Tabu Search vs. Bio-inspired Algorithms for Antenna Selection in Spatially Correlated Massive MIMO Uplink Channels

Zaid Abdullah, Charalampos C. Tsimenidis and Martin Johnston
School of Electrical and Electronic Engineering
Newcastle University
Newcastle Upon Tyne, NE1 7RU, England, UK
Emails: {z.a.abdullah, charalampos.tsimenidis, martin.johnston}@newcastle.ac.uk

***Abstract*—Massive Multiple Input Multiple Output (MIMO) systems can significantly improve the system performance and capacity by using a large number of antenna elements at the base station (BS). To reduce the system complexity and hardware cost, low complexity antenna selection techniques can be used to choose the best antenna subset while keeping the system performance at a certain required level. In this paper, Tabu Search (TS) and three bio-inspired optimization algorithms were used for antenna selection in Massive MIMO systems. The three bio-inspired algorithms were: Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and Artificial Bee Colony (ABC). Simulations showed promising results for the TS by achieving higher capacity with GA than PSO and ABC, and much shorter CPU time than any of the bio-inspired techniques.**

***Index Terms*—Massive MIMO, Antenna selection, Bio-inspired algorithms, Particle Swarm Optimization (PSO), Genetic algorithm (GA), Artificial Bee colony (ABC), Tabu search (TS).**

I. Introduction

Massive Multiple Input Multiple Output (MIMO) systems are a state of the art research area in wireless communications, and have received much attention over the last few years [1] [2]. Thanks to the great advantages of Massive MIMO and improved performance, they have been considered as a potential technology for fifth generation wireless communications systems (5G) [3]. Massive MIMO refers to a system where tens to hundreds of antennas are used at the Base Station (BS). To obtain the full advantage of using Massive MIMO, every antenna should be associated with its own Radio Frequency (RF) chain. However, this increases the system complexity, power consumption and hardware cost [4], since every RF chain consists of low noise amplifier, mixer, and Analog to Digital Converter (ADC) [5]. Antenna selection techniques can be used to reduce the system cost, complexity and power consumption. To achieve maximum capacity performance, Exhaustive Search (ES) over all possible subsets is required, which can be applied for a conventional MIMO system but might be infeasible approach for a system with hundreds of antennas. Recently, a considerable amount of work has been published on low complexity antenna selection techniques. In this paper, Tabu Search (TS) algorithm and three bio-inspired algorithms were implemented to tackle the antenna selection problem. TS and bio-inspired optimization techniques are commonly used in many different engineering applications and are known for their low complexity, while at the same time finding near optimal solution for any certain optimization problem.

Particle Swarm Optimization (PSO) algorithm was first developed by [6], and is a class of evolutionary algorithms (EAs) based on the intelligent behavior of biological organisms. The term “swarm” refers to a collection of interacting agents. For example, a flock of birds can be thought of as a swarm whose individual agents are birds, or a crowd is a swarm whose agents are people, and so forth [7].

Genetic Algorithms (GAs) have been used since the 1950s. One of the first people who worked on these algorithms and also had the most influence on this field than any other was John Holland of the University of Michigan [7]. Holland represented GA as a method for moving from a certain population of “chromosomes” to another population by using genetics-inspired operations such as $Crossover$, $Reproduction$, and $Mutation$ [8].

Artificial Bee Colony (ABC) algorithm is among the recent EAs, and was developed by [9] to tackle optimization problems based on the intelligent behavior of honey bees on finding food sources. In ABC algorithm, the colony of artificial bees consists of three different types of bees, and they are used to search for the best solution. These bees are: Employed, Onlooker, and Scout bees [10].

Metaheuristic search methods such as greedy [11] and tabu search [12] algorithms are known as low complexity optimization techniques. However, tabu search outperforms the greedy approach by using the memory to avoid revisiting the previous moves (or solutions) to ensure an efficient search of the neighborhood. A special matrix, called Tabu matrix, is used to save the previous visited solutions in the neighborhood, and forbid using them for a certain number of upcoming iterations.

Our contribution in this paper is that we developed the ABC as well as the TS algorithms for antenna selection in massive MIMO systems, and compared their capacity performance and complexity with the well known PSO and GA algorithms.

978-0-9928-6265-7/16/$31.00

TS technique outperformed all the bio-inspired algorithms by requiring a much lower complexity, and in the same time achieving higher capacity with GA compared to PSO and ABC algorithms.

II. System Model

Consider an uplink MIMO system as depicted in Fig. 1 with N_r receive antennas at the base station and N_t transmit antennas ($N_r \gg N_t$). This system can be represented by the following equation

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{x} \in \mathbb{C}^{N_t \times 1}$ is the transmitted signal vector, $\mathbf{H} \in \mathbb{C}^{N_r \times N_t}$ is the channel matrix, $\mathbf{n} \in \mathbb{C}^{N_r \times 1}$ is the additive Gaussian noise with zero mean and variance of σ^2, and $\mathbf{y} \in \mathbb{C}^{N_r \times 1}$ is the received signal vector. Throughout this work, we consider spatial correlation between the antennas at the BS.

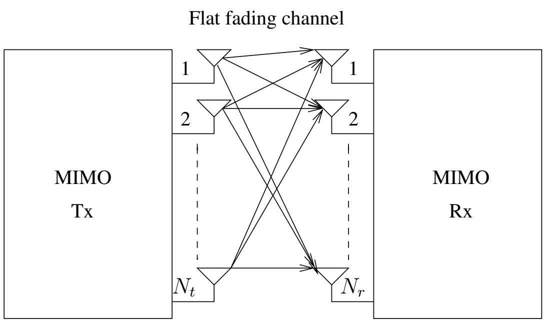

Fig. 1. Block diagram of an uplink MIMO system

A. Spatial correlation channel model

The correlated channel matrix $\mathbf{H}$ in (1) can be described using the Kronecker model as follows [13]

$$\mathbf{H} = \mathbf{R}_R^{1/2}\mathbf{G}\ \mathbf{R}_T^{1/2}, \tag{2}$$

where $\mathbf{G} \in \mathbb{C}^{N_r \times N_t}$ is a Gaussian matrix, with coefficients assumed to be independent and identically distributed (i.i.d.), with zero mean and unit variance. $\mathbf{R}_R$ and $\mathbf{R}_T$ are the receive and transmit correlation matrices, respectively. It should be clarified that the operator $(\ .)^{1/2}$ in (2) represents the Hermitian square root of a matrix. In this paper, we are considering correlation among antennas at the BS only, so the spatially correlated channel matrix can be given as

$$\mathbf{H} = \mathbf{R}_R^{1/2}\mathbf{G}. \tag{3}$$

The model of the $N_r \times N_r$ correlation matrix was assumed to have exponential correlation structure, which is a common model and can effectively measure the level of spatial correlation [13]. In this model, the correlation matrix can be implemented using only one coefficient $\phi \in \mathbb{C}$ with $|\phi| \leq 1$ as follows

$$R_{ij} = \begin{cases} \phi^{|j-i|} & , i \leq j \\ (\phi^{|j-i|})^* & , i > j, \end{cases} \tag{4}$$

where R_{ij} is the correlation between the i^{th} and j^{th} receive antennas, and $|\ .\ |$ is the absolute value operator.

III. Antenna Selection Problem Formulation

We consider a BS with massive number of antenna elements ($N_r \geq 100$), and choosing the best subset of these antennas to maximize the system capacity. For a MIMO system, the capacity can be given using the following equation

$$\mathbf{C} = \log_2 \det(\mathbf{I}_{N_r} + \frac{\rho}{N_t}\ \mathbf{H}\mathbf{H}^H), \tag{5}$$

where $\mathbf{I}_{N_r}$ is the $N_r \times N_r$ identity matrix, ρ is the signal to noise ratio, and $\mathbf{H}^H$ is the Hermitian (conjugate transpose) of the channel matrix.
Out of the available N_r antennas at the BS, we employ the optimization algorithms to choose the best N_s antennas that can maximize the capacity.

For simplicity, we will define the antenna selection operator as

$$\mathbf{s} = [s_1, s_2, ...s_{N_r}], \tag{6}$$

where

$$s_i = \begin{cases} 1 & \text{if the antenna is selected} \\ 0 & \text{Otherwise .} \end{cases} \tag{7}$$

At first, $\mathbf{s}$ is initialized with zeros, and once the optimization algorithm choose the best antenna subset, the location of these antennas will become 1s, while the rest of the elements will remain 0s. The optimized capacity can be then calculated as

$$\mathbf{C} = \log_2 \det(\mathbf{I}_{N_r} + \frac{\rho}{N_t}\ .\ \mathrm{diag}(\mathbf{s})\ .\ \mathbf{H}\mathbf{H}^H), \tag{8}$$

where $\mathrm{diag}(\mathbf{s})$ is an $N_r \times N_r$ diagonal matrix with $\mathbf{s}$ is its diagonal entry.

A. PSO algorithm for antenna selection

At first, a certain number of particles are generated randomly. Each particle can be represented as a vector of length N_r with N_s number of 1s located randomly along the vector. The capacity represented by the fitness value is measured for each particle. Additionally, the velocity is calculated, which directs the particle to fly towards the best solution. In PSO, every particle is influenced by its neighbors (called *local best*) as well as by the best particle among the group (called *global best*). The velocity of the particles is given as

$$\begin{aligned} v_i(t) = v_i(t-1) + \ & rand_1 \times k_1(p_{li} - s_i(t-1)) \\ & + \ rand_2 \times k_2(p_{gi} - s_i(t-1), \end{aligned} \tag{9}$$

$$f_i(t) = f_i(t-1) + v_i(t), \tag{10}$$

where $v_i(t)$ represents the velocity of the current iteration for the i^{th} antenna, and $v(t-1)$ is the velocity of the previous iteration. $rand_1$ and $rand_2$ are random numbers drawn from a uniform distribution between 0 and 1. k_1 and k_2 are weighting factors with arbitrary values, they are assumed to have a value of 2 in our simulations. p_{li} represents the local best solution for the i^{th} antenna depending on the two neighbors of the current particle. In our simulations, the first and last particles were assumed to be connected, i.e. the neighbors of the first particle are the second and last particle of the population. Moreover, the neighbors of the last particle are the first and the second last particle in the population. p_{gi} is the global best solution among the whole population for the current antenna, and finally, $\mathbf{f}$ is a vector initialized with zeros.

After each iteration, the global best and the local best solutions are updated before the next iteration starts. Furthermore, at the end of the iterations, the N_s maximum values of $\mathbf{f}$ will be chosen as the surviving antennas, while the rest of antennas will be ignored.

B. GA algorithm for antenna selection

At the beginning, a certain number of chromosomes are generated randomly. Every chromosome can be represented as a vector of “genes” (in this case bits, 0s or 1s), and the number of 1s in each chromosome equal to N_s. The fitness value for each chromosome will then be calculated and the best K chromosomes will be chosen to a mating pool for the reproduction process.

1) Reproduction process: In the reproduction process, the best K chromosomes will be paired off randomly into pairs of chromosomes, these chromosomes will then go through certain operations to produce a new population of chromosomes. In our simulations, K was equal to half the number of chromosomes.

2) Crossover process: In this process, a mask of length N_r is generated randomly with values of 0s and 1s, where the probability of each bit being 0 is equal to the probability of being 1 (50% each). In the $crossover$ process, and for each gene of the chromosome, if the values for the two chromosomes in each pair were not equal, and the value of the mask was 1 at the location of the current gene, then the two chromosomes will exchange their genes to produce a new chromosome.

However, this might cause a problem, since the total number of 1s in the new chromosome might be less or more than N_s. To overcome this issue, after generating each chromosome, the number of 1s within this chromosome will be checked. If it is less than N_s, then random locations of the chromosome will change their values from 0 to 1, until the total number of 1s is equal to N_s. In contrast, if the total number of 1s within any generated chromosome is greater than N_s, then random genes will be ignored so that the total number of 1s in any chromosome will be equal to N_s.

3) Mutation process: The last process of the GA algorithm is the $Mutation$ process, where a mutation mask will be generated that consists of 0s and 1s according to the mutation probability P_m. In our simulation, P_m was set equal to 0.09, if the element of the mask was equal to 1. Subsequently, two random genes in the corresponding chromosome will exchange their information. If the two genes have the same information, implying that both of them were zeros or ones, then the chromosome will remain the same after the $mutation$ process.

After finishing all the steps, the fitness value will be calculated for the new population and the best K chromosomes will go through the same process in the next iteration until the maximum number of iterations has been reached. In the final step, the chromosome with the highest fitness value will be chosen.

C. ABC algorithm for antenna selection

In the ABC algorithm, every bee represents a possible solution for the optimization problem. There are three different types of bees used in this algorithm, they are: Employed bees (EB), Onlooker bees (OB), and Scouts. At first, a certain number of employed bees (initial solutions) are generated randomly and their nectar amount, which is the capacity in this case, is measured. The total number of solution, or food sources, is equal to the number of employed bees. Every solution can be represented as a vector of length N_r, which is the number of parameters ($0s$ and $1s$) in the solution, and the number of $1s$ in any solution is equal to N_s. These bees share their nectar amount with the bees waiting on the dance area in the hive. Every employed bee will return to the same food position visited by itself after sharing its nectar amount, and modifies its solution by changing the parameters randomly, i.e. changing the location of the $0s$ and $1s$, then measures the modified fitness value. If the value of the modified solution for every bee is better than the previous one, then the bee will forget its old solution and memorizes the position of the new food source, otherwise the bee will return to the initial position.

The onlooker bee will then choose a food source (solution) depending on the nectar amount measured by the employed bee by using the following equation

$$\text{Source}_i = \frac{f_i}{\sum_{n=1}^{N} f_n}, \tag{11}$$

where f_i represents the fitness value of the source i, and N is the total number of possible solutions (employed bees). Once the onlooker bee has chosen the food source, then it will try to improve its solution using other food sources by the following equation

$$v_{ij} = \left| x_{ij} - x_{kj} \right|, \tag{12}$$

where $k \in \{1, 2, ...N\}$ is a randomly selected index, and $i \neq k$. $j \in \{1, 2, ...N_r\}$ represents the parameter index of the food source i. The total number of onlooker bees (OB) is equal to the number of solutions N.

This modification on the food sources might cause the problem of having more or less than N_s number of $1s$ in any solution. To tackle this issue, random parameters will be chosen to change their values so that the total number of $1s$ is equal to N_s in all the food sources. After that the fitness value of the modified solution will be calculated, and if it shows an improvement compared to the old solution, then it will memorize the modified solution, otherwise, the old solution will be used.

Finally, in order to search the area for the best food source and not getting stuck in limited number of solutions, one scout bee will be sent at each iteration to perform random search and calculate the fitness value and compare it with the worst solution in the population, if it was better then that food source will be replaced with the new food source found by the scout bee, otherwise the population will remain the same without any further changes for the next iteration.

For the next iteration, the onlooker bees will modify the solutions provided by the employed bees and a scout bee will be sent to perform a random selection until a certain number of iterations has been reached, and the best food source will be chosen for the antenna selection operation.

TABLE I
CPU TIME REQUIRED FOR THE DIFFERENT ALGORITHMS AT SNR = 0 DB, N_t = 10, N_r = 400, N_s = 100, AND 50 ITERATIONS

	Algorithm Specifications	Capacity (Bits/sec/Hz)	CPU time (Minutes)
PSO	20 particles	34.4627	7.8985
GA	20 chromosomes	35.5471	6.9756
ABC	20 food sources	34.5867	9.4569
TS	10 neighbors	35.9624	3.7001

TABLE II
CPU TIME REQUIRED FOR THE DIFFERENT ALGORITHMS AT SNR = 0 DB, N_t = 10, N_r = 400, N_s = 100, AND 75 ITERATIONS

	Algorithm Specifications	Capacity (Bits/sec/Hz)	CPU time (Minutes)
PSO	20 particles	34.4775	12.5534
GA	20 chromosomes	36.1473	10.3469
ABC	20 food sources	34.6487	14.0196
TS	10 neighbors	36.0814	5.5620

TABLE III
CPU TIME REQUIRED FOR THE DIFFERENT ALGORITHMS AT SNR = 0 DB, N_t = 10, N_r = 400, N_s = 100, AND 50 ITERATIONS

	Algorithm Specifications	Capacity (Bits/sec/Hz)	CPU time (Minutes)
PSO	40 particles	34.3620	15.7600
GA	40 chromosomes	36.2724	13.9459
ABC	40 food sources	34.7633	18.5817
TS	20 neighbors	36.5691	7.0808

D. *TS algorithm for antenna selection*

In the TS algorithm, an initial solution will be generated randomly. This solution can be represented as a vector of length N_r with N_s number of $1s$ and the rest are $0s$. At each iteration, a certain number of neighbors will be generated and their fitness values will be calculated. The best among these neighbors will be chosen as the next move for the next iteration, even if its fitness value is less than the fitness value of the current solution. The reason behind this is to ensure exploring the area as wide as possible without getting stuck in certain locations.

We define the neighbor in this algorithm as a solution differs with the current solution by a very few number of antenna locations, we call them the tabu antennas. For example: choosing two out of the N_s antennas and change their locations while keeping the locations for the rest of the antennas fixed.

After choosing the best among the neighbors, the old and new locations of the tabu antennas will be stored in the tabu matrix, and they can not be used for the next L iterations, where L is the length of the tabu matrix. The first set of tabu antennas to enter the tabu matrix will be the first one to leave it.

In the next iteration, the new solution will be used and new neighbors will be generated, and the best one will be considered as the next move, and the tabu antennas will be stored in the tabu matrix and so on until we reach the maximum number of iterations. At the end, the solution with the best fitness value in all the iterations will be declared as the final solution for the antenna selection problem.

IV. PROCESSING TIME EVALUATION

To address the underlying complexity, the CPU time was measured for the different algorithms on a 3.4 GHz intel Core i5 PC, with 8 GB of RAM using the MATLAB R2014a software program. In Tables I, II, and III the SNR value was fixed at 0 dB, and the simulations were carried out 50 times for each algorithm. In Tables I and III, the number of iterations were 50, while in Table II, 75 iterations were used. On the other hand, the number of initial solutions were 20 for PSO, GA and ABC respectively, and 10 neighbors for TS in Tables I and II, while 40 initial solutions and 20 neighbors were considered in Table III. The reason behind this is that the complexity level of these algorithms depends on both the number of iterations and the population size. In each case, the capacity as well as the CPU time were captured and compared for the different algorithms.

As Tables I, II, and III show, TS requires approximately 50% shorter CPU time than any of the bio-inspired algorithms, and achieves at the same time matching or higher capacity performance.

V. SIMULATION RESULTS AND DISCUSSION

In this section, the system capacity after applying the different antenna selection techniques has been measured. The system has 10 transmit antennas ($N_t = 10$) and 400 receive antennas ($N_r = 400$). Two different scenarios have been applied. The first one, the best 100 ($N_s = 100$) antennas were selected, while in the second scenario, only 50 ($N_s = 50$) antennas were chosen out of the 400 antennas. In both cases, we assume spatially correlated MIMO channels, with the correlation coefficient $|\phi|$ being equal to 0.85. Before we discuss the results, the specifications of the different algorithms will be introduced. In PSO, GA and ABC, the population consisted of 20 random initial solutions (i.e. 20 particles for PSO, 20 chromosomes for GA, and 20 food sources for ABC). For the TS algorithm, the number of tabu antennas in each iteration was equal to 2, and the number of neighbors was 10, while the length of the tabu matrix was equal to 50 for the first scenario and 25 for the second scenario. And the number of iterations was 50 for all the algorithms.

Fig. 2 shows that TS and GA achieve higher rates than ABC and PSO in both cases. Moreover, TS shows a slightly higher capacity than GA when the number of chosen antennas is 100, and the same capacity when N_s is equal to 50.

Fig. 3 shows the effect on the capacity by changing the correlation coefficient from 0.5 to 1. It is obvious from the figure that the performance is highly degraded when $|\phi|$ is ≥ 0.9.

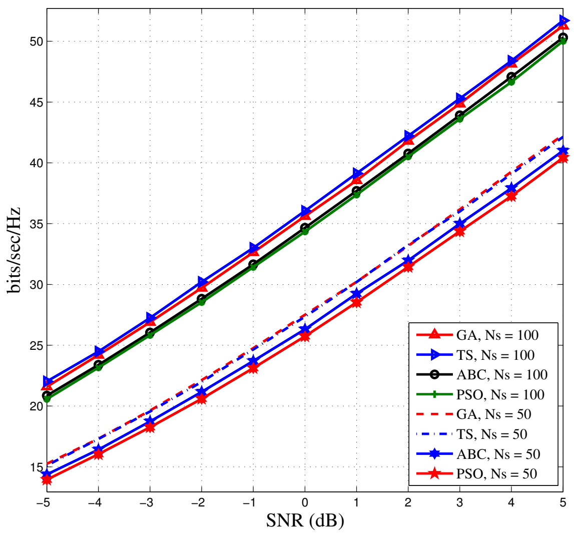

Fig. 2. Capacity Vs SNR for PSO, GA, ABC and TS algorithms with N_t = 10, and N_r = 400.

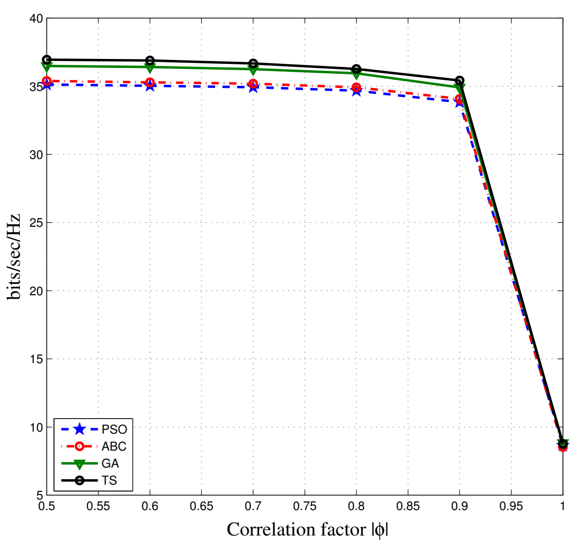

Fig. 3. Capacity Vs $|\phi|$ for different algorithms at SNR = 0 dB, $N_t = 10$, $N_r = 400$, and $N_s = 100$.

VI. Conclusions

In this paper, TS and three bio-inspired algorithms were used for antenna selection in spatially correlated massive MIMO uplink channels. The bio-inspired algorithms were: PSO, GA, and ABC. Two types of results were presented, the capacity, and the computational complexity using the CPU time required for different scenarios. For the capacity, TS and GA achieved higher rates than ABC and PSO. While for the CPU required time, TS recorded a much shorter time than any of the bio-inspired algorithms used in this paper.

References

[1] L. Lu, Geoffrey Ye Li, A Lee Swindlehurst, Alexei Ashikhmin, and Rui Zhang. An overview of massive mimo: benefits and challenges. *Selected Topics in Signal Processing, IEEE Journal of*, 8(5):742–758, 2014.

[2] Erik Larsson, Ove Edfors, Fredrik Tufvesson, and Thomas Marzetta. Massive mimo for next generation wireless systems. *Communications Magazine, IEEE*, 52(2):186–195, 2014.

[3] Mouncef Benmimoune, Elmahdi Driouch, Wessam Ajib, and Daniel Massicotte. Joint transmit antenna selection and user scheduling for massive mimo systems. In *Wireless Communications and Networking Conference (WCNC), 2015 IEEE*, pages 381–386. IEEE, 2015.

[4] Hui Li, Lingyang Song, and Mérouane Debbah. Energy efficiency of large-scale multiple antenna systems with transmit antenna selection. *Communications, IEEE Transactions on*, 62(2):638–647, 2014.

[5] Andreas F Molisch and Moe Z Win. Mimo systems with antenna selection. *Microwave Magazine, IEEE*, 5(1):46–56, 2004.

[6] J Kennedy and R Eberhart. Particle swarm optimization. In *Proc of IEEE International Conference on Neural Network. Piscataway: IEEE Press*, pages 1942–1948. sn, 1995.

[7] James Kennedy, James F Kennedy, Russell C Eberhart, and Yuhui Shi. *Swarm intelligence*. Morgan Kaufmann, 2001.

[8] Melanie Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[9] Dervis Karaboga. An idea based on honey bee swarm for numerical optimization. Technical report, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, 2005.

[10] Dervis Karaboga and Bahriye Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of global optimization*, 39(3):459–471, 2007.

[11] Thomas H.. Cormen, Charles Eric Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*, volume 6. MIT press Cambridge, 2001.

[12] Fred Glover. Tabu search-part i. *ORSA Journal on computing*, 1(3):190–206, 1989.

[13] Symeon Chatzinotas, Muhammad Ali Imran, and Reza Hoshyar. On the multicell processing capacity of the cellular mimo uplink channel in correlated rayleigh fading environment. *Wireless Communications, IEEE Transactions on*, 8(7):3704–3715, 2009.