An Improved EigenAnt Algorithm that Achieves a Balance between Intensification and Diversification

MAHAN MAHRUEYAN
Federal University of Rio de Janeiro
mmahrueyan@gmail.com

AMIT BHAYA
Federal University of Rio de Janeiro
amit@nacad.ufrj.br

June 16, 2019

Abstract

The EigenAnt algorithm has been proposed to find the shortest path between two nodes with a proof of convergence. Two novel features are noticeable in EigenAnt compared to the conventional ant colony optimization algorithms. Pheromone evaporation in the EigenAnt algorithm is done locally for the selected path and it does not use heuristic information in its selection phase. On the other hand, a simple version of ant colony optimization that also does not use heuristic information has been used for analyzing convergence, considering parameter impact analysis, in the problem of finding the shortest path for 1*–node binary chain problems. In this paper, we propose to improve the EigenAnt algorithm by adding additional parameters in such a way as to be able to take advantage of the analysis developed for the simple ant colony optimization algorithm. We demonstrate through our analysis that since the Improved EigenAnt algorithm has one parameter more than ant colony optimization algorithms, tuning of convergence speed is decoupled from tuning of convergence features in the algorithm. Thus, IEigenAnt, which can be interpreted as having two uncorrelated intensification and diversification components, makes it possible to achieve a balance between intensification and diversification. In order to illustrate the advantages of this decoupling, we present experimental results for the routing network problem.*

keywords: *EigenAnt, parameter impact analysis, intensification and diversification balance, routing networks*

1. INTRODUCTION

Theoretical development accompanying the design of algorithms is ideal because long-term behavior of an algorithm and influence of certain parameters in the algorithm can be predicted [1]. In the area of Ant Colony Optimization (ACO) algorithms, researchers have developed some theoretical approaches. In [2] an analytical model called ant programming is proposed for the ACO algorithms in order to analyze the convergence of ACO through optimal control theory. In [3–6] probability of convergence to optimal solution in the ACO algorithm is analyzed. In [1] an analysis for the Simple ACO (SACO) [7] algorithm applied to the 2–nodes 2–paths problem (1–node Binary Chain Problem (BCP)), is extended to the N–nodes 2–paths problem (N–node BCP). The paper [8] proposes a novel ACO based algorithm entitled EigenAnt for the shortest path problems (paths of different lengths between a source and a destination node) with a proof of convergence of pheromone concentration to the shortest path.

This paper proposes to combine the two latter approaches considering the similarities and the differences between the SACO and the EigenAnt algorithm. They are similar since both are in the ACO algorithm family and do not use heuristic information in their solution construction phase. EigentAnt is different from SACO due to its evaporation process. EigenAnt's evaporation process is done locally, i.e. the pheromone trails related to the selected path are evaporated, while, in SACO, the pheromone evaporation is done globally—i.e. all pheromone trails are evaporated. Local pheromone evaporation has also been used in the Ant Colony System (ACS) algorithm [9]. Furthermore, the pheromone update phase in EigenAnt is different from SACO. Unlike ACO, EigenAnt does not use pheromone amplification parameter α. EigenAnt is also different from SACO in using only one ant at each iteration, while SACO uses M number of ants. However, the analysis in [1] also uses one ant to analyze SACO, which makes it similar to EigenAnt.

In this paper, we update the analysis relating to a proposal to modify EigenAnt [10] by including additional parameters in such a way that it is possible to take advantage of the analysis in [1] and gain the important advantage of allowing independent adjustment of the stability of a desired equilibrium point and the speed of convergence to this equilibrium. This improved algorithm, which is described in detail below, has been entitled IEigenAnt [10], abbreviating the term improved EigenAnt. Note that independent adjustment of stability of equilibrium points and speed of convergence is not possible either in EigenAnt or SACO.

The decoupling of tuning of speed from convergence behavior is an important aspect of designing metaheuristic algorithms, including ACO, also referred to in terms of achieving intensification and diversification balance. Diversification refers generation of diverse solutions so as to explore the search space globally while intensification refers to focusing on the search in a local region by exploiting the information that a current good solution found in this region [11]. In [12] it is explained that diversification is increased by having a slowly convergent algorithm while a rapidly converging algorithm increases intensification. Furthermore, the impact of parameter α on the convergence and speed of convergence in SACO is analyzed in [12] where it is concluded that the pheromone amplification factor $\alpha > 1$ results in convergence to an undesired equilibrium because the intensification is too much while the diversification is too little (very fast convergence behavior); $\alpha = 1$ results in a single pass and lacks the ideal diversification (fast enough to converge); $\alpha < 1$ results in a distribution of paths (high diversification and too little intensification or too slow convergence behavior). The paper [1] arrived at the same conclusions as [12] in regard to the impact of parameter α on the SACO algorithm. From what we have just explained about the conclusions in [12], we can deduce that intensification and diversification have opposing interactions.

In [13] the concepts of intensification and diversification are considered as two powerful forces that have a great impact on the performance of metaheuristic algorithms. The paper [13] defines an intensification and diversification component (I & D) as any algorithmic or functional component that has an intensification and/or a diversification effect on the search process. For the ACO algorithms, the paper [13] considers the probabilistic selection in the solution construction phase and the pheromone update dynamic in the pheromone update phase as its I & D components. Considering the conventional ACO algorithms and the parameter impact analysis done in [1] and [12], the transition probability function is the only I & D component in which the intensification and diversification have an opposing interaction; hence, a strategy is needed to balance between intensification and diversification. One simple strategy is to increase diversification by another diversification component in the pheromone update phase while choosing the intensification for the selection phase. Max-Min Ant System [14], and the local evaporation in the ACS algorithm are the examples of such simple strategies. Both EigenAnt and IEigenAnt have such a diversification component due to their local evaporation process. However, if these strategies are not powerful enough to balance between intensification and diversification as in [15], a dynamical Max-Min strategy in order to boost the diversification; and in [16] a strategy to balance between intensification and diversification for the ACS algorithm is suggested in its transition probability function although the algorithm has the local evaporation. Choosing intensification in the selection phase is important because the ACO algorithms are anytime algorithms[1] [17]. Thus, the ACO algorithms with fast speed of convergence (more intensification in the selection I & D component) have better performance when there is not enough computation time available.

The paper [18] suggests a population based strategy in which the parameter α in the transition probability function relating to the best iteration ant increases gradually while the one relating to the worst ant takes the

[1]Anytime algorithms are algorithms whose qualities improve gradually as computation time increases.

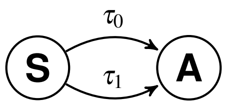

Figure 1: *The* 1*-node BCP of finding the shorter of the two paths from the start node* $\textcircled{S}$ *to the arrival node* $\textcircled{A}$*.* $\tau_x, x = \{0,1\}$ *denotes the pheromone concentration on each path.*

value of the best ant for the next iteration. By doing so, the I & D component of the solution construction phase gradually switches from the diversification to the intensification. For the I & D component in the pheromone update phase of ACO, [18] suggests having a regulated pheromone evaporation parameter value proportional to its associated pheromone trail value in order to make the evaporation parameter as a diversification component although the proposed ACO algorithm in [18] has global pheromone evaporation. In addition, the deposition parameter is chosen to be dependent on the evaluated cost in a way that an ant allocates a greater amount of pheromone to a newly found solution, whereas a relatively smaller amount of pheromone is laid on the iteration-best solution, which is no better than the global best. Thus, the deposition process considered by this strategy acts as an intensification component. Therefore, the strategy in [18] considers an I& D component for the selection phase, a D component for the evaporation process, and an I component in the deposition process of the pheromone update phase in order to balance between intensification and diversification in ACO.

Although the strategy in [18] possesses sufficient degree of freedom with its I & D components, it does not have any analytical support. Moreover, the strategy neglects having a fast convergence during the run of the algorithm since in the beginning of the algorithm the parameters α for each ant have small values leading to slow convergence behavior and high diversification. In this paper we will propose an Improved EigenAnt algorithm, with analytical support, that has one I & D parameter in its solution construction phase allowing adjustment of the speed of convergence; one D component in its evaporation process of the pheromone update phase because of its local pheromone evaporation; one I & D component in its pheromone deposition process of the pheromone update phase allowing adjustment of stability of the desired equilibrium point. Therefore, IEigenAnt has enough I & D components to achieve a degree of freedom that enables the algorithm to balance between intensification and diversification. In order to illustrate the advantage of decoupling feature in IEigenAnt for balancing between intensification and diversification, we compare the performance of IEigenAnt with EigenAnt and SACO in the application of Routing Network (RN) problem.

In Section 2, we review the SACO algorithm and the formulation of its associated analytical model, from [1], for 1-node BCPs. Next, we review EigenAnt and justify the need to introduce the Improved EigenAnt (IEigenAnt). Finally, convergence and parameter impact analysis are discussed for the 1-node BCP application. In section 3, we illustrate the benefit of the decoupling feature in IEigenAnt by empirically comparing its performance with the EigenAnt and the SACO algorithms. In Section 4, conclusions and suggestions for future works are presented.

2. Analytical model for 1-node BCP

2.1. The SACO Algorithm for 1-node BCP

The application of SACO to the 1-node BCP (Fig. 1) includes the three following phases:

2.1.1 Solution Construction Phase

At every iteration, M ants select between paths indexed as $x = \{0,1\}$ through a probabilistic procedure entitled Roulette-Wheel Selection (RWS) [19]. A transition probability function is used in the RWS for the ant m as follows:

$$p_x^m(t) = \frac{\tau_x(t)^\alpha}{\tau_0(t)^\alpha + \tau_1(t)^\alpha} \tag{1}$$

where τ_x is the pheromone trail concentration corresponding to the edge x and α is the pheromone amplification parameter.

2.1.2 Cost Evaluation Phase

After ant m has selected an edge x, a function is defined to evaluate the cost of the constructed solution. The constructed solution cost for the ant m is denoted by L_x^m. Usually the number of cost function evaluations is a criterion for comparing the complexity between the ACO algorithms.

2.1.3 Pheromone Update Phase

Pheromone update takes place in two steps at each iteration. First, a global process named pheromone evaporation reduces all the τ_x globally for all the edges as follows:

$$\tau_x(t+1) = (1-\rho)\,\tau_x(t) \tag{2}$$

where $\rho \in [0,1]$ is the evaporation parameter. Second, the pheromone deposition process occurs only for the edges included in the constructed solution by each ant m upon returning from the path it traversed as follows:

$$\tau_x^m(t+1) = \tau_x^m(t) + \frac{Q}{L_x^m} \tag{3}$$

where Q is a scaling factor.

These three phases are carried out in each iteration for all the ants, until a stopping criterion is satisfied.

2.2. Implementing SACO on the Analytical Model

The global behavior of the system can be described using the statistical physics idea of ensemble averaging over a large number of copies or instances of the system, each of which represents a possible system state. Considering $Q = 1$ in Eq.(3), this procedure leads to the following deterministic difference equation:

$$\tau_x(t+1) = (1-\rho)\,\tau_x(t) + \left(\frac{1}{L_x}\right)\left(\frac{\tau_x(t)^\alpha}{\tau_0(t)^\alpha + \tau_1(t)^\alpha}\right) \tag{4}$$
$$\left(\frac{\tau_x(t)^\alpha}{\tau_0(t)^\alpha + \tau_1(t)^\alpha}\right) \in [0 \quad 1], x = \{0,1\}$$

It should be mentioned that in Eq. (4) the transition probability is only multiplied by the deposition term because the evaporation is done globally for each edge. In other words, the ensemble averaging considers the deposition of pheromone on the selected edge with the probability equal to the transition probability function while the probability of evaporation is considered as one.

Passing from the discrete representation to the continuous one:

$$\begin{cases} \frac{\mathrm{d}\tau_0}{\mathrm{dt}} = -\rho\tau_0 + C_0P_0 \\ \frac{\mathrm{d}\tau_1}{\mathrm{dt}} = -\rho\tau_1 + C_1P_1 \end{cases} \tag{5}$$

The pheromone deposition coefficients C_0 and $C_1 = \kappa C_0$ have inverse relation with the evaluated cost L_x in Eq. (4). It should be mentioned that $\kappa = \frac{C_1}{C_0} = \frac{L_0}{L_1}$ is the deposition parameter. In [1], Eq. (5) is used as the analytical model for 1-node BCP.

2.3. The EigenAnt Algorithm for 1-node BCP

The EigenAnt algorithm applied to 1-node BCP is composed of the following three phases.

2.3.1 Solution Construction Phase

As mentioned before, unlike SACO, EigenAnt only uses one ant at each iteration. The transition probability function for the EigenAnt's selection phase is as follows:

$$p_x(t) = \frac{\tau_x(t)}{\tau_0(t) + \tau_1(t)} \tag{6}$$

The pheromone amplification parameter α used in SACO (Eq. (1)) is not used in the transition probability function of EigenAnt.

2.3.2 Cost Evaluation Phase

Having selected an edge x by an ant, a function is defined to evaluate the cost of the constructed solution. The constructed solution cost is denoted by L_x.

2.3.3 Pheromone Update Phase

The major difference between EigenAnt and SACO is in this phase in which EigenAnt uses a specific dynamical model to update pheromone concentration trails. The model used to update the pheromone trails in the application of EigenAnt to the 1-node BCP is as follows:

$$\tau_x(t+1) = (1-\rho)\,\tau_x(t) + (1/L_x)\,\frac{\tau_x(t)}{\tau_0(t) + \tau_1(t)} \tag{7}$$

It can be noticed that, unlike the SACO algorithm, the pheromone evaporation process only takes place for the selected edge. In other words, the pheromone evaporation process is done locally in EigenAnt.

2.4. Implementing EigenAnt on the Analytical Model

The EigenAnt algorithm is similar to SACO, in the sense that it does not use the heuristic information in its transition probability function. In contrast with the pheromone transition probability function for SACO (Eq. (1)), EigenAnt's transition probability function lacks the pheromone amplification parameter α (Eq. (6)). Hence, we propose an improved version of EigenAnt which has the same pheromone transition probability function as SACO, with the pheromone amplification parameter α. Therefore, the following deterministic difference equation is derived through the ensemble averaging for the improved version of the EigenAnt algorithm:

$$\tau_x(t+1) = \left((1-\rho)\,\tau_x(t) + \frac{1}{L_x}\left(\frac{\tau_x(t)}{\tau_0(t) + \tau_1(t)}\right)\right)\left(\frac{\tau_x(t)^{\alpha}}{\tau_0(t)^{\alpha} + \tau_1(t)^{\alpha}}\right) \tag{8}$$

It should be mentioned that in Eq. (8) the transition probability function is multiplied by the whole pheromone update expression in contrast with SACO. This is due to the local pheromone evaporation in EigenAnt. In other words, the pheromone trail related to the selected edge x is evaporated and deposited with the probability equal to the transition probability function under the ensemble averaging consideration.

The most general model of this type, which is henceforward entitled Improved EigenAnt (IEigenAnt) is obtained by introducing parameters α_1 and α_2 and is thus written as follows:

$$\tau_x(t+1) = \left((1-\rho)\,\tau_x(t) + \frac{1}{L_x}\left(\frac{\tau_x(t)^{\alpha_2}}{\tau_0(t)^{\alpha_2} + \tau_1(t)^{\alpha_2}} \right) \right) \left(\frac{\tau_x(t)^{\alpha_1}}{\tau_0(t)^{\alpha_1} + \tau_1(t)^{\alpha_1}} \right) \tag{9}$$

The large brackets on the left hand side of Eq. (9) contain an expression that is the same as the discrete version of the analytical model in Eq. (4). Passing from the discrete representation to the continuous one, gives the following analytical model for the IEigenAnt:

$$\begin{cases} f : \frac{\mathrm{d}\tau_0}{\mathrm{dt}} = \left(-\rho\tau_0 + C_0 P_0(\alpha_2)\right) P_0(\alpha_1) \\ g : \frac{\mathrm{d}\tau_1}{\mathrm{dt}} = \left(-\rho\tau_1 + C_1 P_1(\alpha_2)\right) P_1(\alpha_1) \end{cases} \tag{10}$$

The parameter α_1 is associated with the transition probability function in the solution construction phase of the IEigenAnt while the parameter α_2 is associated with the pheromone update dynamic in IEigenAnt. As a result, the following transition probability function and pheromone update dynamic is assumed for the application of IEigenAnt to solve 1-node BCPs:

$$p_x(t) = \frac{\tau_x(t)^{\alpha_1}}{\tau_0(t)^{\alpha_1} + \tau_1(t)^{\alpha_1}} \tag{11}$$

$$\tau_x(t+1) = (1-\rho)\,\tau_x(t) + (1/L_x)\,\frac{\tau_x(t)^{\alpha_2}}{\tau_0(t)^{\alpha_2} + \tau_1(t)^{\alpha_2}} \tag{12}$$

2.5. Discussion

In [1], it is concluded from the stability analysis of the equilibrium points of the dynamic in Eq. (5) that with $\alpha = 1$ the system converges to the shortest path. However, it is also concluded that the speed of convergence is the slowest with $\alpha = 1$. Therefore, choosing $\alpha = 1$ in SACO application to the 1-node BCPs guarantees the convergence to the shortest path at the expense of increased convergence time. It is also concluded that with $\alpha > 1$ the system might have a premature convergence to the non-optimal path, although the speed of convergence is faster than in the case $\alpha = 1$. Finally, it is concluded that the system does not demonstrate convergence behavior with $\alpha < 1$ performing continuous exploration for new paths. It should be mentioned that the conclusions about the speed of convergence in [1] are only derived from the numerical simulations.

The analytical model of IEigenAnt (Eq. (10)) has the same equilibrium points as SACO (Eq. (5)). It should be mentioned that the equilibrium points in IEigenAnt are dependent on the α_2 which is associated to the pheromone update dynamic in IEigenAnt while in SACO the parameter α is associated with the transition probability function. Therefore, any result achieved for the impact of α on the convergence features of SACO in [1] is also true for the impact of α_2 on the convergence features of IEigenAnt. The only distinction between the two analytical models is in the additional factor $P(\alpha_1)$, associated with the transition probability function in IEigenAnt that affects the speed of convergence without affecting the nature of the stability of the equilibrium points. From Eq. (10), it can be concluded that an increase in the value of $P(\alpha_1)$ leads to the faster convergence.

Reconsidering the transition probability function in IEigenAnt ($\frac{\tau_x(t)^{\alpha_1}}{\tau_0(t)^{\alpha_1}+\tau_1(t)^{\alpha_1}}$), we assume that the transition probability function has two paths to select from. Thus, we modify the denominator of the formula for the transition probability function with two paths trails of which one will be selected and the other will be not. We refer the former to x and the latter to $\bar{x}$. Therefore, we have the following analysis for the impact of α_1 on $P(\alpha_1)$:

$$\lim_{\alpha_1\to\infty} \frac{\tau_x(t)^{\alpha_1}}{\tau_x(t)^{\alpha_1}+\tau_{\bar{x}}(t)^{\alpha_1}} = \lim_{\alpha_1\to\infty} \frac{1^{\alpha_1}}{1^{\alpha_1}+\left(\frac{\tau_{\bar{x}}(t)}{\tau_x(t)}\right)^{\alpha_1}} \mapsto \begin{cases} 0, \; if\; \tau_x < \tau_{\bar{x}} \\ 1, \; if\; \tau_x > \tau_{\bar{x}} \\ \frac{1}{2}, \; if\; \tau_x = \tau_{\bar{x}} \end{cases} \tag{13}$$

We have to define the meaning of each condition in Eq.(13). The condition *if* $\tau_x < \tau_{\bar{x}}$ refers to the case where the pheromone trail τ_x associating with the selected path by the RWS is smaller than the trail associating with the path not selected by RWS. This case happens less frequently as RWS is designed to more frequently select the paths with greater pheromone trails. On the other hand, the condition *if* $\tau_x > \tau_{\bar{x}}$ refers to the case where the pheromone trail associating with the selected path has greater amount than the trail which is not selected by RWS. This case is more frequently happening. Finally, the last condition *if* $\tau_x = \tau_{\bar{x}}$ refers to the case that the two alternative paths has equal pheromone trails. Such a conditions general happens in the beginning of the run of the IEigenAnt algorithm where the initial values for the pheromone trails are equal.

For our deterministic analysis, we are interested in ensemble averaging where the most frequent case is only considered. Therefore, it is enough to only describe the limit for the second case in order to analyze the impact of α_1 on $P(\alpha_1)$. The second case in Eq.(13) means that when α_1 increases to infinity, the transition probability function converges to one, which is the largest value it can have as a probability function. Considering $\alpha_1 = 0$, then the transition probability function would be $\frac{\tau_x(t)^0}{\tau_x(t)^0+\tau_{\bar{x}}(t)^0} = \frac{1}{2}$. Therefore, the transition probability function in the ensemble averaging is increasing from $\frac{1}{2}$ to 1 when the parameter α_1 increases from 0 to ∞. Considering the analytical model in Eq.(10), the multiplicative term $P(\alpha_1)$ has its largest value one by increasing the parameter α_1, which means that the speed of convergence is increasing when the parameter α_1 increases, which is the same as the conclusions in [12].

The conclusions about the impact of parameter α on the convergence features for SACO in [1,12] are also true for the impact of the parameter α_2 on IEigenAnt. Specifically, choosing $\alpha_2 = 1$ causes the pheromone trails converge to the shortest path in 1-node BCPs which has already been proved for the original EigenAnt algorithm [8]. Therefore, we can tune the speed of convergence through α_1 in the transition probability function independently from the convergence features associated with the α_2 in the pheromone update dynamic of the IEigenAnt algorithms.

Such a decoupling allows us to design the highest speed of convergence (selecting intensification) by choosing $\alpha_1 > 1$ while balancing between intensification and diversification by choosing $\alpha_2 < 1$. This decoupling feature favors IEigenAnt over SACO because choosing $\alpha > 1$, in SACO, in order to increase the convergence speed might cause the premature convergence to a non-optimal solution (too much intensification).

3. Experiment on Routing Networks

An empirical experiment is necessary to check which choice of parameters α_1 and α_2 is ideal. Moreover, we have to verify the advantage of decoupling feature in the IEigenAnt algorithm over the EigenAnt and the SACO algorithms. In this section, we experimentally confirm the benefit of decoupling feature in IEigenAnt

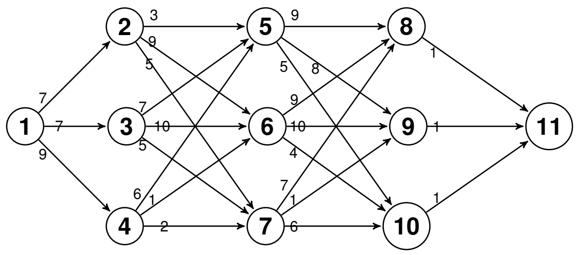

Figure 2: *A Routing Network of size* 3×3 *(a three stage multi-hop network)*

over EigenAnt and SACO for the application of finding the shortest path in the Routing Networks (RNs). A multi-hop network as in the format Fig. 2 is solved via EigenAnt in [8]. Note that, without loss of generality, all edges in the last layer can be chosen to have length one. Thus, the last layer is not counted in the modeling of multi-hop networks with the format of Fig. 2. Generally speaking, the multi-hop network in Fig. 2 can be modeled as an $N \times O$ matrix where $N = 3$ layers and $O = 3$ number of outgoing edges from each node.

3.1. Application of IEigenAnt to Routing Networks

IEigenant application to a 3×3 RN is done through the three following phases:

3.1.1 Solution Construction Phase

One ant at each iteration constructs a solution from the source node $\textcircled{1}$ to the final node $\textcircled{11}$ (Fig. 2).

As for the simple 1–node BCP, each pheromone trail represents an edge of the graph in Fig. 2. At each node of the i^{th} layer, an ant chooses the outgoing edge j from the O choices with the following general transition probability function:

$$P_{ij}\left(\alpha_1\right) = \frac{\tau_{ij}^{\alpha 1}}{\sum_{c=1}^{O} \tau_{ic}^{\alpha_1}} \tag{14}$$

Generally speaking, each ant in the $N \times O$ RN has O edges to choose via RWS, from the 1st layer to the N^{th} layer.

3.1.2 Cost Evaluation Phase

The constructed solution cost, denoted r for the paths with more than one edges, is evaluated by summing the length of the edges that form the solution from the 1st to the N^{th} layer ($L_r = \sum_{i=1}^{N} d_i$) in which d_i is the length of the chosen edge at the i^{th} layer.

3.1.3 Pheromone Update Phase

At each iteration, the pheromone trails relating to the edges of the constructed solution are updated independently. The following dynamic is for IEigenAnt pheromone update of each edge $j \in \{1, \cdots O\}$ at the i^{th} layer that forms the path r:

$$\tau_{ij}\left(t+1\right) = \left(1-\rho\right)\tau_{ij}\left(t\right) + \left(\frac{Q}{L_r}\right) P_{ij}\left(\alpha_2\right) \tag{15}$$

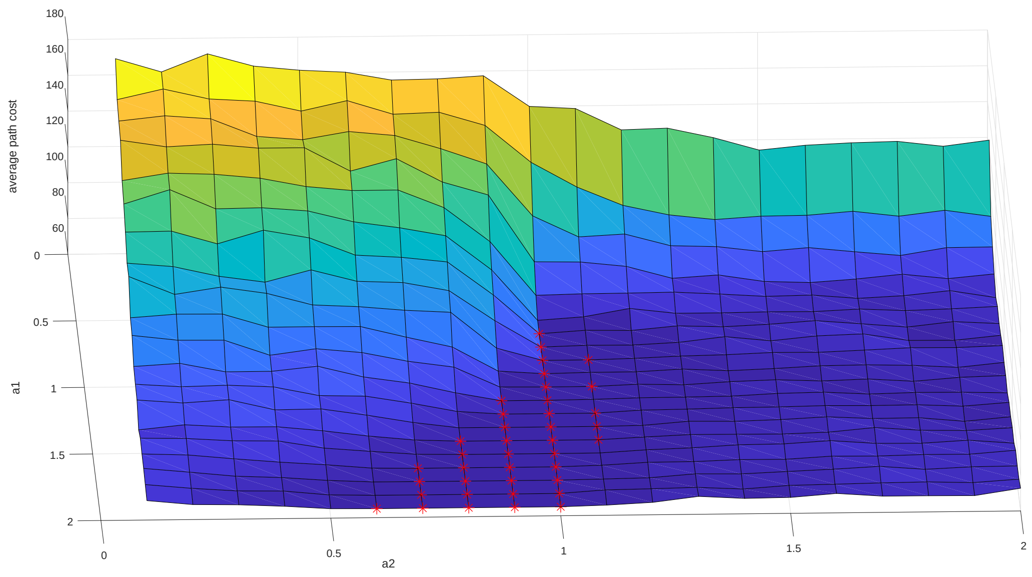

Figure 3: *The average of shortest path cost found by IEigenAnt for the application of solving* 10×10 *RN, with the values of parameters* α_1 *and* α_2 *in the range of* $(0,2)$*, after* 8000 *cost evaluations. The star marks depict the best average results.*

where $P_{ij}(\alpha_2) = \frac{\tau_{ij}^{\alpha 2}}{\sum_{c=1}^{O} \tau_{ic}^{\alpha_2}}$.

3.2. Experimental Results

In this subsection, we illustrate the decoupling advantage of the IEigenAnt over the EigenAnt and the SACO algorithms by experimenting them on RN models. The RN models to which we apply the algorithms are a 10×10 and a 20×20 multi-hop RN with the format of Fig. 2[2]. Dijkstra's algorithm [20] application to the problems results in the optimal solution with the costs of 65 for the 10×10 model and 58 for the 20×20 model.

First we apply the algorithms to the smaller problem of 10×10 RN model. We perform 30 experiments for IEigenAnt applied to the 10×10 RN problem with different possible exponential parameters in the range of $(0,2)$ —α_1, α_2 for IEigenAnt, and α for SACO. The pheromone evaporation parameter is set to the value of $\rho = 0.1$. Parameter Q is set equal to number of layers in the RN model ($N = Q = 10$). Initial pheromone values for each edge are chosen to be equal to the reciprocal of the length of the associated edge. We save the shortest path found in each experiment after 8000 cost evaluations.

Fig. 3 illustrates the average of shortest path cost found by IEigenAnt, with values of parameters α_1 and α_2 in the range of $(0,2)$, after 8000 cost evaluations. Fig. 3 depicts that the average equal to the optimal cost 65 is achieved 39 times. The optimal path is always found by IEigenAnt in all cases with $\alpha_2 = 1$ and $\alpha_1 \geq 0.7$ which means that the success is dependent on setting the algorithm with convergence to the local optimal solution. By increasing the diversification ($\alpha_2 < 1$), the balance between intensification and diversification should be maintained; hence, $\alpha_1 \geq 1.2$ makes IEigenAnt always achieve optimal solution in this case. The most interesting case for balancing between intensification and diversification can be noticed when IEigenAnt always finds the optimal paths with $\alpha_2 = 0.6$ (great diversification) and $\alpha_1 = 2$ (great intensification and very

[2]The models in MATLAB® code are available upon request.

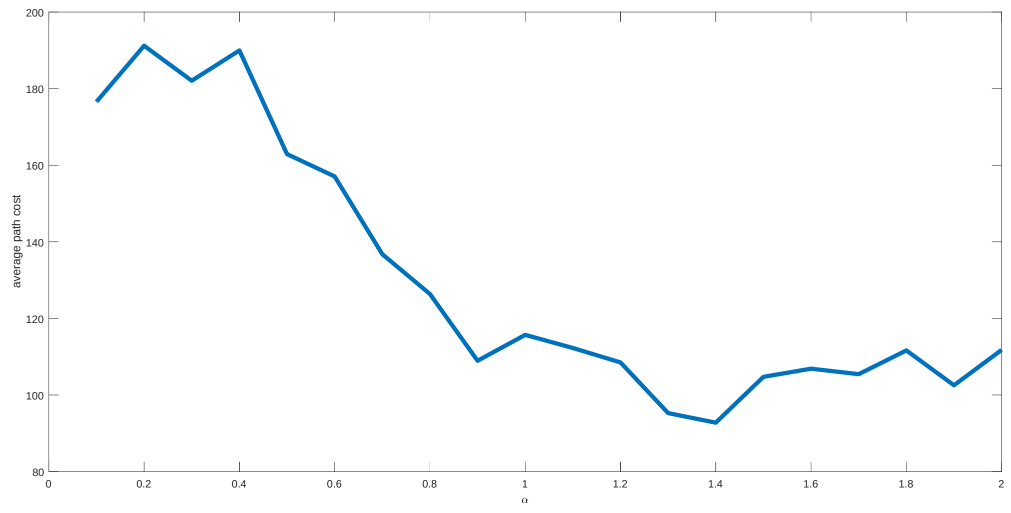

Figure 4: *The average of shortest path cost found by SACO for the application of solving* 10×10 *RN, with the values of parameter* α *in the range of* $(0,2)$*, after* 8000 *cost evaluations.*

fast convergence). For $\alpha_2 = 1.1$ (so much intensification that it might lead to premature convergence), the speed should be limited to the associated parameter $\alpha_1 \leq 1.5$; otherwise, the diversification due to the local optimal evaporation can not individually maintain the balance between the intensification and diversification. It should be noted that EigenAnt ($\alpha_1 = \alpha_2 = 1$) is also successful in always finding the optimal path since the diversification attributed to the local evaporation in EigenAnt can maintain the balance between intensification and diversification although the algorithm is set to converge to the local optimal solution (deposition process in EigenAnt acts as an I component).

Fig. 4 illustrates the average of shortest path found by SACO, with different value of parameter α in the range of $(0,2)$, after 8000 cost evaluations. It should be mentioned that the application of SACO to the RN is the same as the IEigenAnt, except for its pheromone update phase that uses Eq.(2) and Eq.(3) instead of Eq.(15). Moreover, we only use one ant at each iteration in the SACO algorithm ($m = 1$). As can be seen from Fig. 4, the SACO algorithm is incapable of achieving the average path cost results less than 92.77 which is considerably worse than the IEigenAnt algorithm. We also can note that the setting of parameter $\alpha = 1.4$ with the premature convergence and fast speed leads to the best result in SACO.

Furthermore, we apply IEigenAnt and SACO to the larger problem of 20×20 model. All the parameters and pheromone trails initialization are set the same as the smaller model except for the parameter Q which is set equal to $Q = N = 20$. Fig. 5 illustrates the results of solving the problem after 8000 cost evaluations. In the three parameter settings, in which $\alpha_1 \geq 1.8$ and $\alpha_2 = 0.9$, IEigenAnt average cost is minimum with the value of 58.13. IEigenAnt is incapable of always achieving the optimal cost when the problem size is increased and the computation time is still 8000 cost evaluations; however, the minimum average cost is only 0.13 more than the optimal cost. Since the best settings in this experiment do not always achieve the optimal solution, the Standard Deviation (SD) of the results are also illustrated in Fig. 6. It can be noticed that the best SD results with the value of 0.73 in Fig. 6 match the best average results in Fig. 5. In the best parameter settings, the speed of convergence is tuned very fast ($\alpha_1 \geq 1.8$) since the available computational time is small considering the large size problem. Having a very fast convergence behavior, means a great amount of intensification that makes IEigenAnt use its decoupling feature to balance between intensification and diversification by having a non–convergence feature through $\alpha_2 = 0.9 < 1$. This illustrates the power of IEigenAnt in balancing

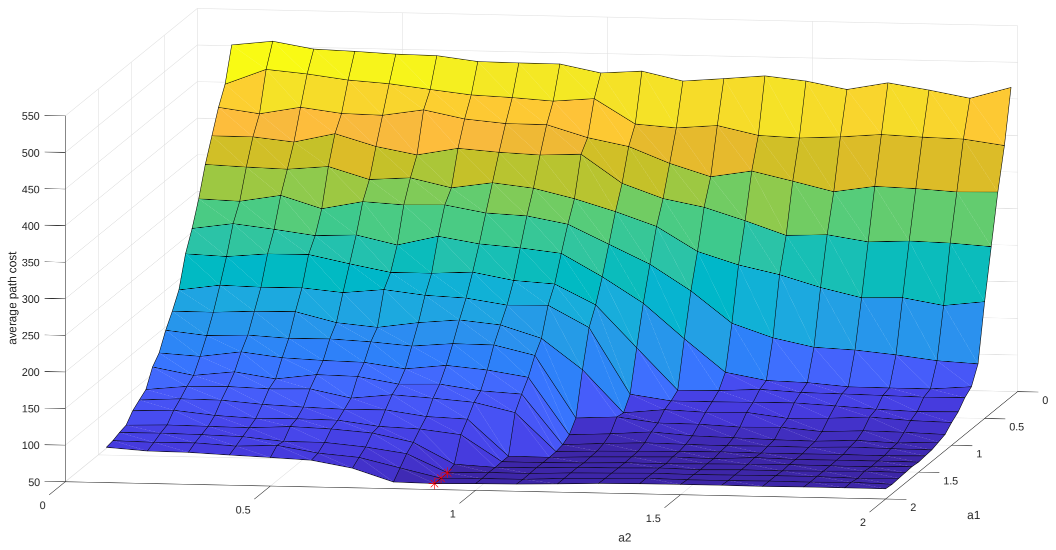

Figure 5: *The average of shortest path cost found by IEigenAnt for the application of solving* 20×20 *RN, with the values of parameters* α_1 *and* α_2 *in the range of* $(0,2)$*, after* 8000 *iterations. The star marks depict the best average results.*

between intensification and diversification while it maintains a fast convergence speed whenever computation time is limited. The average shortest path cost found by the EigenAnt algorithm ($\alpha_1 = \alpha_2 = 1$) is 184.37 which is considerably worse than the IEigenAnt algorithm since EigenAnt is incapable of balancing between intensification and diversification when computation time is limited. Fig. 5 and Fig.6 also illustrate that there is a valley in which the IEigenAnt parameters are set at the values $\alpha_1 > 1$ and $\alpha_2 > 1$ (too much intensification). The existence of this valley illustrates that when computation time is limited, choosing parameters $\alpha_1 > 1$ and $\alpha_2 > 1$ helps to achieve an acceptable sub-optimal solution (through rapid premature convergence).

Fig. 7 illustrates the average of shortest path with different value of of parameter α in the range of $(0,2)$. As we can notice, the SACO algorithm is incapable of achieving average path cost results less than 116.9 which is considerably worse than IEigenAnt. We also can note that the setting of parameter $\alpha = 1.9$ with the premature convergence and fast speed leads to the best result in SACO.

For the experiment performed on 10×10 RN problem in Fig.3, it is difficult to observe the power of achieving a balance between intensification and diversification in IEigenAnt to make the algorithm perform rapidly. Thus, we limit the available computational time for the IEigenAnt algorithm application to the 10×10 RN problem by saving the results after 400 cost evaluations —decreasing the available computational time by a factor of 20.

Fig. 8 illustrates the results of solving the 10×10 RN problem after 400 cost evaluations. The best parameter settings that always achieve the optimal cost of 65 are $(\alpha_2 = 0.6, \alpha_1 = 2)$ and $(\alpha_2 = 0.7, \alpha_1 = 1.8)$. Since the available computational time for the anytime algorithm of IEigenAnt is too small, only in two settings it can always achieve the optimal solutions. The same as the previous experiment with the larger size problem of 20×20, the choices of parameters with very fast speed of convergence and increase in diversification, by setting $\alpha_2 < 1$, to balance the high intensification due to the fast speed of the algorithm is the best tuning of the IEigenAnt algorithm. The EigenAnt average path cost in this experiment is 78.03, which is considerably worse than the optimal cost of 65.

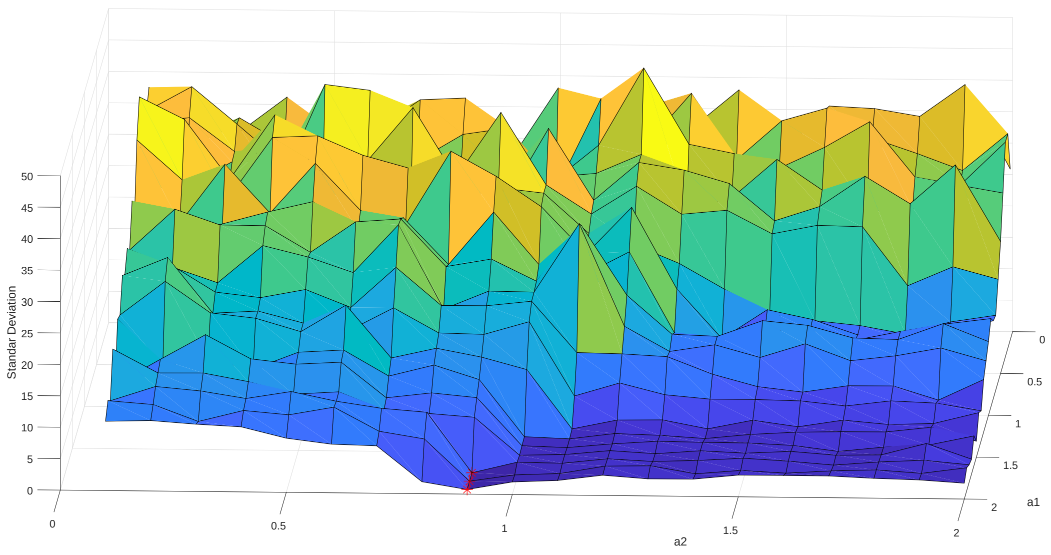

Figure 6: *The Standard Deviation of the shortest path cost found by IEigenAnt for the application of solving* 20×20 *RN, with the values of parameters* α_1 *and* α_2 *in the range of* $(0,2)$*, after* 8000 *iterations. The star marks depict the results with respect to the Standard Deviation.*

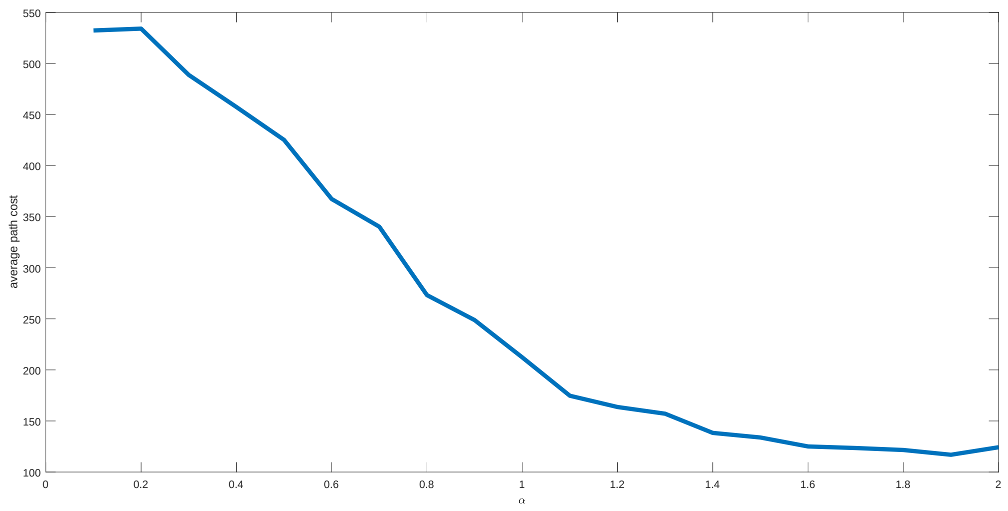

Figure 7: *The average of shortest path cost found by SACO for the application of solving* 20×20 *RN, with the values of parameter* α *in the range of* $(0,2)$*, after* 8000 *cost evaluations.*

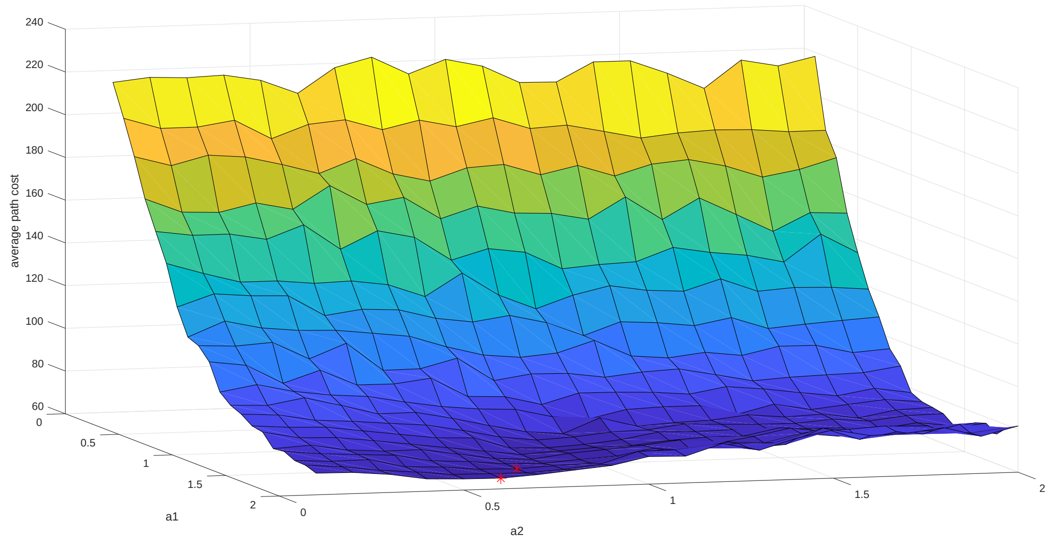

Figure 8: *The average of shortest path cost found by IEigenAnt for the application of solving* 10×10 *RN, with the values of parameters* α_1 *and* α_2 *in the range of* $(0,2)$*, after* 400 *cost evaluations. The star marks depict the best average results.*

We can deduce from the experiments that the advantage of decoupling feature in IEigenAnt over EigenAnt and SACO is much more noticeable when the available computational time for the anytime algorithms is too small. In the smaller size RN problem of 10×10 with enough available computational time, IEigenAnt does not outperform EigenAnt since the impact of decoupling feature on balancing between intensification and diversification is not as strong as the impact of local evaporation. Since SACO lacks the local evaporation, it still has worse performance than IEigenAnt and EigenAnt for the smaller size problem with enough available computational time.

4. Conclusions

Inspired by an analytical model proposed for the Simple Ant Colony Optimization algorithm, we improved the EigenAnt algorithm by adding two parameters in its transition probability function and pheromone update dynamic in order to be able to implement the Improved EigenAnt on the interesting analytical model for the application of 1–node binary chain problems. The parameter relating to the transition probability function affects the speed of convergence while the parameter relating to the pheromone update dynamic affects the convergence features of IEigenAnt. Therefore, IEigenAnt has a decoupling feature in which the speed of convergence can be tuned independent from the convergence features. Both of these parameters are considered as independent intensification and diversification components in such a way that the balance between intensification and diversification can be maintained. IEigenAnt and EigenAnt have an extra diversification component due to their local pheromone evaporation process. The advantage of the decoupling feature in IEigenAnt was shown through an empirical parameter impact analysis and comparing it with the EigenAnt and the SACO algorithms applied to the solution of Routing Network problem. The results show that IEigenAnt and EigenAnt have the same performance for smaller problems with enough available computational time and they both outperform the SACO algorithm due to their local pheromone evaporation. However, for larger problems and smaller problems with limited available computational time, IEigenAnt not only outperforms SACO but also EigenAnt because it has the decoupling feature that allows IEigenAnt be an algorithm that is fast enough to cope with the short computation time available, and simultaneously maintain its diversification behavior.

IEigenAnt analysis will be extended to N–node binary chain problems following the method used for the SACO algorithm in future work. Moreover, the advantage of using the decoupling feature in IEigenAnt can be verified in some problems modeled as N–node binary chain problems especially in problems with constraints. Utilizing the decoupling feature and local pheromone update advantage in IEigenAnt in solving dynamic optimization problems is another suggestion for future research.

Acknowledgement

We would like to thank Prof. Jayadeva of IIT Delhi for useful discussions. Author AB's work is supported by a BPP grant from the Brazilian govt. agency CNPq, while author MM's work was funded by a fellowship from the Brazilian govt. agency CAPES.

References

[1] Iacopino C, Palmer P. The dynamics of ant colony optimization algorithms applied to binary chains. Swarm Intelligence, 2012,6(4): 343-377.

[2] Birattari M, Di Caro G, Dorigo M. Toward the formal foundation of ant programming. In International Workshop on Ant Algorithms., August 2002, pp.188-201.

[3] Stützle T, Dorigo M. A short convergence proof for a class of ant colony optimization algorithms. IEEE Transactions on Evolutionary Computation, 2002, 6(4): 358-365.

[4] Gutjahr W J. A graph-based ant system and its convergence. Future Generation Computer Systems, 2000, 16(8): 873-888 (2000).

[5] Gutjahr W J. ACO algorithms with guaranteed convergence to the optimal solution. Information Processing Letters, 2002, 82(3): 145-153.

[6] Gutjahr W J. On the finite-time dynamics of ant colony optimization. Methodology and Computing in Applied Probability, 2006, 8(1): 105-133.

[7] Dorigo M, Stützle T. An experimental study of the simple ant colony optimization algorithm. In WSES International Conference on Evolutionary Computation (EC'01)., 2001, pp.253-258.

[8] Jayadeva, Shah S, Bhaya A, Kothari R, Chandra S. Ants find the shortest path: a mathematical proof. Swarm Intelligence, 2013, 7(1): 43-62.

[9] Dorigo M, Gambardella L M. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53-66.

[10] Mahrueyan M. A proposal for an improved version of EigenAnt algorithm with applications to classes of combinatorial optimization problems [D.Sc Thesis]. Department of Electrical Engineering, Federal University of Rio de Janeiro (UFRJ), 2017.

[11] Yang X S, Deb S, Fong S. Metaheuristic algorithms: optimal balance of intensification and diversification. Applied Mathematics & Information Sciences, 2014, 8(3): 977-983.

[12] Meyer B. Convergence control in ACO. In Genetic and Evolutionary Computation Conference (GECCO)., 2004, late-breaking paper available on CD.

[13] Blum C, Roli A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison, ACM Computing Surveys (CSUR), 2003, 35(3): 268-308.

[14] Stützle T, Hoos H H. MAX–MIN ant system. Future Generation Computer Systems, 2000, 16(8): 889-914.

[15] Ke L, Feng Z, Ren Z, Wei X. An ant colony optimization approach for the multidimensional knapsack problem. Journal of Heuristics, 2010, 16(1): 65-83.

[16] Randall M. A systematic strategy to incorporate intensification and diversification into ant colony optimisation. In Proceedings of the Australian Conference on Artificial Life., 2003, pp.199-208.

[17] Zilberstein S. Using anytime algorithms in intelligent systems. AI Magazine, 1996, 17(3): 73-73.

[18] Yan Y, Sohn H S, Reyes G. A modified ant system to achieve better balance between intensification and diversification for the traveling salesman problem. Applied Soft Computing, 2017, 60: 256-267.

[19] Goldberg D E. Genetic algorithms. India: Pearson Education, 2006.

[20] Dijkstra E W. A note on two problems in connexion with graphs, Numerische Mathematik, 1959, 1(1): 269-271.