World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:10, No:1, 2016

SMART:Solution Methods with Ants Running by Types

Nicolas Zufferey

Open Science Index, Mathematical and Computational Sciences Vol:10, No:1, 2016 publications.waset.org/10003380.pdf

Abstract—Ant algorithms are well-known metaheuristics which have been widely used since two decades. In most of the literature, an ant is a constructive heuristic able to build a solution from scratch. However, other types of ant algorithms have recently emerged: the discussion is thus not limited by the common framework of the constructive ant algorithms. Generally, at each generation of an ant algorithm, each ant builds a solution step by step by adding an element to it. Each choice is based on the *greedy force* (also called the visibility, the short term profit or the heuristic information) and the *trail* system (central memory which collects historical information of the search process). Usually, all the ants of the population have the same characteristics and behaviors. In contrast in this paper, a new type of ant metaheuristic is proposed, namely SMART (for Solution Methods with Ants Running by Types). It relies on the use of different population of ants, where each population has its own personality.

Keywords—Optimization, Metaheuristics, Ant Algorithms, Evolutionary Procedures, Population-Based Methods.

I. Introduction

As exposed in [1], modern methods for solving complex optimization problems are often divided into *exact* methods and *metaheuristic* methods. An exact method guarantees that an optimal solution is obtained in a finite amount of time. However, for a large number of applications and most real-life optimization problems, which are typically NP-hard, such methods need a prohibitive amount of time to find an optimal solution. For these difficult problems, it is preferable to quickly find a satisfying solution. If solution quality is not a dominant concern, then a simple *heuristic* can be employed, but if quality plays a critical role, then a more advanced *metaheuristic* procedure is recommended. There are mainly two classes of metaheuristics: *local search* and *population based* methods. The former type of algorithm works on a single solution (e.g., descent local search, tabu search, variable neighborhood search), whereas the latter makes a population of (pieces of) solutions evolve (e.g., genetic algorithms, ant colonies, adaptive memory algorithms). The reader is referred to [2] for a recent book on metaheuristics.

Ant algorithms were first introduced in [3], and relevant surveys are [4], [5] and [6]. Different roles are possible for each ant, ranging from a negligible help in the decision process to a refined local search technique [7]. The common points of all the ant algorithms are the following [8].

- A population of N ants is handled.
- Each ant is able of self-adaptation (independently of the other ants).
- The ants are able to collaborate (e.g., exchange information).
- At each generation, solutions are provided based on the ants' activity.
- The output of the method is the best encountered solution during the search process.

Nicolas Zufferey is with the Geneva School of Economics and Management, GSEM - University of Geneva, Blvd du Pont-d'Arve 40, 1211 Geneva 4, Switzerland (e-mail: n.zufferey@unige.ch).

As presented in [7] and [9], in most ant algorithms, the role of each ant is to build a solution step by step. At each step, an ant adds an element to the current partial solution. Each *decision* or *move* m is based on two ingredients: the *greedy force* $GF(m)$ (short-term profit) and the *trail* $Tr(m)$ (information obtained from other ants). The probability $p_i(m)$ that ant i chooses decision m is given by (1), where α and β are parameters, and M_i is the set of admissible decisions that ant i can make.

$$p_i(m) = \frac{GF(m)^\alpha \cdot Tr(m)^\beta}{\sum_{m' \in M_i} GF(m')^\alpha \cdot Tr(m')^\beta} \quad (1)$$

Let M be the set of all possible decisions. When each ant of the population has built a solution, the trails are generally updated as in (2):

$$Tr(m) = \rho \cdot Tr(m) + \Delta Tr(m), \ \forall m \in M \quad (2)$$

$\rho \in]0,1[$ is a parameter representing the evaporation of the trails, which is usually close or equal to 0.9. $\Delta Tr(m)$ is a term which reinforces the trails left on decision m by the ant population. That quantity is usually proportional to the following elements:

- the number of times the ants have made decision m;
- the quality of the obtained solutions when decision m was made.

More precisely, let N be the number of ants, then $\Delta Tr(m)$ is updated as in (3), where $\Delta Tr_i(m)$ is proportional to the quality of the solution provided by ant i if it has made decision m.

$$\Delta Tr(m) = \sum_{i=1}^{N} \Delta Tr_i(m) \quad (3)$$

The pseudo-code of a classical ant method is given in Algorithm 1. A *generation* consists in performing steps (1) to (4). A stopping condition can be a maximum number of generations or a maximum time limit.

The contribution of this paper consists in designing a new type of ant metaheuristics denoted SMART (for Solution Methods with Ants Running by Types). A generic presentation of SMART is proposed in Section II. Then, in Section III, relying on [10], the adaptation of SMART is briefly

Algorithm 1 Classical ant metaheuristic

Initialize the ant system.

While no stopping condition is met, **do:**

(1) for $i = 1$ to N, do: ant i builds a solution s_i step by step based on (1);

(2) *intensification* (optional): apply a local search to some solutions of $\{s_1, \ldots, s_N\}$;

(3) update $s^\star$ (best ever encountered solution found during the search);

(4) update the trails by the use of a subset of $\{s_1, \ldots, s_N\}$;

Output: solution $s^\star$.

described for the *VRP*, which is the well-known *Vehicle Routing Problem*.

II. Presentation of SMART

Two main elements define the personality of an ant: the role assigned to the ant, and the way to use the greedy forces and the trail system in order to select a move (i.e., to make a decision). On the one hand, and as exposed in [7], three possible roles are possible:

(R1) *Constructive ants*: each ant builds a solution step by step. At each step, an ant adds an element to the current partial solution. It is the classical role found in most of the literature on ant algorithms.

(R2) *Local search ants*: a local search algorithm starts with an initial solution and tries to improve it iteratively. At each iteration, a modification (called a move) of the current solution s is performed in order to generate a neighbor solution s'. The definition of a move, that is the definition of the neighborhood structure, depends on the considered problem. Well-known local search methods are the descent local search, simulated annealing, tabu search, and variable neighborhood search.

(R3) *Improving ants*: a single ant can help to make a decision within a procedure which makes only one solution evolve. In other words, each ant helps to move from a current solution to a neighbor solution by performing moves on the current solution.

On the other hand and as detailed in [8], in every ant algorithm, each decision m relies on the greedy force $GF(m)$ and on the trail system $Tr(m)$. The selection of a decision m is based on a tradeoff between $GF(m)$ and $Tr(m)$. Such a tradeoff can have at least three different forms, as listed below.

(T1) *Give a chance to each tradeoff*: use (1) with comparable values for α and β. The more different are these two parameter, the least balanced is the tradeoff. Often, in order to better control the balance between GF and Tr, such values are normalized within interval $[0, 1]$. Most of the ant literature is based on this tradeoff.

(T2) *Focus on the best tradeoff*: in order to be more aggressive and thus giving more power to the move associated with the best tradeoff between $GF(m)$ and $Tr(m)$, the following technique is sometimes used. At each step, select strategy $\mathcal{S}_1$ with probability p (parameter), and strategy $\mathcal{S}_2$ with probability $(1 - p)$. Strategy $\mathcal{S}_1$ consists in selecting a decision m according to (1), whereas strategy $\mathcal{S}_2$ consists in selecting the decision m maximizing $GF(m) \cdot Tr(m)$. The smaller p is, the more aggressive is the method.

(T3) *Use sequentially the greedy forces and the trails*: a selection strategy avoiding the use of a probability function was first proposed in [11]. It works as follows. At each step associated with an ant, let B be the set of decisions with the largest greedy force (resp. trail) values. Then, the selected decision is the one in B with the largest trail (resp. greedy force) value. Of course, this process is only interesting if $|B| > 1$, otherwise the trails (resp. greedy forces) will have no impact on the search.

The performance of a metaheuristic can be evaluated according to several criteria [1]. The most relevant criteria are the following:

(A) *Quality*: the value of the obtained results, according to a given objective function.

(B) *Speed*: the time needed to get good results.

(C) *Robustness*: the sensitivity to variations in problem characteristics and data quality.

(D) *Ability to take advantage of problem structure*, considering that efficiency often depends on making effective use of properties that differentiate a given class of problems from other classes.

(E) *Ease of adaptation*: the ability to organize the method so that it can appropriately apply to different specific classes of problems.

Because of the unlimited number of personalities, the proposed SMART methodology can have a good behavior according to all the above criteria but (E), for which basic constructive methods (e.g., the greedy algorithm, GRASP) or classical local search metaheuristics (e.g., the descent local search, simulated annealing, tabu search) rank better. Indeed, in order to design SMART for a specific problem, all the following ingredients have to be defined:

- A way to encode a solution s.
- An objective function f.
- The greedy force an the trail system.
- The q (parameter) different personalities $P_1, P_2, \ldots, P_q$.
- The stopping condition (e.g., a time limit, a specific number of generations).

Elements are provided below in order to justify the potential of SMART for each of the above criterion, from (A) to (D).

(A) A personality can have an aggressive (and thus efficient in the short term) behavior by focusing on the best moves. For his purpose, the greedy forces should be favored when compared to the trails. Such a personality can quickly allow the involved ants to generate solutions with a fairly good quality. The above strategies (T2) or (T3)

Open Science Index, Mathematical and Computational Sciences Vol:10, No:1, 2016 publications.waset.org/10003380.pdf

Open Science Index, Mathematical and Computational Sciences Vol:10, No:1, 2016 publications.waset.org/10003380.pdf

can be used to reach this goal.

(B) A small number N of ants and aggressive selection strategies (e.g., strategy (T2) with a small value of p, or strategy (T3)) should be favored in order to speed up the method. In addition, a filtering technique (e.g., [12]) could be used to reduce the search space.

(C) SMART has a natural potential to perform well on robustness, because of the following elements: the use of a population of ants; the consideration of the short term profit (namely the greedy force) and the learning mechanism (namely the trail system); the use of various personalities.

(D) SMART is likely to take advantage of problem structure, at least because of the total freedom allowed for encoding the solution.

Let N_j be the number of ants with personality j (for $j \in \{1, \ldots, q\}$). It can be assumed that $\sum_{i=1}^{q} N_i = N$. The pseudo-code of SMART is given in Algorithm 2.

Algorithm 2 SMART

Initialize the ant system.

While no stopping condition is met, **do:**

(1) for $j = 1$ to q, do:
for $i = 1$ to N_j, do: ant i provides a solution s_i^j based on the greedy forces and the trail system;

(2) *intensification*: apply a local search to some s_i^j's;

(3) update $s^\star$ (best ever encountered solution found during the search);

(4) update the trails by the use of some s_i^j's;

Output: solution $s^\star$.

III. SMART for the *VRP*

Based on [10] to which the reader is referred for the details, the main elements involved in the design of SMART for the *VRP* (vehicle routing problem) are now exposed.

First, the *VRP* consists in designing the route of each of the k identical vehicles with the aim of minimizing the total traveled distance f (or the total cost or the total travel time). All vehicles are initially in a depot, where each route starts and ends. Each client v (with demand $D(v)$) has to be visited once by the collection of routes. The problem is defined in an undirected graph $G = (V, E)$, where $V = \{v_0, v_1, \ldots, v_n\}$ is the vertex set and $E = \{(v_i, v_j) \mid v_i, v_j \in V, i < j\}$ is the edge set. Note that v_0 is the depot and the other vertices are clients. The following lexicographical approach is generally used: minimize k, then the total distance f. The two most well-known constraints associated with the VRP are: (1) *capacity*: each vehicle has a limited capacity Q, thus the demand of each route cannot exceed Q; (2) *autonomy*: each vehicle has a limited autonomy A, thus the total duration of each route cannot exceed A. Several extensions of the *VRP* can be found in the literature. In this paper, only the capacity constraint is considered, which is the most studied version of the *VRP*. For survey papers on the *VRP*, the reader is referred to [13]–[18].

The following methods are compared for the *VRP*. They will be summarized below.

- *GR* is a greedy constructive algorithm with restarts. It is the core procedure for the ant algorithm.
- *ANT* is an ant algorithm where each ant is a constructive heuristic derived from *GR* by mainly adding a trail system.
- *AL* is derived from *ANT* by adding local search techniques to improve the solutions provided by the ants.
- *ALM* is derived from *AL* by adding an intensification component at the very beginning of the process, where some parts of the best ever generated solution $s^\star$ are integrated in the solutions constructed by the ants.
- *ALMP* is derived from *ALM* by giving a specific personality to each ant. Four types of personality will be used.

GR is a greedy constructive algorithm with restarts. It consists in sequentially constructing each of the k routes. The procedure starts a new route R by choosing randomly an unserved client $v \in \{v_1, \ldots, v_n\}$, and creates a tour $v_0 - v - v_0$. Let $C(R)$ be the capacity of route R (defined as the vehicle capacity, minus the demands $D(R)$ of all the clients belonging to R). Then, for all the unserved clients v such that $D(v) \leq C(R)$ (called the *R-available* clients), a move $m = (v, p, R)$ can be performed, which consists in inserting client v at position p (between two clients v_i and v_j, or between the depot v_0 and one client v_i) in route R. *GR* is performed until there is no more R-available client v. When this occurs, a new route is started by choosing randomly an unserved client. The process stops when all the clients have been served (feasible solution), or when it is not anymore possible to serve a client with one of the k vehicles (unfeasible solution).

ANT is an ant algorithm. The role of each ant is to build a solution with a 2-phase algorithm denoted *2PH*, where a trail value is associated with each edge. A move is selected based on the trail values, among the ones with the most promising greedy forces (in other words, the technique (T3) is used). In the first phase (P1), the routes are sequentially built and extended as in *GR*, whereas in the second phase (P2), the unserved clients are sequentially considered to fill any of the existing routes. In other words, (P1) works tour by tour, whereas (P2) works client by client (by order of decreasing demands, which improves the likelihood of the solution to be feasible). To move from (P1) to (P2), the key idea is to stop the construction of a route R in (P1) when only poor R-available insertions are possible (i.e., do not fill route R just to fill it, because these R-available clients might be much more efficiently served by other routes).

Often, in order to get competitive results, it is unavoidable to apply a local search method (e.g., a descent method, tabu search) to the solutions provided by the classical constructive

Open Science Index, Mathematical and Computational Sciences Vol:10, No:1, 2016 publications.waset.org/10003380.pdf

ants [8]. *AL* is derived from *ANT* as follows. At the end of each generation, before updating the trail system (i.e., at step (2) of Algorithm 1), the following local search techniques are sequentially applied to the elite solutions: the *2-opt* [19], the forward *Or-exchange* and the backward *Or-exchange* [20]. This sequence of three local search procedures is restarted until no more improvement is encountered by any of the procedure.

ALM is derived from *AL* by adding an intensification component at the beginning of *2PH*, before (P1). This component (P0) consists in copying some of the routes of $s^\star$ when generating a solution s with the considered ant.

ALMP is derived from *ALM* and involves different ant personalities. A specific personality is assigned to each of the N ants of the population. The personality intervenes anytime the ant makes a decision, which consists in selecting a move among the eligible ones. Four ant personalities are proposed: *Normal Ants* (*NA*), *Follower Ants* (*FA*), *Moody Ants* (*MA*) and *Innovative Ants* (*IA*). These characteristics are likely to belong to any group of individuals working together to reach a common goal. In order to work with a well-balanced ant society, $N/4$ ants of each personality are used (with N tuned to 12).

- *NA* corresponds to the classical ant personality as known in the literature. *NA* selects a move proportionally to its trail value (among the ones with promising greedy forces).
- *FA* corresponds to the personality that strictly follows what others have done previously. *FA* always selects the move with the largest trail value. This behavior aims at intensifying the search.
- *MA* corresponds to *NA* with probability $(1 - p_{MA})$, but with a probability p_{MA} (parameter tuned to 0.4), it changes its mood and starts behaving apparently against the goal. *MA* selects a move proportionally to the trail values with probability $(1 - p_{MA})$, and inverse-proportionally to the trail values with probability p_{MA}. This behavior aims at strongly diversifying the search.
- *IA* corresponds to the personality that tends to behave in an unusual way, but with the intention to reach the goal. *IA* corresponds to *FA* with probability $(1 - p_{IA})$ (intensification role), but with a probability p_{IA} (parameter tuned to 0.2), it changes its mood and make a random decision (diversification role). *IA* selects the move with the largest trail values with probability $(1-p_{IA})$, and randomly with probability p_{IA}. This personality favors the exploration of new solutions.

The tests have been run in a Windows 7 PC with an Intel Core2 Quad Q9400 of 2.66GHz and 4MB of RAM in 32-bit. For each proposed algorithm, the stopping condition is $5 \cdot n$ seconds, where n is the number of clients of the considered instance. The results are averaged over 9 runs. The considered instances are all the benchmark instances from [21] and [17] which do not have the autonomy constraint. The results are provided in Table I. Let $f^\star$ be the best known value obtained from [22]. Column 2 indicates the average percentage gap between *GR* and $f^\star$. The next columns provide the same information, but for *ANT*, *AL*, *ALM* and *ALMP*, respectively. It can be safely concluded that every single ingredient (i.e., a trail system, local search procedures, a central memory, and various personalities) successively added to *GR* to derive *ALMP* is useful.

TABLE I
RESULTS ON THE CONSIDERED BENCHMARK INSTANCES

Method	*GR*	*ANT*	*AL*	*ALM*	*ALMP*
Percentage gap	12.50%	11.00%	7.50%	3.70%	3.30%

IV. CONCLUSION

In this paper, a new ant metaheuristics called SMART is proposed, which can be adapted to any combinatorial optimization problem. The paradigm of the ant methodology is thus extended, as in contrast with most of the literature on ant algorithms, a specific personality is assigned to each ant. Because each personality has its own role and characteristics, SMART can find a good balance between exploitation (i.e., the ability to guide the search in the solution space and to take advantage of the problem structure) and exploration (i.e., the ability to visit various zones of the solution space). A solution method having a good behavior according to exploitation and exploration is likely to be a robust method, as robustness can be defined as the sensitivity to variations in problem characteristics and data quality. Future works include the development of SMART for other problems.

REFERENCES

[1] N. Zufferey, "Metaheuristics: some Principles for an Efficient Design," *Computer Technology and Applications*, vol. 3 (6), pp. 446 – 462, 2012.
[2] M. Gendreau and J.-Y. Potvin, *Handbook of Metaheuristics*, ser. International Series in Operations Research & Management Science. Springer, 2010, vol. 146.
[3] M. Dorigo, "Optimization, learning and natural algorithms (in Italian)," Ph.D. dissertation, Politecnico di Milano, Dipartimento di Elettronica, Italy, 1992.
[4] C. Blum, "Ant colony optimization: Introduction and recent trends," *Physics of Life Reviews*, vol. 2(4), pp. 353–373, 2005.
[5] M. Dorigo, M. Birattari, and T. Stuetzle, "Ant colony optimization – artificial ants as a computational intelligence technique," *IEEE Computational Intelligence Magazine*, vol. 1 (4), pp. 28–39, 2006.
[6] M. Dorigo and T. Stuetzle, *Handbook of Metaheuristics*. In F. Glover and G. Kochenberger (Eds), 2003, vol. 57, ch. The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances, pp. 251–285.
[7] N. Zufferey, "Optimization by ant algorithms: Possible roles for an individual ant," *Optimization Letters*, vol. 6 (5), pp. 963 – 973, 2012.
[8] "Design and classification of ant metaheuristics," in *Proceedings of the 22nd Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP 2014)*, Turin, Italy, February 12 – 14 2014.
[9] L. Luyet, S. Varone, and N. Zufferey, "An Ant Algorithm for the Steiner Tree Problem in Graphs," *Lecture Notes in Computer Science*, vol. 4448, pp. 42 – 51, 2007.
[10] N. Zufferey, J. Farres, and R. Glardon, "Ant metaheuristics with adapted personalities for the vehicle routing problem," in *Proceedings of the 6th International Conference on Computational Logistics (ICCL 2015)*, Delft, Nederland, September 23 – 25 2015.
[11] M. Plumettaz, D. Schindl, and N. Zufferey, "Ant local search and its efficient adaptation to graph colouring," *Journal of the Operational Research Society*, vol. 61, pp. 819 – 826, 2010.

[12] A. Hertz, D. Schindl, and N. Zufferey, "Lower bounding and tabu search procedures for the frequency assignment problem with polarization constraints," *4OR*, vol. 3 (2), pp. 139 – 161, 2005.
[13] J.-F. Cordeau, M. Gendreau, A. Hertz, G. Laporte, and J.-S. Sormany, *Logistics Systems: Design and Optimization*. Springer, 2005, ch. New Heuristics for the Vehicle Routing Problem, pp. 270–297.
[14] J.-F. Cordeau, M. Gendreau, G. Laporte, J.-Y. Potvin, and F. Semet, "A Guide to Vehicle Routing Heuristics," *Journal of the Operational Research Society*, vol. 53 (5), pp. 512–522, 2002.
[15] J.-F. Cordeau and G. Laporte, *Metaheuristic Optimization via Memory and Evolution: Tabu Search and Scatter Search*. Boston: Kluwer, 2004, ch. Tabu search heuristics for the vehicle routing problem, pp. 145–163.
[16] M. Gendreau, G. Laporte, and J.-Y. Potvin, *The Vehicle Routing Problem*. Philadelphia: SIAM Monographs on Discrete Mathematics and Applications, 2002, ch. Metaheuristics for the VRP, pp. 129–154.
[17] B. L. Golden, E. A. Wasil, J. P. Kelly, and I.-M. Chao, *Fleet Management and Logistics*. Boston: Kluwer, 1998, ch. Metaheuristics in vehicle routing, pp. 33–56.
[18] G. Laporte and F. Semet, *The Vehicle Routing Problem*. Philadelphia: SIAM Monographs on Discrete Mathematics and Applications, 2002, ch. Classical heuristics for the capacitated VRP, pp. 109–128.
[19] S. Lin, "Computer solutions of the traveling salesman problem," *Bell System Technical Journal*, vol. 44, pp. 2245–2269, 1965.
[20] I. Or, "Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking," Ph.D. dissertation, Nortwester University, USA, 1976.
[21] N. Christofides, A. Mingozzi, and P. Toth, *Combinatorial Optimization*, 1979, ch. The vehicle routing problem, pp. 315 – 338.
[22] D. Mester and O. Braysy, "Active-guided evolution strategies for large-scale capacitated vehicle routing problems," *Computers & Operations Research*, vol. 34 (10), pp. 2964 – 2975, 2007.

Open Science Index, Mathematical and Computational Sciences Vol:10, No:1, 2016 publications.waset.org/10003380.pdf

Nicolas Zufferey Nicolas Zufferey is a full professor of operations management at the University of Geneva in Switzerland. His research activities are focused on designing metaheuristics for difficult and large combinatorial optimization problems, with applications mainly in transportation, scheduling, production, inventory management, network design, and telecommunications. He received his BSc and MSc degrees in Mathematics at EPFL (the Swiss Federal Institute of Technology in Lausanne), as well as his PhD degree in operations research (2002). He was then successively a post-doctoral trainee at the University of Calgary (2003 - 2004) and an assistant professor at Laval University (2004 - 2007). He is the (co)author of more than 70 publications (papers in scientific journals, proceedings of conferences, and book chapters) and has reviewed papers for more than 30 international journals. He has had research activities with about 15 Universities in Europe and North America, as well as with about 10 private companies.