19] D. Goldber, *Genetic Algorithms in Search, in: Optimization and Machine Learning,* Addison-Wesley Pub. Co., 1989.

[20] V. Maniezzo; A. Colorni; M. Dorigo, "Ant System: Optimization by a Colony of Cooperating Agents," *IEEE Transactions on Systems, Man, and Cybernetics–Part B*, vol. 26, no. 1, pp. 1-13, 1996.

[21] B. Basturk. D. Karaboga, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing*, vol. 8, p. 687–697, 2008.

[22] J. Wang, T. Li, R. Ren, "A Real Time IDSs Based on Artificial Bee Colony Support Vector Machine Algorithm," in *Third International Workshop on Advanced Computational Intelligence,* Suzhou, Jiangsu, China, Aug. 25-27, 2010.

[23] M. Salim; T. Vakil-Baghmisheh, "Discrete bee algorithms and their application in multivariable function optimization," *Artificial Intellignece Rev. Springer Journal on,* vol. 35, pp. 73-84, 2011.

Appendix.

Neighborhood Food Source Selection in Discrete ABC

In (3.2), ϕ_i^j is a random number between $[-1,1]$. We choose the minimum and maximum of this interval and apply it to (3.2). For the minimum we have $\phi_i^j = -1$; hence:

$$v_i^j = x_i^j + (-1)\left(x_i^j - x_k^j\right) \quad \text{A.1}$$

$$v_i^j = x_k^j$$

For the maximum we have $\phi_i^j = +1$; as a result:

$$v_i^j = x_i^j + (1)\left(x_i^j - x_k^j\right) \quad \text{A.2}$$

$$v_i^j = 2x_i^j - x_k^j$$

Putting (B.1) and (B.2) together, replacing real parameter values x_i^j with integer parameter values $\underline{x_i^j}$, and using a random integer number generator function $randint(u, v)$ that returns a random integer between u and v, we obtain:

$$v_i^j = randint\left[x_k^j, \quad 2x_i^j - x_k^j\right] \quad \text{A.3}$$

However, there is no guarantee that $x_k^j \leq 2x_i^j - x_k^j$. Therefore, we generalize (B.3) to:

$$v_i^j = randint\left[\min\left\{x_k^j, 2x_i^j - x_k^j\right\}, \max\left\{x_k^j, 2x_i^j - x_k^j\right\}\right]. \quad \text{A.4}$$

4. Hybrid ABC/BBO Algorithm

In this chapter we propose a novel Evolutionary Algorithm for optimization problems in both continuous and discrete domains. This new algorithm intends to combine the advantages of both BBO and ABC algorithms. We refer to this new algorithm as hybrid ABC/BBO algorithm. This algorithm has shown good performance in comparison to other EAs when applied to some optimization problems in wireless communication, including single-objective unconstrained (e.g. MD-STBC-MIMO (Chapter 5)), single-objective constrained (e.g. Relay Assignment in Cognitive Radio (Chapter 6)), and multi-objective constrained optimization problems (e.g. Green Resource Allocation in Cognitive Radio (Chapter 7)). In this chapter starts with an introduction to the hybrid EAs in the next section, and followed be a brief discussion on advantages and disadvantages of BBO and ABC in Section 4.2. Then the novel hybrid ABC/BBO is presented in Section 4.3.

4.1. Introduction

For some problems, a simple evolutionary algorithm might be good enough to find a desired solution. As reported in the literature, there are several types of problems in which an application of a known EA could fail to obtain a good solution [1, 2, 3, 4, 5]. Recently, the hybridization of EAs has become popular due to its capabilities for handling several real world problems involving complexity, noisy environment, imprecision, uncertainty, and vagueness [11]. Some of the possible reasons for hybridization are:

1. To improve the performance of the evolutionary algorithm (e.g. speed of convergence)
2. To improve the quality of the solutions obtained by the evolutionary algorithm
3. To incorporate the evolutionary algorithm as a part of a larger system.

In order to balance the exploration and the exploitation in an EA, in this chapter we propose a hybrid algorithm with ABC and BBO for the global numerical optimization problems. In this hybrid algorithm, a hybrid migration operator is proposed, which combines the exploration of ABC with the exploitation of BBO effectively. **Exploitation** refers to the algorithm's tendency to use the values it already has in order to refine its search for an optimal solution, whereas **exploration** refers to the algorithm's ability to search in previously-unexplored regions of the search space [8].

4.2. Discussion on ABC and BBO

In this section first the strong and weak points of ABC and BBO are discussed, following by the origin of the idea of hybridizing the two algorithms.

4.2.1. BBO's Pros and Cons

BBO's main operator is migration. As illustrated in Table 2.1, during the migration procedure, each SIV (individual component) can be *replaced* with another existing SIV from another habitat (individual); e.g. $H^8(s_3)$ (third component of the 8th individual) can be replaced with $H^i(s_3)$, where H^i is another habitat being selected based on its emigration rate. Thus, all possible values for new $H^8(s_3)$ are other individuals' third components values, and no new value is generated. The migration operator of BBO has a good *exploitation* attribute; i.e., BBO's migration procedure uses the values it already has in order to refine its search for an optimal solution. After the initial population is generated at the beginning of the algorithm, during the next iterations the algorithm shares the individual components (habitats' SIVs) as a result of the migration operator. In this phase, SIVs are copied from a habitat to another; and thus all current SIV values retain in the population, and are highly dependent on the initial population.

BBO is proposed for discrete optimization problems, and its behavior best works for the problems in the integer domain, because there are a few (discrete) possible values for each SIV. However, if the number of choices are fairly large (large discrete domain), or more importantly, if it is applied to the problems in the continuous domain, then this migration feature would be a downside of BBO because it lacks *exploration*.

The migration process in unable to *explore* new SIVs; i.e., it only copies SIVs from one habitat to another, and does not search previously unexplored regions of the search space and cannot generate new values. In fact, BBO takes the advantage of its mutation operator for exploration. Yet, the mutation factor m has to be set to a very small number. Dan Simon mentions in [7] that a high mutation rate of 10% ($m = 0.1$) results in too much exploration. However, the performance of the algorithm increases as the mutation rate decreases to the more reasonable values of 1% and 0.1%; that is $m = 0.1$ and $m = 0.1$, respectively [7]. He also applies BBO to a real world problem in [13] and applies mutation with $m = 0.01$. Fig. 4 in this chapter depicts the BBO simulation result with and without mutation, and it is clear from this figure that even with a well-tuned mutation factor, there is only a slight improvement in the result of the BBO with mutation.

4.2.2. ABC's Pros and Cons

ABC does not use any gradient-based operator. It incorporates a simple mechanism to adapt to the global and local exploration abilities within a short computation time, while it uses just a few control parameters. Hence, this method is can be used for solving multimodal and multidimensional optimization problems [15]. ABC's main evolutionary process is during its employed bee and onlooker bee phases, where the algorithm *explores* the search space for finding new food sources in previously un-explored locations. The algorithm can locate the region of global minimum during the employed bee and onlooker bee phases, as a result of the exploring actions of the employed bees while they are finding new food sources around their current associated sources (equations (3.2) and (3.5)). We refer to these equations as "***ABC's explorer***". This explorer operator effectively generates new values for a component of an individual.

ABC's downside, apart from its great exploring feature, is *exploitation* – i.e. in ABC, the algorithm does not retain the values of its population. In the employed bee and onlooker bee phases of the original (continuous) ABC, all food sources have to be modified by the explorer operator (equation (3.2)). An individual component survives changes (its value would be unmodified during these two phases) only if

ϕ_j^i = -1, then $v_j^i = x_j^i$,

ϕ_j^i = 0, then $v_j^i = x_j^i$.

Only in these two cases, the new food source position v_j^i preserves the values of a component currently inside the population; i.e., x_j^i or x_j^k. However, because ϕ_j^i is a randomly generated real number in the closed interval $[-1,1]$, it is highly unlikely that the value of ϕ_j^i would be exactly -1 or zero.

Another disadvantage of ABC is the *uniform selection* of a neighbor food source during the employed bee and onlooker bees phases. During these two phases, all food sources are equally likely to be selected as a neighbor of a food source. The only condition is that a neighbor has to be different from the food source itself; i.e. $k \neq j$. However, the algorithm could have developed more intelligently such that high quality food sources would be more likely to be chosen as a neighbor. The result of this small yet effective modification is a more efficient exploration. The original ABC has some other drawbacks that are discussed in Chapter 3, and some improvements are proposed in sections 3.4 and 3.5.

4.3. The Hybrid ABC/BBO algorithm

There are certain reasons to choose ABC with BBO for hybridization among other EAs. The first reason is that these two algorithms already have shown good performance in comparison with other EAs for the optimization problems of our interest and similar applications [8, 9, 13, 14, 16,]. Therefore we can predict the hybridization of these two algorithms may outperform other EAs even further. More importantly, BBO and ABC's advantages and disadvantages are such that the disadvantage of one is the strong feature of the other. BBO benefits from exploitation, and lacks exploration; while ABC has a great exploration feature, and its drawback is exploitation. Therefore a wise idea would be to integrate the exploration of ABC with the exploitation of BBO to develop an algorithm that probably outperforms ABC and BBO solely.

We explain the procedure of the new algorithm by taking the general scheme of ABC and implementing BBO's feature into that. The hybrid algorithm consists of three phases: employed bee phase, onlooker bee phase, and scout phase. It benefits from the BBO's migration procedure, therefore it contains emigration rate and immigration rate curves. This new migration operator is referred to as the "Hybrid Migration Operator".

4.3.1. The Hybrid Migration Operator

The main operator of the hybrid algorithm is its hybrid migration operator, which hybridizes the employed bees' behavior with the BBO's migration operator, and is described in Table 4.1. The terminology is the same as Section 3.3.1 for ABC. According to this procedure, the food source position v^i is constituted by three components:

- the ABC's explorer,
- the migration of other solutions,
- its corresponding food sources x^i and x^k.

The core idea of the proposed hybrid migration operator is based on two considerations. First, good solutions would be less destroyed, while poor solutions can accept many new features from good solutions. In this sense, the current population can be exploited sufficiently. Second, the ABC's explorer operator is able to explore the new search space and make the algorithm more robust. Correspondingly, the original BBO migration operator focuses on the intensification; while the ABC's explorer operator emphasizes on the diversification. From this analysis, it can be seen that the hybrid migration operator can balance the exploration and the exploitation effectively.

Table 4.1. Hybrid Migration Operator for the i^{th} individual

1. select a random component j, $j \in \{1, \dots, D\}$
2. **if** λ^i **then,**
3. select x^k with probability $\propto \mu^k$
4. **if** $rand < \mu^k$ **then,**
5. $v_i^j = x_j^i + \phi_i^j \left(x_j^i - x_j^k\right)$ for continuous problems, or
   $v_j^i = randint\left[\min\{x_j^k\ ,\ 2\,x_j^i - x_j^k\}\ ,\ \max\{x_j^k\ ,\ 2x_j^i - x_j^k\}\right]$ for discrete problems
6. end **if**
7. else
8. $v_j^i = x_j^i$
9. end **if**

4.3.2. Main Procedure of the Hybrid Algorithm

We incorporate the above hybrid migration operator into ABC, and develop the hybrid ABC/BBO algorithm as described in Section 4.3. The algorithm needs to calculate λ and μ before running the migration operator. Compared with the ABC algorithm described in Sections 3.3 and 3.4, our approach needs only a small extra computational cost for sorting the population and calculating the migration rates. In addition, the structure of our proposed ABC/BBO is also very simple. Moreover, ABC/BBO is able to explore the new search space with the explorer operator of ABC, and to exploit the population information with the migration operator of BBO. This feature overcomes the lack of exploitation of the original ABC algorithm. In the hybrid ABC/BBO, we have also modified the ABC selection method to benefit from the BBO migration rates. Therefore, this algorithm evolves more intelligently compared with original ABC algorithm and BBO.

Table 4.2. The Main Pseudo-Code for Hybrid ABC/BBO

```
1. Initialize the population of solutions x^i, i = 1,2,...,N,
2. Evaluate F(x^i), ∀i,
3. Sort population
4. Calculate μ^i and λ^i, ∀i,
5. for g = 1 to G %(maximum algorithm iterations)%
6.       Run the modified employed bee phase, (Table 4.3)
7.       Sort the population
8.       Calculate μ^i and λ^i, ∀i,
9.       Run the modified onlooker bee phase, (Table 4.4)
10.      Run the modified scout bee phase, (Table 4.5)
11.      Save the best results,
12. end for,
```

Table 4.3. The Hybrid Algorithm's Employed Bee Phase Pseudo Code

1. $trial^i = 0, \ \forall i$
2. **for** each food source $x^i, \ i = 1,2,\dots,N$,
3. run the Hybrid Migration Operator (Table 4.1)
4. $trial^i = trial^i + 1$,
5. **if** $v^i \neq x^i$ **then**,
6. Evaluate $F(v^i)$,
7. $x^i \leftarrow v^i$,
8. $trial^i = 0$,
9. end **if**,
10. end **for**,

Table 4.4. The Hybrid Algorithm's Onlooker Bee Phase Pseudo Code

1. Calculate probability values p_i for $x^i, \forall i$ using (3.9~3.12),
2. $c = 1; i = 1;$
3. **for** $t = 1,\dots,N$, %(t corresponds to the t^{th} onlooker bee)%
4. **if** $rand > p_i$ **then**,%(select the i^{th} employed bee to follow)%
5. run the Hybrid migration operator (Table 4.1)
6. $trial^i = trial^i + 1$,
7. **if** $v^i \neq x^i$ **then**,
8. Evaluate $F(v^i)$,
9. $x^i \leftarrow v^i$,
10. $trial^i = 0$,
11. end **if**,
12. $c = c + 1$,
13. end **if**
14. $i = i + 1;$
15. **if** $i > N$ **then** $i = 1;$ %(reset i)%
16. end **for**,

Table 4.5. The Hybrid Algorithm's Scout Bee Phase Pseudo Code

1. **for** $\forall x^i \mid \left\{trial^i > t\right\}$,
2. **for** each component $j,\ j \in \{1,2,\dots,D\}$,
3. $\hat{x}_j^i = randint\,[x_{min}, x_{max}]$,
4. end **for**,
5. Evaluate $F\left(\hat{x}^i\right)$,
6. $x^i \leftarrow \hat{x}^i$,
7. $trial^i = 0$,
8. end **for**,

It is worth pointing out that the computational complexity of this algorithm, based on the number of fitness function evaluations, is similar to the ABC's complexity; i.e. the hybrid algorithm only has some more if conditions and numerical operators, and does not include any more fitness evaluations. However, its complexity is higher than BBO; in the sense that BBO runs with a total of GN fitness evaluations, where G is the number of algorithm iterations and N is the population size. But both ABC and hybrid algorithm once evaluate all the population in the employed bee phase, and once again in the onlooker bee phase, in addition to evaluations for some mutated individuals in the scout bee phase. However, because of the if conditions in lines 5 of Table 3.3 and 7 of Table 3.4, the number of evaluations reduced to only those individuals which have been modified during the exploring phase in ABC, and the hybrid migration in the hybrid algorithm. The stochastic nature of these two algorithms does not allow us to find a closed form expression for their complexity. More discussion on the issue of algorithms computational complexity is provided in Section 5.5, where we derive the complexity of the original ABC algorithm as well.

4.3.3. Configuring the Algorithm

Based on different options and a number of improvements discussed for BBO and ABC in the previous two chapters, the hybrid algorithm can be developed into

various configurations. Some of these configuration options, according to the previous chapters, are as follows:

- **Continuous and discrete problems:** To apply the algorithm to continuous or discrete problems, the expression in line 5 of Table 4.1 and line 3 of Table 4.5 has to be chosen correspondingly. Our results in Part II of this thesis illustrate that the hybrid algorithm outperform other EAs in both discrete and continuous problems.
- **The migration scheme:** the hybrid migration operator's immigration and emigration rates curves can be any of the three schemes presented in Section 2.3. However, we use the Linear Immigration (Piece-wise Emigration scheme (2.3.3)) because of its reasonable performance and less complexity than the third model (2.3.1).
- **Employed bee selection probability:** onlooker bees can select employed bees at the beginning of the onlooker bees phase, based on different probability expressions discussed in Section 3.5.2. In our implementations presented in Part II we choose the expression that results in the best performance.
- **Mutation operator:** The hybrid algorithm has different ways to handle mutation. The scheme mentioned in Table 4.5, and is employed in our implementations, is the enhanced scout bee phase described in Section 3.5.3. Another scheme is the original ABC's scout bees phase, which is not as efficient as the aforementioned operator. The BBO's mutation operator is another choice, where each food source has the chance to be mutated, as discussed in Section 2.2.2. The mutation scheme in Table 4.5 is more efficient than BBO's mutation operator, since it only replaces those individuals that have not been improved during a certain number of iterations.

The algorithm's settings are given for each of the application implementations in Part II of the thesis, and the simulation results demonstrate the superiority of the algorithm over BBO and ABC, as well as other EAs.

4.4. Algorithms' Computational Complexity

The complexity of the introduced EAs – BBO, ABC and hybrid – in terms of the $\mathcal{O}(\ )$ notation is presented in this section. This complexity has been calculated based on the algorithms' pseudo codes discussed in sections 2.2, 3.4 and 4.4. Note that the

complexities of ABC and hybrid algorithms are presented for the discrete version of the algorithms. These two algorithms are running the simple scout bees phase, where the algorithm selects only one individual that has exceeded the predetermined number of trials; similar to the scout bees procedure introduced in the original ABC paper [15]. Table 4.6 demonstrates the computational complexities of these algorithms and their comprising procedures.

In this table $\mathcal{F} = \mathcal{O}(F)$ refers to the computational complexity of the objective function, n denotes the number of feasible discrete numbers between x_{min} and x_{max}, and G, N and D have been already defined as the maximum number of generations, population size and the number of components in an individual vector, respectively. BBO and hybrid include one or more sorting procedures. The computational complexity of some sorting algorithms such as selection sort, insertion sort, bubble sort and cycle sort is $\mathcal{O}(N^2)$, while the computational complexity of some other sorting algorithms such as binary tree sort, tournament sort and merge sort is $\mathcal{O}(N \log N)$ [17]. In Table 4.6 the first group of sorting algorithms has been assumed that has a complexity of $\mathcal{O}(N^2)$.

Table 4.6. Computational Complexity of BBO, ABC and hybrid algorithms

BBO	Migration	$\mathcal{O}(N^2 D)$
	Mutation	$\mathcal{O}(N)$
	Complete Algorithm	$\mathcal{O}(GN^2 D + G\mathcal{F})$
ABC	Employed Bees Phase	$\mathcal{O}(Nn + N\mathcal{F})$
	Onlooker Bees Phase	$\mathcal{O}(N^2 + Nn + N\mathcal{F})$
	Scout Bees Phase	$\mathcal{O}(D + \mathcal{F})$
	Complete Algorithm	$\mathcal{O}(GN^2 + GNn + GN\mathcal{F})$
Hybrid	Hybrid Migration Procedure	$\mathcal{O}(N + n)$
	Employed Bees Phase	$\mathcal{O}(N^2 + Nn + N\mathcal{F})$
	Onlooker Bees Phase	$\mathcal{O}(N^2 + Nn + N\mathcal{F})$
	Scout Bees Phase	$\mathcal{O}(D + \mathcal{F})$
	Complete Algorithm	$\mathcal{O}(GN^2 + GNn + GN\mathcal{F})$

The difference in the computational complexity order between BBO's migration and ABC/hybrid employed/onlooker bees phases is because the migration procedure

alters every component of an individual using migration rates, but employed bees only select one component randomly and run line 5 of Table 4.1 to find a new food source. If the employed bees were set to change all components of an individual, the complexity of these phases would have contained the term N^2D instead of N^2. It is also interesting to see that ABC and hybrid algorithm have the same complexity order. In Part II of this thesis, the computational complexity of the applications have been compared based on other methods such as elapsed time or the average number of fitness function evaluation (e.g. Section 5.6.2).

References

[1] S. Ashrafinia, "Critical Node Detection Problem," *Directed Studies Course Report,* Simon Fraser University, Burnaby, BC, 2011.

[2] LY Tseng and SC. Liang, "A hybrid metaheuristic for the quadratic assignment problem," *Comp. Opt. and Applications*, vol. 34, no. 1, pp. 85-113, 2005.

[3] R. Lakshmiramanan, K. Kuppusamy P. Somasundaram, "Hybrid algorithm based on EP and LP for security constrained economic dispatch problem," *Electric Power Systems Research,* vol. 76, no. 1-3, pp. 77-85, 2005.

[4] R. Morgan, and D. Williams F. Li, "Hybrid genetic approaches to ramping rate constrained dynamic economic dispatch," *Electric Power Systems Research*, vol. 43, no. 11, pp. 97-103, 1997.

[5] WH. Chang CC. Lo, "A multiobjective hybrid genetic algorithm for the capacitated multipoint network design problem," *IEEE Transactions on Systems, Man and Cybernetics - Part B*, vol. 30, no. 3, pp. 461-470, 2000.

[6] WG. Macready DH. Wolpert, "No free lunch theorems for optimization*," IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67-82, 1997.

[7] D. Simon, M. Ergezer, D. Du, and R. Rarick, "Markov Models for Biogeography-Based Optimization," *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, vol. 41, no. 1, pp. 299–306, February 2011

[8] Simon, D., his brief discription is available at the website: . http://academic.csuohio.edu/simond/courses/eec693b/homework.html, last accessed summer 2012.

[9] Ashrafinia, S.; Pareek, U.; Naeem, M.; Lee, D.; "Biogeography-based optimization for joint relay assignment and power allocation in cognitive radio systems", *Swarm Intelligence (SIS), 2011 IEEE Symposium on,* 11-15 April 2011, pp 1 – 8, Paris, France.

[10] T. Bäck, *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*, Oxford University Press, Oxford, UK, 1996.

[11] A. Abraham,H. Ishibuchi C. Grosan, *Hybrid Evolutionary Algorithms,* Springer-Verlag Berlin Heidelberg, Berlin, Germany, 2007.

[12] C. García-Martínezb M. Lozanoa, "Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report," *Computers and Operations Research*, vol. 37, no. 3, pp. 481-497, Mar. 2010.

[13] D. Simon, "Biogeography-Based Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702-713, Dec 2008.

[14] B. Basturk D. Karaboga, "Powerful And Efficient Algorithm For Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459-471, 2007.

[15] B. Basturk D. Karaboga, "On the performance of Artificial Bee Colony (ABC) Algorithm," *Applied Soft Computing*, vol. 8, no. 1, pp. 687–697, 2008.

[16] Ashrafinia, S.; Naeem, M.; Lee, D.; "A low complexity evolutionary algorithm for multi-user MIMO detection", *Computational Intelligence in Multicriteria Decision-Making (MDCM), 2011 IEEE Symposium on,* 11-15 April 2011, pp. 8 – 13, Paris, France.

[17] D. Knuth; *The Art of Computer Programming, Volume 3: Sorting and Searching,* Third Edition. Addison–Wesley, 1997.

Part II: Applications of Evolutionary Algorithms to Wireless Communication Problems

5. Computationally Efficient Symbol Detection Using EAs in Multi-User STBC-MIMO Systems

5.1. Introduction

The Multi Input Multi Output (MIMO) communication system has a significantly higher channel capacity than the Single-Input-Single-Output (SISO) system for the same total transmission power and bandwidth [1] [2]. The system considered in this chapter is assumed to comprise one receiving station and multiple transmitting devices. The receiver's front end has multiple antennas, and each transmitting device has multiple transmit antennas. It is known that the use of Space Time Block Code (STBC) can increase the capacity of MIMO systems and thus improve data throughput and spectral efficiency [3]. Multi-antenna systems are widely used because of their ability to dramatically increase the channel capacity in fading channels [4]. Each transmit device uses a STBC, the receiver side performs the joint signal detection. In this thesis, such a system is referred to as a Multi-Device (MD) STBC-MIMO system. In a MD-STBC-MIMO system, the number of receive antennas is typically smaller than the cumulative number of transmit antennas used by all transmitting devices in the system. An example of MD-STBC-MIMO, with a smaller number of antennas at the base station or access point, would be the uplink multiple access communication in cellular systems.

This chapter addresses the symbol detection problem in MD-STBC-MIMO systems. The Maximum A-Posteriori probability (MAP) detection, which reduces to the Maximum Likelihood (ML) detection in the case of a priori equally likely symbol blocks, minimizes the probability of detection error. The ML detector returns optimal results, and is further explained in section 5.3. However, a computationally efficient algorithm for achieving MAP or ML detection is not known. Some studies with Sphere-Decoding (SD) algorithms exhibit that their expected computational complexity grows polynomially with the problem size $\mathcal{M}$ up to some value of $\mathcal{M}$ for the cases of small constellation sizes [5], but it grows exponentially for the cases of large constellation sizes. In addition, for some SD algorithms, operating at a low SNR requires inordinately high computation; yet operation at a high SNR is efficient. In fact, reference [6] shows that even the expected computational complexity of the SD grows exponentially with the problem size in MIMO

communication systems. In any case, an algorithm with polynomial growth of expected complexity for all values of the problem size $\mathcal{M}$ has not yet been found.

Due to unavailability of a computationally efficient algorithm for finding the codeword that maximizes the MAP, a number of heuristic algorithms have been suggested, such as BBO, ABC and Hybrid, as well as some mainstream EAs such as GA and EDA. In this chapter, we consider applying Evolutionary Algorithms (EAs) to MD-STBC-MIMO codeword detection.

Many EAs are inspired by biological evolution and mutation. We have applied some EAs to the problem with much less computational complexity than the ML method. For this purpose, we choose the Biogeography-Based Optimization (BBO), Artificial Bee Colony (ABC), and a new developed hybrid ABC / BBO algorithm. This optimization technique has some features in common with other bio-inspired optimization methods, like Genetic Algorithm (GA) [9], and we have included the performance results of GA and EDA for comparison. The MD-STBC-MIMO detection problem is a discrete optimization problem and thus requires a discretized version of the ABC algorithm. Our simulation results show that BBO, ABC and the hybrid algorithm can meet the best known detector (i.e., SD) with less complexity, and have better performance than other methods such as Minimum Mean Square Error (MMSE), Zero Forcing (ZF), Semi-Definite Relaxation (SDR) [11], as well as EDA and GA.

In the rest of this chapter, the system model is presented in Section 5.2. The application of existing symbol detection algorithms is discussed in Section 5.3. In Section 5.4 the idea of applying EAs on the symbol detection problem is presented. Section 5.5 compares the computational complexities, and the simulation results are presented in section 5.6. Section 0 includes the conclusion and the future work.

5.2. System Model

Figure 5.1 shows an MD-STBC-MIMO system [21]. The system consists of K mobile devices transmitting signals and one receiver. Each mobile device has N_T transmitting antennas that apply STBC, whereas the receiver front end has N_R receive antennas. The multiple mobile devices in the proposed systems can cause co-channel

interference. An IQ-modulation scheme (e.g. $\mathcal{M}$-PSK, $\mathcal{M}$-QAM, etc.) maps source information into complex numbers. Even if each transmit device employs an orthogonal space-time code, the absence of coding across different wireless devices cannot guarantee the orthogonality among their signals. In the case of a single mobile device $K = 1$, the wireless device transmits using N_T transmit antennas, and communicates with a receiver that has N_R antennas. The number of time slots in the space-time code block is denoted by T.

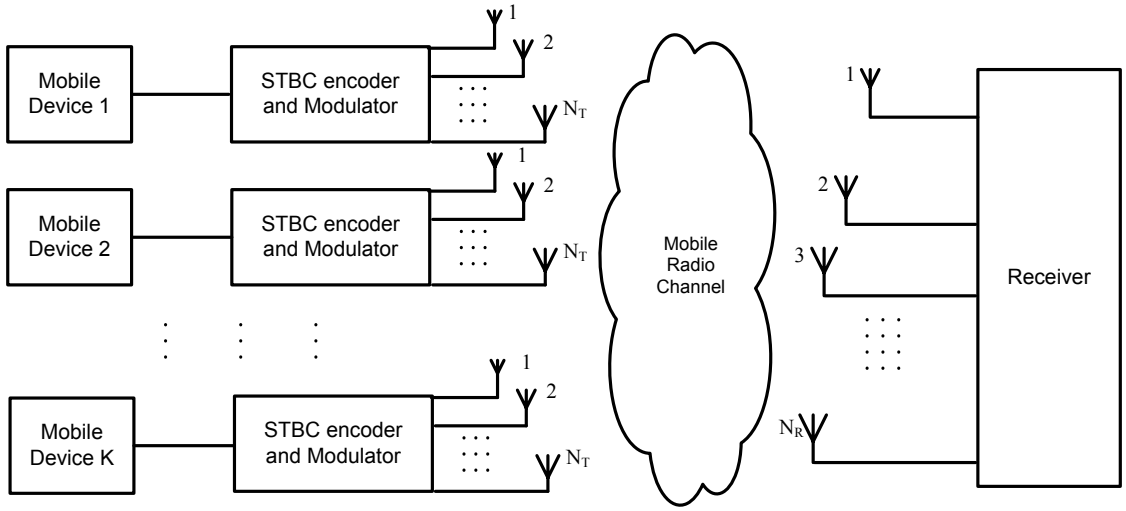


Figure 5.1. A block diagram of MD-STBC-MIMO system

The channel is assumed to be quasi-static; i.e., the channel gain remains constant during each time block of data. It is also assumed that the channel gain at each time block is known to the receiver. This assumption is often used in literature and reasonable if training or pilot signals are used. A complex $N_T \times N_R$ dimensional matrix $\boldsymbol{H}$ represents the MIMO channel and another complex $T \times N_T$ dimensional matrix $\boldsymbol{S}$ represents the input signal in a space-time code block. The relationship between the input and output signal can be expressed as:

$$\tilde{Y} = \boldsymbol{SH} + \tilde{Z} \tag{5.1}$$

where $\tilde{Y}$ is the $T \times N_R$ dimensional complex output matrix, and $\tilde{z}$ represents the additive white noise matrix.

Equation (5.1) describes the relation between the input (transmitted signals) and the output (received signals) in terms of a complex-valued matrix equation. The relation between the input and the output of the channel in a system with linear dispersion space-time coding can be equivalently expressed in terms of a real-valued matrix equation. We now briefly discuss that real-valued matrix equation. The input signal in Equation (5.1) in the case of the linear dispersion code [12] is denoted by a complex-valued matrix $\boldsymbol{S}$ that takes the form:

$$\boldsymbol{S} = \sum_{q=1}^{Q} \left[(\alpha_q + j\beta_q) C_q + (\alpha_q - j\beta_q) D_q \right] \tag{5.2}$$

Here Q indicates the number of symbols conveyed in a space time code block, and $\alpha_q + j\beta_q$ $(q = 1, \ldots, Q)$ is the complex number that represents the q^{th} symbol, where α_q and β_q correspond to the real and imaginary parts of the symbol, respectively. In the IQ constellation diagram, α_q and β_q are discrete valued variables, such that $\alpha_q + j\beta_q$ corresponds to a symbol in the constellation diagram. In 4-QAM for example, each of these two variables can take values of ± 1, and thus $\alpha + j\beta$ determines one of the four possible symbols arranged in the square grid of $(1, i), (-1, i), (-1, -i)$ and $(1, -i)$. These Q symbols $\alpha_q + j\beta_q$ can be represented as a $2Q$-dimensional real-valued row vector χ, whose components are constituted by $\alpha_q, \beta_q, q = 1, \ldots, Q$. The real and imaginary parts of matrix $\tilde{\Upsilon}$'s components can be arranged as a $2TN_R$-dimensional real-valued row vector y. The relation between χ and y in this new alternative form can be expressed as:

$$y = \chi \, \Omega + Z \tag{5.3}$$

where $2Q \times 2TN_R$ real-valued matrix Ω is derived from the component of the matrices $\boldsymbol{H}$, C_q, D_q, q=1,…,Q, and Z is the $2TN_R$-dimensional real-valued vector representing noise.

In the case of multiple mobile devices, equation (5.1) is naturally generalized to

$$\tilde{Y} = \sum_{k=1}^{K} \boldsymbol{S}_k \boldsymbol{H}_k + \tilde{Z} \tag{5.4}$$

where the $T \times N_T$-dimensional complex matrix, $\boldsymbol{S}_k$, is the input signal from mobile device k, and the $N_T \times N_R$-dimensional complex matrix $\boldsymbol{H}_k$ represents the channel from the k^{th} device to the receiver. Correspondingly, (5.3) is naturally generalized to

$$y = [\chi_1 \quad \chi_2 \quad \cdots \quad \chi_K] \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \vdots \\ \Omega_K \end{bmatrix} + Z \tag{5.5}$$

where χ_k is a $2Q_k$-dimensional real-valued row vector that represents the Q_k complex symbols sent from mobile device k in a space time code block. Note that (5.5) can model the case in which different mobile devices use different code rates Q_k /T and different space time codes. We denote by $N_S = \sum_{k=1}^{K} Q_k$ the total number of symbols (from all mobile devices) transmitted in a space-time coded block through all of their transmit antennas.

5.3. Signal Detection

The ML detection is known to yield the lowest symbol error probability in the case of a-priori equally likely symbols. In the case of our problem, the detector at the receiver has to choose from $\mathcal{M}^{N_S}$ possible sequences of symbols transmitted in a space-time code block, where $\mathcal{M}$ is the size of the symbol constellation associated with the modulation scheme. ML detection chooses transmitted symbols $[\chi_1, \chi_2, \dots, \chi_K]$ that maximize $P(y|\chi_1, \chi_2, \dots, \chi_K)$. In the case of additive white Gaussian noise Z, the ML detection is reduced to choosing the vector $[\chi_1, \chi_2, \dots, \chi_K]$ from $\mathcal{M}^{N_S}$ possibilities that has the shortest Euclidean distance $\hat{l}$ that is expressed as:

$$\hat{l} = \left\| y - \sum_{k=1}^{K} \chi_k \Omega_k \right\|. \tag{5.6}$$

The ML detection scheme can be implemented by searching through all $\mathcal{M}^{N_S} = 2^{bN_S}$ possible symbol sequences, where $b = \log_2 \mathcal{M}$, and $\mathcal{M}$ is the size of the symbol constellation. Performing such an exhaustive search to find the minimum of (5.6) is computationally inefficient, especially for large N_S. Computational complexity increases exponentially with $N_S = \sum_{k=1}^{K} Q_k$. High-speed communication requirements demand a low-complexity detection scheme. For low-complexity near-optimal detection, in this chapter the ABC algorithm is applied to this MD-STBC-MIMO detection problem. Section 5.4 describes how EAs are applied to the signal detection. In subsequent sections, we compare the performance of the BBO-based algorithms with other low-complexity suboptimal algorithms such as MMSE, SDR and SD.

5.4. Evolutionary Algorithms for solving MD-STBC-MIMO problem

In this section, we present a MD-STBC-MIMO detector that utilizes EAs presented in the first part of this thesis. The aim of applying discrete EAs to the MD-STBC-MIMO symbol detection problem is to minimize the Euclidian distance defined in (5.6). Therefore, the Euclidian distance in Equation (5.6) represents the fitness function; and shorter Euclidian distance means better fitness. An EA individual corresponds to a possible solution to the joint symbol detection problem; i.e., a set of conveyed symbols from the K transmit devices.

In the MD-STBC-MIMO system discussed in this chapter, transmitted symbols are chosen from an IQ-modulation such as $\mathcal{M}$-QAM or $\mathcal{M}$-PSK constellation diagram. In order to implement EAs, we represent each of the$\mathcal{M}$ possible points in the constellation by a unique integer in the set $\{0, \dots, \mathcal{M} - 1\}$. The system comprises K transmit devices, each device indexed by *k* transmitting Q_k $\mathcal{M}$-QAM symbols in a space-time code block. Therefore, an EA individual $\mathbb{x}$ can be defined as a $N_S = \sum_{k=1}^{K} Q_k$-dimensional ($D = N_S$)

integer row vector $\mathbb{x} \triangleq [\mathbb{x}_1, \mathbb{x}_2, \dots, \mathbb{x}_{N_S}]$ where $\mathbb{x}_v \in \{0, \dots, \mathcal{M}-1\}, v \in \{1, \dots, N_S\}$. The integer vector x represents the vector $[\chi_1, \chi_2, \dots, \chi_K]$ in equation (5.6) and the fitness function is translated accordingly.

The integer vector $\mathbb{x}$ requires implementation of discrete versions of EAs to the MD-STBC-MIMO problem. The BBO algorithm's implementation for problems in both discrete and continuous domain is almost the same, and presented in Table 2.1. The applied migration scheme is the linear immigration-constant emigration model presented in 2.3.2. We used a linear decreasing λ (immigration rate) curve with a maximum of I and a constant μ (emigration rate) in order to reduce complexity. This constant emigration rate reduces the complex process of selecting habitats by assigning a constant (uniform) probability to each habitat to be chosen for sharing its SIVs.

However, the problem requires the implementation of the discrete ABC algorithm presented in section 3.4. The DABC algorithm's pseudo code is given in **Error! Reference source not found.**, and the special version proposed for MD-STBC-MIMO employs operators (3.5) and (3.12). Similarly, the hybrid algorithm uses the same discrete approach. A complexity comparison between the algorithms applied to the MD-STBC-MIMO is discussed in the subsequent section.

5.5. Computational Complexity

A motivation for applying the proposed near-optimal algorithms to a MD-STBC-MIMO problem is their low computational complexity. In this section, the computational complexities of BBO, ABC and the hybrid algorithm proposed for MD-STBC-MIMO symbol detection are compared with that of MMSE, SD, SDR, GA and the exhaustive search. The computational complexity of exhaustive search (an implementation of the ML detector) is $O(\mathcal{M}^{N_S})$ or $O(2^n)$, $n \triangleq N_S \log_2 \mathcal{M}$; so exhaustive search is usually impractical for real-time operations of symbol detection. A number of suboptimal detection schemes with better computational complexity have been presented in literature.

The worst-case complexity of SD is exponential, and its expected complexity depends on problem size and SNR [15]. SD has high complexity of $O(\tilde{n}^6)$ [16] at low SNR, where $\tilde{n} \triangleq n \log_2 \mathcal{M}$. However, it has polynomial complexity, often roughly cubic complexity, at high SNR [15]. MMSE is one of the sub-optimal detectors that involve inverting a matrix, and its computational complexity is $O(\tilde{n}^3)$ [17]. The computational complexity of SDR [19] in each iteration is $O(N_T^{3.5})$ where N_T stands for the number of transmit antennas.

Typically, the computational complexity of population-based algorithms is analyzed in terms of the number of fitness function evaluations, which in our problem would be (5.6). One important reason is that their complexity is highly dependent on their implementation and coding efficiency. The number of function evaluations in three algorithms BBO, GA and EDA are the same and equal to GN; where G and N represent the total number of generations, and the population size, respectively [9, 18].

In the original ABC algorithm presented in [8], there is more than one fitness function evaluation phase for each individual during one generation. In the ABC, during the employed bees phase, each employed bee tests a neighbor food source for its quality, thus the fitness function evaluation has to be run once for the whole N food sources in this phase. By the same token, during the onlooker bees phase, there are N fitness function evaluations for every food source. As a result, the overall number of fitness function evaluations for these two phases in one algorithm generation is $2N$. In the scout bees phase, ABC randomly selects one food source (individual) that hasn't improved after t trials for elimination. Then the algorithm replaces its associated employed bee with a scout that randomly selects a new food source location and keeps its nectar quality in her memory. The $trial$ values of all of the solutions in the population increase twice during an algorithm generation: once during the employed bees phase and once during the onlooker bees phase, unless a solution's quality improves, or one turned into a scout. Moreover, an employed bee turns into a scout after t trials. Therefore, the first individual to exceed the t trials would be at the $t/2^{\text{th}}$ generation. After the $t/2^{\text{th}}$ generation in the worst case scenario, every generation sends one scout that runs the function evaluation procedure. As a result, the total number of fitness function evaluations for the original ABC algorithm would be:

$$2G.N + \left(G - \frac{t}{2}\right). \quad (5.7)$$

This complexity is of course higher than the complexity of other aforementioned EAs.

The complexity of our proposed discrete ABC (**Error! Reference source not ound.**) is yet less than (5.7) because this algorithm does not run the function evaluation procedure for all the individuals in the employed bees and onlooker bees phases. Due to the stochastic nature of the algorithm, there wouldn't be a closed form expression for the number of food source locations that do not change with equation (10). Moreover, the discrete ABC algorithm doesn't have the limit of randomly selecting one food source to abandon in the scout phase. Yet it abandons all solutions whose $trial > t$, and replaces their associated bee with a scout. The number of these replacements is unknown and random due to the heuristic nature of the algorithm. Therefore, we only present an average number of fitness function evaluation for each run of discrete ABC from our simulations in section 5.7. This number clearly demonstrates that this enhancement to the original ABC has a great effect on reducing the complexity of the algorithm, such that its total number of fitness function evaluations would become less than other EAs.

The complexity of the hybrid algorithm follows the same approach as the DABC. This algorithm takes the advantage of the aforementioned enhancements of DABC, and there would not be a closed form solution for its complexity. However, since the hybrid algorithm has more effective procedures than DABC, it is predictable that it has more number of fitness function evaluations than DABC with equal G and N. The reason is that less number of solutions remains unchanged during the employed and onlooker phases, which results in more number of fitness evaluation. This prediction is further confirmed in section 5.6 when we present the comparison result between the average numbers of fitness function evaluations of different EAs.

5.6. Simulation Results

In this section we present the simulation results of the proposed EA-based detection applied to a MD-STBC-MIMO system, and its comparison with other detection

techniques. The system model used in our simulations is depicted in Figure 5.1. The channels are assumed to be quasi-static, and different channels in MD-STBC-MIMO assumed to be independent. In all our simulations, it is assumed that the mobile data is transmitted in a form of 4-QAM modulation from all wireless devices $(\mathcal{M} = 4)$. For simulation experiments we assumed that each of the K devices transmit the same number of symbols $Q_k = Q$. Therefore, there are $N_S = KQ$ symbols conveyed from the K transmit devices to the receiver. Each point in the plots of Figures 5.4 − 5.10 is a value averaged over multiple independent runs. In each trial, the set of transmitted signals ($[\chi_1 \quad \chi_2 \quad \dots \quad \quad]$ in Eqn. (5.5)), channel matrices ($[\Omega_1 \quad \Omega_2 \quad \cdots \quad \quad]^T$ in Eqn. (5.5)), and noise (z in Eqn. (5.5)) are generated randomly and independently of other trials. Therefore, in each simulation trial received signal *y* in Eqn. (5.6) set from those randomly generated variable in accordance with Eqn. (5.5). Then, the algorithms are run to seek the value of $[\chi_1, \chi_2, \dots, \chi_K]$ that minimizes $\hat{l}$. Therefore, the averaged results over different simulation trials are in fact averaged over the different channel and noise realizations, and also different realizations of the algorithm evolution in the case of probabilistic algorithms such as GA, EDA, DABC, hybrid and BBO. This experimental setup enables us to compare different algorithms in terms of the performance averaged over different channel and noise realizations.

In order to present a fair comparison between EAs, all of them have the same number of generations and population size. They also share the same initial population matrix, and all these setting is kept constant during all simulations. The BBO parameters set for implementation are: *I* = 1 and *m* = 0.1. The DABC algorithm's trial parameter is set to $1.35 \times$ the number of algorithm iterations. The hybrid algorithm has the same parameters as BBO and ABC for its related BBO and ABC procedures, respectively. Detailed system configuration and algorithm parameters are given in a table next to each figure.

5.6.1. BER Performance Comparison

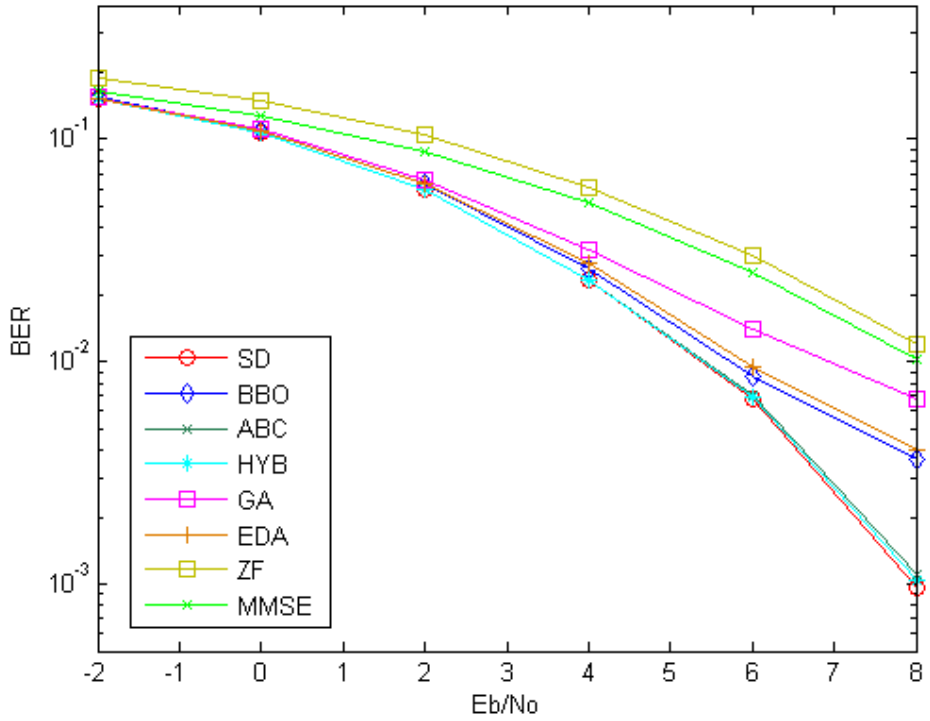
The simulation results in Figures 5-2 through 5-6 show the BER performance comparison between MMSE, SDR[1], SD[2], GA, BBO, ABC and Hybrid BBO/ABC detectors. The MD-STBC-MIMO system configuration, (K,N_T,N_R, $\mathcal{M}$,T), is set (4,2,6,4,2), (5,2,8,4,2), (6,2,10,4,2), (7,2,14,4,2) and (3,4,4,4,2) for Figures Figure 5.2, Figure 5.3, Figure 5.4, Figure 5.5, and Figure 5.6, respectively. The Alamouti space-time coding [7] is used in Figures Figure 5.2, Figure 5.3, Figure 5.4, and Figure 5.5, but for Figure 5-6 a non-orthogonal four transmit antennas configuration is used for each mobile device. EA shared parameters, (G, N), which denote the number of iterations and the population size (the number of island), are set to (60, 60), (100,100), (100,120), (120,200), and (120,200) for Figure 5.2, Figure 5.3, Figure 5.4, Figure 5.5, and Figure 5.6, respectively. For these figures, the total numbers N_S of symbols transmitted from all users are set 8, 10, 12, 12, and 14, which indicates that search spaces of 4^8, 4^{10}, 4^{12}, 4^{14} and 4^{12} possible solutions, respectively.

For each simulation, we tried to pick a pair of (G,N) not only to make the EAs' results close to SD's, but also to choose the smallest possible pair of G *and* N in order to reduce the computational complexity of the algorithm. Moreover, in order to compare the results more precisely, G and N are identical for all EAs. From these figures we observe that the best algorithm, that almost always return the same result as the SD is the hybrid algorithm, and after that the ABC decoder returns the result with over 97% of the SD's. The third place is for BBO, following by EDA and GA. Comparing the BER performance results of various detectors, it can be observed that EAs outperform other sub-optimal detection methods in all the five figures, and can meet the optimal result by searching through a much smaller set of individuals by selecting a feasible pair of (G,N).

[1] For Semi-definite Relaxation (SDR) simulation we have used the software provided by Dr. Zhi-Quan Luo [13]

[2] For Sphere Decoder, we have implemented the algorithm mentioned in [15].

a.



b.

System												
K	N_T	N_R	$\mathcal{M}$	T	Search space	STBC type	Channel Type			No. of Simulation runs		
4	2	6	4	2	4^8	Alamouti	Quasi-static fading			2000		
Common EAs			**BBO**			**ABC**	**GA**			**EDA**		
Generation	Pop	I	m	Migration		trial	Pxover	Pmut	Psel	Pxover	Pmut	Psel
60	60	1	0.015	Constant		0.4× *pop*	0.9	0.5	0.5	0.99	0.95	0.5

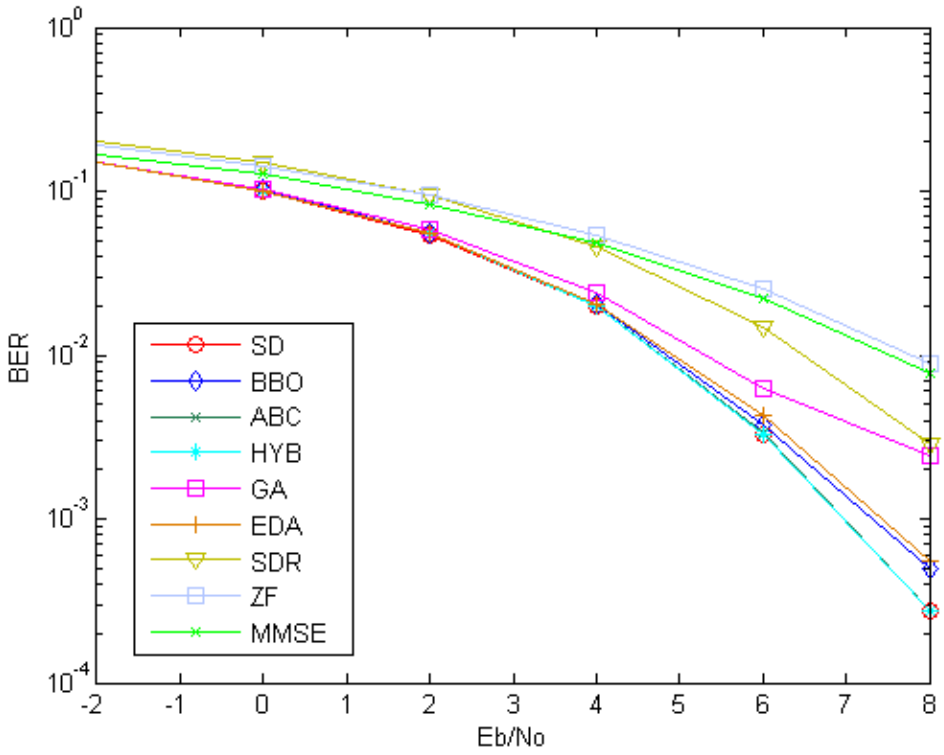
c.

	2	0	-2	-4	-6	-8
ZF	89	86	80	74	70	64
MMSE	96	92	86	78	74	66
SDR	84	84	80	78	78	77
GA	100	99	96	91	85	72
EDA	100	100	98	95	93	80
BBO	100	100	98	97	96	81
ABC	100	100	100	100	99	98
Hybrid	100	100	100	100	100	99

Figure 5.2. Performance comparison for K = 4

a. BER performance comparison for (K,N_T,N_R, $\mathcal{M}$) = (4,2,6,4),
b. simulation parameters,
c. decoders' percentage of the SD results

a.



a.

b.

System											
K	N_T	N_R	*M*	*T*	Search space	STBC type	Channel Type	No. of Simulation runs			
5	2	8	4	2	4^{10}	Alamouti	Quasi-static fading	2000			
Common EAs		**BBO**			**ABC**	**GA**			**EDA**		
Generation	Pop	*I*	*m*	Migration	trial	*P*xover	*P*mut	*P*sel	*P*xover	*P*mut	*P*sel
100	100	1	0.015	Constant	0.4× *pop*	0.9	0.5	0.5	0.99	0.95	0.5

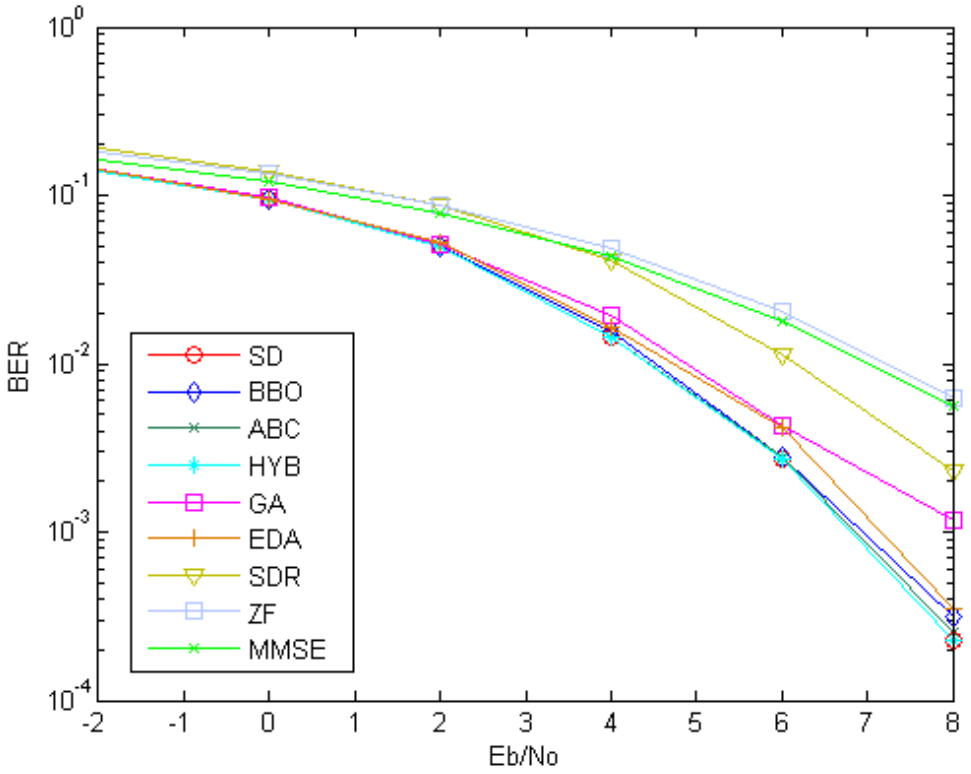
c.

	2	0	-2	-4	-6	-8
ZF	88	85	81	75	64	58
MMSE	95	90	85	78	67	59
SDR	85	82	81	79	74	71
GA	100	99	98	95	89	73
EDA	100	100	100	99	96	92
BBO	100	99	100	100	98	93
ABC	100	100	100	100	99	100
Hybrid	100	100	100	100	100	100

Figure 5.3. Performance comparison for K = 5

a. BER performance comparison for $(K,N_T,N_R,\mathcal{M}) = (5,2,8,4)$,
b. simulation parameters,
c. decoders' percentage of the SD results

a.



b.

System												
K	N_T	N_R	$\mathcal{M}$	T	Search space	STBC type	Channel Type			No. of Simulation runs		
6	2	10	4	2	4^{12}	Alamouti	Quasi-static fading			2000		
Common EAs			**BBO**			**ABC**	**GA**			**EDA**		
Generation	Pop		I	m	Migration	trial	Pxover	Pmut	Psel	Pxover	Pmut	Psel
100	120		1	0.015	Constant	0.4× *pop*	0.9	0.5	0.5	0.99	0.95	0.5

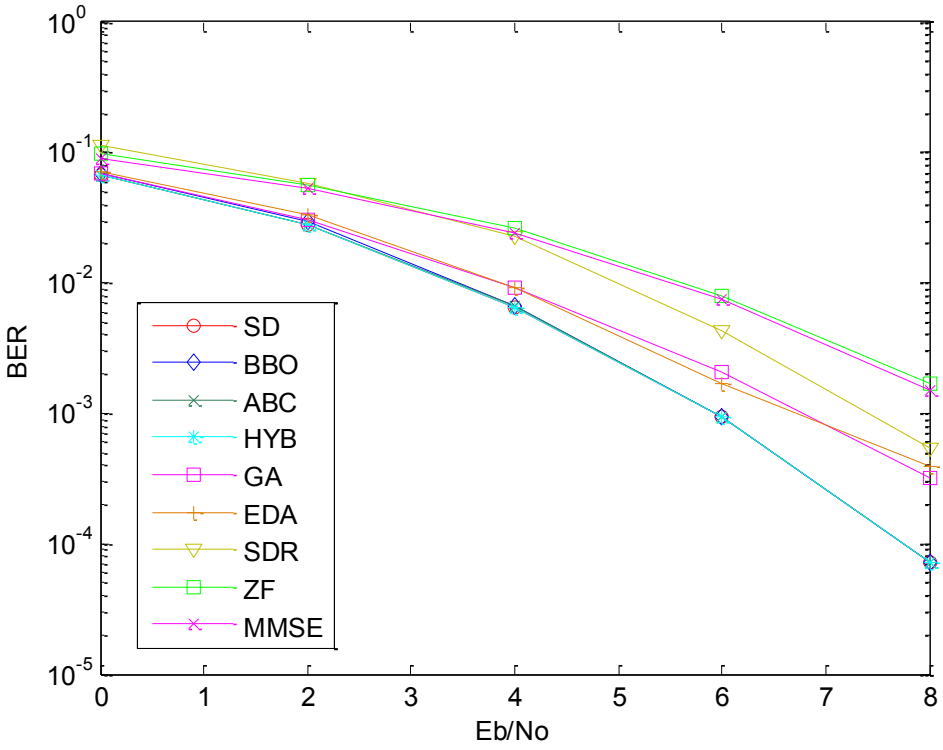
c.

	-2	**0**	**2**	**4**	**6**	**8**
ZF	87	85	81	72	66	61
MMSE	93	90	85	74	68	62
SDR	84	84	81	75	76	72
GA	99	99	98	93	92	81
EDA	100	100	98	97	93	95
BBO	100	100	100	98	99	96
ABC	100	100	100	100	100	99
Hybrid	100	100	100	100	100	100

Figure 5.4. Performance comparison for K=6

a. BER performance comparison for $(K,N_T,N_R,\mathcal{M}) = (6,2,10,4)$,
b. simulation parameters,
c. decoders' percentage of the SD results

a.



b.

System								
K	N_T	N_R	$\mathcal{M}$ T	Search space	STBC type	Channel Type	No. of Simulation runs	
7	2	14	4 2	4^{14}	Alamouti	Quasi-static fading	1000	

Common EAs		BBO			ABC	GA			EDA		
Generation	Pop	I	m	Migration	trial	Pxover	Pmut	Psel	Pxover	Pmut	Psel
120	150	1	0.015	Constant	0.4× *pop*	0.9	0.5	0.5	0.99	0.95	0.5

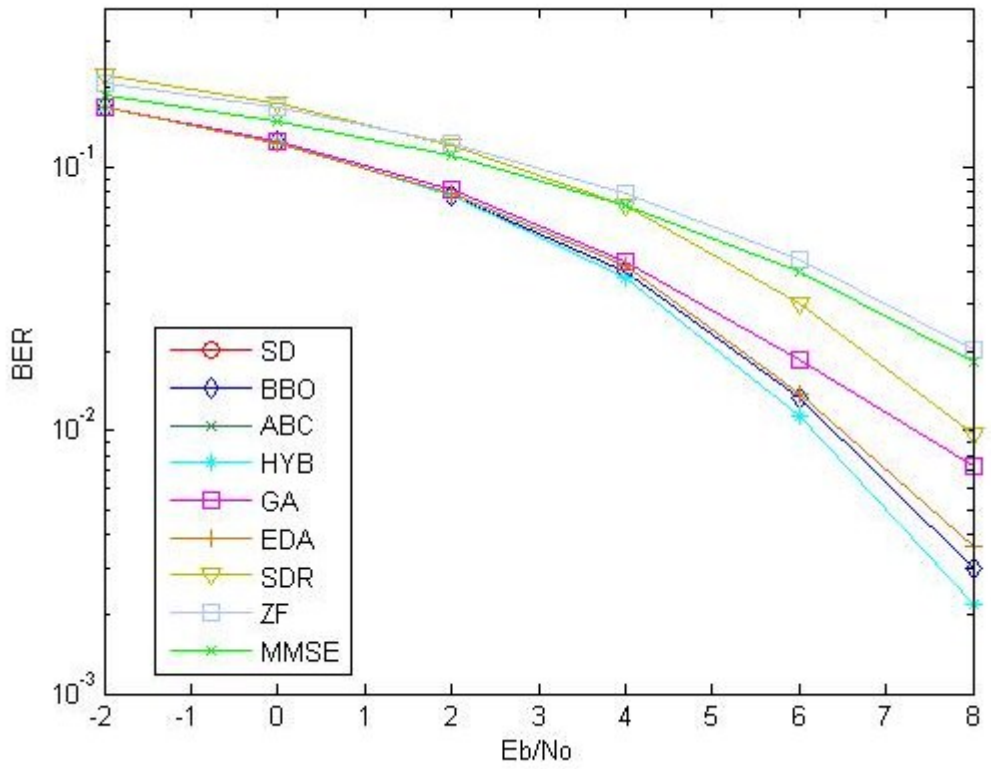
c.

	0	2	4	6	8
ZF	68	49	22	14	2
MMSE	74	54	24	15	2
SDR	60	48	26	25	7
GA	36	91	84	86	100
EDA	36	94	71	76	14
BBO	33	100	64	94	100
ABC	100	100	100	100	100
Hybrid	100	100	100	100	100

Figure 5.5. Performance comparison for K = 7

a. BER performance comparison for $(K,N_T,N_R,\mathcal{M})$ = (7,2,14,4),
b. simulation parameters,
c. decoders' percentage of the SD results

a.



b.

System								
K	N_T	N_R	$\mathcal{M}$	T	Search space	STBC type	Channel Type	No. of Simulation runs
3	4	4	4	2	4^{12}	Alamouti	Quasi-static fading	2000

Common EAs		BBO			ABC	GA			EDA		
Generation	Pop	I	m	Migration	trial	Pxover	Pmut	Psel	Pxover	Pmut	Psel
120	200	1	0.015	Constant	0.4× *pop*	0.9	0.5	0.5	0.99	0.95	0.5

c.

	-2	0	2	4	6	8
ZF	88	85	81	75	64	58
MMSE	95	90	85	78	67	59
SDR	85	82	81	79	74	71
GA	100	99	98	95	89	73
EDA	100	100	100	99	96	92
BBO	100	99	100	100	98	93
ABC	100	100	100	100	99	100
Hybrid	100	100	100	100	100	100

Figure 5.6. Performance comparison For K = 3

a. BER performance comparison for (K,N_T,N_R, $\mathcal{M}$) = (3,4,4,4),
b. simulation parameters,
c. decoders' percentage of the SD results

The experiment for Figure 5-6 was performed on a non-orthogonal space-time code, whereas the experiments for other figures were performed on the Alamouti code (simple and orthogonal). The total number N_S of symbols transmitted from all users in a space time code block is 12, and 4-QAM is used, so the size of the search space is 4^{12}. The population size is 150 and the number of generations (iterations) is 120, so BBO, GA and EDA are exploring only 18,000 points in the search space, which is a reasonably small portion of the search space. Similar to other figures, SD and BBO has the best BER performance. In higher SNRs, GA's performance diminishes notably, while BBO pursues the near-optimal SD. GA requires 1.8 dB less SNR than SD and BBO to achieve BER of 10^{-2}.

From the computational complexity point of view in EAs, finding the optimal pair of (G, N) is essential in order to minimize the processing power and the required memory. According to the computational complexity order of these algorithms, with a fixed population size (N), more iteration until termination means more computation. Figure 5.7 and Figure 5.8 show number of iterations required by each detection scheme to achieve a desirable BER. The MIMO system configurations are (K,N_T,N_R,M) = (6,2,10,4) and (5,2,8,4) for Figure 5.7 and Figure 5.8, respectively, using the Alamouti STBC and quasi-static channel, and the SNR is fixed to 8 dB. Figure 5.7 shows that the hybrid algorithm with the population size fixed to 100 is the first algorithm achieves the sphere decoding performance in less than 50 iterations. After the hybrid algorithm, the ABC decoder reaches SD in iteration 64. Other EAs whether cannot reach the SD results, or require much more iterations to reach meet the SD results. In Figure 5-8 we observe that the hybrid algorithm reaches the SD in about 32 iterations, while ABC cannot reach earlier than its 80^{th} iteration. This improved performance is consistently observed in several other simulations with different system configurations. As a result, not only the hybrid algorithm outperforms other sub-optimal algorithms, it delivers better results than other well-known EAs such as GA and EDA, its predecessors BBO and ABC, and can reach SD.

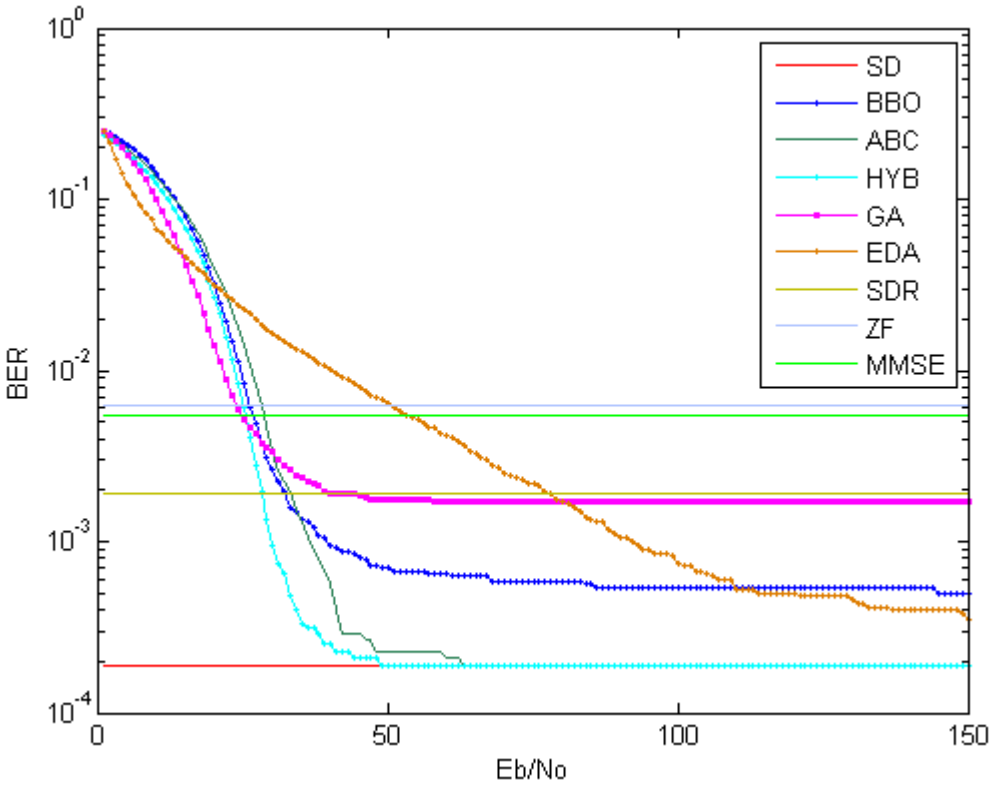
Furthermore, increasing the number of population N per iteration up to some point tends to hasten finding an acceptable solution; i.e., decreases the number of iterations, G, until termination. Figures Figure 5.9 to Figure 5.12 show the trade-off

between the population size and the iterations required to achieve a desired BER in GA, BBO, ABC and the hybrid algorithm. The MIMO system configuration is (K,N_T,N_R,M,) = (4,2,4,4), using the Alamouti STBC and quasi-static channel, and the SNR is 8 dB. The detailed system configuration is given in Table 5.1.This trade-off is useful from the system design point of view. If a hardware system has high processing capabilities and low memory, then we can set the population size low to get same BER performance and vice versa. (Higher N and G needs more memory.)

Table 5.1. System parameters for iteration – population size trade-off

System												
K	N_T	N_R	$\mathcal{M}$	T	Search space	STBC type	Channel Type			No. of Simulation runs		
5	2	8	4	2	4^{10}	Alamouti	Quasi-static fading			2000		
Common EAs			**BBO**			**ABC**	**GA**			**EDA**		
Generation		Pop	I	m	Migration	trial	Pxover	Pmut	Psel	Pxover	Pmut	Psel
1 ~ 120		100	1	0.015	Constant	0.4× *pop*	0.9	0.5	0.5	0.99	0.95	0.5

a.



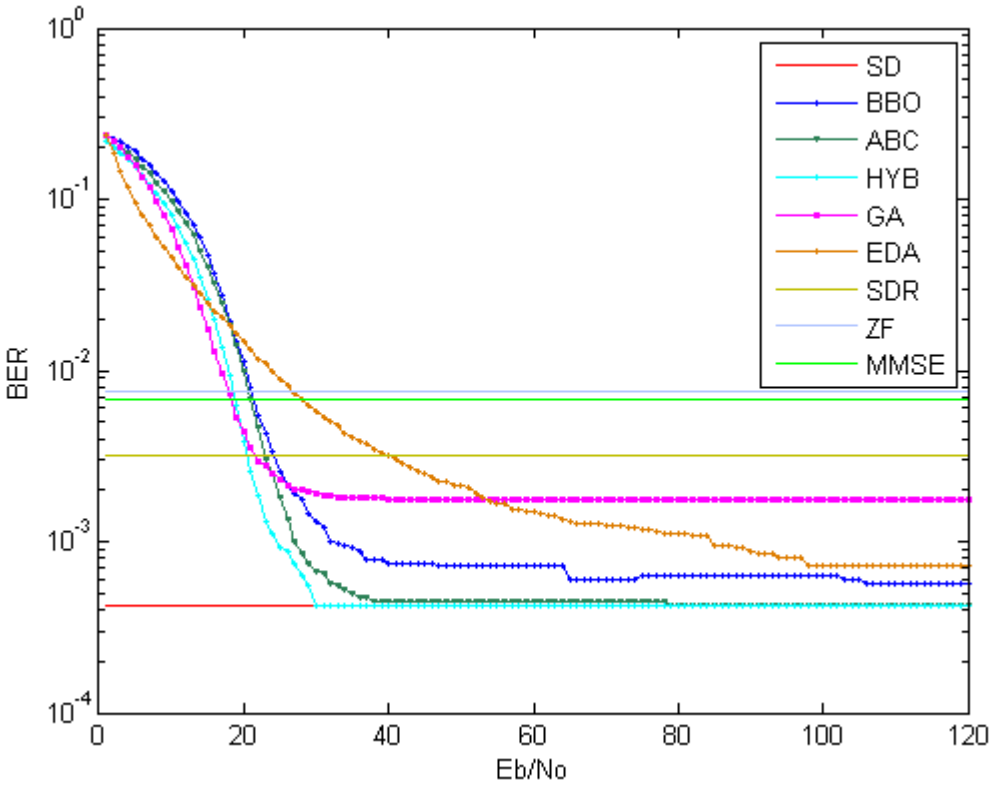
b.

System												
K	N_T	N_R	$\mathcal{M}$	T	Search space	STBC type	Channel Type	No. of Simulation runs				
6	2	10	4	2	4^{12}	Alamouti	Quasi-static fading	2000				
Common EAs		**BBO**			**ABC**	**GA**			**EDA**			
Generation	Pop	I	m	Migration	trial	Pxover	Pmut	Psel	Pxover	Pmut	Psel	
1 ~ 150	100	1	0.015	Constant	0.4× pop	0.9	0.5	0.5	0.99	0.95	0.5	

Figure 5.7. BER vs. algorithm iteration comparison

a. BER performance comparison vs. iterations for (K,N_T,N_R, $\mathcal{M}$) = (6,2,10,4),
b. simulation parameters

a.



b.

b.

System												
K	N_T	N_R	$\mathcal{M}$	T	Search space	STBC type	Channel Type	No. of Simulation runs				
5	2	8	4	2	4^{10}	Alamouti	Quasi-static fading	2000				
Common EAs		**BBO**			**ABC**	**GA**			**EDA**			
Generation	Pop	I	m	Migration	trial	Pxover	Pmut	Psel	Pxover	Pmut	Psel	
1 ~ 120	100	1	0.015	Constant	0.4× pop	0.9	0.5	0.5	0.99	0.95	0.5	

Figure 5.8. BER vs. algorithm iteration comparison

a. BER performance comparison vs. iterations for (K,N_T,N_R, $\mathcal{M}$) = (5,2,8,4),
b. simulation parameters

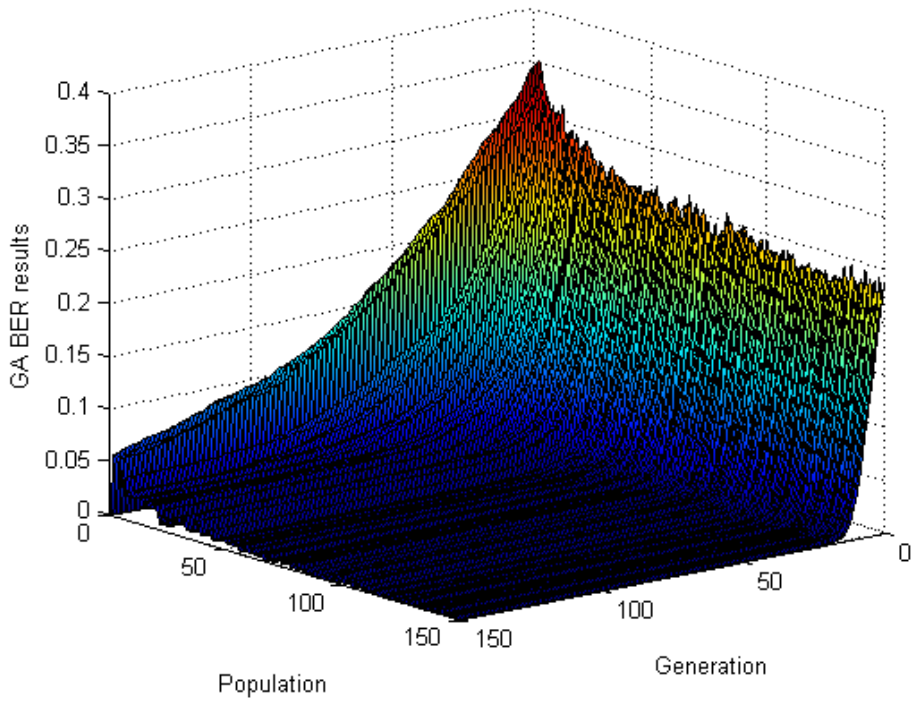


Figure 5.9. Population size and iterations trade-off for GA with K = 4

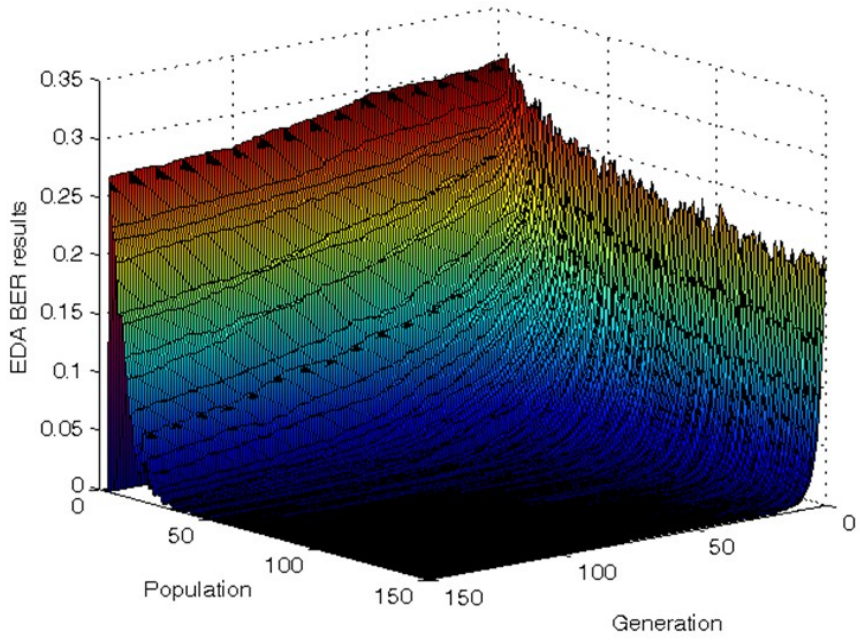


Figure 5.10. Population size and iterations trade-off for EDA with K = 4

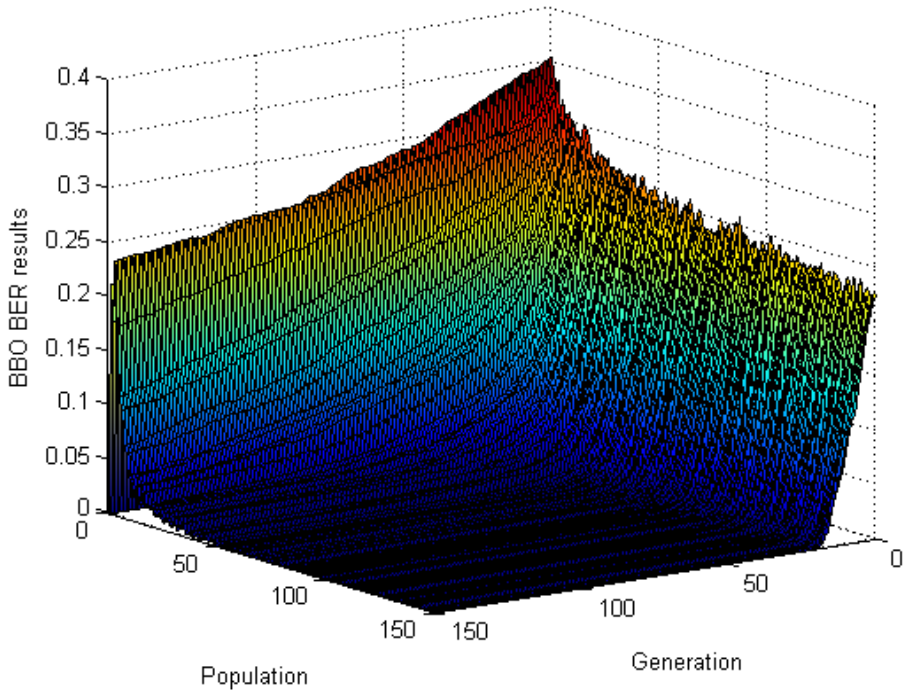


Figure 5.11. Population size and iterations trade-off for BBO with K = 4.

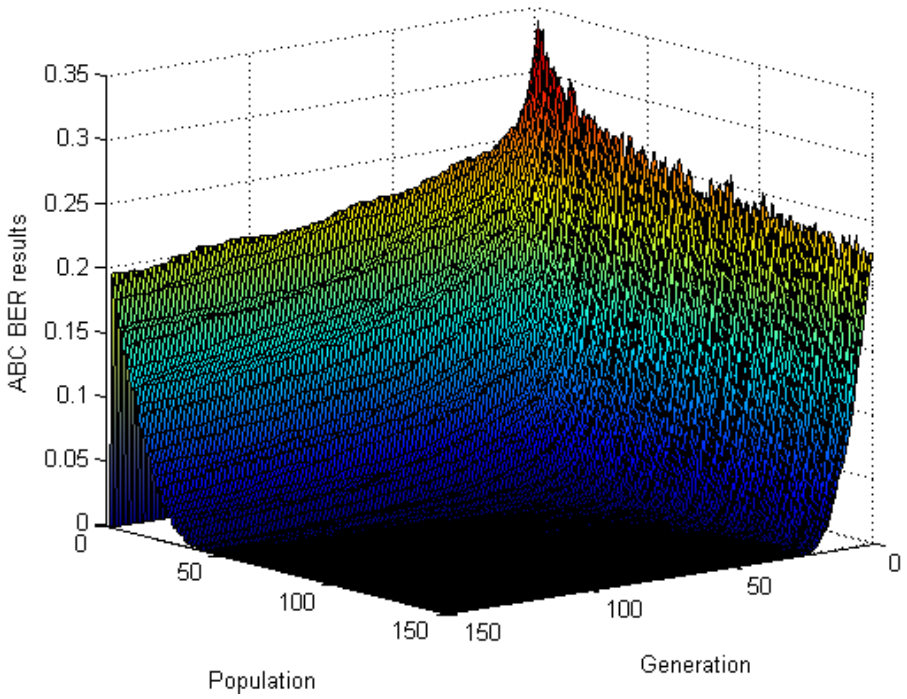


Figure 5.12. Population size and iterations trade-off for ABC with K = 4.

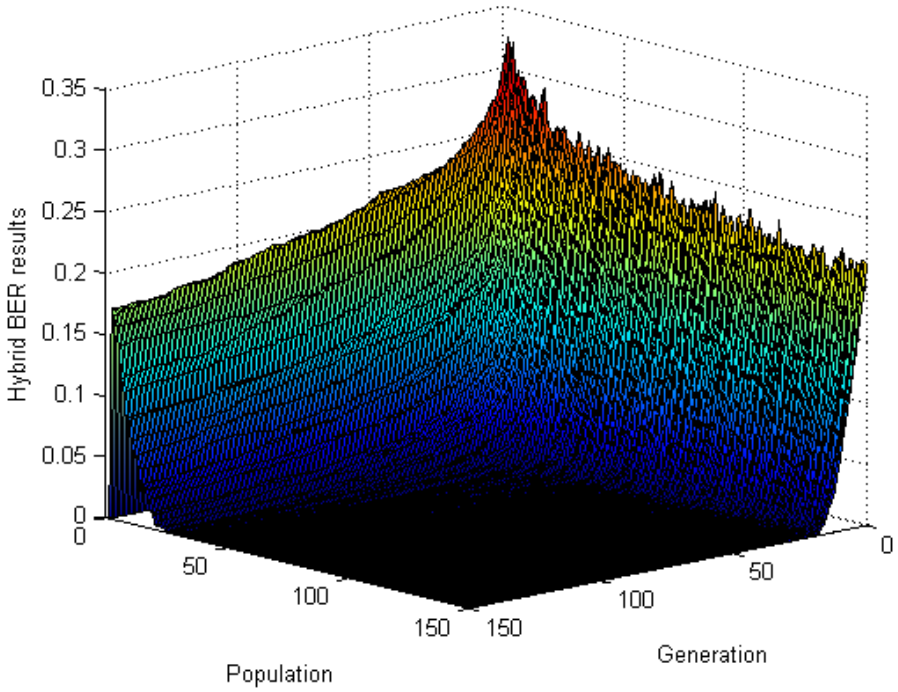


Figure 5.13. Population size and iterations trade-off for Hybrid with K = 4.

From the above four 3D plots, the fastest decreasing algorithm is the hybrid algorithm. GA is also a rapid algorithm to reach the minimum, but it doesn't reach the optimal value, and according to Figure 5.2 till Figure 5.6 at the best it can reach 80% of the SD results. From the last three plots we conclude that the hybrid algorithm reaches its minimum value faster than its predecessors ABC and BBO both in the directions of iterations and population size.

5.6.2. Complexity Comparison

There are two considerable issues while dealing with optimization algorithms and especially Evolutionary Algorithms. First issue is the algorithms' performance comparison in terms of the elapsed time during each simulation runs. Second issue which is a concern about the EAs is the number of fitness function evaluations. The most complex procedure of an EA is where it runs the fitness function evaluation procedure, which has to be run at least once in a generation (sometimes more than once such as in ABC and the hybrid). The other procedures of an EA are usually simple additions,

multiplications, if conditions, taking minimum or maximum, which are not usually as much complex as the fitness function.

In order to compare the complexities of the algorithms mentioned earlier in Section 5.5 more practically, we run simulations for four different system configurations, to compare their complexity in terms of computation time. Table 5.2 compares SD , ZF, MMSE and SDR with other EAs in terms of the average elapsed time. The system configurations are the same as those in Figure 5.2 to Figure 5.5. For the purpose of these simulations, we employed Matlab® R2010b, running on PCs with Intel® Quad-core 2.83 GHz CPUs and 3 GB of RAM. The results show that all EAs need much less time than SD at very low SNRs. The reason for this difference is that SD has a high computational complexity especially at very low SNRs, which also depends on the search space. Therefore the case of -2 dB for five, six and seven users SD spends the highest time during each simulation run among all other detectors in the same and above SNRs. However, the time elapsed in every EA simulation run is almost constant in different SNRs; because their complexities does not depend on SNR. These results demonstrate that some EAs, especially hybrid and DABC are the best choices in the low and mid SNRs. As the number of transmit devices increases, SD detector's execution time grows exponentially, and make it impracticable particularly for low to mid SNRs.

Table 5.3 contains the average number of fitness function evaluations for one simulation run of each EA. The system parameters are the same as Figure 5.2, Figure 5.3, and Figure 5.4. We observe that the number of fitness evaluations is identical for GA, EDA and BBO and is equal to GN, as discussed earlier in section 5.5. The original ABC algorithm requires $2GN + (G - t/2)$ fitness function evaluations, which is equal to 7,220, 20,033 and 24,019 for four, five and six transmit devices, respectively. These numbers are more than twice the number of fitness evaluations of BBO, GA and EDA. However, after our enhancement to the DABC algorithm, we observe that these numbers have been reduced and become closer to GN. We further observe the outstanding results of the hybrid algorithm: despite of its performance results equal to SDs', its number of fitness function evalutions is mostly less than other EAs. In conclusion, the hybrid algorithm would be a significantly considerable choice for joint symbol detection in MD-STBC-MIMO systems.

Table 5.2. Comparison between detectors' execution time (in seconds).

No. of Devices	4			5		
SNR	**-2**	**2**	**6**	**-2**	**2**	**6**
ZF	0.0015	0.0013	0.0013	0.0015	0.0013	0.0013
MMSE	0.0012	0.0011	0.0011	0.0012	0.0011	0.0011
SDR	0.0046	0.0040	0.0033	0.0071	0.0060	0.0050
GA	0.2685	0.2684	0.3682	0.4820	0.4814	0.4819
EDA	0.4170	0.4168	0.4167	0.7361	0.7363	0.7361
BBO	0.0603	0.0601	0.0598	0.1036	0.1035	0.1035
ABC	0.2500	0.2530	0.2547	0.4176	0.4232	0.4252
Hybrid	0.3432	0.3442	0.3441	0.5636	0.5652	0.5649
SD	0.4462	0.1062	0.0223	3.8743	0.5213	0.0563

No. of Devices	6			7		
SNR	**-2**	**2**	**6**	**-2**	**2**	**6**
ZF	0.0081	0.0024	0.0014	0.0109	0.0016	0.0016
MMSE	0.0015	0.0015	0.0012	0.0039	0.0013	0.0013
SDR	0.0111	0.0092	0.0069	0.0226	0.0108	0.0082
GA	0.5186	0.5214	0.5180	0.8181	0.8116	0.8116
EDA	0.7762	0.7802	0.7756	1.1836	1.1824	1.1814
BBO	0.1083	0.1083	0.1077	0.1690	0.1648	0.1650
ABC	0.4204	0.4292	0.4270	0.6125	0.6180	0.6194
Hybrid	0.5790	0.5849	0.5813	0.8627	0.8591	0.8617
SD	32.0401	2.3918	0.1875	356.9387	8.5781	0.2243

Table 5.3. The Average Number of EAs' Fitness Function Evaluations.

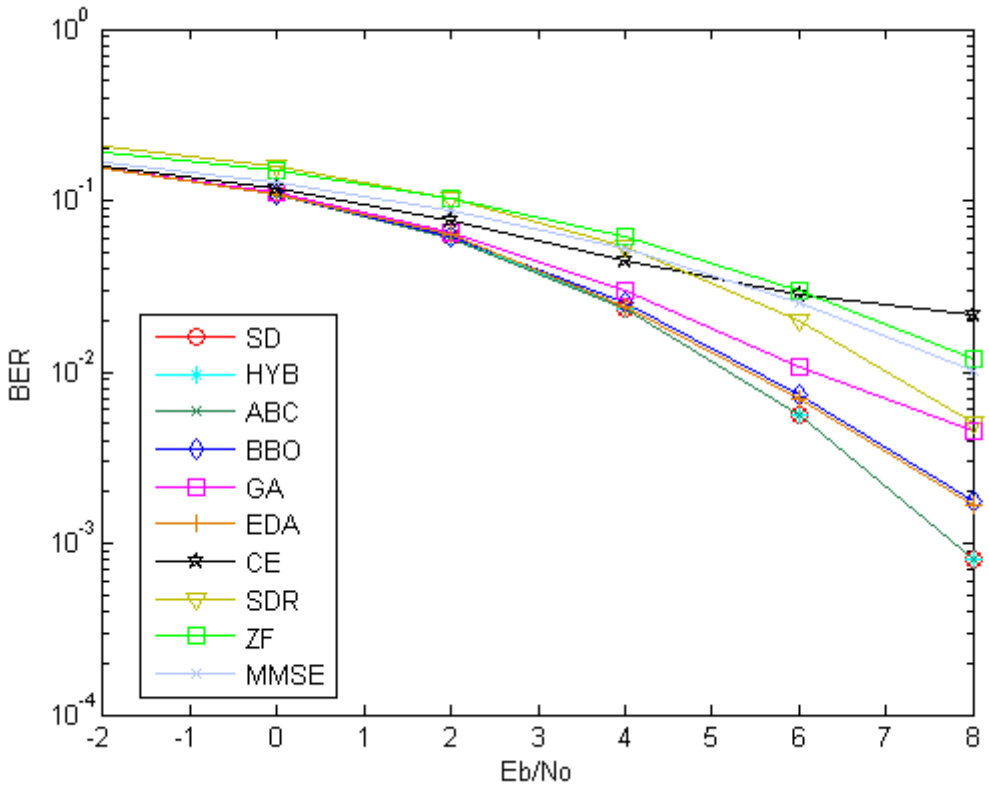
No. of Transmit Devices	4	5	6	7
GA	3,600	10,000	12,000	24,000
EDA	3,600	10,000	12,000	24,000
BBO	3,600	10,000	12,000	24,000
ABC	5,071	13,852	16,063	31,151
Hybrid	3,857	9,985	11,866	21,473

5.6.3. Related Work

The author has published the results of this chapter in [19, 20]. Naeem has applied another EA to a similar system model presented in [21]. The EA he uses is Central Entropy (CE) [22]. The simulation results of this paper show CE has a close performance to SD. Note that as mentioned in Section 5.5, computational complexity is a major concern. In the EAs discussed in this the thesis and in CE [21], computational complexity, by the definition of number of fitness function evaluations, is GN. In the simulation results of this chapter, small numbers for G and N have been selected to demonstrate hybrid algorithm approaches SD with less number of GN thus less number of fitness evaluations. Moreover, the simulation results in figures 5-7 until 5-13 show that other EAs may have a closer approach to the SD if the number of iterations G or population size N is increased. This can be intuitively explained as the more number of G and N, the higher chance of the algorithm to find the global optima. The results in [21] include the simulations for $K = 4$ and $K = 5$. However, for $K = 4$ and other simulation problems the same, the result of Figure 4 in [21] is obtained with $N = 80$ where in Figure 5.2 N is set to 60. Also in Figure 3 of [21], with all other parameters the same, $N = 200$, while in Figure 5.3 N is set to half of that amount. Therefore, a conclusion from the above points is that not only for CE, but for other EAs presented in this chapter, if the number of population size or the number of iterations is increased, most of them have the potential to have a very close results as SD. But the computational complexity is a major concern and the main goal is to choose the EA that meet SD results with less number of fitness function evaluations. A simulation has been run to compare the results of CE with the existing algorithms in this paper and is presented in Figure 5.14. This

simulation has been run for three times, where each run was an average BER of 2000 independent trials, and all these curves were quite identical. Note that as the number of population size N is 100 and is half of the number in the results presented in [21]. The mediocre result for CE is the effect of decreasing the algorithm parameters G and N (thus the number of fitness function evaluation), while keeping other simulation parameters the same.

a.



b.

System												
K	N_T	N_R	*M*	*T*	Search space	STBC type	Channel Type	No. of Simulation runs				
5	2	8	4	2	4^{10}	Alamouti	Quasi-static fading	2000				
Common EAs			**BBO**			**ABC**	**GA**			**EDA**		
Generation	Pop		*I*	*m*	Migration	trial	*P*xover	*P*mut	*P*sel	*P*xover	*P*mut	*P*sel
100	100		1	0.015	Constant	0.4× *pop*	0.9	0.5	0.5	0.99	0.95	0.5

Figure 5.14. Performance Comparison with CE for K = 5

a. BER performance comparison for $(K,N_T,N_R,M) = (5,2,8,4)$,
b. simulation parameters

Naeem has published another paper with the same system model in [22], where he applies EDA to the optimization problem. Similar to the previous discussion, his results are obtained with higher number of population size N; while even Figure 5.7 and Figure 5.8 show that EDA can have a closer result to SD with higher N or G. In fact, the simulation results in this chapter demonstrate a fairly good performance for EDA that is about 90% of the SD results. Yet, the hybrid algorithm is the one that returns the best results with the lowest fitness function evaluations. Another similar work is published in [23] where authors applied EDA and BBO. Similar to the above discussion, it is clear that the solver presented in this chapter outperform the BER performance of EDA and BBO.

5.7. Conclusion

In this chapter, we proposed three EAs discussed earlier in Part I of this thesis for Multi-Device (MD) Space-Time Block Coded (STBC) Multi Input Multi Output (MIMO) Communication System. The complexity of these algorithms is low as compared with optimal ML detector, so they are suitable for high-speed real-time communications. In addition, compared to the Sphere Decoding, other Evolutionary Algorithms like GA and EDA, and decoding schemes such as MMSE, ZF and SDR, these EA detectors show significantly better performance in MD-STBC-MIMO. The proposed algorithms also have consistently better performance-complexity trade-off at low SNRs, in comparison to existing algorithms. Even at high SNRs, these algorithms have relatively good performance-complexity trade-off.

Among the proposed algorithms, the BBO decoder requires the least time to return the results, and the hybrid algorithm usually returns the same results as the SD, and it returns the results through the least number of fitness function evaluations. Therefore, we conclude that the proposed EAs, particularly the hybrid algorithm, are suitable for high-speed real-time communications.

The hybrid algorithm is a potential solution to be applied to the same type of computationally complex problems in wireless communication because of its simplistic model, low implementation complexity, and convergence to a nearly optimal solution with a small number of iterations.

References

[1] G. J. Foschini and M. J. Gans, “On limits of wireless communications in a fading environment when using multiple antennas,” *Wireless Personal Communications,* vol. 6, pp. 311-335, 1998.

[2] E. Telatar, “Capacity of Multi-antenna Gaussian Channels,” *European Trans. on Telecomm.* vol. 10, pp. 569-709, Nov. 1999.

[3] V. Tarokh, N. Seshadri, and A. R. Calderbank, “Space-time codes for high data rate wireless communications: performance criterion and code construction,” *IEEE Trans. Information Theory*, vol. 44, pp. 744-765, Mar. 1998.

[4] H. Vikalo and B. Hassibi, “On the sphere decoding algorithm: Part II, generalizations, second-order statistics and applications to communications,” *IEEE Trans. on Signal Processing,* vol 53, no. 8, pp. 2819-2834, Aug 2005.

[5] J. Jalden and B. Ottersten, “On the complexity of sphere decoding in digital Communications,” *IEEE Trans. on Signal Processing,* vol. 53, no 4, pp. 1474-14844, April 2005.

[6] S.Verdú, “Minimum Probability of Error for Asynchronous Gaussian Multiple Access Channels,” *IEEE Trans. Information Theory*, vol. 32, pp. 85-96, Jan. 1986.

[7] S. M. Alamouti, “A Simple Transmit Diversity Technique for Wireless Communications,” *IEEE journal on select areas in communications,* vol. 16, no. 8, pp. 1451-1458, Oct. 1998.

[8] D. Karaboga, B. Basturk, “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm”, *Journal of Global Optimization*, vol.39, no. 3, pp. 459-471, 2007.

[9] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning,* Addison-Wesley, 1989.

[10] J. Proakis, *Digital Communication*, McGraw-Hill, 5th Ed. 2007.

[11] M. Nekuii, M. Kisialiou, T.N. Davidson, and Z. Q. Luo, “Efficient Soft Demodulation of MIMO QPSK via Semidefinite Relaxation," *Proceedings of 2007 IEEE International Conference on Acoustics, Speech, and Signal Processing,* pp. 2665-2668, April 2008.

[12] B. Hassibi and B.M. Hochwald, “High-rate codes that are linear in space and time,” *IEEE Trans. Information Theory,* vol.48, no.7, pp. 1804−1824, Jul. 2002.

[13] http://www.ece.umn.edu/~luozq/software/sw_about.html; last accessed Fall 2012.

[14] M. Kisialiou and Z.-Q. Luo, “Performance analysis of quasi-maximum-likelihood detector based on semi-definite programming,” *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 3, pp. 433-436, 2005.

[15] B. Hassibi, H. Vikalo, “On the sphere decoding algorithm: Part I, the expected complexity*,” IEEE Trans. Signal Processing,* vol.53, no.8, pp. 2806-2818, Aug 2005.

[16] O. Damen, A. Chkeif, and J. C. Belfiore, “Lattice code decoder for space-time codes*,” IEEE Communications Lett.,* vol. 4, no. 5, pp. 161–163, May 2000.

[17] C. Comaniciu, N.B. Mandayam, and H.V. Poor, “Wireless Networks: Multiuser Detection in Cross-Layer Design,” *Springer*, NY, May 2005.

[18] Ashrafinia, S.; Pareek, U.; Naeem, M.; Lee, D.; "Biogeography-based optimization for joint relay assignment and power allocation in cognitive radio systems," *Swarm Intelligence (SIS), 2011 IEEE Symposium on*, pp.1-8, 11-15 April 2011.

[19] S. Ashrafinia; M. Naeem;D. Lee; “A low complexity evolutionary algorithm for multi-user MIMO detection”, *Computational Intelligence in Multicriteria Decision-Making (MDCM), 2011 IEEE Symposium on,* 11-15 April 2011, pp. 8 – 13, Paris, France.

[20] S. Ashrafinia; M. Naeem; D. Lee;“Discrete Artificial Bee Colony for Computationally Efficient Symbol Detection in Multi-Device STBC MIMO Systems”, accepted in *the Journal of Advances in Artificial Intelligence*, Nov. 2012.

[21] M. Naeem; D.C. Lee; "A joint symbol detection algorithm efficient at low SNR for a Multi-Device STBC-MIMO system," *Radio and Wireless Symposium (RWS), 2010 IEEE* , pp.440-443, 10-14 Jan. 2010.

[22] M. Naeem; D.C. Lee; "Efficient symbol detection in Multi-Device STBC-MIMO System," *Communications and Information Technology, 2009. ISCIT 2009. 9th International Symposium on*, pp.578-583, 28-30 Sept. 2009.

[23] D. C. Lee and M. Naeem, “Computationally Efficient Symbol Detection Schemes in Multi-Device STBC-MIMO Systems,” in *MIMO Systems, Theory and Applications*, Vienna, Austria: INTECH, 2011.

6. EAs for Joint Relay Assignment and Power Allocation in Cognitive Radio Systems

In this chapter, an EA-based (BBO, ABC and hybrid ABC/BBO algorithms) low-complexity interference aware relay assignment scheme with power control is presented for a relay-assisted cognitive radio network comprising one source node, multiple relays and multiple destination nodes. Optimally relay assignment using the Exhaustive Search Algorithm (ESA) has a high computational complexity, which grows exponentially with the number of relays and users. The joint relay assignment formulation is presented with source and relays' power allocation as a mixed integer non-linear programming problem, which is further reduced to a simpler integer programming problem. The EA-based relay assignment scheme with discrete power control at source and relays is presented for the integer programming problem with the three algorithms, and compared with other EAs such as EDA and BACO. The superiority of the hybrid algorithm over other EAs is confirmed through computational experiments, and we present these results.

6.1. Introduction

Official reports show that spectrum lies fallow in certain areas, at certain times, and on certain frequencies [1]. For example, a licensee may have exclusive use of spectrum in a particular geographic area, but choose not to make use of the spectrum over the entire area. Likewise, a licensee may fully utilize spectrum during times of peak usage, but utilize only half of its spectrum during off-peak hours. The spectrum utilization efficiency can be improved by allowing secondary (unlicensed) users to access the band unused or partially occupied by the primary (licensed) users (PUs), under the condition that the secondary users' signals do not exceed the interference threshold at the PU [2]. Relay assisted cognitive radio networks of the secondary users [3-5] take the advantage of cooperative communication [6], which reduces the source to the destination transmission power, and consequently shrinks the interference at the primary users.

The cognitive radio system presented in this chapter comprises a single source node, multiple destination nodes and multiple relays. We employ the spectrum underlay [3] as a technique that allows the secondary users to share the whole licensed spectrum with the PUs. Relays use amplify and forward (AF) relaying [7], in which the relay amplifies the received signal from the source and simply forwards it to the destination. We further assume that these relays can only transmit on discrete power levels. This assumption simplifies the control channel traffic from source to destination, and eliminates employing sophisticated circuit to support communication at arbitrary power levels [8].

Researchers have shown interest to cooperative communications for wireless networks, due to its ability to mitigate fading in wireless communication through achieving spatial diversity [20]. However, using multiple relays rather than a single relay raises the problem of how to assign each relay to receivers. In a system with multiple destination nodes, one has to consider the issue of optimal assignment of relays to different destinations. Running on the underlay mode is followed by constraints on the relay transmission power, due to the interference constraints that need power control at relays. In this chapter, our main objective for both problems is the optimal assignment of relays to the secondary users, in a cognitive radio network with discrete power levels working under AF mode. Therefore the sum capacity of the system is maximized under the constraint that the interferences on the primary users should be below their specified threshold.

The issue of optimal relay assignment has been proposed for cellular networks and some work has been done on relay assignment and power allocation schemes [9]. Nonetheless, these schemes cannot be applied to cognitive radio systems as they may violate the interference constraints at the PUs. Some works have also been done on the optimal relay assignment in ad-hoc networks comprising multiple source-destination pairs [10, 11]. A relay assignment schemes for cognitive radio networks with single source node, single multiple destinations and multiple relays is proposed in [12]; yet the power to source and relays are not assigned optimally.

We formulate the optimization problem of optimal relay assignment and power control as a non-linear mixed integer programming with these variables: the source

transmission power, relays' transmission power levels and the assignment of relays to receivers. We observe that the optimization of source and relays' power is separable which enables us to present a closed-form expression for the optimum source power. This further reduces our initial formulation into an integer programming problem.

We can use the Exhaustive Search Algorithm (ESA) to obtain the optimal solution, due to the combinatorial nature of the problem. Although ESA provides the optimal solution to the problem, it has a high computational complexity. Therefore, we take the advantage of heuristic algorithms, particularly the EAs, and their ability to solve optimization problems efficiently and come to an optimal solution as rapid as possible – i.e. with relatively-low computational complexity.

In the rest of this chapter, we present the cognitive radio system model and formulation in section 6.2, followed by the EAs' implementations in 6.3. The simulation results are presented in 6.4, and section 6.5 contains the conclusion.

6.2. System Model

This section presents the system model of a cognitive radio with a single source node, multiple relays and multiple destinations in 6.2. The formulation of the joint source and relay power allocation is presented as a mixed integer non-linear programming problem, and further reduced to an integer programming problem. The EAs' implementation is discussed in 6.3, and the simulation results are presented in Section 6.4.

Our model of the relay assisted cognitive radio network comprises one source (transmitting) node, K destination (receiving) nodes or secondary users, and L relay nodes. There are M primary users (PUs) in this system, and these M primary users can be interpreted as M geographic locations, where the strengths of the cognitive radio signals must be limited. Figure 6.1 shows a block diagram of a multi destination cooperative cognitive radio network.

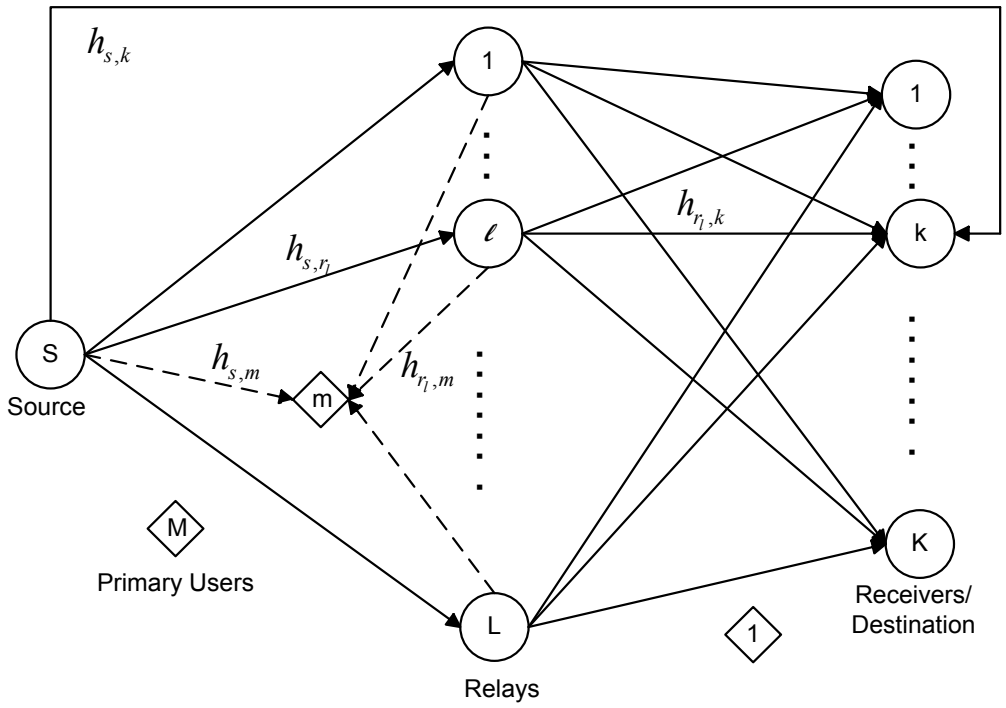


Figure 6.1. Relay Assisted Cognitive Radio Network

Each transmitter, receiver and relay has a single antenna. We denote by $h_{s,k}$ the complex-valued channel gain from the source to the k^{th} receiver, $h_{s,l}$ the channel from the source to the l^{th} relay, and $h_{l,k}$ the channel from the l^{th} relay to the k^{th} receiver, as depicted in Figure 1. Also we denote by $h_{s,m}$ the channel gain from the source to the m^{th} primary user and $h_{l,m}$ the channel gain from the l^{th} relay to the l^{th} primary user. We assume that the transmitter of the source and the receivers of the destinations have knowledge of their incident channel states (channel gains). It is also assumed that the complex-valued channel gains $h_{s,l}$ and $h_{l,k}$ are known to the l^{th} relay, and all the relays are perfectly synchronized, as assumed in [8]. In our system we assume that each relay can transmit to the PUs at a finite number of transmission power levels between 0 and p_l^{max}, where p_l^{max} is the maximum power which the l^{th} relay is allowed to transmit; and

the l^{th} relay uses a fixed transmission power p_l per dimension. The interference power from the l^{th} relay to the m^{th} PU is denoted by $I_{l,m} = p_l \left| h_{l,m} \right|^2$.

In this cognitive radio system a two-step Amplify-and-Forward (AF) scheme is employed for cooperative communication [8]. Each symbol is conveyed from the source to destinations in two steps (time slots). In the first time slot, the source transmits its data carrying signals, and all relays and destination nodes are able to receive these signals. It is assumed that each source-destination pair of this cognitive radio network has been allocated equal bandwidth, and each destination node receives its data on a separate frequency band. This separation of receiver nodes' frequency band paves the way of assuming that different receiver nodes' signals do not interfere with one another. Each relay is assumed to transmits its received signal at the same frequency band it received the signal. This models a low-cost relay that simply amplifies the signal and forwards it.

As mentioned above, the interference to each PU must be constrained by a specific threshold in each user band and in each time slot. We denote by P_s^k the transmission power of the source to the k^{th} user band. The received signal at the l^{th} relay is $\sqrt{P_s^k} h_{s,l} s + Z_l$; where s is normalized complex-valued transmitted symbol – i.e. $E(|s|^2) = 1$, and Z_l represent the complex white Gaussian noise with the power spectral density of $N_0/2$. The power of this white Gaussian noise is expressed as $N = \frac{N_0}{2} 2W = N_0 W$ in each user band [13], where W denotes the bandwidth of each user band. In the second time slot, relays transmit the amplified received signal. In our system model, the relay or relays assigned to the k^{th} user filter in the signal received in the band indexed by k, amplify, and then transmit it to the k^{th} user [8]. In the system being studied in this chapter, a receiver can receive data from multiple relays, while each relay can only transmit to one receiver. We define $\varepsilon_{l,k}$ as a binary assignment indicator with the following definition:

$$\varepsilon_{l,k} = \begin{cases} 1 & \textit{if } l^{th} \textit{ relay is assigned to the } k^{th} \textit{ receiver} \\ 0 & \textit{otherwise} \end{cases} \tag{6.1}$$

Now, we can express the channel capacity for the k^{th} user in shared bandwidth [14] amplify and forwarding mode [8] as:

$$C_s^k(\varepsilon_{l,k}, p_l) = \frac{1}{2}\log\left[1 + \frac{P_s^k\left|h_{s,k}\right|^2}{N} + \frac{P_s^k}{N} \times \frac{\left(\sum_{l=1}^{L}\varepsilon_{l,k}\left|h_{s,l}h_{l,k}\right|\beta_l\sqrt{p_l}\right)^2}{1+\sum_{l=1}^{L}\left(\beta_l\left|h_{l,k}\right|\sqrt{p_l}\right)^2}\right] \tag{6.2}$$

where

$$\beta_l = \frac{1}{\sqrt{P_s^k\left|h_{s,l}\right|^2 + N}}$$

The goal of the optimal relay assignment problem formulation is to maximize the sum capacity at the receivers, under the interference constraints of the PUs.

If we implement the interference constraints to the relay assignment problem, we'll have the following mixed integer non-linear programming problem:

$$OP1: \max_{P_s^k, p_l, \varepsilon_{l,k}} \sum_{k=1}^{K}\frac{1}{2}\log\left[1 + \frac{P_s^k\left|h_{s,k}\right|^2}{N} + \frac{P_s^k}{N}\left(\frac{\left(\sum_{l=1}^{L}\varepsilon_{l,k}\left|h_{s,l}h_{l,k}\right|\beta_l\sqrt{p_l}\right)^2}{1+\sum_{l=1}^{L}\left(\beta_l\left|h_{l,k}\right|\sqrt{p_l}\right)^2}\right)\right] \tag{6.3}$$

subject to

$$C1: \sum_{k=1}^{K}\varepsilon_{l,k} \le 1, \quad \forall l = 1,2,\ldots,L$$

$$C2: \sum_{l=1}^{L}\varepsilon_{l,k}p_l\left|h_{l,m}\right|^2 \le I_{m,k}^{max}, \quad \forall(m,k)$$

$$C3: P_s^k\left|h_{s,m}\right|^2 \le I_{m,k}^{max}, \quad \forall(m,k)$$

$$C4: P_s^k \le P_s^{max}, \quad \forall(k)$$

$$C5: p_l \le \sum_{k=1}^{K}\varepsilon_{l,k}p_l^{max}, \quad \forall l = 1,2,\ldots,L$$

$$C6: \varepsilon_{l,k} \in \{0,1\}, p_l \in P_L P_s^k \ge 0, \quad \forall(l,k)$$

where I_m^{max} is the maximum tolerable interference for the m^{th} primary user, P_L is a set of relay power levels $\left\{0, \frac{p_l^{max}}{\delta}, \frac{2p_l^{max}}{\delta}, \ldots, p_l^{max}\right\}$ with the cardinality of $|P_L|$ [12], and p_l is a discrete value from the set P_L representing the discrete power level.

Since the relays power levels are discrete, they can only operate on finite number of transmission power levels. The advantage of discretizing the relay power levels is twofold: first, fewer choices of transmission power – i.e. fewer bits in control messages to indicate the relay power level, and second, employing inexpensive relays. The constraint $C1$ ensures assignment of each relay to the maximum one user. The constraints $C2$ and $C3$ define the interference constraints for the source and relay transmission. The constraints $C4$ and $C5$ limit the power for the source and relays respectively. Moreover, the constraints $C1$, $C5$ and $C6$ together ensure that only the selected relay transmits under its own specified power levels. Due to the fact that the source transmission is in a different time slot than relays', $C3$ and $C4$ as source power constraints, as well as $C2$ and $C5$ as relays' power constraints are decoupled.

We can re-write *C3* as:

$$P_s^k \leq \frac{I_{m,k}^{max}}{|h_{s,m}|^2}, \quad \forall m = 1,2,\ldots,M. \tag{6.4}$$

The optimum source power satisfying all M PUs at the k^{th} band is

$$P_{s,opt}^k = \min\left\{P_s^{max}, \left(\frac{I_{1,k}^{max}}{|h_{s,1}|^2}, \frac{I_{2,k}^{max}}{|h_{s,2}|^2}, \ldots, \frac{I_{M,k}^{max}}{|h_{s,M}|^2}\right)\right\}. \tag{6.5}$$

If $P_{s,opt}$ is known, the optimization problem in (6.3) can be rewritten as the following integer programming problem:

$$OP2: \quad \max_{p_l,\varepsilon_{l,k}} \sum_{k=1}^{K} \frac{1}{2} \log \left[1 + \frac{P_{s,opt}^{k} \left| h_{s,k} \right|^2}{N} + \frac{P_{s,opt}^{k}}{N} \left(\frac{\left(\sum_{l=1}^{L} \varepsilon_{l,k} \left| h_{s,l} h_{l,k} \right| \beta_l \sqrt{p_l} \right)^2}{1 + \sum_{l=1}^{L} \left(\beta_l \left| h_{l,k} \right| \sqrt{p_l} \right)^2} \right) \right] \tag{6.6}$$

subject to

$$C1 \quad : \quad \sum_{k=1}^{K} \varepsilon_{l,k} \le 1, \quad \forall l = 1,2,\ldots,L$$

$$C2 \quad : \quad \sum_{l=1}^{L} \varepsilon_{l,k} p_l \left| h_{l,m} \right|^2 \le I_{m,k}^{max}, \quad \forall (m,k)$$

$$C3 \quad : \quad p_l \le \sum_{k=1}^{K} \varepsilon_{l,k} p_l^{max}, \quad \forall l = 1,2,\ldots,L$$

$$C4 \quad : \quad \varepsilon_{l,k} \in \{0,1\}, p_l \in P_L \quad \forall (l,k).$$

Equation (6.6) can be bounded below by adding one-to-one constraint in relay assignment to reduce its feasible set. This one-to-one constraint means that a relay can only transmit data to at most one user, and a user can be assigned to at most one relay. Thus, the modified version of (6.6) would be:

$$OP3: \quad \max_{p_l,\varepsilon_{l,k}} \sum_{k=1}^{K} \frac{1}{2} \log \left[1 + \frac{P_{s,opt}^{k} \left| h_{s,k} \right|^2}{N} + \frac{P_{s,opt}^{k}}{N} \left(\frac{\left(\sum_{l=1}^{L} \varepsilon_{l,k} \left| h_{s,l} h_{l,k} \right| \beta_l \sqrt{p_l} \right)^2}{1 + \sum_{l=1}^{L} \left(\beta_l \left| h_{l,k} \right| \sqrt{p_l} \right)^2} \right) \right] \tag{6.7}$$

subject to

$$C1 \quad : \quad \sum_{l=1}^{L} \varepsilon_{l,k} \le 1, \quad \forall k = 1,2,\ldots,L$$

$$C2 \quad : \quad \sum_{k=1}^{K} \varepsilon_{l,k} \le 1, \quad \forall l = 1,2,\ldots,K$$

$$C3 \quad : \quad \sum_{l=1}^{L} \varepsilon_{l,k} p_l \left| h_{l,m} \right|^2 \le I_{m,k}^{max}, \quad \forall (m,k)$$

$$C4 \quad : \quad p_l \le \sum_{k=1}^{K} \varepsilon_{l,k} p_l^{max}, \quad \forall l = 1,2,\ldots,L$$

$$C5 \quad : \quad \varepsilon_{l,k} \in \{0,1\}, p_l \in P_L \quad \forall (l,k).$$

The Exhaustive Search Algorithm (ESA) evaluates all $(K+1)^{L \times (|P_L|-1)}$ possible relay assignments for $OP2$ (6.6) to reach to an optimal solution, while $OP3$ needs

$\sum_{i=0}^{min\left(K,L|P_L|\right)} K_i \left(\frac{L|P_L|!}{\left(L|P_L|-i\right)!} \right)$ relay assignments. The former number is computationally inefficient because it grows exponentially with the number of relays and power levels; while the latter has a lower complexity that increases with the number of users.

6.3. Evolutionary Algorithms-Based Relay Assignment with Greedy Power Allocation

We present our contributed algorithm, the binary EA-based relay assignment with greedy power allocation. This algorithm employs the binary EAs to assign relays to receivers, by detecting suboptimal value of binary assignment indicators $\varepsilon_{l,k}$. Subsequently, based on the values of the binary assignment indicators $\varepsilon_{l,k}$, the algorithm utilizes a greedy algorithm to allocate the relays' discrete power levels. We propose a general description of BBO in the following subsection, and will continue by presenting our implementation of BBO to the relay assignment problem to determine the suboptimal solutions for the optimization problem discussed in (6.6).

The optimization problem discussed in (6.6) is a constraint optimization problem. Thus it needs to run a procedure to ensure that the interference and power allocation constraints are satisfied. We present a “Constraint Check with Power Allocation” (CCPA) procedure to perform this checking. This procedure’s task is to convert the non-feasible candidate solutions to feasible solutions that satisfy the (6.6) constraints, and greedily assign power to the relays. The pseudo code of CCPA is given in Table 6.1.

For implementing EAs and finding the matrix ε of the size $L \times K$, the ε matrix is modified to a $1 \times LK$-dimensional vector $\bar{\varepsilon}$ and expressed as: $\bar{\varepsilon} = \left[\varepsilon_{1,1}, \dots, \varepsilon_{1,K}, \varepsilon_{2,1}, \dots, \varepsilon_{2,K}, \dots, \varepsilon_{L,1}, \dots, \varepsilon_{L,K}\right]$. If any candidate solution violates the constraints *C1* and *C2* of the optimization problem presented in (6.6), CCPA intelligently corrects the violation in the lines 3-9 of Table 6.1 by placing zeros and ones in some position. The binary assignment indicator vector $\bar{\varepsilon}$ in the lines 7-8 helps relays to be assigned to users. In the next step, the procedure sets the corresponding relays’ initial power levels to the maximum possible discrete value. In the occasion of the PU interference constraint *C4* violation, the procedure selects the relay with the highest

individual sum interference in the lines 12-14. This is the same as writing $l^* = \text{argmax}\,\Gamma^l$, $l = 1,2,\dots,L$, where $\Gamma^l = \sum_m p_l \left|h_{l,m}\right|^2$, and if $p_{l^*} = p_{l^*}^{max}$, then its power level is reduced by one $p_{l^*} = (\delta - 1)p_{l^*}^{max}/\delta$. Lastly, the procedure recalculates the PU interference, and ensures the interference constraints are satisfied at every PU through the repetition of the lines 12-15.

Table 6.1. Pseudo code of the CCPA algorithm

1: Initialization: $\Gamma^l = 0, \quad \forall(l)$

2: **for** $j = 1$ to N

3: **for** $l = 1$ to L

4: **if** $\sum_k \varepsilon_{l,k}^j > 1$

5: $x = rand\_perm\ of\ \ find\left(\varepsilon_{l,k}^j = 1\right)$

6: $\varepsilon_{l,k}^j = 0, \quad \forall k, \varepsilon_{l,x(1)}^j = 1$

7: **end if**

8: $pFactor(l) = \sum_k \varepsilon_{l,k}^j \times \left(\left|P_L\right| - 1\right)$

9: $p_l^j = p_l^{max} \times pFactor(l) / \left(\left|P_L\right| - 1\right)$

10: **end for**

11: $\Delta_m^k = \sum_l p_l^j \varepsilon_{l,k}^j \left|h_{l,m}\right|^2$

12: **while** $\Delta_m^k \geq I_{max}^{m,k}, \quad \forall(m,k)$

13: $\Gamma^l = \sum_m p_l^j \left|h_{l,m}\right|^2$

14: $l^* = \underset{l=1,2,L}{\operatorname{argmax}}\ \Gamma^l$

15: $p_{l^*}^j = p_{l^*}^j \left(pFactor\left(l^*\right) - 1\right)\left(\left|P_L\right| - 1\right)$

16: $\Delta_m^k = \sum_l p_l^j \varepsilon_{l,k}^j \left|h_{l,m}\right|^2$

17: **end while**

18: **if** $p_l^j = 0$

19: $\varepsilon_{l,k}^j = 0, \quad \forall k$

20: **end if**

21: $\bar{\varepsilon} = \left[\varepsilon_{1,1}, \ldots, \varepsilon_{1,K}, \varepsilon_{2,1}, \ldots, \varepsilon_{2,K}, \ldots, \varepsilon_{L,1}, \ldots, \varepsilon_{L,K}\right]$

22: **end for**

23: return

6.4. Simulation Results

In this section we illustrate our simulation results for the proposed EA-based relay assignment with greedy power allocation problem, which provides a suboptimal solution to the constraint optimization problem discussed in (6.6). The performance comparison includes the Exhaustive Search Algorithm (ESA) that returns the exact maximal capacity in the optimization problem (6.7). We also employ other bio-inspired EAs, such as EDA and ACO, and compare them with BBO, ABC and hybrid, as well as another scheme, called *ESA one-to-one*. The ESA one-to-one scheme serves as the optimal solutions lower bound for the constraint optimization problem in (6.6), which yields to the optimal solution of the constraint optimization problem presented in (6.7). We compare these four schemes with different transmission power levels. The signal to noise ratio is fixed to $P^{max}/N = 10$ dB, where P^{max} denotes the maximum allowed transmission power from the source. Each relay can either operate on two, or four different transmission power levels -- i.e. $P_L \in \{0, p_l^{max}\}$ or $P_L \in \{0, p_l^{max}/3, 2p_l^{max}/3, p_l^{max}\}$.

6.4.1. Algorithms' Performance Results

Figure 6.2 till Figure 6.7 illustrate the implementations results of the aforementioned algorithms, obtained by averaging the individual capacity of different randomly-generated scenarios based on above values. We compared these simulation results based on different system parameters such as: L, K, M, and I_m^{max}. The channel gain between source, relays and destinations are randomly generated in accordance with the assumption of independent (i.i.d.) channel gain drawn from a complex Gaussian distribution. As a consequence, the presented results averaged over different simulation trials are in fact the average over different channel and noise realizations and also different realizations of the algorithm evolution in case of EAs.

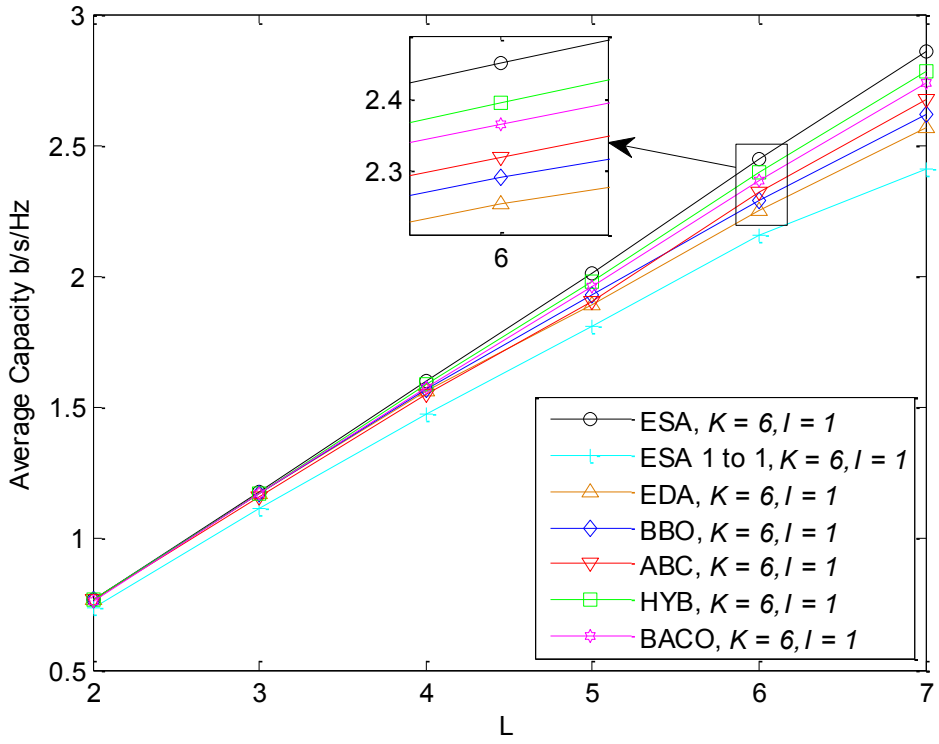
EA parameters are kept constant through all simulations, and they share the same initial population for a fair comparison. BBO uses the piece-wise constant migration explained in 2.3.3, and is based on the partial immigration-based BBO.

We present the plot for the capacity versus the number of relays in Figure 6.2 and Figure 6.3 for the set of parameters $\left(M, K, I_m^{max}, p_l^{max}, |P_L|\right) = \left(2,6,1mW, P^{max}/10,2\right)$ and $\left(4,5,1mW, P^{max}/10,4\right)$, respectively. We provide simulation results for the case of $P_L \in \left\{0, p_l^{max}\right\}$ and $P_L = \left\{0, p_l^{max}/3, 2p_l^{max}/3, p_l^{max}\right\}$. Here we observe that capacity increases with the number of relays, because more relays increase the choices of relay assignment.

Figure 6.4 and Figure 6.5 depict the capacity vs. number of secondary users, for systems $\left(L, M, I_m^{max}, p_l^{max}, |P_L|\right) = \left(6,5,100mW, P^{max}/10,2\right)$ and $\left(5,4,100mW, P^{max}/10,4\right)$, respectively. In these two figures, we observe that the capacity increases with the number of users.

Figure 6.6 and Figure 6.7 show the performance plots of the capacity versus the interference threshold I_m^{max} for the case of $\left(L, M, K, p_l^{max}, |P_L|\right) = \left(6,1,4, P^{max}/10,1\right)$ and $\left(4,3,4, P^{max}/10,4\right)$ respectively. Here again we observe that the capacity increases as we amplify the interference threshold, because a feasible set of the optimization problem with lower I_m^{max} is a subset of a feasible set with higher I_m^{max}.

a.



b.

System							Common EAs	
K	M	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No of Simulation runs	Generations	Pop
6	2	1mW	2	$p^{max}/10$	$7^2 \sim 7^7$	200	20	20

BBO				ABC	ACO		EDA		
I	m	Migration	Levels	trial	ε	δ	Pxover	Pmut	Psel
1.4	0.8	Piece-wise constant	10	0.4× pop	0.5	0.9	0.99	0.95	0.3

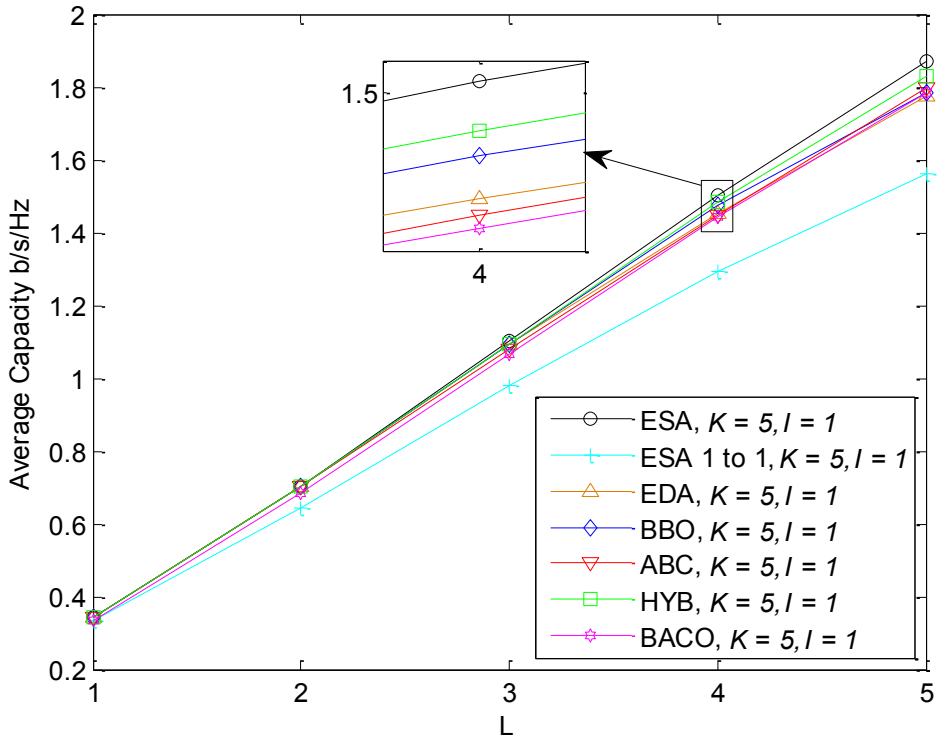
c.

	2	3	4	5	6	7
ESA 1 to 1	97	95	93	90	89	85
EDA	100	100	98	95	92	90
BACO	100	100	99	98	97	96
BBO	100	100	99	96	94	92
ABC	100	99	97	95	95	94
Hybrid	100	100	100	99	98	98

Figure 6.2. Sum rate vs. number of relays For K = 6

a. Sum rate vs. number of relays for $(\mathrm{K}, \mathrm{M}, \mathrm{I}_{\mathrm{m}}^{\max}, |\mathrm{P}_{\mathrm{L}}|) = (6,2,1\ \mathrm{mW}, 2)$,
b. simulation parameters,
c. algorithms' percentage of the ESA results

a.



b.

System							Common EAs	
K	M	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No of Simulation runs	Generations	Pop
5	4	1mW	4	$p^{max}/10$	$6^3 \sim 6^{15}$	200	20	20

BBO				ABC	ACO		EDA		
I	m	Migration	Levels	trial	ε	δ	Pxover	Pmut	Psel
1.4	0.8	Piece-wise constant	10	0.4× pop	0.5	0.9	0.99	0.95	0.3

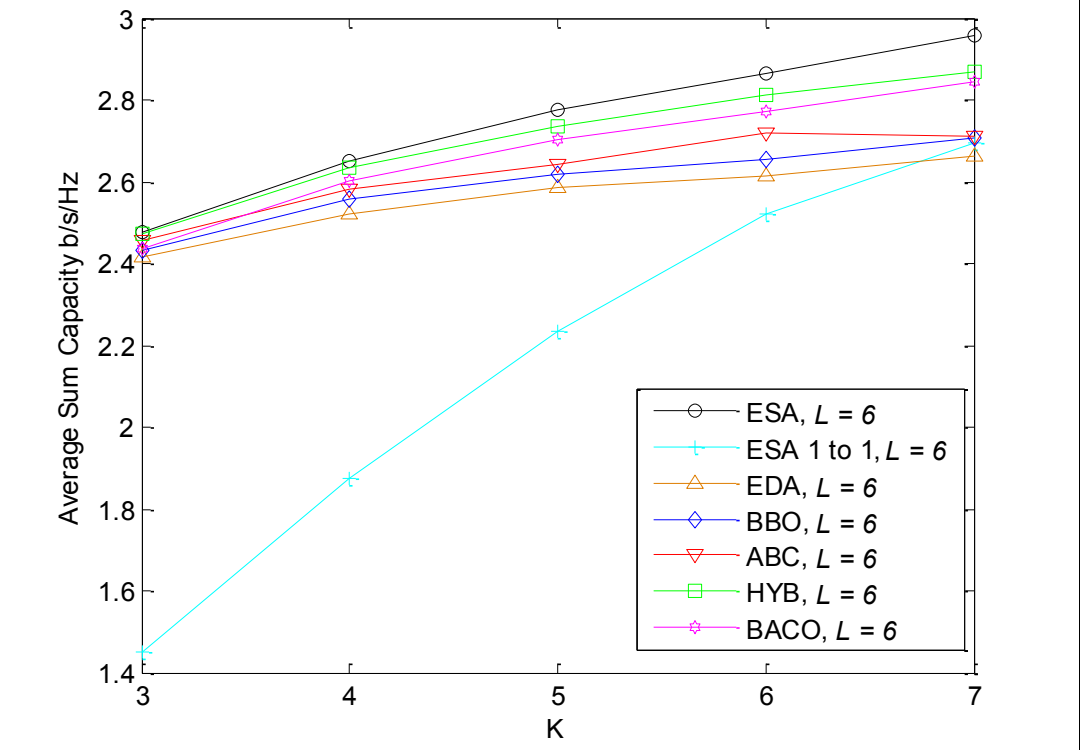
c.

	1	2	3	4	5
ESA 1 to 1	97	92	89	86	84
EDA	100	100	100	97	95
BACO	97	98	97	96	96
BBO	100	100	100	98	96
ABC	100	100	99	97	97
Hybrid	100	100	100	99	98

Figure 6.3. Sum rate vs. number of relays For K = 5

a. Sum rate vs. number of relays for $(K, M, I_m^{max}, |P_L|) = (5,4,1\ mW, 4)$,
b. simulation parameters,
c. algorithms' percentage of the ESA results

a.



b.

System							Common EAs	
L	M	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No of Simulation runs	Generations	Pop
6	5	100mW	2	$p^{max}/10$	$4^6 \sim 8^6$	200	20	20

BBO				ABC	ACO		EDA		
I	m	Migration	Levels	trial	ε	δ	Pxover	Pmut	Psel
1.4	0.8	Piece-wise constant	10	0.4× pop	0.5	0.9	0.99	0.95	0.3

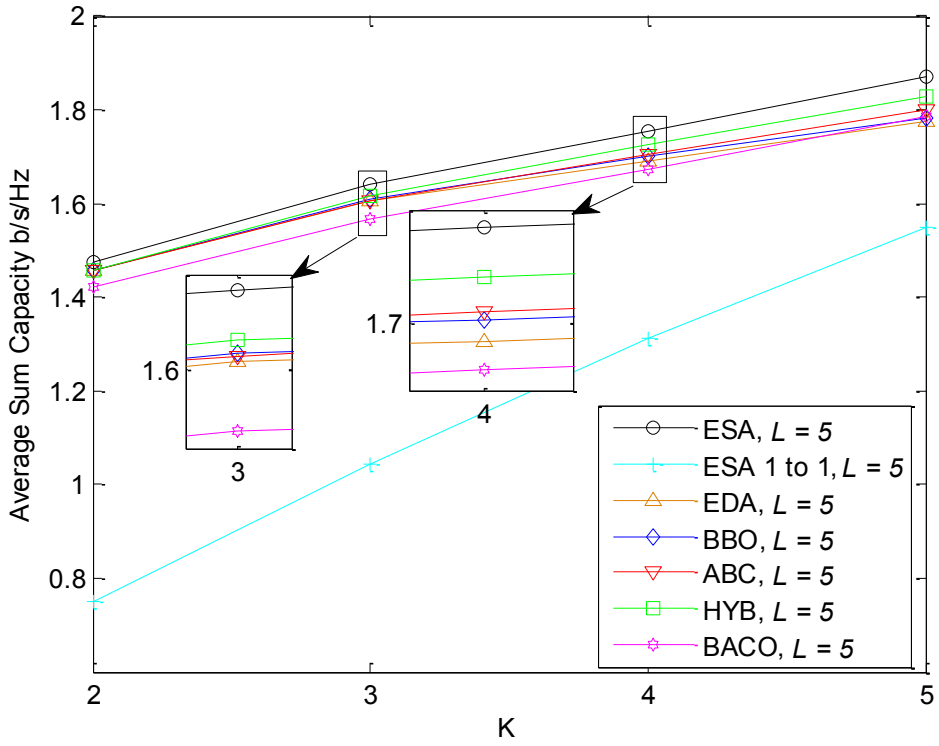
c.

	3	4	5	6	7
ESA 1 to 1	59	71	81	88	92
EDA	98	96	94	92	91
BACO	99	99	98	97	97
BBO	99	97	95	93	92
ABC	100	98	96	95	92
Hybrid	100	100	99	99	98

Figure 6.4. Sum rate vs. number of users For L = 6

a. Sum rate vs. number of users for $(L, M, I_m^{max}, |P_L|) = (6,5,100\,mW, 2)$,
b. simulation parameters,
c. algorithms' percentage of the ESA results

a.



b.

System							Common EAs	
L	M	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No of Simulation runs	Generations	Pop
5	4	100mW	4	$p^{max}/10$	$3^{15} \sim 5^{15}$	200	20	20

BBO				ABC	ACO		EDA		
I	m	Migration	Levels	trial	ε	δ	*P*xover	*P*mut	*P*sel
1.4	0.8	Piece-wise constant	10	0.4× *pop*	0.5	0.9	0.99	0.95	0.3

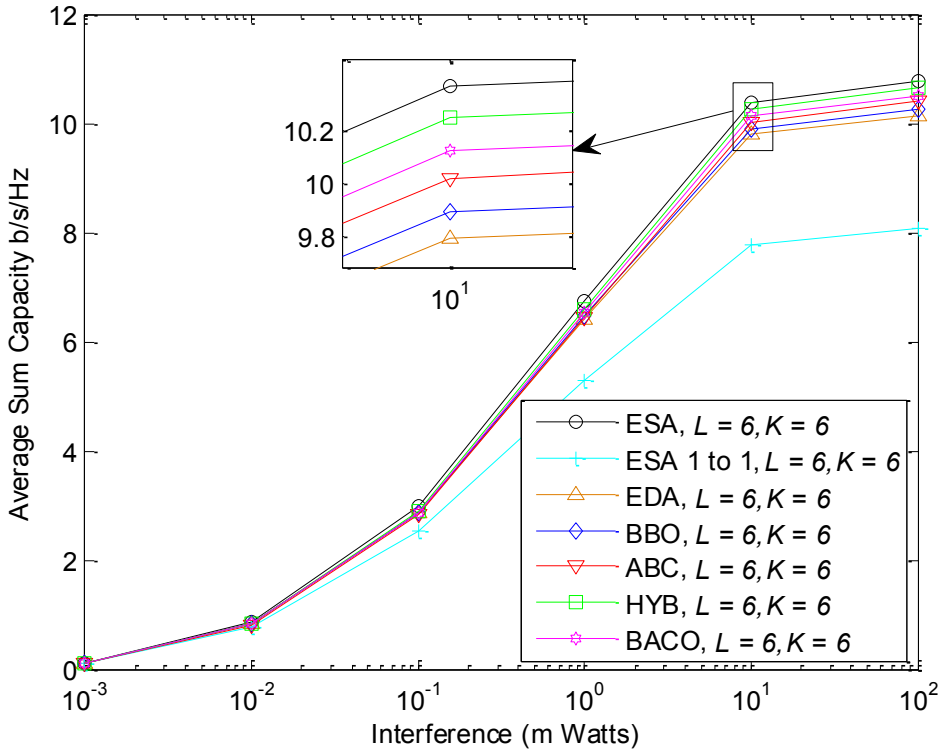
c.

	2	3	4	5
ESA 1 to 1	51	64	75	83
EDA	99	98	97	95
BACO	97	96	96	96
BBO	99	98	97	96
ABC	99	98	98	97
Hybrid	99	99	99	98

Figure 6.5. Sum rate vs. number of users For L = 5

a. Sum rate vs. number of users for $(L, M, I_m^{max}, |P_L|) = (5,4,100\,mW, 4)$,
b. simulation parameters,
c. algorithms' percentage of the ESA results

a.



b.

System							Common EAs	
L	M	K	$\|P_L\|$	p_l^{max}	Search space	No. of Simulation runs	Generations	Pop
6	5	6	2	$p^{max}/10$	7^6	200	20	20

BBO				ABC	ACO		EDA		
I	m	Migration	Levels	trial	ε	δ	Pxover	Pmut	Psel
1.4	0.8	Piece-wise constant	10	0.4× pop	0.5	0.9	0.99	0.95	0.3

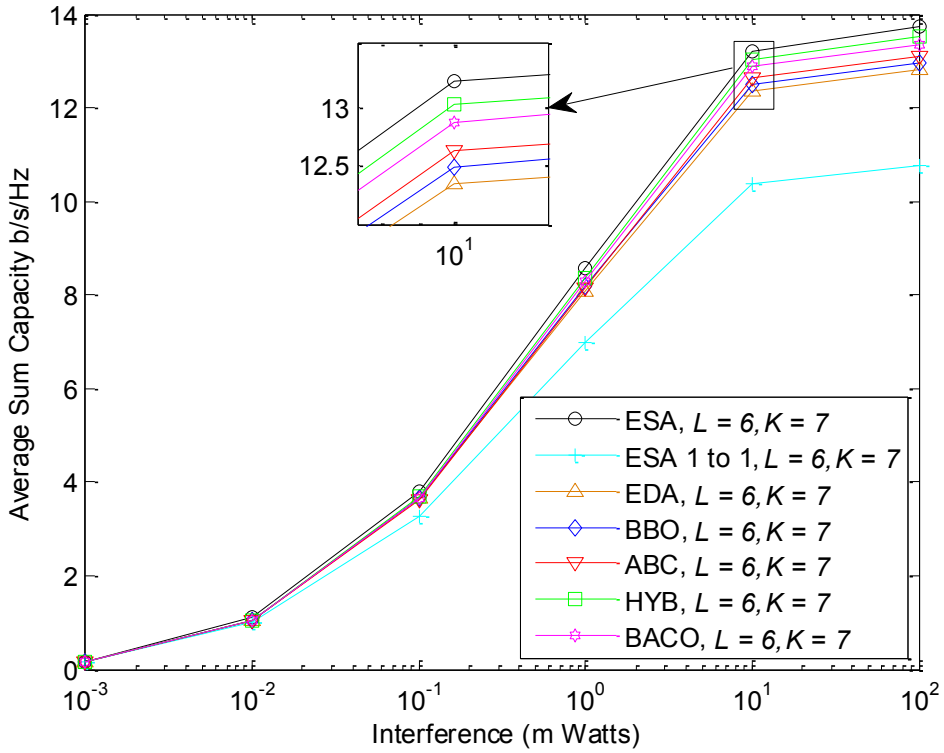
c.

	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
ESA 1 to 1	99	90	85	79	75	76
EDA	96	96	97	95	95	95
BACO	96	96	97	97	98	98
BBO	96	96	97	96	96	96
ABC	96	95	96	96	97	97
Hybrid	96	95	97	98	99	99

Figure 6.6. Sum rate vs. interference threshold For K = 6

a. Sum rate vs. interference threshold for $(K, L, M, |P_L|) = (6,6,5,2)$,
b. simulation parameters,
c. algorithms' percentage of the ESA results

a.



b.

System							Common EAs	
L	M	K	$\|P_L\|$	p_l^{max}	Search space	No. of Simulation runs	Generations	Pop
6	5	7	2	$p^{max}/10$	8^6	200	20	20

BBO				ABC	ACO		EDA		
I	m	Migration	Levels	trial	ε	δ	Pxover	Pmut	Psel
1.4	0.8	Piece-wise constant	10	0.4× pop	0.5	0.9	0.99	0.95	0.3

c.

	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
ESA 1 to 1	**99**	**91**	**87**	**82**	**79**	**79**
EDA	**96**	**96**	**96**	**95**	**94**	**94**
BACO	**96**	**95**	**97**	**97**	**98**	**98**
BBO	**96**	**96**	**97**	**96**	**95**	**95**
ABC	**96**	**95**	**95**	**95**	**96**	**96**
Hybrid	**96**	**95**	**97**	**98**	**99**	**99**

Figure 6.7. Sum rate vs. interference threshold For K = 7

a. Sum rate vs. interference threshold for$(K, L, M, |P_L|) = (7,6,5,2)$,
b. simulation parameters,
c. algorithms' percentage of the ESA results

In the above simulation results, the performance of all EAs is comparable to ESA, while the hybrid algorithm consistently returns results closer to the optimal results. In all results, the complexity of all EAs is less than the optimal ESA with power control. For instance, according to section 6.2, for the scenario of Figure 6.5 and the system parameters of $\left(L, K, |P_L|\right) = (5,5,4)$ and a pair of population size and generations $(G, N) = (20,20)$, the number of ESA iterations over all possible iterations is $5^{15} \cong 3 \times 10^{10}$; whereas the number of iterations for BBO and GA is $G.N = 20^2 = 400$. Thus the computational complexity of the EA-based schemes are significantly lower than ESA with fairly competitive results, while sometimes the hybrid algorithm returns the same result as the optimal ESA with the power control. The hybrid algorithm consistently returns the closest results to the ESA, and following by ACO, BBO, ABC and GA. In short, the hybrid-based scheme performs close to ESA, with significant lower computational complexity.

6.4.2. EAs' Evolution Comparison Results

From the practical point of view, computational complexity is a major concern for any algorithm applied to the problem. Therefore we present some comparison result that helps to choose the best algorithm and its settings that returns the favorable results with a relatively low computational complexity. Similar to section 5.6.1, we present some specific results for comparing the performance of EAs in the relay assignment problem.

The first set of results is the comparison between the EAs' performances (the average sum capacity in this problem) vs. the number of iterations for each algorithm that is depicted in Figure 6.8 and Figure 6.9. These figures show the number of iterations required by each algorithm to achieve a certain capacity. The system configurations are $\left(L, M, k, I_m^{max}, |P_L|\right) = (6,4,3,10\, mW, 2)$ and $(5,4,5,10\, mW, 2)$ for Figure 6.8 and Figure 6.9, respectively. Figure 6.8 shows that the hybrid algorithm has the fastest approach towards the optimal ESA results, and becomes very close to ESA at its 17th iteration. Figure 6.9 also demonstrates the superiority of the hybrid algorithm over other EAs, which constantly is the closest algorithm to the optimal ESA. As a result, the hybrid algorithm outperforms other EAs such as BACO, GA and EDA, as well as its predecessors BBO and DABC.

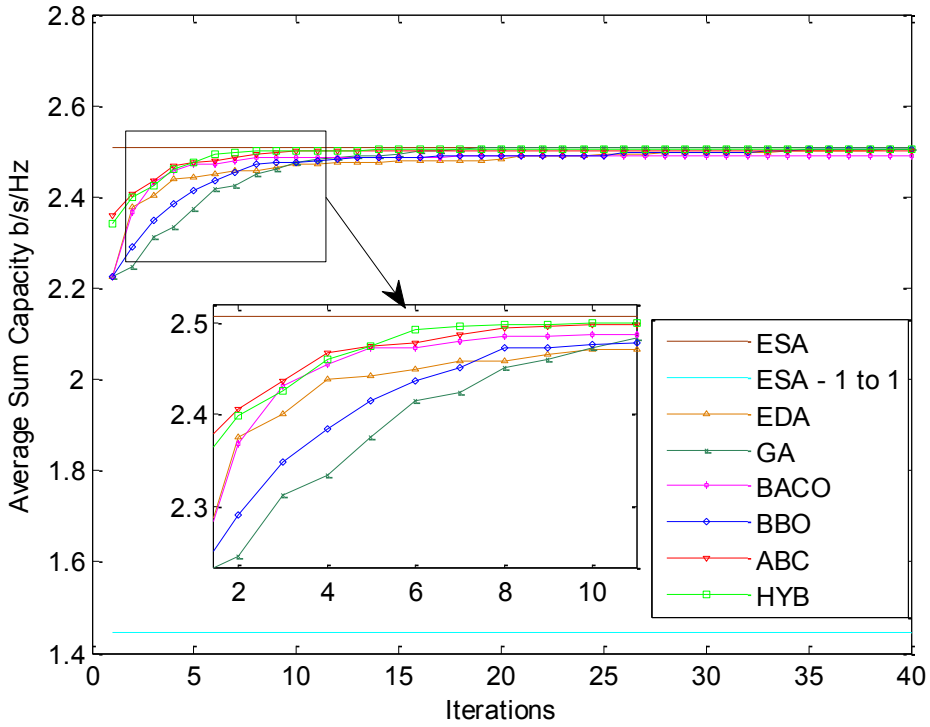
Furthermore, a better understanding of the EAs' evolution towards the optimal result can be observed from a three-dimensional plot that depicts the algorithm performance vs. the number of iterations and population size. Figure 6.10 show the trade-off between the population size and the iterations required to achieve a desired average sum capacity for EDA, BACO, BBO, ABC and the hybrid algorithm. The system configuration is $\left(L, M, k, I_m^{max}, |P_L|\right) = (5,2,4,10\ mW, 4)$. The detailed system configuration is given in Table 6.2.This trade-off is useful from the system design point of view. If a hardware system has high processing capabilities and low memory, then we can set the population size low to get same performance and vice versa. (Higher N and G needs more memory.) These figures also show that EAs has faster evolution towards the optimal result in the relay assignment problem than the MD-STBC-MIMO results depicted in Figure 5.9 to Figure 5.13. We further observe that for population size above 10, the hybrid algorithm has the fastest convergence. As a result, we conclude that the hybrid algorithm has the closest results to the optimal ESA, while it has the fastest approach towards the results as well.

Table 6.2. System parameters for iteration–population size trade-off

System								**Common EAs**	
L	*M*	*K*	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No. of Simulation runs	Generation	Pop
5	2	4	10mW	4	$p^{max}/10$	6^5	50	1 ~ 40	1~40

BBO				**ABC**	**ACO**		**EDA**		
I	*m*	Migration	*Levels*	trial	ε	δ	P_{xover}	P_{mut}	P_{sel}
1.4	0.8	Piece-wise constant	1	0.4× *pop*	0.5	0.9	0.99	0.95	0.3

a.



b.

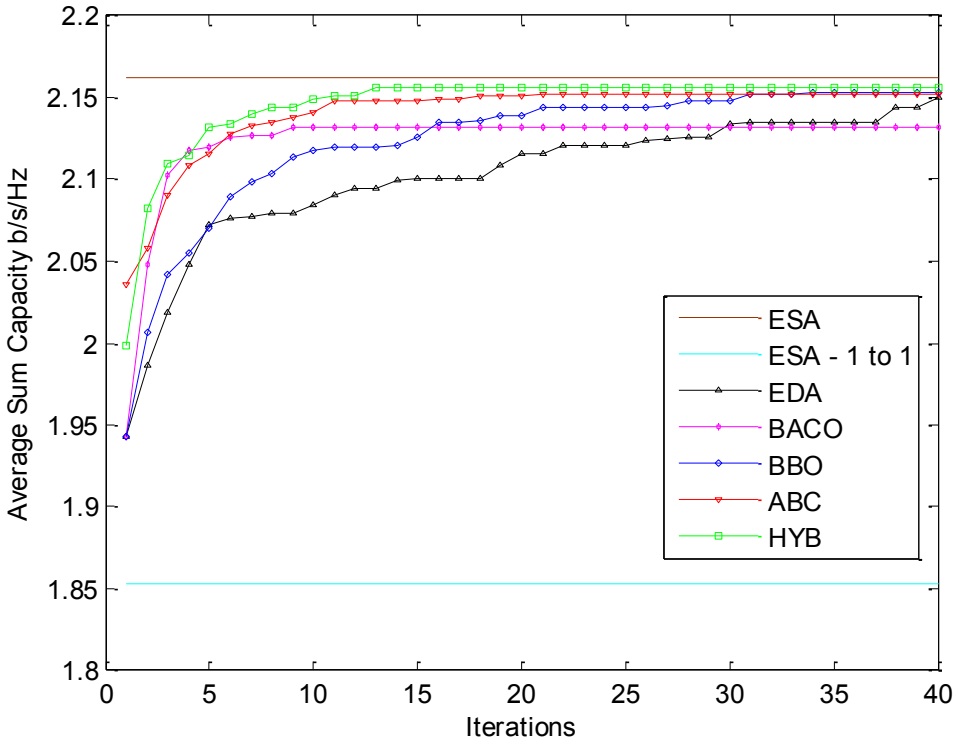
System								Common EAs	
L	*M*	*K*	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No. of Simulation runs	Generation	Pop
6	4	3	10mW	2	$p^{max}/10$	4^6	20	1 ~ 40	20

BBO				ABC	ACO		EDA		
I	*m*	Migration	*Levels*	trial	ε	δ	P_{xover}	P_{mut}	P_{sel}
1.4	0.8	Piece-wise constant	1	0.4× *pop*	0.5	0.9	0.99	0.95	0.3

Figure 6.8. Sum rate vs. algorithms' iteration For L = 6

a. Sum rate vs. algorithms' iteration for $\left(\boldsymbol{L, M, k, I_m^{max}, |P_L|}\right) = (\mathbf{6, 4, 3, 10\ mW, 2})$,
b. simulation parameters,

a.



b.

System								Common EAs	
L	*M*	*K*	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No. of Simulation runs	Generation	Pop
5	4	5	10mW	2	$p^{max}/10$	6^5	20	1 ~ 40	50

BBO				ABC	ACO		EDA		
I	*m*	Migration	*Levels*	trial	ε	δ	P_{xover}	P_{mut}	P_{sel}
1.4	0.8	Piece-wise constant	1	0.4× *pop*	0.5	0.9	0.99	0.95	0.3

Figure 6.9. Sum rate vs. algorithms' iteration For L = 5

a Sum rate vs. algorithms' iteration for $\left(\boldsymbol{L, M, k, I_m^{max}, |P_L|}\right) = \boldsymbol{(5, 4, 5, 10\ mW, 2)}$,
b. simulation parameters,

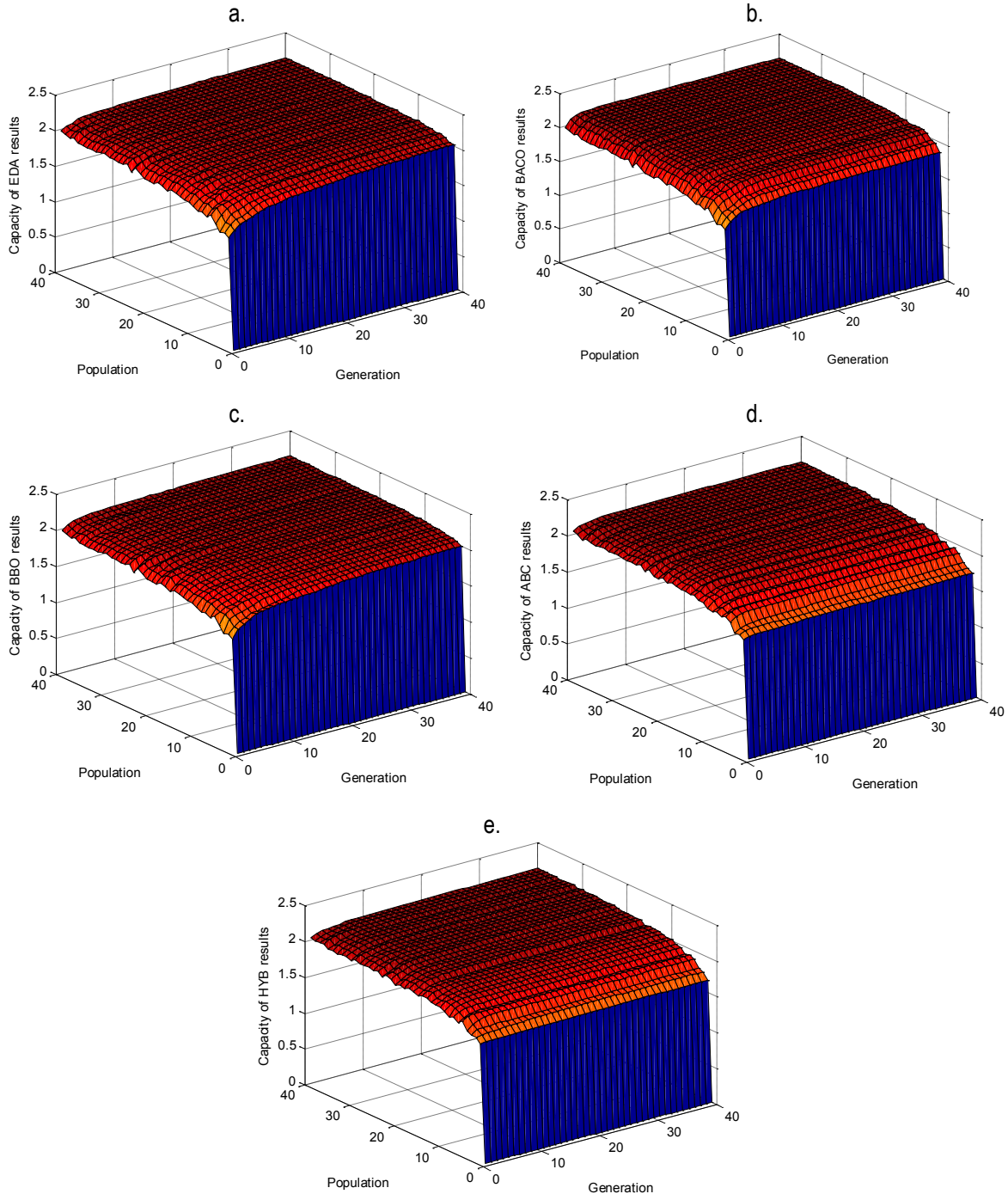


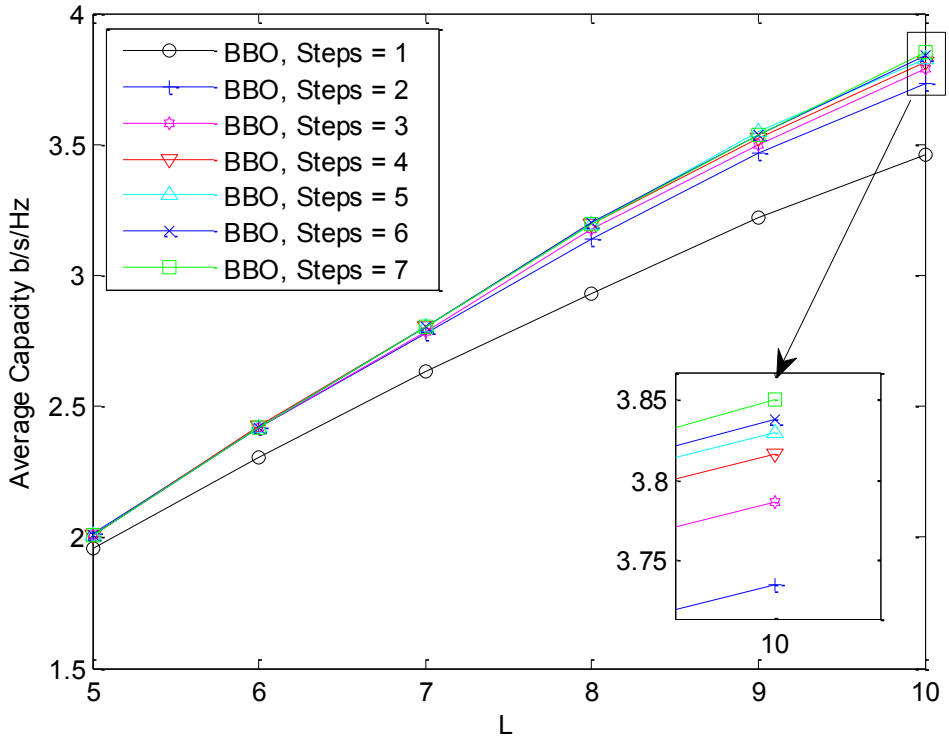
Figure 6.10. Sum rate vs. algorithms' population size and iteration

a. EDA BER performance comparison for $\left(L, M, k, I_m^{max}, |P_L|\right) = (5,2,4,10\ mW, 4)$,
b. BACO, c. BBO, d. ABC, e. Hybrid algorithm

6.4.3. BBO Migration Tuning Result

At the end of this section, we present a plot to demonstrate the effect of number of steps for the proposed linear immigration, piece-wise constant emigration BBO migration scheme. The plot and detailed system model is presented in Figure 6.11.

a.



b.

System						
K	M	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No of Simulation runs
7	4	1mW	4	$p^{max}/10$	$8^{15} \sim 8^{30}$	200

BBO parameters					
I	m	Migration	Levels	Generations	Pop
1.4	0.8	Piece-wise constant	1 ~ 7	30	30

Figure 6.11. Comparison between the number of BBO migration steps

a. Comparison between the number of BBO migration steps for $(K, M, I_m^{max}, |P_L|) = (7,4,1\ mW, 4)$,
b. simulation parameters,

In this figure, “steps = 1” refers to the linear immigration, constant emigration scheme, which has the lowest performance. This figure clearly demonstrates the effect of more steps in the piece-wise constant immigration scheme that results in better overall algorithm performance, without increasing much complexity. Therefore, this scheme would be a considerable setting for BBO to return high performance result with low complexity.

6.4.4. EAs’ Complexity Comparison Results

Lastly, we present a complexity comparison result between the EAs applied to this relay assignment problem. We compare these algorithms in terms of their numbers of fitness function evaluations. As discussed in section 5.6.2, BBO and EDA both have a fixed number of $G.N$ fitness function evaluations. Similarly, BACO has the same number of $G.N$ evaluations.

A simulation is run with a cognitive radio system with the system parameters $\left(L, M, k, I_m^{max}, |P_L|\right) = (6,4,5,10\ \mathrm{mW}, 2)$, and $G = N = 20$, where BBO, EDA and BACO return their result by evaluating the fitness function $GN = 400$ times. Yet one simulation run of DABC and hybrid algorithm needs an average of 665 and 560 fitness function evaluations, which is 65% and 44% more than that number for the first three EAs respectively. Note that all these EAs’ complexities are still much lower than the complexity of the optimal ESA.

We observe that unlike the MD-STBC-MIMO problem, the hybrid algorithm doesn’t beat other algorithms in terms of both performance and complexity (in terms of the number of fitness function evaluations). This algorithm still returns the best performance results compare to other EAs, yet it costs more number of function evaluations. We conclude that at the presence of sufficient processing power that can handle the 44% complexity of the hybrid algorithm, it would be the best choice to achieve the closest results to the optimal ESA.

6.4.5. Related Work

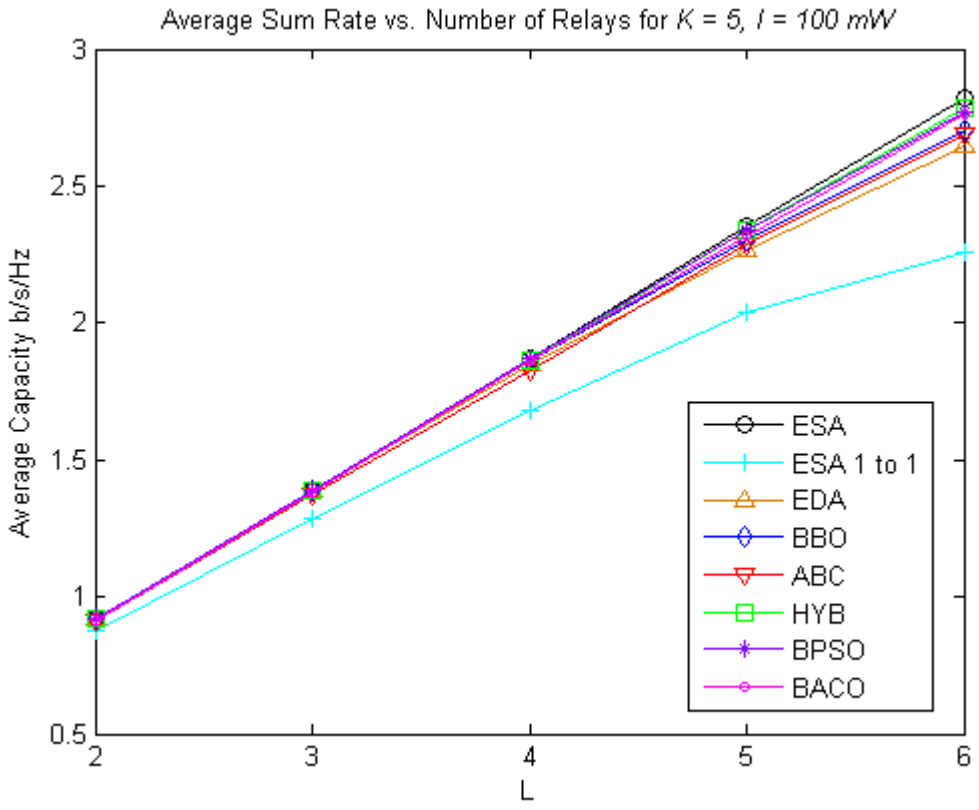
The author has published two papers from the results of this chapter in [21] and [22]. The issue of optimal relay assignment schemes has been proposed for cellular networks with some work on relay assignment and power allocation schemes [23]. However, these schemes cannot be applied to cognitive radio, because they may violate the interference constraints at the PUs. There are also some works on the optimal relay assignment in ad-hoc networks that comprises multiple source-destination pairs [24], [25]. Pareek et al proposed a relay assignment schemes for cognitive radio networks with single source node, single multiple destinations and multiple relays in [26]; but the power to source and relays are not assigned optimally. A similar work has been published in [27] and a binary Particle Swarm Optimization (BPSO) [28] applied to the cooperative cognitive radio system. A comparison between BPSO and other EAs mentioned in 6.4.1 is presented in Figure 6.12. For a fair comparison between the algorithms, the simulation parameters are tuned to the algorithms' best performance for solving the optimization problem. This figure shows that the hybrid algorithm and BPSO curves are close together. Other papers are published with similar system models, comprising only one receiver node such as [29] [30]. The work presented in this thesis considers a more general case of a cognitive radio system with multiple destinations.

6.5. Conclusion

In this chapter, we present the optimization problem formulation for a relay assignment problem in a multi-user cognitive radio network with discrete power control. The cognitive radio network consists of a single source node, multiple relays and multiple destination nodes. In section 6.2 we demonstrate the separation of the source transmitted power and the relays' transmission power levels, in addition to the optimization of the source and relays' transmission power levels in (6.3) are reduces to the optimization of the relays' power levels, as in (6.6). Then we introduced EA-based relay assignment algorithm with low computational complexity. We also propose a constraint checker algorithm to ensure the interference threshold is satisfied in the procedure of EA relay assignment. We observe that reduction in search space results in low computational complexity and faster convergence to an acceptable solution. The

performance comparison results of the proposed schemes are comparable to the optimal ESA, and can beat other mainstream EAs such as EDA and BACO. Using other efficient constraint optimization techniques inside the bio-inspired heuristic algorithms, multiuser cognitive radio network with imperfect channel gain, as well as constant power level for relays and source are left for the future work.

a.



b.

System							Common EAs	
K	M	I_m^{max}	$\|P_L\|$	p_l^{max}	Search space	No of Simulation runs	Generations	Pop
5	2	100mW	2	$p^{max}/10$	$6^2 \sim 6^6$	150	24	24

BBO				ABC	ACO		EDA		
I	m	Migration	Levels	trial	ε	δ	Pxover	Pmut	Psel
1.4	0.8	Piece-wise constant	10	0.4× pop	0.5	0.9	0.99	0.95	0.3

Figure 6.12. Sum rate vs. number of relays comparison with BPSO for K = 5

a. BER comparison vs. number of relays for $(K, M, I_m^{max}, |P_L|) = (5,2,0.1W,2)$,
b. simulation parameters,

References

[1] Federal Communications Commission, *Spectrum Policy Task Force Report*, FCC 02-135, 2002.

[2] J. Mitola and G. Q. Maguire, “Cognitive radio: making software radios more personal,” *IEEE Personal Communications Magazine,* vol. 6, no. 4, pp. 13-18, Aug. 1999.

[3] S. Haykin, “Cognitive radio: brain-empowered wireless communications,” *IEEE Journal on Selected Areas in Communications,* vol. 23, no. 2, pp. 201-220, Feb. 2005.

[4] L. Berlemann and S. Mangold, *Cognitive radio and Dynamic Spectrum Access*, Wiley, 2009.

[5] L. E. Doyle, *Essentials of Cognitive Radio,* Cambridge University Press, 2009.

[6] A. Nosratinia, T. E. Hunter, and A. Hedayat, “Cooperative Communication in Wireless Networks,” *IEEE Communication Magazine,* vol. 42, no. 10, pp. 68-73, Oct. 2004.

[7] J. Laneman, D. Tse, and G. Wornell, “Cooperative Diversity in Wireless Networks: Efficient Protocols and Outage Behavior,” *IEEE Trans. Inform. Theory,* vol. 50, no. 12, pp. 3062-3080, Dec. 2004.

[8] Y. Jing and H. Jafarkhani, “Single and multiple relay selection schemes and their achievable diversity orders,” *IEEE Transactions on Wireless Communications,* vol. 8, no. 3, pp. 1414-1423, Mar. 2009.

[9] S. Kadloor, R. Adve, “Optimal Relay Assignment and Power Allocation in Selection Based Cooperative Cellular Networks,” *ICC '09 IEEE International Conf. on,* pp. 1-5, 14-18 Jun. 2009.

[10] Yi Shi, Sushant Sharma, Y. Thomas Hou, and Sastry Kompella. “Optimal relay assignment for cooperative communications” In Proceedings of *the 9th ACM international symposium on Mobile ad hoc*

[11] P. Zhang, Z. Xu, F. Wang, X. Xie, L. Tu, “A Relay Assignment Algorithm With Interference Mitigation For Cooperative Communication”, *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE* , vol., no., pp.1-6, 5-8 April 2009.

[12] M. Naeem, U. Pareek, D. C. Lee, “Interference Aware Relay Assignment Schemes For Multiuser Cognitive Radio Systems,” *Proc. IEEE Vehicular Technology Conf.* Ottawa, Canada, Sep. 2010

[13] T. Cover and J. Thomas, *Elements of Information Theory*, 2nd Ed. Wiley, NY, 2006.

[14] I. Maric and R. D. Yates, “Bandwidth and Power Allocation for Cooperative Strategies in Gaussian Relay Networks,” *IEEE Trans. Inform. Theory,* vol. 56, no. 4, pp. 1880-1889, Apr. 2010.

[15] R. Eberhart, Y. Shi, and J. Kennedy, *Swarm Intelligence,* Morgan Kaufmann, 2001.

[16] H. Ma, “An analysis of the equilibrium of migration models for biogeography-based optimization,” *Information Sciences,* vol. 180, no. 18, pp. 3444-3464, Sep. 2010.

[17] D. E. Knuth, *The Art of Computer Programming, Vol. 3: Sorting and Searching,* Addison-Wesley, 1998.

[18] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning,* Addison-Wesley, 1989.

[19] J. Proakis, *Digital Communication*, McGraw-Hill, 5th Ed. 2007.

[20] Ibrahim, A.S.; Sadek, A.K.; Weifeng Su; Liu, K.J.R.;, "Cooperative communications with relay-selection: when to cooperate and whom to cooperate with?," *Wireless Communications, IEEE Transactions on* , vol.7, no.7, pp.2814-2827, July 2008.

[21] S. Ashrafinia; U. Pareek; M. Naeem; D. Lee; “Biogeography-based optimization for joint relay assignment and power allocation in cognitive radio systems”, *Swarm Intelligence (SIS), 2011 IEEE Symp. on*, 11-15 April 2011, pp 1 – 8, Paris, France.

[22] S. Ashrafinia; U. Pareek; M. Naeem; D. Lee ; “Binary Artificial Bee Colony for Cooperative Relay Communication in Cognitive Radio Systems”, *IEEE International Conf. on Comm., ICC 2012*, 10-15 Jun. 2012, Ottawa, Canada

[23] S. Kadloor, R. Adve, “Optimal Relay Assignment and Power Allocation in Selection Based Cooperative Cellular Networks,” *Commun., 2009. ICC ’09. IEEE International Conf. on,* pp. 1-5, 14-18 Jun. 2009.

[24] Yi Shi, Sushant Sharma, Y. Thomas Hou, and Sastry Kompella. “Optimal relay assignment for cooperative communications” In Proceedings of *the 9th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc ’08)*. ACM, New York, NY, USA, pp. 3-12, 2008.

[25] P. Zhang, Z. Xu, F. Wang, X. Xie, L. Tu, A Relay Assignment Algorithm With Interference Mitigation For Cooperative Communication, Wireless Communications and Networking Conference, WCNC 2009. IEEE ,pp.1-6, 5-8 April 2009.

[26] M. Naeem, U. Pareek, D. C. Lee, “Interference Aware Relay Assignment Schemes For Multiuser Cognitive Radio Systems,” *Proc. IEEE Vehicular Technology Conf.* Ottawa, Canada, Sep. 2010.

[27] Pareek, U.; Naeem, M.; Lee, D.C.; , "An efficient relay assignment scheme for multiuser cognitive radio networks with discrete power control," *Wireless and Mobile Computing, Networking and Communications (WiMob), 2010 IEEE 6th International Conference on* , vol., no., pp.653-660, 11-13.

[28] J. Kennedy and R.C. Eberhart , “Particle Swarm Optimization,” *Proceedings of IEEE Conf. on Neural Networks*, Piscataway, NJ, USA, pp. 1942-1948, 1995.

[29] S. Ashrafinia; U. Pareek; M. Naeem; D. Lee ; “Source and Relay Power Selection Using Biogeography-Based Optimization for Cognitive Radio Systems”, *IEEE VTC 2011 – fall*, 5-8 Sep. 2011, San Francisco, CA.

[30] U. Pareek, M. Naeem, and D. C. Lee, “Quantum inspired evolutionary algorithm for joint user selection and power allocation for uplink cognitive MIMO systems,” *Proc. IEEE Symposium Series in Computational Intelligence 2011 - Track: CISched - 2011 IEEE Symposium on Computational Intelligence in Scheduling*.

7. Green Resource Allocation in Cognitive Radio Systems

In this chapter, we formulate a resource allocation optimization problem for a cooperative relay-assisted cognitive radio system, comprising a single source node, multiple relays and multiple destinations. Our formulation takes into account the effects of the resource allocation on CO_2 emission, and we refer to it as a green resource allocation problem. The green resource allocation problem is formulated as a non-linear multi-objective optimization problem. We modify the objective function by applying the weighted sum method, which results in a non-convex mixed integer non-linear programming problem. We propose a hybrid evolutionary scheme that utilizes different EAs such as GA, EDA, ABC and hybrid BBO/ABC to solve this optimization problem. Simulation results demonstrate the efficiency of the hybrid algorithm approach in comparison to other schemes such as ABC, GA and EDA.

7.1. Introduction

The Information and Communication Technology (ICT) has become one of the 21^{st} century's biggest industries and accordingly has a huge carbon foot print. According to the Smart 2020 report, this industry will emit 1.4 Giga tons (10^9) of carbon dioxide (CO_2) emissions or 2.8% of global emissions by 2020 [1] [2] [3]. This sector is responsible for approximately five per cent of the global electricity demand and CO_2 emission [6] [7].

It is estimated that the ICT industry alone produces CO_2 emission equivalent to the carbon output of the entire aviation industry [2]. ICT emissions growth fastest of any sector in society: doubling about every 4 – 6 years [5]. Currently ICT represent 8 – 9.4% of total US electricity consumption, and 8% of the global electricity consumption, and it is

projected to grow to as much as 20% of all electrical consumption in the US [5]. Future Broadband Internet alone is expected to consume 5% of all electricity.

The main aim of green ICTs is to minimize the CO_2 emissions. Research in green ICTs will enable the communication system designer to develop and design the communication systems that will use power more efficiently and thus contribute to reducing the CO2 emissions.

In the last few years, there have been increasing efforts towards green ICTs. A comprehensive survey on green networking is presented in [4]. In [6], authors presented the concept of energy efficiency in telecommunication networks. A detailed discussion about ICTs footprint and its impact on the environment is presented in [8] [9] and [10]. In [11], authors described a variable power/bandwidth efficient modulation strategy to save the battery life of the communication device. Information and technology companies like Google and Microsoft have already started working towards green ICTs [21] [22].

In the context of green communication, cooperative communication can contribute to reducing the CO_2 emissions. Cooperative communication is a powerful concept for extending coverage and improving system's efficiency [23]. Green communications can utilize cooperative paradigms in order to reduce energy consumption for signal transmission [24]. In this chapter, we present a multi-objective optimization framework that jointly solves the problem of spectrum sharing and reducing CO_2 emissions. In particular, we propose a green multi-objective optimization framework for joint relay assignment and power allocation in a cooperative Cognitive Radio System (CRS). Then, we present evolutionary algorithms to solve the green multi-objective optimization.

7.2. Multi-objective Optimization

Multi-objective optimization (MOO) is used in many complex engineering optimization problems [12] − [15]. In typical MOO problems, different objectives can conflict with each other. Optimization with respect to any particular objective can give unacceptable results with respect to other objectives [14]. For resource allocation in Green Cooperative Cognitive Radio Network (GCCRN), in this chapter we consider two

conflicting objectives: to maximize the sum-capacity and to minimize the CO_2 emissions. Determining the optimal set of decision variables' values for a single objective (CO_2) emission minimization can result in a non-optimal solution with respect to other objectives, e.g. sum-capacity maximization.

Two widely used methods to solve multi-objective optimization, along with other methods, are weighted sum method and constraint objective method [12] – [15]. In the Weighted Sum Method (WSM), a weighted sum of the multiple objective function is considered as the metric to minimize (maximize). In WSM, the weight of each objective is proportional to its importance placed for decision making. A general WSM multi-objective optimization problem is expressed as follows:

$$\min f(x) = \sum_{i=1}^{Q} w_i f_i(x)$$

Subject to:

$$g_j(x) \leq 0, j = 1,2, \dots, U$$

$$h_j(x) = 0, j = 1,2, \dots, V$$

where the weights are such that $\sum_{i=1}^{Q} w_i = 1$, Q is the number of objective functions, U is the number of inequality constraints, and V is the number of equality constraints. In the constraint objective method [14], each objective is transformed into a constraint. In our formulation, we will use weighted sum method.

In formulating the weighted sum MOO, we will normalize each objective function $f_i(x)$ so that each objective function has the same range of values. The main reason for normalization is that the objective functions can have different dimensions (e.g., for the GCCRN problem, one is bits/Hz, and the other is power (Watts)) – they become dimensionless after normalization, and this enables their addition in the weighted sum expression. Furthermore, in a weighted-sum method [14] for MOO, without normalization, we cannot specify the bias toward a particular objective with weights alone. For instance, if the value of one objective function is in the range of [0, 1], and the value of second objective is in the range [0, *x*] (where $1 \leq x \leq \infty$), then the second

objective produces bias in the weighted fitness function, even if we use equal weights $w_1 = w_2 = 0.5$. In this work, all of the objective function values are normalized within the range close to [0, 1]. In the case in which we do not have exact maximal and minimal value of an individual objective function, we will normalize the objective function on the basis of its upper bound and lower bound. The GCCRN MOO is formulated so that the range of combined objective function is always within 0 and 1.

7.3. Green Relay Assignment for GCCRN

We consider a two-hop wireless network with one transmitter (source), *K* receivers (secondary users), *L* relays, and *M* primary users, as illustrated in Figure 7.1. Each relay, transmitter, and receiver is equipped with a single antenna. We denote by $h_{s,l}$, the channel from the source to the l^{th} relay, $h_{l,k}$the channel from the l^{th} relay to the k^{th} secondary user, and $g_{l,m}$ the channel from the l^{th} relay to the m^{th} primary user. We denote by p_l, the l^{th} relay's transmission power. We consider a two-step amplify-and-forward (AF) scheme [16]. We assumed in our cognitive radio network that data is received by each destination node on a separate frequency band and that each source-destination pair has been allocated equal bandwidth. We further assume that signals destined for different users do not interfere with one another. We also assume that each relay transmits its received signal at the same frequency band in which it received the signal. This models a low-cost relay that simply amplifies the signal and forwards it. We define $\varepsilon_{l,k}$as a binary assignment indicator variable such that:

$$\varepsilon_{l,k} = \begin{cases} 1 & \text{if the } l^{\text{th}} \text{ relay is assigned to the } k^{\text{th}} \text{ receiver} \\ 0 & \text{otherwise} \end{cases}$$

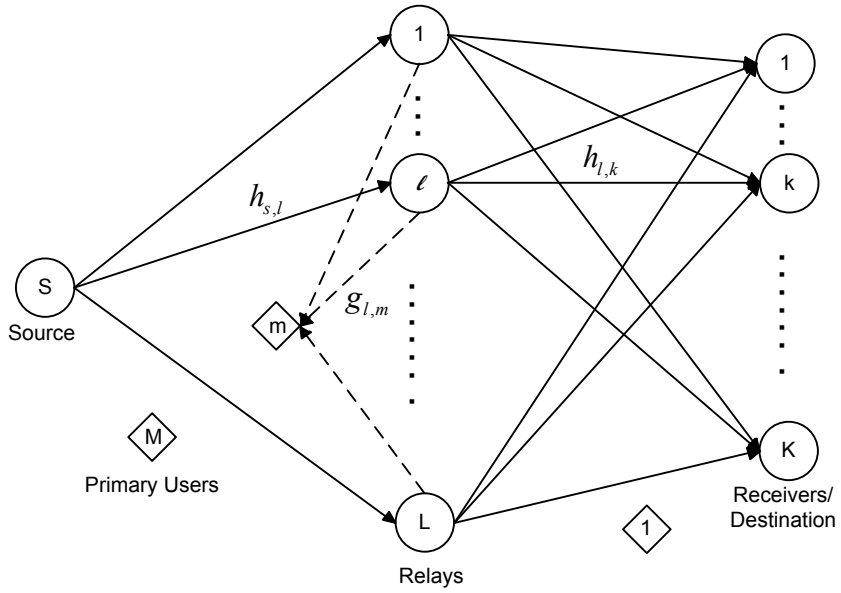


Figure 7.1. Cooperative Cognitive Radio Network

The channel capacity of the k^{th} user for amplify and forward relaying is [16] [17] [25]:

$$C_k(\boldsymbol{\varepsilon}, \boldsymbol{p}) = \frac{1}{2}\log\left[1 + \frac{P_s^k}{N}\left(\frac{\left(\sum_{l=1}^{L} \varepsilon_{l,k}\left|\, h_{s,l}h_{l,k}\right|\beta_l\sqrt{p_l}\right)^2}{1 + \sum_{l=1}^{L}\left(\beta_l\left|\, h_{l,k}\right|\sqrt{p_l}\right)^2}\right)\right] \tag{7.1}$$

where $\beta_l = \left(\sqrt{P_s^k\left|h_{s,l}\right|^2 + N}\right)^{-1}$, $\boldsymbol{\varepsilon} = \left[\varepsilon_{l,k}\right]_{L\times K}$ is an $L \times K$ binary matrix indicating relay to secondary users connectivity, and $\boldsymbol{p} = [p_l]_{1\times L}$is an L-dimensional vector comprising L relays' power levels. Our first objective is to maximize the sum-rate capacity $\sum_{k=1}^{K} C_k$. As mentioned in section 7.2, the division of the sum-rate capacity with $\sum_{k=1}^{K} C_k^{max}$ normalizes the first objective between 0 and 1, where C_k^{max} is an upper bound on the capacity of the k^{th} secondary user. The upper bound on the sum capacity can be obtained as follows:

$$\log\left[1+\frac{P_s^k}{N}.\frac{\left(\sum_{l=1}^{L}\varepsilon_{l,k}\left|\,h_{s,l}h_{l,k}\right|\beta_l\sqrt{p_l}\right)^2}{1+\sum_{l=1}^{L}\left(\beta_l\left|\,h_{l,k}\right|\sqrt{p_l}\right)^2}\right]$$

$$\leq\ \log\left[1+\frac{P_s^k}{N}.\frac{\left(\sum_{l=1}^{L}\left|\,h_{s,l}h_{l,k}\right|\beta_l\sqrt{p_l}\right)^2}{1+\sum_{l=1}^{L}\left(\beta_l\left|\,h_{l,k}\right|\sqrt{p_l}\right)^2}\right]$$

$$\leq\ \log\left[1+\frac{P_s^k}{N}.\frac{\left(\sum_{l=1}^{L}\left|\,h_{s,l}\right|^2\right)\left(\sum_{l=1}^{L}\left(\left|\,h_{l,k}\right|\beta_l\right)^2 p_l\right)}{1+\sum_{l=1}^{L}\left(\beta_l\left|\,h_{l,k}\right|\sqrt{p_l}\right)^2}\right]$$

$$\leq\ \log\left[1+\frac{P_s^k}{N}.\frac{\left(\sum_{l=1}^{L}\left|\,h_{s,l}\right|^2\right)\left(\sum_{l=1}^{L}\left(\left|\,h_{l,k}\right|\beta_l\right)^2 p_l^{max}\right)}{1+\sum_{l=1}^{L}\left(\beta_l\left|\,h_{l,k}\right|\sqrt{p_l^{max}}\right)^2}\right]\triangleq C_k^{max}$$

The second inequality is obtained from the Schwartz inequality. The last expression is greater or equal than the third one because the third expression is an increasing function of p_l. A proof is provided in Appendix 7.A. The latter expression is referred to as C_k^{max}. Note that if the term $\left(\sum_{l=1}^{L}\left(\left|\,h_{l,k}\right|\beta_l\right)^2 p_l^{max}\right)$ is canceled out from the numerator and denominator of the latter expression, the resulted upper bound is even looser and less accurate than the current C_k^{max}.

Mathematically, the objective of the sum-rate capacity can be expressed as:

$$\bar{f}_c(\boldsymbol{\varepsilon},\boldsymbol{p})=\frac{\sum_{k=1}^{K}C_k(\boldsymbol{\varepsilon},\boldsymbol{p})}{\sum_{k=1}^{K}C_k^{max}} \tag{7.2}$$

The second objective is to reduce the CO_2 emissions. The CO_2 emissions are measured in grams. If *P* is the transmission power and *X* is a constant in grams/KWh, then the product, *PX*, of *P* and *X* represents the CO_2 emissions in grams/hour. The value of *X* is different for different types of material (fuel) used in electricity generation. There are three major sources of fuel for electricity generation: oil, gas, and coal. The value of

X for lignite/brown coal, natural gas, crude oil and diesel oil is 940, 370, 640, and 670 grams/KWh, respectively [6] − [8]. The CO_2 emissions due to the l^{th} relay would be $E_l^{CO_2}(p_l) = Xp_l$. Therefore, the objective of CO_2 emissions can be written as:

$$f_{CO_2}(\boldsymbol{p}) = \frac{\sum_{l=1}^{L} E_l^{CO_2}(\boldsymbol{p})}{\sum_{l=1}^{L} E_{l^{max}}^{CO_2}} \tag{7.3}$$

where $E_{l^{max}}^{CO_2} = Xp_l^{max}$. To define a single objective, the maximization objective $\bar{f}_c$ is transformed into minimization using the relation $f_c = 1 - \bar{f}_c$. Mathematically, the MOO for GCCRN can be expressed as:

$$\begin{aligned}
&OP1: \min_{\boldsymbol{\varepsilon},\boldsymbol{p}} \{w_1 f_c(\boldsymbol{\varepsilon},\boldsymbol{p}) + w_2 f_{CO_2}(\boldsymbol{p})\} \\
&\text{subject to} \\
&C1: \sum_{k=1}^{K} \varepsilon_{l,k} \le 1, \ \ \forall l \\
&C2: \sum_{l=1}^{L} \varepsilon_{l,k} p_l \left| g_{l,m} \right|^2 \le I_m^{max}, \quad \forall (m,k) \\
&C3: \ \ 0 \le p_l \le \sum_{k=1}^{K} \varepsilon_{l,k} p_l^{max}, \quad \forall l \\
&C4: \ \ \varepsilon_{l,k} \in \{0,1\}
\end{aligned} \tag{7.4}$$

Both denominators of the two objective functions f_c and f_{CO_2} require p_l^{max} to be evaluated. The definition of p_l^{max} is $\max[p_1, p_2, \dots, p_L]$ which is non-differentiable at the points $p_1, p_2, \dots, p_L$, thus the objective functions would be non-differentiable at these points too. Therefore, the optimization problem *OP1* is non-convex. Moreover, p_l is a continuous variable, while $\varepsilon_{l,k}$ is discrete; thus the problem is mixed integer. Furthermore, the objective function f_c , and the constraint *C*2 both are nonlinear expressions of p_l and $\varepsilon_{l,k}$. As a result, the formulation in (7.4) is a non-convex mixed integer non-linear programming problem. The objective function in (7.4) is bounded by zero and one. In this equation, the constraint *C*1 ensures that a relay can only be

assigned to one secondary user, *C*2 is the interference constraint, the constraints *C*3 and *C*4 jointly ensure that if the l^{th} relay is not assigned to any secondary user, then the transmission power of the lth relay should be zero. In the next section, we present a low-complexity hybrid scheme, comprising an EA and an iterative greedy algorithm, for the GCCRN MOO problem.

7.4. Hybrid Solver for GCCRN MOO Problem

In this section, we present a solver for the GCCRN multi-objective problem. The proposed scheme is an integration of an EA and an Iterative Greedy Algorithm for relay assignment and power allocation such that the constraints in (7.4) are satisfied. The optimization variables of the GCCRN problem in (7.4) are $\boldsymbol{p}$ (relays' powers variable) and $\boldsymbol{\varepsilon}$ (relays assignment variable). The hybrid solver handles both optimization variables $\boldsymbol{\varepsilon}$ and $\boldsymbol{p}$, such a way that the EA operator modifies the variable $\boldsymbol{p}$ towards higher fitness value, and the Iterative Greedy algorithm verifies that the relays powers $\boldsymbol{p}$ (being modified by the EA) and the relays assignment variable $\boldsymbol{\varepsilon}$ satisfy the constraints in the optimization problem (7.4).

In the GCCRN problem, there are L relays that can handle continuous power levels, which demands continuous EAs to be applied to the problem. We implement ABC (Chapter 3) and the hybrid ABC/BBO algorithm (Chapter 4) for the EA part of the solver for the GCCRN problem. We have not included the BBO algorithm, since this algorithm is primarily presented for problems in the integer domain (Section 2.2). A continuous version of BBO would be the hybrid algorithm, which is included in our implementations.

The Iterative Greedy Algorithm receives one EA individual $\boldsymbol{p}$ as an input, and performs two operations on it. It modifies the relay power levels $\boldsymbol{p}$ such that the constraints *C2* and *C3* in (7.4) are satisfied; while it also determines the relays' assignment variable $\boldsymbol{\varepsilon}$ for each received EA individual $\boldsymbol{p}$ heuristically, and ensures that all constraints in (7.4) are satisfied.

Figure 7.2 illustrates the flowchart of the hybrid solver, which consists of an EA and the Iterative Greedy Algorithm. In the rest of this section, first we discuss our EA

implementation for the GCCRN problem and in Section 7.4.1, and then we discuss the Iterative Greedy Algorithm procedures in Section 7.4.2.

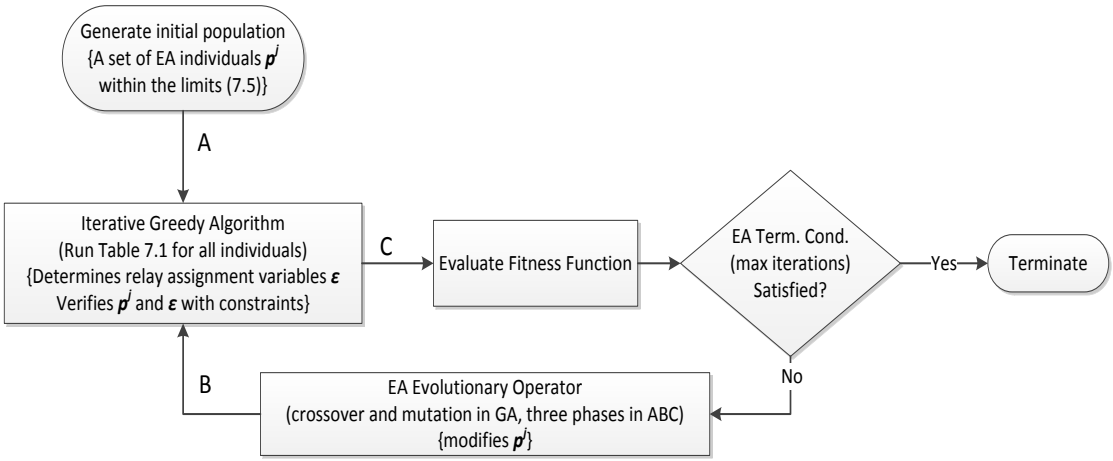


Figure 7.2. Flowchart of the Hybrid solver for GCCRN problem

The flowchart comprises an EA and an iterative greedy algorithm

7.4.1. Evolutionary Algorithms for the Hybrid Solver

EAs in general have been often used to solve MOO problems. Candidate solutions to a multi-objective optimization problem are represented as individuals in the population. In EAs, the objective function value of a candidate solution indicates the fitness of the individual, which is associated with the concept of natural selection [18]. Each **EA's individual** represents the **relays' transmission powers** $\boldsymbol{p}$. We denote by $\boldsymbol{p}^j$ the j^{th} individual in the population. Each individual $\boldsymbol{p}^j = [p_1^j, p_2^j, \dots, p_L^j]$ is a vector of L real components, where $p_l^j, l \in \{1, \dots, L\}$ represents the power of the l^{th} relay. Relays powers are bounded by $[P_{Low}, P_{High}]$, where P_{Low} and P_{High} denote the lower and upper limits of the EA's search window. Therefore, each individual of an EA is a vector of **continuous** real numbers between P_{Low} and P_{High}.

The first step of the hybrid solver is to generate a random population of EA individuals $\boldsymbol{p}^j$s. These relays' powers have to satisfy the constraints of the optimization problem (7.4) as well. According to the constraint *C2* of this optimization problem, relay powers should satisfy $\sum_{l=1}^{L} \varepsilon_{l,k} p_l \left| g_{l,m} \right|^2 \leq I_m^{max}, \forall(m,k)$. If the initial population is

generated randomly within $[P_{Low}, P_{High}]$, there is a possibility that some randomly generated p_l cause excessive interference to the PUs that violates the constraint *C2*. The algorithm could have generated the initial population within the aforementioned interval and further verify the constraints through an alternative procedure (Iterative Greedy Algorithm to ensure that individuals satisfy the constraint *C2*; nonetheless, we redefine the above generation interval to ensure that the randomly generated population lies within $[P_{Low}, P_{High}]$, while it also satisfies the constraint *C2*. The new interval for randomly generating the initial population is expressed as:

$$[P_{Low}, P_{Upper}] \tag{7.5}$$

where $P_{Upper} = \min\left\{\frac{I_1^{max}}{|g_{l,1}|^2}, \frac{I_2^{max}}{|g_{l,2}|^2}, \ldots, \frac{I_M^{max}}{|g_{l,M}|^2}, P_{High}\right\}, \forall l$

The new limits in (7.5) results in a smaller yet more effective interval for randomly generating the individuals. The smaller interval lets EAs to focus their exploration and exploitation on the feasible domain, rather than to handle an individual that lies above P_{Upper} and has to be repositioned by the Iterative Greedy Algorithm.

Although the initial population is ensured to satisfy the constraint *C2*, the population in the rest of EA iterations may violate the constraint. For instance, expression (3.2) for locating new food sources, in addition to line 5 of Table 3.3 and line 7 of Table 3.4 for ABC, and line 5 of Table 4.1 for the hybrid algorithm may cause a food source to lay out of the limits mentioned in (7.5). Consequently, some individuals from the population may require some modifications. The solver passes the population to the Iterative Greedy Algorithm that determines the relay assignment variables $\boldsymbol{\varepsilon}$, while it also ensures that all relays powers $\boldsymbol{p}^j$s and relay assignment variables $\boldsymbol{\varepsilon}$ satisfy the constraints of (7.4), and then passes the verified initial population to the next step to evaluate its fitness. Later on, in every iteration, after the population is modified by EA through its evolutionary operator, it will be passed to the Iterative Greedy algorithm for determining the relay assignment variables and constraints verification.

7.4.2. Iterative Greedy Algorithm

In each EA iteration, after the algorithm modifies the population through its specific evolutionary procedure (Point "B" in Figure 7.2), the solver has to determine $\boldsymbol{\varepsilon}$ and verify $\boldsymbol{\varepsilon}$ and $\boldsymbol{p}^j$ with the constraints in (7.4) before the population is passed towards the fitness function evaluation step (Point "C" in Figure 7.2). We propose an Iterative Greedy Algorithm to repair each EA individual $\boldsymbol{p}^j$ such that constraints *C2* and *C3* are satisfied, and to generate a feasible assignment variable $\boldsymbol{\varepsilon}$ for each individual heuristically that satisfies all constraints in (7.4). At the end of this procedure, the algorithm's output is an individual with feasible relays' power levels and the associated assignment variables $\boldsymbol{\varepsilon}$ (Point "C" in Figure 7.2), which it is ready to be passed to the fitness function evaluation process. This procedure has to be run for N times (number of individuals in the EA's population) to determine the relay assignment variable and ensure that all individuals satisfy the constraints.

Table 7.1 shows the pseudo-code of the iterative greedy algorithm. The algorithm runs the procedure in Table 7.1 for every individual $\boldsymbol{p}^j$ of the population in every EA iteration. This algorithm consists of two steps. In the first step, the algorithm greedily assigns relays to the secondary users based on the channel conditions. However, the assigned relays in this step may not satisfy all the constraints. In the second step, the algorithm verifies the assigned relays with the constraints and finalizes the power allocation to ensure that the interference constraint at the PUs is satisfied.

7.4.2.1. Step 1: Partial Relay Assignment

For describing the basic idea behind the proposed suboptimal algorithm, we view $|h_{s,l}|^2|h_{l,k}|^2$ (the product of the channel gain from the source to the l^{th} relay and the channel gain from the l^{th} relay to the k^{th} secondary user) as the profit from investing (assigning) the l^{th} relay to the k^{th} secondary user (because of the channel gain's positive effect on the throughput). Step 1 of the algorithm temporarily assigns each relay to the secondary users that return the maximum profit. Note that according to *C1* in problem (7.4), each relay can be assigned only to one secondary user. Mathematically, for each relay l, the algorithm temporarily assigns secondary users as follows:

Table 7.1. Iterative greedy relay assignment for each EA individual

Initialization:

1. $S(l) = 0, \forall l$
2. $C(k) = 0, \forall k$

Step 1:

3. **for** $l = 1, \dots, L$,
4. $S(l) = \arg\max_{k=1,\dots,K} \left|h_{l,k}\right|^2$,
5. end **for**

Step 2:

1. **for** $l = 1, \dots, L$
2. $p_l^j = \min\left\{\frac{I_1^{max}}{\left|g_{l,1}\right|^2}, \frac{I_2^{max}}{\left|g_{l,2}\right|^2}, \dots, \frac{I_M^{max}}{\left|g_{l,M}\right|^2}, p_l^j\right\}, \forall l$ (power of the l^{th} relay of the j^{th} individual)
3. end **for**,
4. **for** $k = 1, \dots, K$
5. $\mathcal{L}_k = \{l \mid S(l) = k\}$
6. **if** $\mathcal{L}_k \neq \emptyset$ **then**,
7. $flg = 0$,
8. **while** $flg = 0$,
9. **if** {constraint C2 is not satisfied with $\mathcal{L}_k$} **then**,
10. $\bar{l} \leftarrow$ find the relay that causes the highest interference,
11. $\mathcal{L}_k = \mathcal{L}_k \setminus \{\bar{l}\}$,
12. **else**
13. $flg = 1$,
14. $C(k) \leftarrow$ calculate capacity from (7.1) using $\mathcal{L}_k$,
15. end **if**
16. end **while**
17. end **if**
18. end **for**

$$S(l) = \arg\max_{k=1,\dots,K} \left|h_{s,l}\right|^2 \left|h_{l,k}\right|^2 \quad (7.6)$$

where S is an L-dimensional vector that stores this temporary assignment. The variable $h_{s,l}$ is not dependent on k; thus we can rewrite (7.6) as:

$$S(l) = \arg\max_{k=1,\dots,K} \left|h_{l,k}\right|^2 \quad (7.7)$$

At the end of Step 1, every relay is assigned to one secondary user. However, the relay powers, along with the temporarily relay assignment variables from (7.6), still need to satisfy the interference constraint at the PUs. In Step 2 of the algorithm, based on temporary relay assignment in Step 1, the algorithm performs a joint relay assignment and power allocation such that the constraints are satisfied at all PUs.

7.4.2.2. Step 2: Final Relay Assignment with Interference Constraint

In the second step, the algorithm performs a final assignment to ensure that the interference constraints at the PUs are all satisfied. Note that the relays' power levels randomly generated by the EA (Point "B" in Figure 7.2) can violate the constraint of the limited interference to the PUs, which is to be taken care of by Step 2 of the iterative greedy algorithm.

At the beginning of the second step, the algorithm repairs the power of any relay that violates the interference constraint. For this purpose, first the algorithm examines whether the transmission power of each relay l violates any interference constraint. We denote by p_l^j the power of the l^{th} relay in the j^{th} individual of the population. If p_l^j violates any of the interference constraint, then the algorithm performs the following adjustment:

$$p_l^j = \min\left\{\frac{I_1^{max}}{\left|g_{l,1}\right|^2}, \frac{I_2^{max}}{\left|g_{l,2}\right|^2}, \dots, \frac{I_M^{max}}{\left|g_{l,M}\right|^2}, p_l^j\right\}, \qquad \forall l \quad (7.7)$$

The algorithm then continues to iterate over all of the secondary users to complete the final assignment of the relays. During every iteration over secondary users, the algorithm

collects the set of relays that has been temporarily assigned to the k^{th} secondary user during Step 1 in the variable $\mathcal{L}_k$. Then it checks whether the relays in the set $\mathcal{L}_k$ satisfy the interference constraint. If the relays set $\mathcal{L}_k$ violate the interference constraint at any PU, the algorithm greedily removes the relay from the set $\mathcal{L}_k$that causes the maximum interference to the PUs. This removal process continues until $\mathcal{L}_k$ satisfies the interference constraint. The whole algorithm in Step 2 then continues to run until relays are assigned to all secondary users.

7.4.3. More Discussion on the Hybrid Solver

The cognitive radio discussed in the problem of this chapter operates in the underlay mode. According to [25], two shared-use models are introduced for Dynamic Spectrum Access: the spectrum overlay mode and the spectrum underlay mode. In spectrum overlay, first the secondary users sense the spectrum to find a spectrum hole (vacant frequency band). The secondary users transmit in these vacant frequency bands. Nonetheless, the secondary users should be aware that once PUs start to transmit over these bands, they have to stop utilizing them. In the spectrum underlay technique, the secondary users can transmit over the frequency band utilized by the PUs as long as they do not cause unacceptable interference to the PUs. One difference between these two schemes is that in the overlay mode, either PUs or secondary users are allowed to transmit over the spectrum, and when PUs are utilizing the bend, secondary users have to cease transmitting. However, it is possible that both PUs and secondary users transmit simultaneously over the same spectrum in the underlay mode as long as the interference threshold is not violated. Therefore, by utilizing the spectrum underlay mode in our problem, it is plausible to assume $I_{m,k}^{max} = I_m^{max}$; that is, the interference does not dependent on the secondary users' operation band.

As mentioned before, the Iterative Greedy Algorithm is heuristic in nature. Thus it has the issue of efficiency-complexity trade-off. In other words, the algorithm can be designed to allocate power to the relays more efficiently, but this effectiveness comes with a price of higher complexity. For instance, instead of removing the relay $\hat{l}$ in line 10 of Table 7.1, the algorithm could have reallocated the power by dividing the profit to cost ratio. In this scenario, although the algorithm's performance increases by better utilizing

the relays rather than to simply eliminate it, but we should consider the increase in algorithm's complexity after this modification. This division has to be run for $\hbar$ times in line 10 inside the while loop starting at line 8, where $0 \leq \hbar \leq K$. Then it should run for K times because of the **for** loop in line 4, and the whole Iterative Greedy Algorithm is run for $N.G$ times, where N is the population size and G is the maximum number of EA generations. Consequently, one modification adds $\hbar.K.N.G$ more calculation to the algorithm. However, because the complexity of the problem matters for us, we developed the algorithm with the less complex scheme proposed in Table 7.1.

7.5. Simulation Results

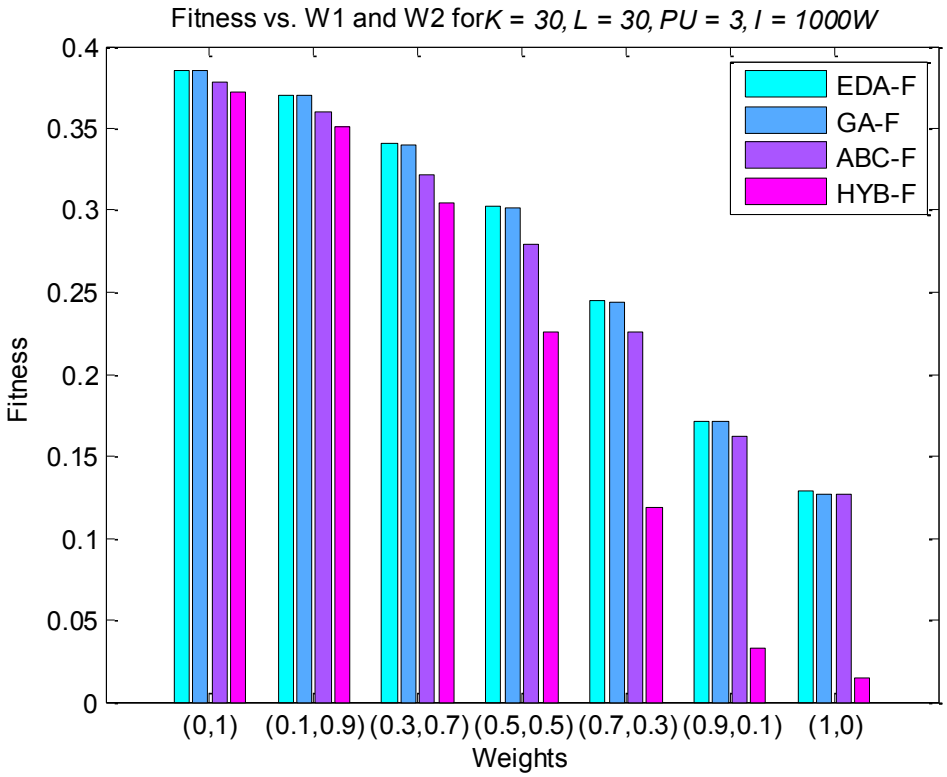
In this section we present the simulation results of EAs applied to a GCCRN MOO problem. In all simulations, the channel gains between source, relays, PUs and destinations have been randomly generated from independent complex Gaussian distribution. Each result is an average of 250 independent simulation runs. We compare the results of Hybrid EDA and ABC with EDA and the standard continuous GA [20]. All algorithms have the same settings as given in Table 7.2.

In Figure 7.3 and Figure 7.4, depicts the trade-off plots of fitness vs. different weights to the MOO objective function. These two figures demonstrate the effect of green communication for different values of weights w_1 and w_2. The results show that when w_2 is greater than w_1, there is more reduction in CO_2 emissions (percentage decrease in power). The reduction in CO_2 emissions comes at the cost of throughput reduction. The different weights settings are suitable for different geographical conditions and regulatory policies. The results also show that hybrid algorithm dominantly returns better results than other EAs. Moreover, because f_{CO_2} is larger than f_c, we observe as (w_1, w_2) moves towards, $(1,0)$, the fitness value decreases. Furthermore, we observe that in both figures, the difference between the EAs' results for (0,1) are much smaller than the EAs' results for (1,0). This observation shows that the major challenge between the EAs' performances is solving f_c, which is a much more complicated constraint optimization problem. We have also included the 95% confidence interval of the fitness results for the system models of these two figures. Table 7.3 and Table 7.4 contain the confidence intervals for different weights of Figure 7.3 and Figure 7.4, respectively.

Table 7.2. EA Settings for Implementation

GA			EDA			ABC	Hybrid	
P_{Xover}	P_{mut}	P_{sel}	P_{Xover}	P_{mut}	P_{sel}	$trial$	I	Migration type
0.99	0.8	0.5	0.1	-	0.5	1	1	Piecewise-Constant

a.



b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
30	3	300	10^3W	$10W$	*0 ~ 1*	*0 ~ 1*	24	30	250

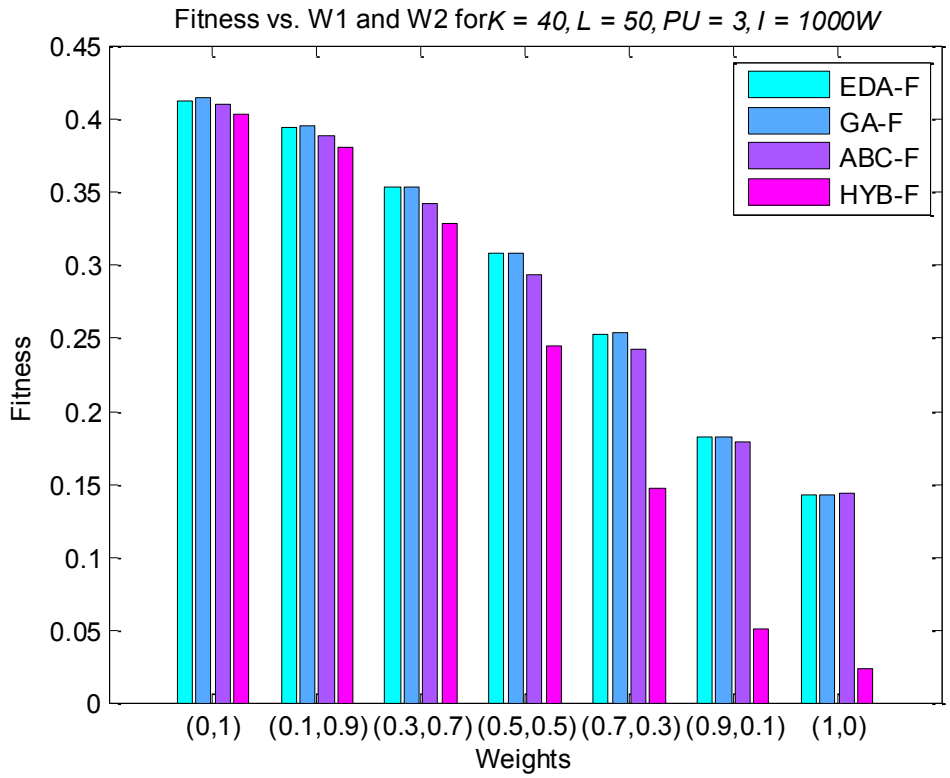
Figure 7.3. Fitness vs. different MOO power settings for K = 30

a. Fitness vs. different MOO power settings for $(K, M, L, I_m^{max}, p_l^{max}) = (30,3,30, 10^3W, 10W)$**,**
b. simulation parameters,

Table 7.3. 95% Confidence Interval of Figure 7.3 results

(w_1, w_2)		EDA	GA	ABC	Hybrid
$(0, 1)$	**Fitness**	**0.4053±0**	**0.4032±0.0004**	**0.3988±0.0037**	**0.399±0.0019**
	f_c	0.2079±0	0.1998±0.0017	0.1871±0.0378	0.1234±0.0204
	f_{co_2}	0.4053±0	0.4032±0.0004	0.3988±0.0037	0.4015±0.0137
$(0.1, 0.9)$	**Fitness**	**0.3859±0.0007**	**0.3803±0.0019**	**0.3804±0.0045**	**0.3698±0.0063**
	f_c	0.2414±0.0023	0.238±0.0016	0.2294±0.0007	0.1351±0.021
	f_{co_2}	0.4019±0.001	0.3961±0.0023	0.3972±0.0049	0.4123±0.0325
$(0.3, 0.7)$	**Fitness**	**0.3473±0**	**0.3473±0**	**0.3385±0.0027**	**0.3081±0.0128**
	f_c	0.3424±0	0.3424±0	0.3364±0.0019	0.2196±0.0331
	f_{co_2}	0.3494±0	0.3494±0	0.3394±0.0031	0.3549±0.0541
$(0.5, 0.5)$	**Fitness**	**0.2987±0.0014**	**0.3054±0**	**0.2678±0.0093**	**0.2278±0.0384**
	f_c	0.1959±0.012	0.1398±0	0.1781±0.0126	0.1304±0.0833
	f_{co_2}	0.4016±0.0148	0.4711±0	0.3576±0.0242	0.3873±0.0707
$(0.7, 0.3)$	**Fitness**	**0.2727±0**	**0.2685±0.0024**	**0.2444±0.0064**	**0.1786±0.0316**
	f_c	0.1953±0	0.1918±0.0241	0.1855±0.0035	0.1641±0.0653
	f_{co_2}	0.4534±0	0.4474±0.0628	0.3817±0.0156	0.4868±0.0276
$(0.9, 0.1)$	**Fitness**	**0.1955±0**	**0.1905±0.0015**	**0.1825±0.0046**	**0.1085±0.0538**
	f_c	0.1689±0	0.1543±0.0045	0.1544±0.0133	0.1264±0.0784
	f_{co_2}	0.4346±0	0.5162±0.0251	0.4352±0.0932	0.5326±0.0339
$(1, 0)$	**Fitness**	**0.1144±0.001**	**0.1145±0.0009**	**0.1143±0.0041**	**0.0251±0.0311**
	f_c	0.1144±0.001	0.1145±0.0009	0.1143±0.0041	0.0803±0.0738
	f_{co_2}	0.565±0.0057	0.5608±0.0102	0.5526±0.0415	0.5195±0.0337

a.



b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
40	3	50	10^3W	$10W$	*0 ~ 1*	*0 ~ 1*	24	30	500

Figure 7.4. Fitness vs. different MOO power settings for K = 40

a. Fitness vs. different MOO power settings for $(K, M, L, I_m^{max}, p_l^{max}) = (40,3,50, 10^3W, 10W)$
b. simulation parameters,

Table 7.4. 95% Confidence Interval of Figure 7.4 results

(w_1, w_2)		EDA	GA	ABC	Hybrid
$(0, 1)$	**Fitness**	**0.4254±0**	**0.4254±0**	**0.4225±0.0021**	**0.4208±0.0019**
	f_c	0.1939±0	0.1939±0	0.198±0.0026	0.1783±0.0088
	f_{co_2}	0.4254±0	0.4254±0	0.4225±0.0021	0.4247±0.0199
$(0.1, 0.9)$	**Fitness**	**0.4254±0**	**0.4254±0**	**0.4225±0.0021**	**0.4208±0.0019**
	f_c	0.1939±0	0.1939±0	0.198±0.0026	0.1783±0.0088
	f_{co_2}	0.4254±0	0.4254±0	0.4225±0.0021	0.4247±0.0199
$(0.3, 0.7)$	**Fitness**	**0.3344±0**	**0.3243±0.0022**	**0.331±0.0011**	**0.3038±0.0099**
	f_c	0.2278±0	0.2112±0.0035	0.2318±0.0017	0.1436±0.0265
	f_{co_2}	0.3801±0	0.3727±0.0016	0.3735±0.002	0.3761±0.025
$(0.5, 0.5)$	**Fitness**	**0.3168±0**	**0.3069±0.0021**	**0.2961±0.0063**	**0.271±0.016**
	f_c	0.1611±0	0.2041±0.0092	0.1888±0.0108	0.0716±0.0962
	f_{co_2}	0.4725±0	0.4097±0.0134	0.4035±0.0147	0.4844±0.0688
$(0.7, 0.3)$	**Fitness**	**0.2342±0**	**0.2298±0.0009**	**0.2342±0**	**0.1842±0.0196**
	f_c	0.1095±0	0.1361±0.0057	0.1095±0	0.167±0.0529
	f_{co_2}	0.5252±0	0.4484±0.0163	0.5252±0	0.4906±0.0196
$(0.9, 0.1)$	**Fitness**	**0.1678±0.004**	**0.1784±0.0026**	**0.1863±0.001**	**0.0756±0.0441**
	f_c	0.1282±0.0052	0.1373±0.0047	0.1536±0.0021	0.0476±0.0898
	f_{co_2}	0.5241±0.0063	0.5489±0.0167	0.4804±0.0284	0.5097±0.0219
$(1, 0)$	**Fitness**	**0.1364±0.0032**	**0.1414±0.0021**	**0.1504±0.0019**	**0.0688±0.029**
	f_c	0.1364±0.0032	0.1414±0.0021	0.1504±0.0019	0.157±0.0259
	f_{co_2}	0.5166±0.0164	0.6353±0.0088	0.555±0.0517	0.494±0.0389

Figure 7.5, Figure 7.6 and Figure 7.7 depict the performance result of fitness vs. number of secondary users for $(M, L, I_m^{max}, p_l^{max}, W_1, W_2) = (9{,}30{,}10\text{W}, 10\text{W}, 0.5\,, 0.5)$, $(2{,}20{,}1\text{W}, 10\text{W}, 0.5\,, 0.5)$, and $(2{,}30{,}1\text{W}, 10\text{W}, 0.5\,, 0.5)$, respectively. The hybrid algorithm returns lower fitness value compared with other EAs. We observe that there is an increase in the fitness value as the number of secondary users grows, because increasing the number of users increases the sum capacity

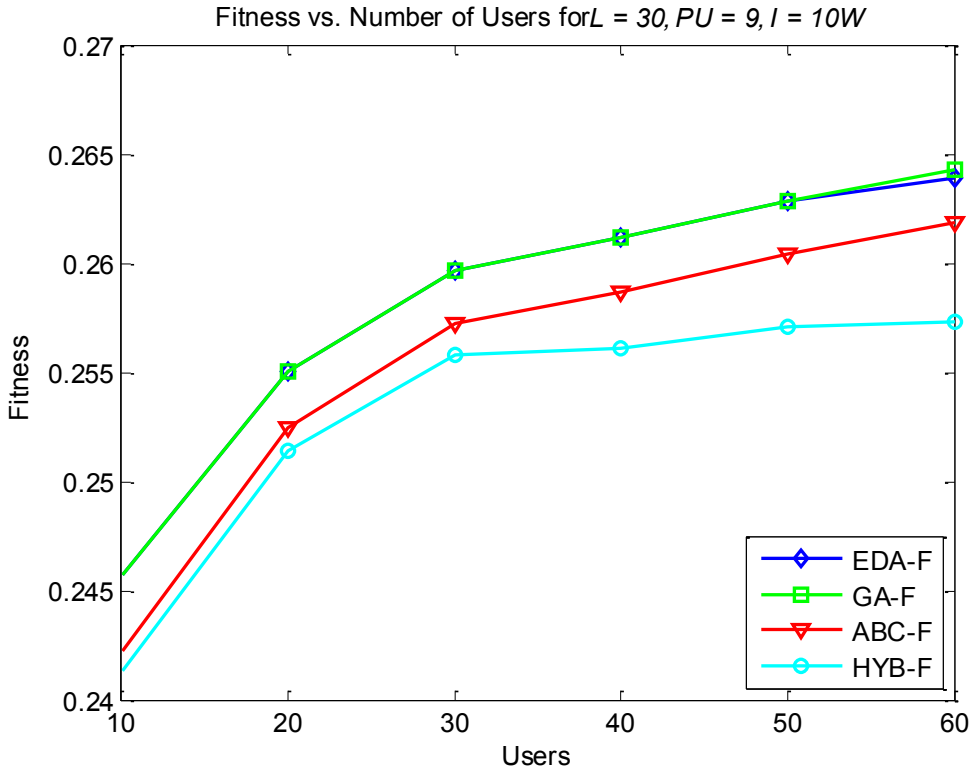
Figure 7.8, Figure 7.9 and Figure 7.10 illustrate the performance of fitness vs. the number of relays for $(K, M, I_m^{max}, p_l^{max}, W_1, W_2) = \left(30{,}5, 10^3 W, 10\text{W}, 0.5\ , 0.5\right)$, $(10{,}2, 10^4\text{W}, 10\text{W}, 0.5\ , 0.5)$ and $(10{,}2{,}10\text{W}, 10\text{W}, 0.5\ , 0.5)$, respectively. An increase in the number of relays results in the fitness growth. We observe that the first two figures have much larger I_m^{max} that have not yet affected by the sum capacity; whereas the non-smooth fitness plot in the last figure implies that the fitness result has been affected by the interference constraint that in fact has a lower I_m^{max}.

Figure 7.11 and Figure 7.12 demonstrate the performance results of fitness vs. number of Primary Users for $(K, L, I_m^{max}, p_l^{max}, W_1, W_2) = \left(50{,}40, 10^3 W, 10\text{W}, 0.5\ , 0.5\right)$ and $(60{,}40, 10^3\text{W}, 10\text{W}, 0.5\ , 0.5)$. We observe that these two figures have constant fitness values. The reason is that I_m^{max} is large enough (10^3) that even increasing the number of PUs does not cause the interference constraint to affect f_c.

Figure 7.13 and Figure 7.14 depicts the EAs' evolution as the algorithm iteration increases for $(K, M, L, I_m^{max}, p_l^{max}, W_1, W_2) = (40{,}7{,}40, 10^3\text{W}, 10\text{W}, 0.5\ , 0.5)$ and $(60{,}7{,}40{,}1\text{W}, 10\text{W}, 0.5\ , 0.5)$, and it contains the weighted single objective function F, along with the two objective functions f_c and f_{CO_2}. We observe that the hybrid algorithm dominantly outperforms other EAs, and as the number of iterations grows, the hybrid results improve even further. The reason is that other EAs cannot effectively exploit or explore their population to advance like the hybrid algorithm. This behavior shows how effectively the hybrid algorithm benefits from its both advantages, exploration and exploitation, to keep improving, while other EAs inconsiderably tend to progress any further.

Finally, Figure 7.15 contains 3D plots of EAs' advancement as the number of relays increases for $(K, M, I_m^{max}, p_l^{max}, W_1, W_2) = (60{,}5{,}40, 10^3\text{W}, 10\text{W}, 0.5\ , 0.5)$. We observe EDA starts with higher fitness value at the beginning. The reason is that this algorithm does not improve its results only after its second iteration. We further observe that the hybrid algorithm results decreases rapidly as the number of iterations grows, which was also observed in Figure 7.13 and Figure 7.14.

a.



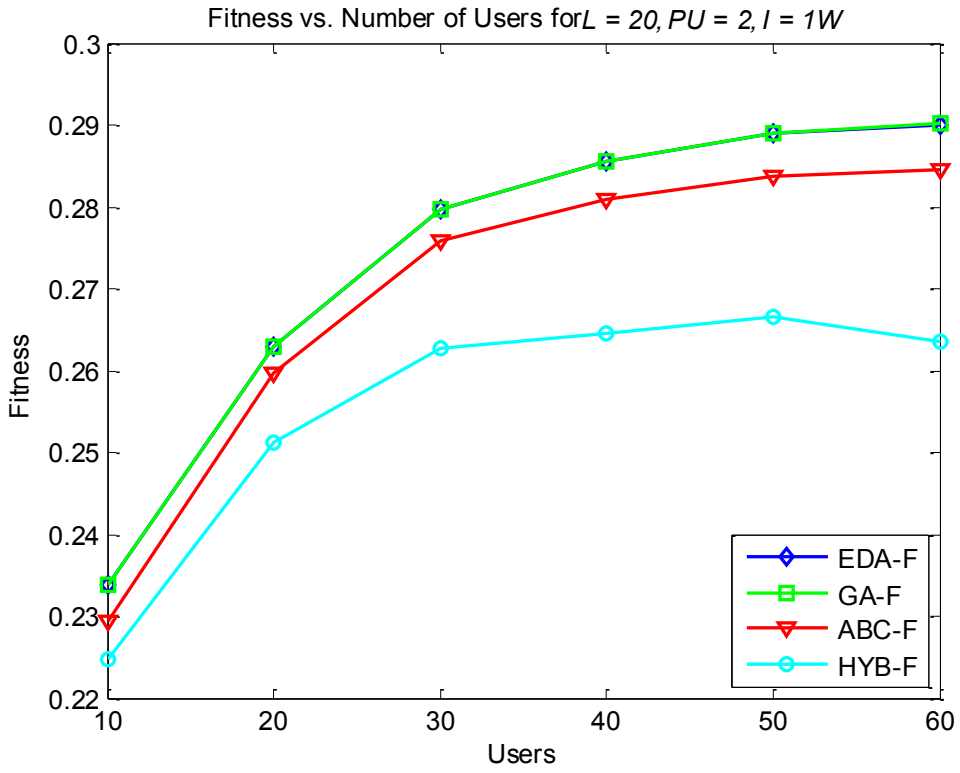
b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
10~60	9	30	$10W$	$10W$	0.5	0.5	24	24	250

Figure 7.5. Fitness vs. number of users for L = 30

a. Fitness vs. number of users for $(\boldsymbol{M}, \boldsymbol{L}, \boldsymbol{I_m^{max}}, \boldsymbol{p_l^{max}}, \boldsymbol{W_1}, \boldsymbol{W_2}) = (\mathbf{9}, \mathbf{30}, \mathbf{10W}, \mathbf{10W}, \mathbf{0.5}, \mathbf{0.5})$,
b. simulation parameters,

a.



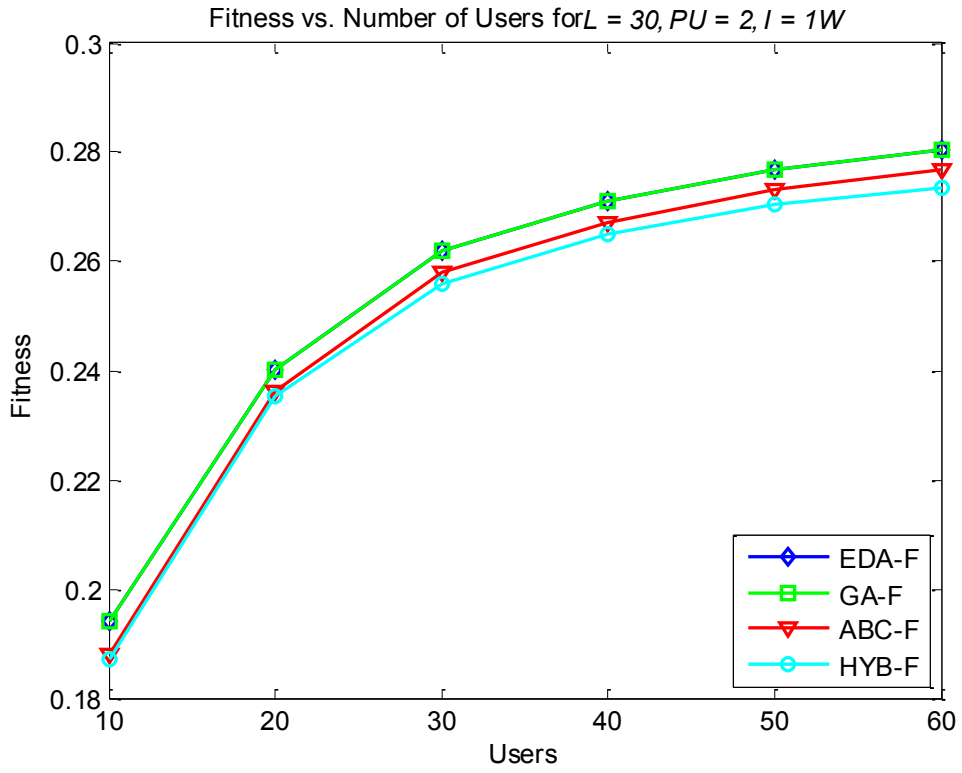
b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
10~60	2	20	$1W$	$10W$	0.5	0.5	24	24	250

Figure 7.6. Fitness vs. number of users for L = 20

a. Fitness vs. number of users for $(M, L, I_m^{max}, p_l^{max}, W_1, W_2) = (2,20,1W, 10W, 0.5, 0.5)$,
b. simulation parameters,

a.



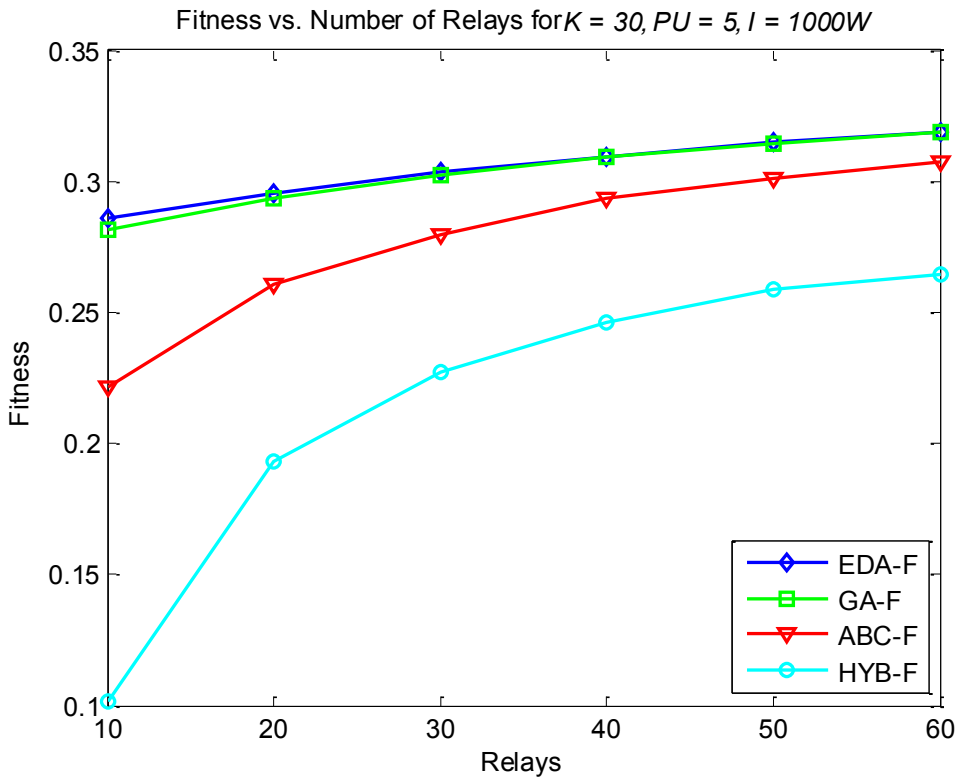
b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
10~60	2	30	$1W$	$10W$	0.5	0.5	24	30	500

Figure 7.7. Fitness vs. number of users for L = 30

a. Fitness vs. number of users for $(M, L, I_m^{max}, p_l^{max}, W_1, W_2) = (2,30,1W, 10W, 0.5, 0.5)$,
b. simulation parameters,

a.



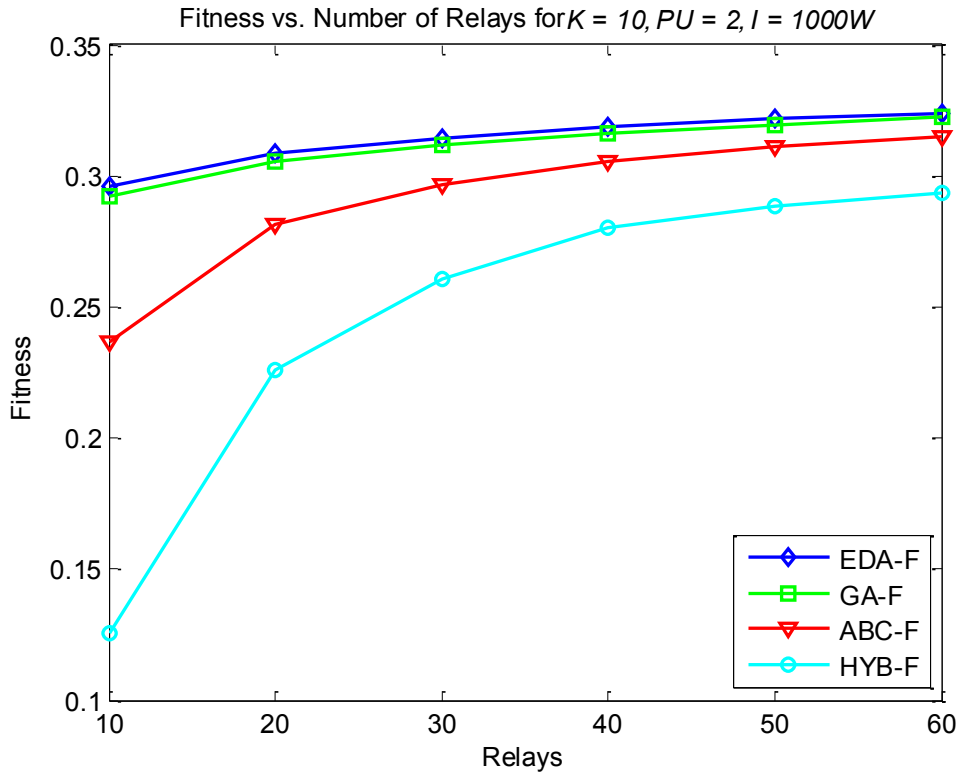
b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
30	5	10~60	10^3W	$10W$	0.5	0.5	24	24	500

Figure 7.8. Fitness vs. number of relays for K = 30

a. Fitness vs. number of relays for $(\boldsymbol{K, M, I_m^{max}, p_l^{max}, W_1, W_2}) = \boldsymbol{\left(30, 5, 10^3 W, 10W, 0.5, 0.5\right)}$,
b. simulation parameters,

a.



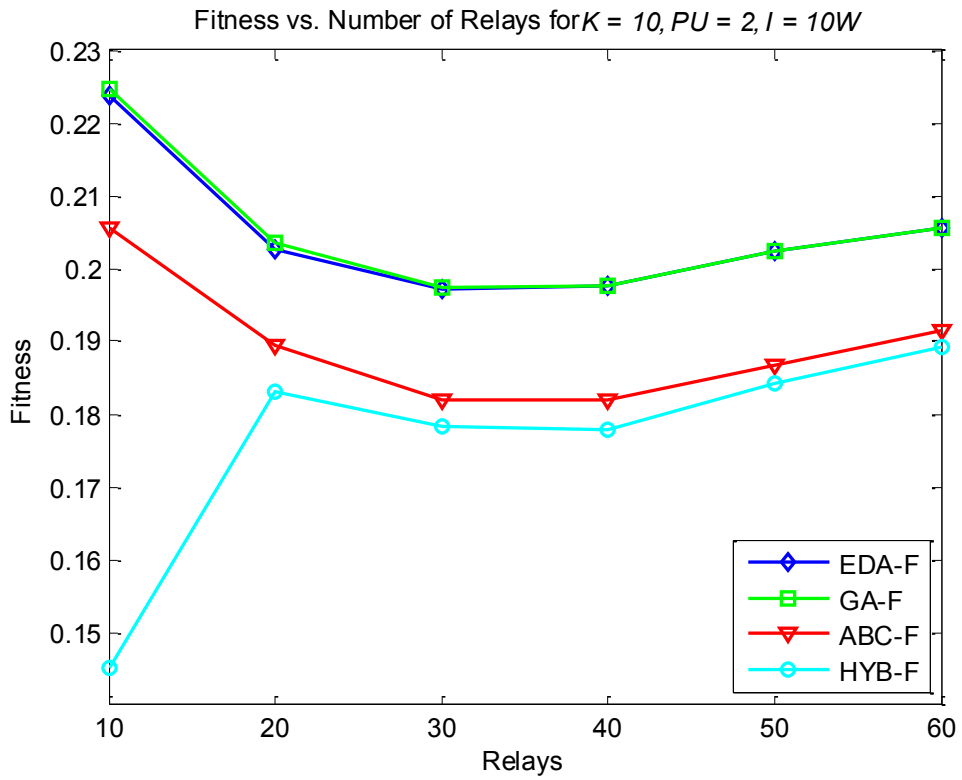
b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
10	2	10~60	10^4W	$10W$	0.5	0.5	24	24	400

Figure 7.9. Fitness vs. number of relays for K = 10

a. Fitness vs. number of relays for $(K, M, I_m^{max}, p_l^{max}, W_1, W_2)$ $(10,2, 10^4W, 10W, 0.5, 0.5)$,
b. simulation parameters,

a.



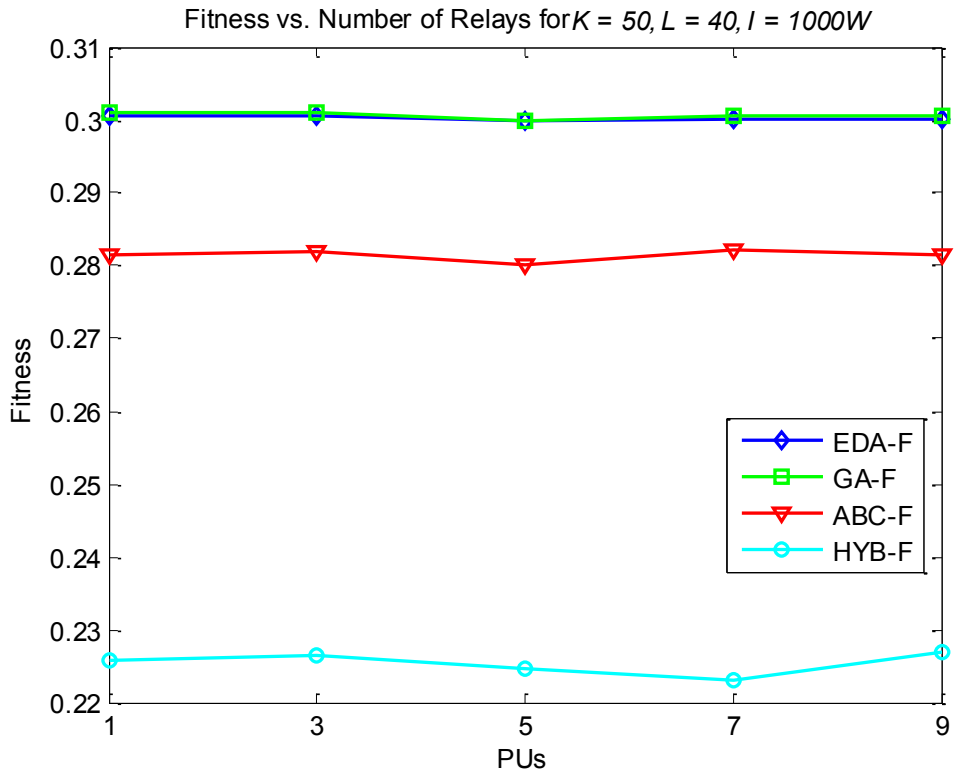
b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
10	2	10~60	$10W$	$10W$	0.5	0.5	24	24	250

Figure 7.10. Fitness vs. number of relays for K = 10

a. Fitness vs. number of relays for $(K, M, I_m^{max}, p_l^{max}, W_1, W_2)$ $(10,2,10W, 10W, 0.5, 0.5)$,
b. simulation parameters,

a.



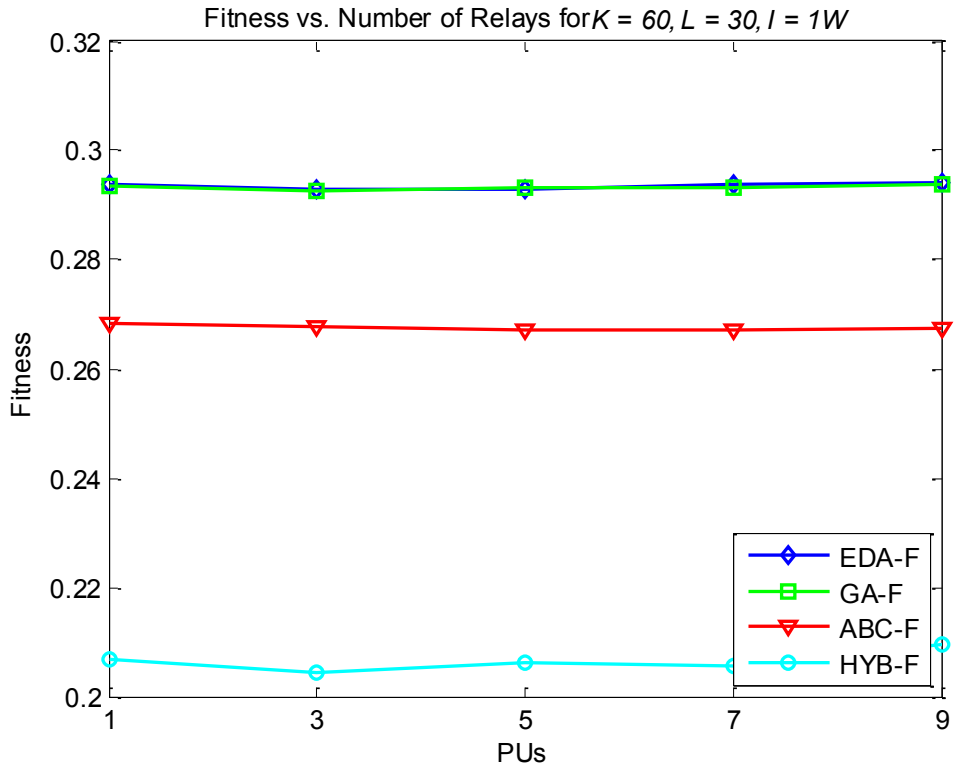
b.

b. System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
50	1~9	40	10^3W	$10W$	0.5	0.5	24	24	500

Figure 7.11. Fitness vs. number of primary users for K = 50

a. Fitness vs. number of primary users for $(K, L, I_m^{max}, p_l^{max}, W_1, W_2) = (50,40, 10^3W, 10W, 0.5, 0.5)$,
b. simulation parameters,

a.



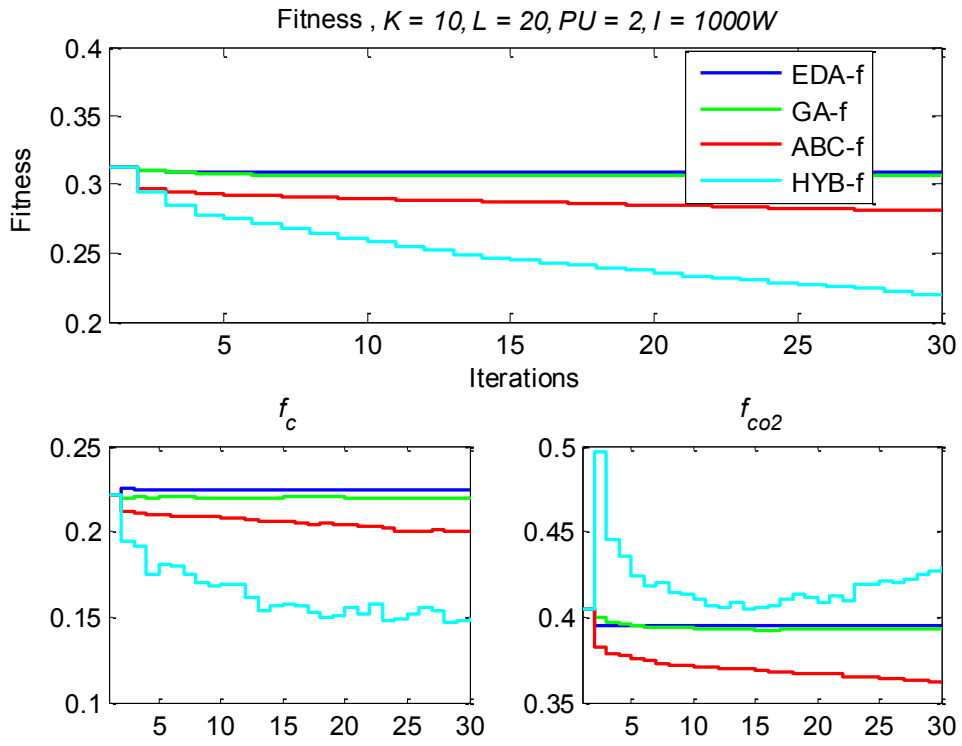
b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
60	1~9	30	$1W$	$10W$	0.5	0.5	24	24	500

Figure 7.12. Fitness vs. number of primary users for K = 60

a. Fitness vs. number of primary users for $(K, L, I_m^{max}, p_l^{max}, W_1, W_2) = (60,40, 10^3W, 10W, 0.5, 0.5)$,
b. simulation parameters,

a.



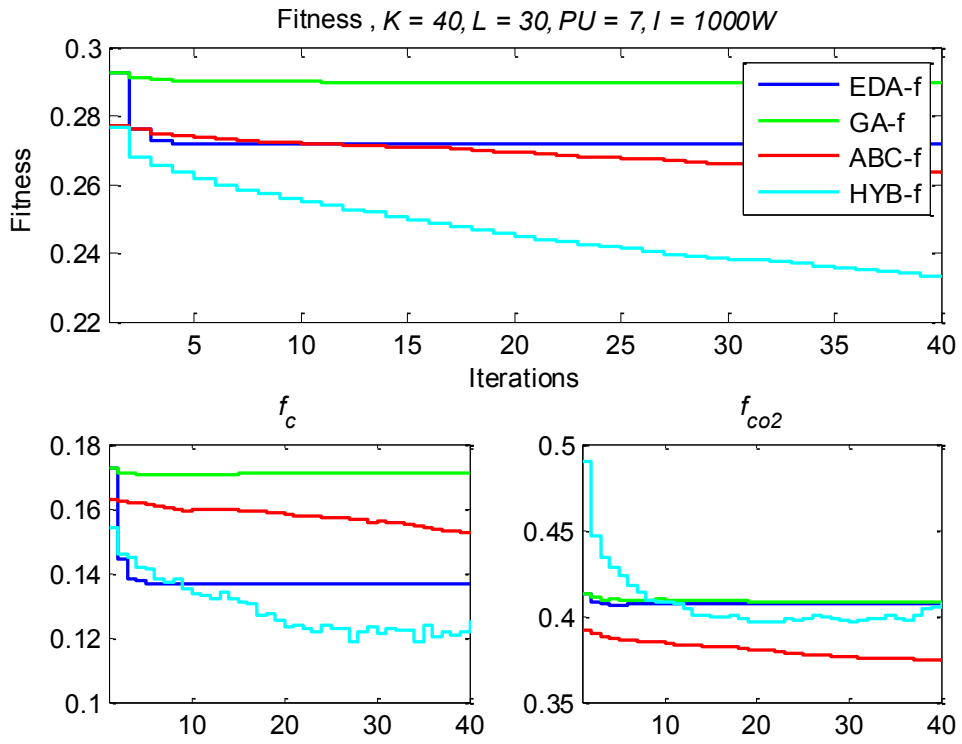
b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
10	2	20	10^3W	$10W$	0.5	0.5	24	30	400

Figure 7.13: Fitness vs. algorithms' iterations for K = 10

a. Fitness vs. EAs' iteration for $(K, M, L, I_m^{max}, p_l^{max}, W_1, W_2) = (10,2,20, 10^3W, 10W, 0.5, 0.5)$,
b. simulation parameters,

a.

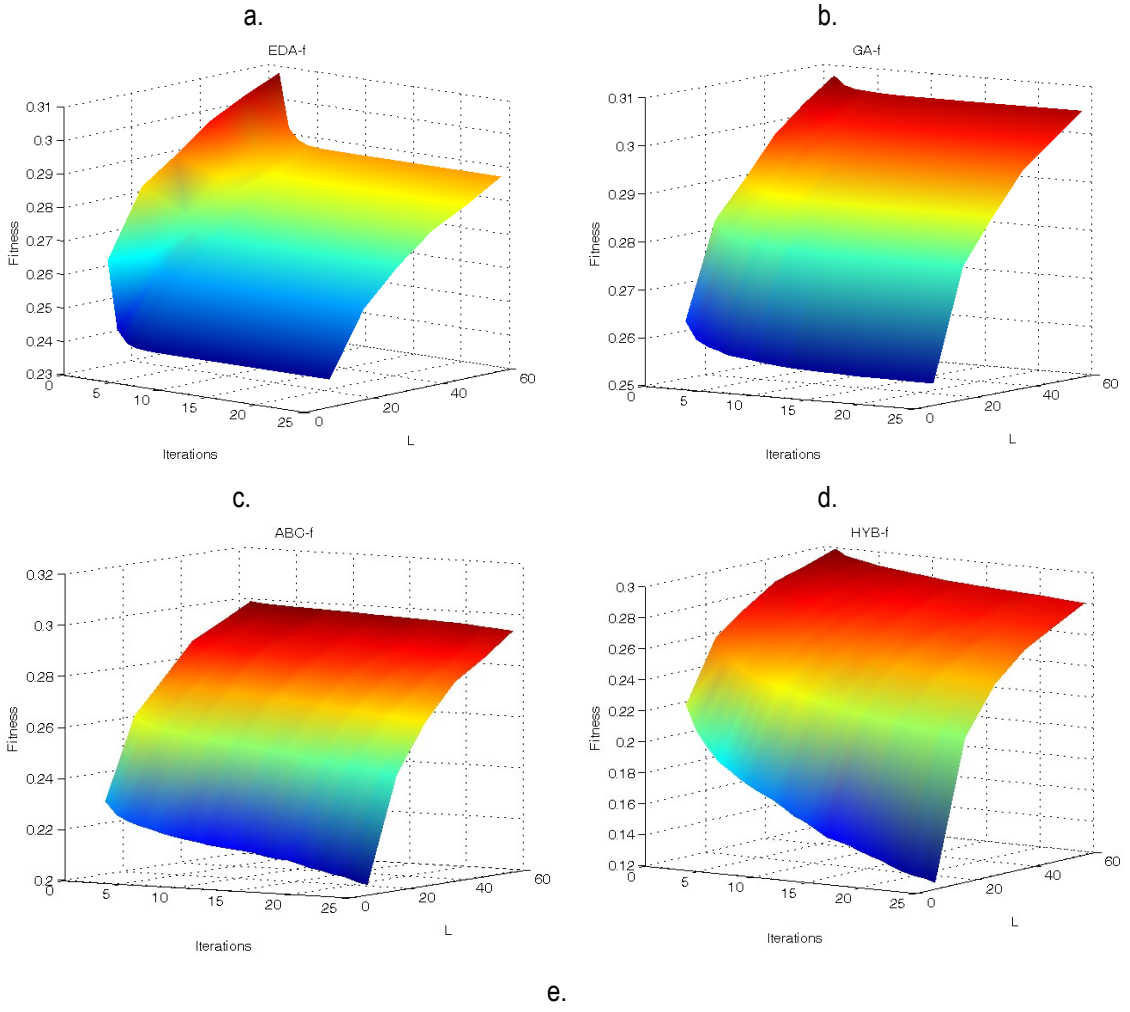


b.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
60	7	40	$1W$	$10W$	0.5	0.5	24	30	400

Figure 7.14. Fitness vs. algorithms' iterations for K = 60

a. Fitness vs. EAs' iteration for $(K, M, L, I_m^{max}, p_l^{max}, W_1, W_2)$ $(60,7,40,1W, 10W, 0.5, 0.5)$,
b. simulation parameters,



e.

System Parameters					Weights		Common EA parameters		Number of Simulation runs
K	M	L	I_m^{max}	p_l^{max}	W_1	W_2	Generation	Pop size	
60	5	10~60	10^3W	$10W$	0.5	0.5	24	30	500

Figure 7.15. Fitness vs. number of relays and algorithms' iterations for K = 60

a. EDA, b. GA, c. ABC, d. Hybrid algorithm,
e. system parameters for $(K, M, I_m^{max}, p_l^{max}, W_1, W_2) = (60,5,40, 10^3W, 10W, 0.5, 0.5)$

7.6. Conclusion

In this chapter, we presented a multi-objective framework for green resource allocation in the multiuser cognitive radio network. We present the constrained optimization formulation of the relay assisted cognitive radio system. Our formulation includes effect transmission power on CO_2 emission, which is a multi-objective optimization in nature. We approached this problem by applying the weighted sum method, which results in a non-convex mixed integer non-linear programming problem. We proposed a hybrid continuous evolutionary scheme comprising an EA and a greedy algorithm to solve this optimization problem. We apply four different EAs (GA, EDA, ABC and hybrid), and the results demonstrate that in all combinations of system parameters and weight values the hybrid algorithm outperforms other EAs. The simple underlying concept and ease of implementation of our proposed algorithm make it a suitable candidate for green resource allocation.

This chapter presented a simple optimization problem that takes into account the effects of communication resource allocation on the environment. We believe that the more system optimization models that take into account the system's effect on the environment will be developed and enhanced. The results of this chapter indicate that our evolutionary algorithms proposed may be useful for various continuous multi-objective optimization problems for green communication.

The future extensions of this research include considering different constraints on the PUs for different frequency bands (different $I_{m,k}^{max}$), multiple source, other signal transmitting scenarios, relaying strategies (compress and forward), and other more complex or realistic system models.

References

[1] Global e-Sustainability Initiative (GeSI), *SMART 2020: Enabling the low carbon economy in the information age*, available at www.gesi.org/Initiatives/ClimateChange/tabid/71/Default.aspx (Last accessed July 2012).

[2] International Telecommunication Union (ITU), *Report on Climate Change,* Oct. 2008.

[3] IEA report, *CO2 Emissions from Fuel Combustion 2010 – Highlights, http://www.iea.org/co2highlights/co2highlights.pdf*

[4] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, , "Energy Efficiency in the Future Internet: A Survey of Existing Approaches and Trends in Energy-Aware Fixed Network Infrastructures*," IEEE Communications Surveys & Tutorials*, (Accepted), (http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5522467&isnumber=5451756)

[5] B. Arnaud, "ICT and Climate Change: A Foundation for Innovation in Canada", CANAIR Inc., available at https://www-950.ibm.com/events/wwe/ca/canada.nsf/vLookupPDFs/Bil_Arnaud/$file/Bil_Arnaud.pdf (last accessed July 2012)

[6] G. Koutitas, "Green Network Planning of Single Frequency Networks," *IEEE Transactions on Broadcasting,* vol.56, no.4, pp.541-550, Dec. 2010.

[7] G. Koutitas, P. Demestichas, "A Review of Energy Efficiency in Telecommunication Networks," *Journal of Telecommunication Forum (TELFOR),* 2010.

[8] K. Li, "Mobile Communications 2008: Green Thinking Beyond TCO Consideration," *In-Stat White paper*, May 2008.

[9] L. Herault,E. C. Strinati ,O. Blume ,D. Zeller, Muhammad A. Imran, R. Tafazolli, Y. Jading ,J. Lundsjö and Michael Meyer, "Green Communications: a Global Environmental Challenge," In *Proceeding of 12th International Symposium on Wireless Personal Multimedia Communications*, 2009.

[10] W. Vereecken, W. V. Heddeghem, D. Colle, M. Pickavet, and P. Demeester, "Overall ICT footprint and green communication technologies," *In Proceedings of 4th IEEE International Symposium on Communications, Control and Signal Processing (ISCCSP),* 2010.

[11] D. Grace, J. Chen, T. Jiang, and P.D. Mitchell, "Using cognitive radio to deliver Green communications," In *Proceedings of 4th IEEE International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM),* 2009.

[12] R.T. Marler and J.S. Arora, “Survey of multi-objective optimization methods for engineering,” *Structural and Multidisciplinary Optimization*, vol. 26, no.6, pp. 369-395, April 2004.

[13] M. Elmusrati, H. El-Sallabi, and H. Koivo, “ Applications of multi-objective techniques in radio resource scheduling of cellular communication systems,” *IEEE Transaction of Wireless Communication,* vol. 7, no. 1,pp. 343-353 Jan. 2008.

[14] K. Deb, *Multi-objective optimization using evolutionary algorithms*, Wiley*,* New York, 2001.

[15] T. Newman, R. Rajbanshi, A. Wyglinski, J. Evans, and G. Minden, “Population Adaptation for Genetic Algorithm-based Cognitive Radios,” *ACM/Springer Mobile Ad Hoc Networks -- Special Issue on Cognitive Radio Oriented Wireless Networks and Communications,* vol. 13, no. 5, pp. 442-451, 2008.

[16] J. N. Laneman, D. N. C. Tse, and G. W. Wornell, “Cooperative Diversity in Wireless Networks: Efficient Protocols and Outage Behavior,” *IEEE Trans. Inform. Theory,* vol. 50, no. 12, pp. 3062-3080, Dec. 2004.

[17] I. Maric and R. D. Yates, “Bandwidth and Power Allocation for Cooperative Strategies in Gaussian Relay Networks,” In *Proceedings of 38th Asilomar Conference on Signals, Systems and Computers,* Pacific Grove, CA, Nov. 2004.

[18] A.E. Eiben and J.E. Smith, *Introduction to Evolutionary Computing*, Springer Verlag, 2003.

[19] P. Larrañaga and J. A Lozano, “Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation,” *Kluwer Academic Publishers*, 2001.

[20] R.L. Haupt and S.E. Haupt, *Practical Genetic Algorithms,* Wiley-Interscience, 2004.

[21] R. H. Katz, “Tech Titans Building Boom,” *IEEE Spectrum*, vol. 46,pp. 40–54, Feb. 2009.

[22] R. Barga, “Cloud Computing – A Microsoft Research Perspective.” *Keynote Speech at IEEE P2P 2009*, Sep. 2009.

[23] F. H. P. Fitzek and M. D. Katz, “Cooperation in wireless networks: principles and applications; real egoistic behavior is to cooperate,” *Springer–Verlag,* New York, 2006.

[24] COST (European Cooperation in Science and Technology), “Cooperative Radio Communications for Green Smart Environments”, *ICT COST Action IC1004*, available at: http://www.cost.eu/domains_actions/ict/Actions/IC1004, (last accessed July 2012).

[25] M. Naeem, “Computationally Efficient Algorithms for Resource Allocation in Cognitive Radio and Green Communication Systems”, *PhD thesis*, Simon Fraser University, Summer 2011.

Appendix.

In this section, we show that the following function is an increasing function of p_l.

$$f(p_l) = \log\left[1 + \frac{P_s^k}{N}.\frac{\sum_{l=1}^{L}|h_{s,l}|^2 \sum_{l=1}^{L}(|h_{l,k}|\beta_l)^2 p_l}{1 + \sum_{l=1}^{L}(\beta_l|h_{l,k}|\sqrt{p_l})^2}\right] \quad \text{A.1}$$

The $\log(\ )$ function is a monotonically increasing function; so proving that its argument is increasing with p_l is enough for the objective function to be monotonically increasing with p_l. Knowing that $\frac{P_s^k}{N}$ is non-negative, it would be enough to show that

$$\frac{\sum_{l=1}^{L}|h_{s,l}|^2 \sum_{l=1}^{L}(|h_{l,k}|\beta_l)^2 p_l}{1 + \sum_{l=1}^{L}(\beta_l|h_{l,k}|\sqrt{p_l})^2} \quad \text{A.2}$$

is increasing with p_l:

$$\begin{aligned}
&\frac{\sum_{l=1}^{L}|h_{s,l}|^2 \sum_{l=1}^{L}(|h_{l,k}|\beta_l)^2 p_l}{1 + \sum_{l=1}^{L}(\beta_l|h_{l,k}|\sqrt{p_l})^2} \quad \text{A.3}\\
&= \frac{\left(\sum_{l=1}^{L}|h_{s,l}|^2\right)\left(\sum_{l=1}^{L}(|h_{l,k}|\beta_l)^2 p_l\right) + \sum_{l=1}^{L}|h_{s,l}|^2 - \sum_{l=1}^{L}|h_{s,l}|^2}{1 + \sum_{l=1}^{L}(\beta_l|h_{l,k}|\sqrt{p_l})^2}\\
&= \left(\sum_{l=1}^{L}|h_{s,l}|^2\right).\left(1 - \frac{1}{1 + \sum_{l=1}^{L}(\beta_l|h_{l,k}|\sqrt{p_l})^2}\right)
\end{aligned}$$

The left summation is non-negative, and so is the summation in the denominator. Therefore, this expression is increasing with p_l, and as mentioned earlier, the $\log(\ )$ function is monotonically increasing. As a result, the function $f(p_l)$ in (A.1) is an increasing function. ■