INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 12(2):265-275, April 2017.

Balancing Between Exploration and Exploitation in ACO

A.E. Negulescu, S.C. Negulescu, I. Dzitac

Alina Eugenia Negulescu
University POLITEHNICA of Bucharest
Romania, 060042 Bucharest, sector 6, Splaiul Independentei, 313
alina.lascu@gmail.com

Sorin Constantin Negulescu
Lucian Blaga University of Sibiu
Romania, 550024 Sibiu, Victoriei Bvd., 10
sorin.negulescu@ulbsibiu.ro

Ioan Dzitac
1. Aurel Vlaicu University of Arad
Romania, 310330 Arad, Elena Dragoi, 2
ioan.dzitac@uav.ro
2. Agora University of Oradea
Romania, 410526 Oradea, Piata Tineretului, 8
rector@univagora.ro

Abstract: In order to balance the preference of the artificial entities towards *exploration* or *exploitation* (in their transition rule), a novel technique is proposed for replacing the random function used by the classical Ant Colony Optimization (ACO) algorithms for solving the Traveling Salesman Problem (TSP). The proposed Beta Distribution function (**B**), or $random.betavariate(a, b)$ has the proven capability (depicted through test-runs) of influencing the algorithm's solution quality and convergence speed. Consequently, this paper will introduce in the related work section the classical ACO algorithm, with a focus on the *transition rule* used for choosing the next node in the problem's associated graph, followed by the related work on this topic, and it will continue with the introduction of the **B** function which will be presented both from a theoretical and practical perspective in relation with the scope: balancing between exploration and exploitation in order to improve the performance of the ACO algorithm for the TSP. The paper concludes that the B-EAS has the ability to find better solution than EAS for a set of benchmarks from the TSPLib library.

Keywords: ACO, TSP, **B**, Elitist Ant System (EAS), Beta-Elitist Ant System (B-EAS), transition rule.

1 Introduction

Swarm inspired algorithms, and specifically the ACO area, represent a steady interest over the past five years, as reflected in searches and recorded by Google Trends [7]. The constant focus on this family of algorithms is due to their simplicity in terms of modeling and, implicitly, the ease of understanding their functionality, which makes them, in turn, very appealing for computer scientists aiming to solve optimization problems.

Since they are inspired from nature, the principles of the ACO algorithms are easy to grasp. Their well-known applicability is for solving the TSP, which can also be very easily explained but, unfortunately, not that simple to solve: a traveling salesman has a list of cities which he must visit and return back to the starting point; there are several conditions that must be met: he must pass through each city only once, and he must find the shortest route to fulfill this task, as otherwise, longer routes will be associated with bigger costs. This problem can be extrapolated to the world of ants which, foraging for food, need to find the shortest path between the nest and the food source. Because of these resemblances in terms of tasks, the ant colony inspired algorithms are directly applicable for solving the TSP.

Pertaining the parameters, by mirroring the natural ant colony model, the classical ACO algorithm operates with artificial *entities* (i.e. the ants), *positive feedback* using pheromone deposit in the environment, *heuristic information* regarding distances in the search-space and *negative feedback* resulted from pheromone evaporation. For all intents and purposes, these parameters, which are inspired from nature, are the most influential ones in any ACO algorithm.

Other common traits between ACO algorithm variants are the initialization steps that include:

- loading the problem's associated graph;
- setting up the initial parameter's values;
- initialization of the pheromone trails on the graph's arches (using a minimum quantity of pheromone) as to "kick-start" the algorithm as otherwise, it would take a couple of iterations until the artificial ants would do this instead and this would result in additional computational costs.

Even though the last described step is obviously not characteristic to real ants, the model has been slightly altered so as to increase the convergence speed of the algorithm towards a solution.

After performing the initialization, as long as the algorithm's stop condition is not met (represented by, for instance, a maximum number of iterations or time limitation), every artificial ant needs to perform a set of steps, part of the construct solutions function:

- *choose a starting node on the graph* – predictably (ant 1 in node 1, ant 2 in node 2 and so on) or randomly;
- *choose next node* until all nodes are visited only once (function on which this paper is centered), during which the "artificial ant has to decide" which nodes are more attractive and choose one of them based on a probability formula that will be presented shortly;

After all artificial ants have constructed solutions:

- *update pheromone trail* function is employed for depositing a pheromone quantity on all the arches of the graph visited by the artificial ants during a tour, which, following the natural model, represents the positive feedback, or reinforcement; in this way the arches that have been used by multiple ants will have a greater pheromone intensity;
- *evaporate pheromone trail* function will act as negative feedback in order for the algorithm to "forget" old and inefficient trails; this step is paramount as without it, the algorithm would converge too fast and get trapped in local optima;
- *store best solution* function will compare all tour lengths of the artificial ants and record the smallest length value along with the order in which the nodes have been visited, for future comparisons.

Having presented the context, the paper will focus on the transition rule of ACO algorithms which is based on a probability formula (as mentioned earlier). The state of the art is presented in "Origins and related work", while the "Rationale and approach" section pinpoints the drawbacks of current approaches and presents the *beta distribution function* as a possible solution. The section entitled "The model", describes the changes that have been applied to the EAS algorithm in order to support the identified approach for improvement: Beta-Elitist Ant System (B-EAS) algorithm. The "Conclusions and intentions" section rounds up the paper by presenting the strengths and weaknesses of the B-EAS algorithm, together with the identified future research intentions in this direction.

2 Origins and related work

Initial experiments, conducted with real ants (*Linepithaema humile*), proved that when ants have to choose between two possible paths of different lengths (when going between a food source and the nest) they will initially travel on any of them with an equal probability but, as time passes, the shorter one will have a higher amount of deposited pheromone (reinforced by multiple, shorter trips) whereas the longer one will have a lower amount of pheromone (due to fewer, longer trips). Moreover, considering that the evaporation speed of the pheromone is equal, the intensity of the pheromone on the longer path will decrease faster. Even though the pheromone intensity on the two paths was different, a very small number of ants were still choosing the one with smaller pheromone intensity from time to time, which lead to the conclusion that the ants were deciding in a *probabilistic* manner.

The findings in this "double-bridge experiment" [8] were adopted when the initial Ant System (AS) algorithm was modeled [5]. As such, almost all ACO algorithms are using a probability formula for determining the next node. The formula coined by AS, which is reproduced below (unaltered) defines, as called by its authors, a *random-proportional rule*:

$$p_{i,j}^{k}\left(t\right)=\begin{cases}\frac{[\tau_{ij}(t)]^{\alpha}\cdot[\eta_{ij}(t)]^{\beta}}{\sum_{l\in N_i^k}[\tau_{il}(t)]^{\alpha}\cdot[\eta_{il}(t)]^{\beta}} & if j\in N_i^k\\ 0 & if j\notin N_i^k\end{cases}\qquad(1)$$

Formula 1 corresponds to the probability p at the moment t for an ant k visiting node i to choose node j as the next node in which it will go. α and β are two general parameters used for weighting the pheromone intensity τ_{ij} and the visibility η_{ij} (defined as $1/distance$) between nodes i and j when computing (at the moment t) the cost of choosing node j. The set N_i^k contains the nodes the ant k is allowed to choose (i.e. all the nodes which were not yet visited, as constrained by the TSP). So, the formula can be expressed as being the ratio between the cost of choosing node j and the sum of costs for all the other nodes not yet visited. Of course, this probability is 0 if node j is already visited.

Before the above presented formula is applied, a random real number between 0 and 1 is generated, representing the probability value. Afterwards, $p_{ij}^{k}\left(t\right)$ is calculated for as many nodes $j\in N_i^k$ as necessary, until the sum of these computed costs is equal or greater than the selected probability and the last j node is chosen to move to. In this way, in many cases the formula is not applied for all the nodes not yet visited, sparing some time. The steps above are repeated until no more nodes are left to visit, resulting a list of nodes visited by the ant k.

As a further improvement of the initial AS algorithm, Dorigo and Gambardella proposed in the Ant Colony System (ACS) algorithm a new transition rule that included *exploration* and *exploitation* traits [6]. This improvement, as described by its authors, is due to the *pseudo-random-proportional rule* for choosing the next node in the itinerary, where the ant k in the node i selects the next node j with a probability of $0\% \leq q_0 \leq 100\%$. As such, a q_0 value is a priori selected when the algorithm starts, whereas the next node is selected by applying one of the following formulas depending on a randomly generated value q:

- if $q < q_0$ then the node for which $\tau_{ij}\left(t\right)\cdot\left[\eta_{ij}\left(t\right)\right]^{\beta}$ is maximum, is used;

- else, the node is selected by using the same formula introduced in the AS algorithm.

As described in [6], "ACS algorithm's state transition rule provides a direct way to balance between *exploration of new edges* and *exploitation of a priori and accumulated knowledge* about the problem". Along with other several improvements, ACS' proven efficiency was demonstrated by its authors through a series of experiments whose results had shown that this algorithm outperformed other algorithms inspired from nature, as for instance, simulated annealing and evolutionary computation.

Another improvement of the AS was the Elitist Ant System (EAS) [5] that was reinforcing the pheromone intensity on the arches belonging to the tour of the ants that found the best solutions at the moment t. In this way one can state that the elitist ants can deposit a greater amount of pheromone. The experiments of the before mentioned authors indicated that a relatively small number of elitist ants can help the performance of the algorithm.

Nevertheless, as depicted in [1], the author claims that "the transition rule suggested in the original Ant System and used extensively in modified versions [...] uses a power law scaling procedure to define the relative importance of the pheromone intensity and local information. [...] This form of transition rule, however, often leads to premature convergence of the ant algorithms to suboptimal solution when used with elitist strategy of pheromone updating". The author proposed, as a result, an alternative transition rule to be used with elitist strategy of pheromone updating in ant algorithms. Moreover, the author of [1] considers that the classical transition rule presented in formula 1 would be the source of the local optima issue which called for the design of the MAX-MIN Ant System (MMAS) algorithm. Further explained in [1], the reason for which the transition rule is not efficient is that the influence of the pheromone intensity (α) will play a main role in the decision making process of an ant choosing the next node to visit, while the heuristic information will have no impact. This is determining, as a result, the stagnation of the search. Consequently, the author proposes an alternative method of transition that would prevent the pheromone intensity from "dominating the ants decision table at all stages of computation". This formula is represented as:

$$p_{i,j}^{k}(t) = \begin{cases} \frac{\alpha\tau_{ij}(t)\cdot\beta\eta_{ij}}{\sum_{l\in N_i^k}[\alpha\tau_{il}(t)\cdot\beta\eta_{il}]} & ifj \in N_i^k \\ 0 & ifj \notin N_i^k \end{cases} \tag{2}$$

The author in [1] claims that, this new transition rule based on addition, will ensure that for “the properly adjusted values of the pheromone, and local heuristics sensitivity parameters, the local heuristic term η_{ij} will always have a saying in forming the decision table, irrespective of the distribution of pheromone intensities”. Moreover, this method has the advantage of "reducing the number of controlling parameters, as one can always assume the value of unity for one of these parameters and adjust the value of the other one for the best performance of the method", as described by the aforementioned author.

Other solutions that were targeting the improvement of the original AS algorithm were the Rank-Based Ant System (RBAS) [3], Best-Worst Ant System (BWAS) [4], HyperCube-Ant Colony Optimization (HC-ACO) [2] and Grahp-Based Ant System (GBAS) [9], to name a few of the most common. Also, a respectable number of approaches went beyond the initial paradigm, implementing pheromone correction strategies [14] or creating hybrid solutions involving evolutionary computation [15] and neural networks [13].

Whereas in the classical ACO algorithm, α and β parameters are global, in the model presented in [11], they become local and "personal" to each individual ant. In this way, ants can behave differently in their transition rule. Considering that the resource costs associated with this approach is minimal (in terms of memory) the benefits are various: this variant of ACO algorithm has the ability to self-tune depending on the problem type and size (if local α and β parameters are adjusted based on the tour lengths found by each ant), the convergence speed can be improved and the parameters values can be easily adjusted automatically. All these benefits can assist in addressing the stagnation issue (i.e. local optima) that is typical to all meta-heuristic algorithms.

3 Rationale and approach

As it can be depicted from the selected examples provided in the previous section, there is an on-going interest regarding the improvement of the ACO algorithms, due to their proven potential for solving combinatorial optimization problems. Some improvement approaches are focusing on the *update pheromone intensities rule*, others are looking towards the parameters' values setting, while this paper investigates the *transition rule* which the ants use to move through the nodes in the graph.

All algorithms from the meta-heuristic class are relaying on a probability that implies the use of randomly generated numbers in different manners. Without this intrinsic randomness factor the behavior would tend to be the same as the best-first algorithm and the solutions quality would be poor.

The approach presented in this paper, that would allow to change the algorithm's behavior towards exploration or exploitation, is based on another type of probability: **B**, that would allow a *beta-proportional rule.*

Beta distribution can be defined, according to [10] as "a mean of distribution probabilities", that essentially, represents "all the possible values of an unknown probability". This method is reasonably employed by statistics and probability theory and is defined in $[0,1]$ domain, having two positive shape parameters: a and b. Its application is widely used in diverse areas as a technique that allows influencing the behavior of random variables limited to ranges of limited lengths. Given these features, it can be used as a proper model for random behavior (pertaining the uncertainty of an outcome).

For all practical purposes, a test-run may only have two possible results: either success (with a probability of x) or failure (with a probability of $1-x$). As explained by the same author, this can be probabilistically illustrated by assigning to x a uniform distribution in the $[0,1]$ domain, as x being a probability, it can only take values between 0 and 1. Also, the uniform distribution assigns respective probability density to every entity in the domain, which effectively means, that no possible value of x is *a priori*, more likely than the other. As such, for n independent iterations performed during an experiment, there can result both k successful outcomes, as well as $n-k$ failed ones. Consequently, the conditional distribution of x, based on the observation of k successful outcomes resulted from n iterations, represents, actually, a beta distribution with $k+1$ and $n-k+1$ parameters.

Therefore, as depicted in [10], for x, absolutely continuous random variable: $R_x = [0,1]$ and $a, b \in \mathbb{R}_{++}$, x has a beta distribution with a, b as shape parameters of x's probability density function, presented through the following formula, where $\mathbf{B}\left(a,b\right)$ represents the $\mathbf{B}$ function:

$$f\left(x\right) = \begin{cases} \frac{1}{B(a,b)} x^{a-1}\left(1-x^{b-1}\right) & if x \in R_x \\ 0 & if x \notin R_x \end{cases} \tag{3}$$

To visually grasp the behavior of this function for different parameter values, below are presented several histograms. The data set is represented by 10000 samples with values within the $[0,500]$ domain on the abscisa and the number of randomly generated samples is displayed on the ordinate.

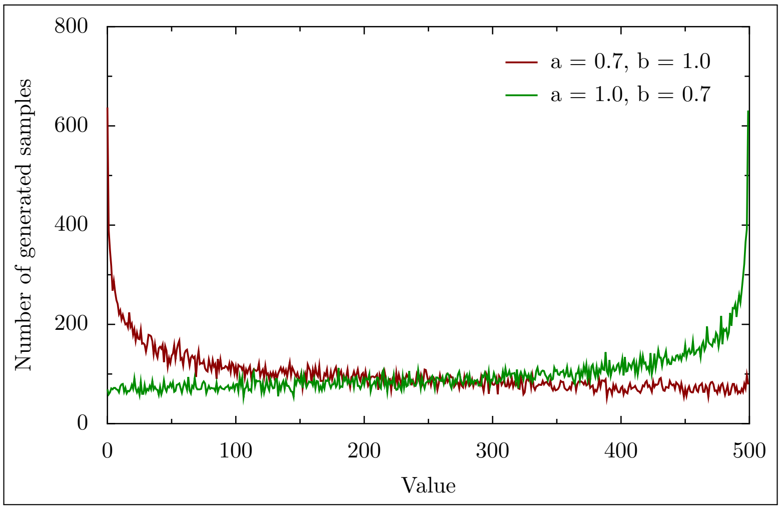

Figure 1: Histogram for beta distribution function with one sub-unitary and one unitary argument

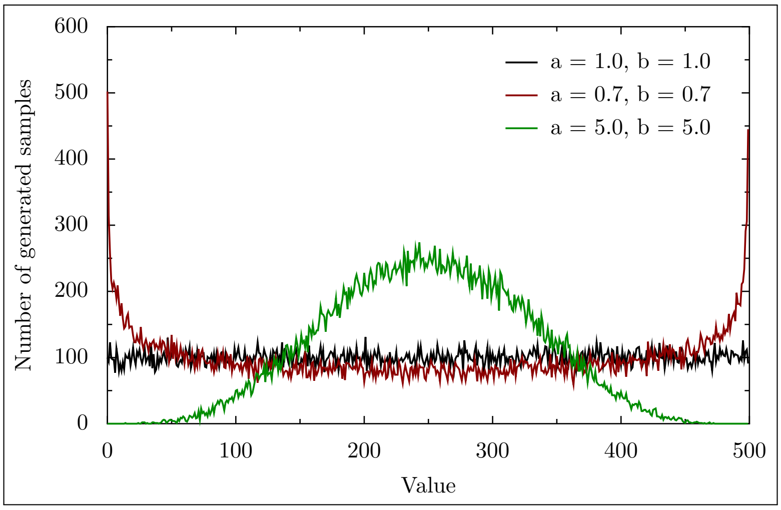

Figure 2: Histogram for beta distribution function with equal arguments values

According to the visual representation of **B** in figures 1 and 2, one can easily depict the fact that this function is highly flexible since it allows, depending on the values of its parameters, to randomly generate a higher number of values at one end or the other of the interval, a uniform distribution and a normal (inverse) distribution. Also, it is important to mention that, if the a and b parameters equal 1, this function is behaving exactly like the classic random function where all the values have an equal probability of being generated and the algorithm would perform exactly like the AS algorithm.

4 The model

Based on the previously introduced approach, consisting in replacing the *random-proportional* transition rule with the *beta-proportional* one, this section describes in detail, the model and the method used for testing it. The foundation of the model is the EAS algorithm where the herein proposed transition rule has been modified to include the following steps:

- Determine the costs of moving from the current node to all the nodes not yet visited by applying the formula for computing $p_{ji}^{k}(t)$ using the same formula used by EAS (as presented in the origins and related work section).
- Store these costs together with the corresponding node number in a vector by inserting them in such a way that the vector is ordered increasingly (so the nodes that have smaller costs are at the beginning of the vector). Obviously, generating this vector in an ordered fashion has the advantage that it is much faster than generating and then sorting it.
- Generate a random number according to **B** that will indicate the position (in the vector) of the node selected as the next node to move to. The values of the parameters for the function must be chosen in such a way that it will generate with a higher probability small values so the elements at the beginning of the vector have a higher chance to be selected.

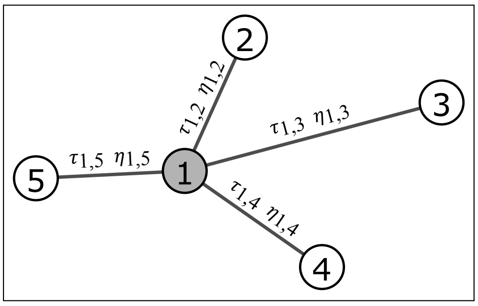

Figure 3: Example of associated graph with heuristic and pheromone information

For a better understanding of the steps involved in the new model called Beta-Elitist Ant System (B-EAS), the graph in figure 3 is provided as an example. Presuming that an ant starts in node 1, the cost of going to nodes $2, 3, 4$ and 5 is taking into consideration the pheromone intensity and the visibility on each arch.

The ordered vector, resulted after computing the costs, is presented in figure 4, together with the probability of selection according to **B** function. As it can be depicted in the example, node 3 which involves the smallest cost in terms of visibility and pheromone intensity, has a higher probability of being selected than $2, 4$ and 5 if the arguments of **B** are properly chosen.

Test-runs were conducted for a full factorial experiment in order to determine appropriate values for the a and b parameters of **B**. Subsequently, in order to generate smaller values with

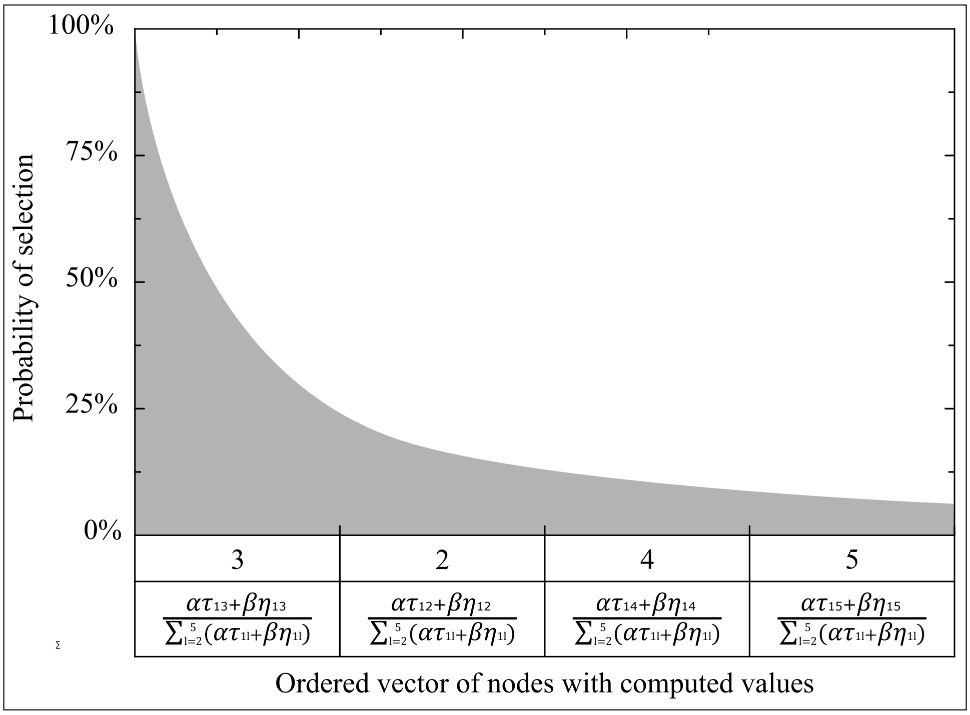

Figure 4: Example of selection probability for the ordered vector's elements

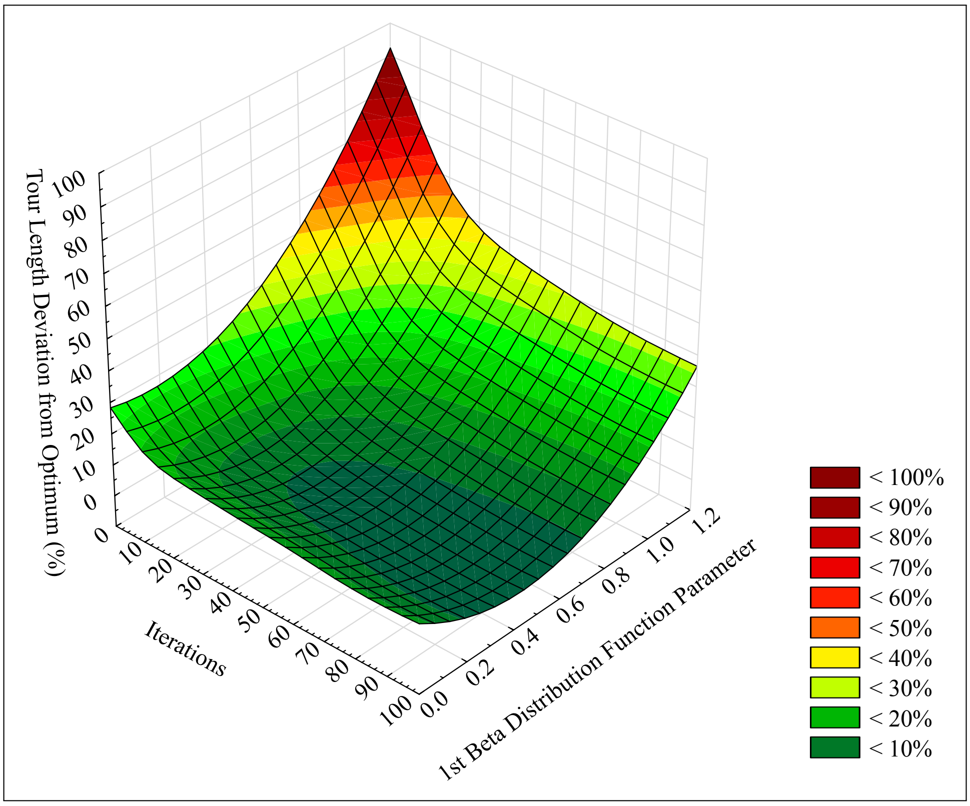

Figure 5: Solution length versus 1^{st} parameter of beta distribution function)

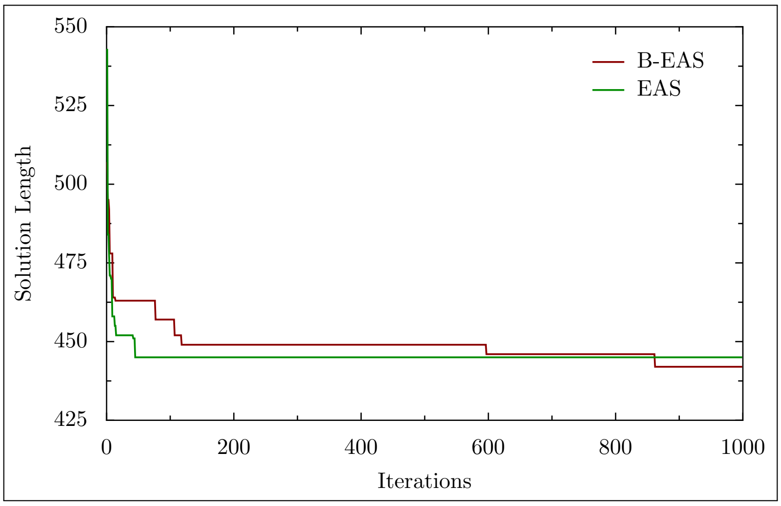

Figure 6: Comparison between EAS and B-EAS algorithms running on eil51 benchmark

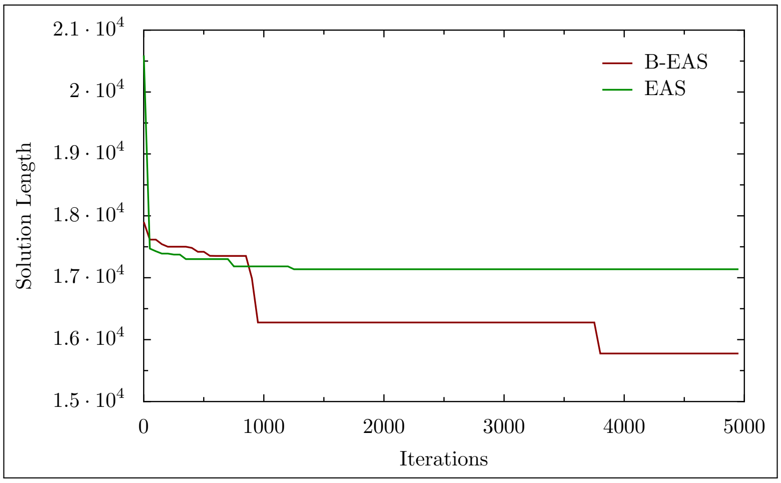

Figure 7: Comparison between EAS and B-EAS algorithms running on d198 benchmark

a higher probability, the values for parameter a have been tested within $[0.1, 1]$ domain, while b was maintained constant at 1. The results are presented further on in figure 5, where, for a set of problems with known solutions, the algorithm was constrained to run for 100 iterations. The quality of the obtained solutions were averaged and compared in a percentual manner to the known solutions so, the optimal solution obtained is 0% deviating from the best known ones (represented in darker color in the graph).

The conclusion of this experiment is that the optimal parameters' values of **B** function, used for generating random numbers, are $a = 0.4$ and $b = 1$. As displayed in figure 5 (where the dependent variable *tour length* is expressed versus parameter a and the *number of iterations* dependent variables), these values are driving the algorithm towards finding better solutions (with shorter lengths), and with a higher convergence speed.

A comparison between the EAS and B-EAS algorithms using standard benchmarks from the TSPLib [12] in regards to solutions quality and convergence speed is presented in figures 6 and 7. We have observed that without modifying the a and b parameters during the test-runs, the B-EAS was converging slower than EAS, but it continued to find better solutions, whereas EAS would have stopped in a local optima.

As stated before, when parameters have the values $a < 1$ and $b = 1$, ants tend to choose with a higher probability the nodes that give smaller costs of travel (*exploitation*), but there is also the probability to choose nodes that imply higher costs (*exploration*). This is why the graphs are displaying large improvements that are not further more developed over a high number of iterations, especially when a is much smaller than 1.

Conclusions and intentions

The paper presents a new algorithm (B-EAS) that can be stirred towards *exploitation* or *exploration* by modifying the a and b parameters used in the *beta-proportional transition rule*. Specifically, the hereby authors propose replacing the classical random function for choosing the next node with the beta distribution function which allows a more flexible way of generating random numbers.

To validate the proposed model, the optimal parameter values of **B** were determined for a set of benchmark problems using a full factorial experiment. The next logical step was to compare through test-runs the performance of B-EAS with the original algorithm from which it was derived (EAS). The results have shown that B-EAS finds better solutions than EAS, given enough time or iterations.

Considering the approach introduced by the same authors in [11], where the global parameters of the algorithm (α, β and ϵ) were transformed into local parameters for each artificial ant, creating thus a "population" of agents with individual characteristics (personalities), the same approach could be applied for the a, and b parameters of the **B** function, as well. We consider that this research direction has the potential to create a genetic-featured ACO, where the algorithm could evolve and "tune" itself.

Bibliography

[1] Afshar, M.H. (2005); A new transition rule for ant colony optimization algorithms: application to pipe network optimization problems, *Engineering Optimization*, 37 (5): 525-540.

[2] Blum, C.; Roli, A.; Dorigo, M. (2001); HC-ACO: The Hyper-Cube Framework for Ant Colony Optimization, *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 399-403.

[3] Bullnheimer, B.; Hartl, R.F.; Strauss, C. (1999); A New Rank Based Version of the Ant System: A Computational Study, *Central European Journal for Operations Research and Economics*, 25-38.

[4] Cordón, O.; de Viana, I.F.; Herrera, F.; Moreno, L. (2000); A new ACO model integrating evolutionary computation concepts: The best-worst Ant System. *In M. Dorigo, M. Middendorf, & T. Stuützle (Eds.), Abstract Proceedings of ANTS 2000-From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms*, Brussels, IRIDIA, Université Libre de Bruxelles, 22-29.

[5] Dorigo, M.; Maniezo, V.; Colorni, A. (1996); Ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 26 (1): 29-41.

[6] Dorigo, M.; Gambardella, L. (1997); Ant Colony System: A Cooperative Learning. *IEEE Transactions of Evolutionary Computation*, 1 (1): 53-66.

[7] Google (2016). Google Trends, Retrieved from `https://www.google.com/trends/explore?q=AntColonyOptimization`.

[8] Goss, S.; Aron, S.; Deneubourg, J.L.; Pasteels, J.M. (1989); Self-organized shortcuts in the Argentine ant, *Naturwissenschaften*, 76, 579-581, Springer-Verlag.

[9] Gutjahr, W.J. (2000). A Graph-based Ant System and its convergence, *Future Generation Computer Systems*, 873-888.

[10] Kerns, G. (2011); Introduction to Probability and Statistics Using R, *IPSUR*, ISBN: 978-0-557-24979-4.

[11] Negulescu, S.; Dzitac, I.; Lascu, A. (2010); Synthetic Genes for Artificial Ants. Diversity in Ant Colony Optimization Algorithms, *International Journal of Computers Communication & Control*, 5 (2): 216-223.

[12] Reinelt, G. (1991); TSPLIB - A Traveling Salesman Problem Library. *ORSA Journal on Computing*, 376-384.

[13] Susnea, I.; Axenie, C. (2015); Cognitive maps for indirect coordination of intelligent agents, *Studies in Informatics and Control*, 24(1): 111-118.

[14] Tuba, M.; Jovanovic, R.(2013); Improved ACO Algorithm with Pheromone Correction Strategy for the Traveling Salesman Problem, *International Journal of Computers Communications & Control*, 8(3): 477-485.

[15] Wei, X.; Han, L.; Hong, L. (2014); A modified ant colony algorithm for traveling salesman problem, *International Journal of Computers Communications & Control*, 9(5): 633-643.