Ant Colony Optimization Approach to Tokens' Movement within Generalized Nets

Vassia Atanassova[1] **and Krassimir Atanassov**[2]

[1] Institute of Information Technologies - Bulgarian Academy of Sciences
"Acad. G. Bonchev" Str., Block 2, Sofia 1113, Bulgaria
e-mail: *vassia.atanassova@gmail.com*

[2] Centre of Biomedical Engineering - Bulgarian Academy of Sciences
"Acad. G. Bonchev" Str., Block 105, Sofia 1113, Bulgaria
e-mail: *krat@bas.bg*

Abstract: This work proposes a novel approach to combining the concepts of Ant Colony Optimization and Generalized Nets. It discusses the possibility for optimization of the tokens' movement throughout the net following the idea behind the Ant Colony algorithm.
Keywords: Ant colony optimization, Generalized net, Modelling

1 Introduction

Generalized Nets (GNs, see [1, 2]) is a concept extending the concept of Petri nets and the rest of its modifications. One of the aspects of generalization is the fact that the GN transitions possess an index matrix of predicates, determining the conditions for tokens' transfer from any input place of the transition to any output place. On the other hand, the tokens enter the GN with their initial characteristics and during their transfer from the input to the output places of the transition, they are assigned new characteristics by means of special characteristic functions.

GNs have been applied to modelling of processes in the field of artificial intelligence (expert systems, neural networks, pattern recognition, machine learning, etc.), and in particular to metaheuristic methods for solving of optimizational problems like the transportational problem, the travelling salesman problem, the knapsack problem. An important venue of application of GN is the area of Ant Colony Optimization (ACO, see [4, 5, 7]). So far, GN have been used as a method for description of the ACO procedures.

The present article for the first time adopts the opposite approach: it discusses the possibility for optimization of the GN tokens' movement, using ACO algorithms.

2 Short remarks on GN theory

Broadly speaking, the GN is a bipartite directed graph consisting of a set of vertices called transitions and another set of vertices called places. Both of them reflect the static nature, or the infrastructure, of the modeled process, while its dynamic nature is represented by a set of tokens initialized with certain starting characteristic which move from the input to the output places of the model having their characteristics

changed. In other words, the tokens are instances of the modeled process, or individuals, who keep track of their history. The tokens, their characteristic functions and the conditions for transfer (as coded in index matrices of the transitions' predicates) reflect the logic of the modeled process.

The formal definition of the GN requires firstly the definition of the net's building block, namely the transition.

$$TR = \langle P_{IN}, P_{OUT}, time, dur, IM_P, IM_C, bool \rangle$$

where

- P_{IN} is the finite nonempty set of the input places (obligatory).
- P_{OUT} is the finite nonempty set of the output places (obligatory).
- *time* is the current time of the transition's activation (optional).
- *dur* is the current duration of the transition's active state (optional).
- IM_P is the index matrix of predicates, determining the conditions for tokens' transfer through the transition (obligatory).
- IM_C is the matrix, determining the number of tokens that may transfer from the i-th input to the j-th output of the transition (optional).
- *bool* is the Boolean type of the transition (optional).

On this basis, the formal definition of the GN is given, comprising of four sets of components demonstrating respectively the static, dynamic, temporal nature of the net and its memory:

$$\begin{aligned} GN = \quad & \langle \langle TRS, \pi_{TR}, \pi_P, c, f, \theta_{ACT}, \theta_{DUR} \rangle, \\ & \langle TKN, \pi_{TKN}, \theta_{TKN} \rangle, \\ & \langle T, t^0, t^* \rangle, \\ & \langle X_{INIT}, X_{NEW}, n \rangle \rangle \end{aligned}$$

Static components of the net

- TRS - set of transitions (obligatory);
- π_{TR} - function, giving the priorities of the transitions (optional);
- π_P - function, giving the priorities of the places (optional);
- c - function, giving the capacities of the places (optional);
- f - function that evaluates the degree of the predicates in IMP (it may be restricted to the $\{false, true\}$-set, or in the $[0;1]$ interval or in the multiset $[0;1] \times [0;1]$ (optional);
- θ_{ACT} - function, giving the next time moment when a given transition would get activated; the value is calculated only when the transition has stopped being active (optional);
- θ_{DUR} - function, giving the duration of the active state (optional).

Dynamic components of the net

- TKN - set of tokens (obligatory);
- π_{TKN} - function, giving the tokens' priorities (optional);
- θ_{TKN} - function, giving the time moment when a given token will enter the net (optional).

Temporal components of the model, defined according to a global time scale

- T - the moment of time when the net would start functioning (optional);
- t^o - the elementary incremental step of the time scale (optional);
- t^* - the total duration of net's functioning (optional).

Characteristic components (memory) of the net

- X_{INIT} - the set of initial characteristics that tokens acquire on entering the net (obligatory);
- X_{NEW} - the characteristic function, which assigns new characteristics of the tokens on their transferring via given transition (obligatory);
- n - function, giving the maximal number of characteristics for storing in a token's memory (optional):
  - $n = 0$ - the token stores no characteristics in its memory;
  - $n = 1$ - the token stores only its current characteristic;
  - $n = k$ - the token stores only the last k acquires characteristics;
  - $n = \infty$ - all token's characteristics are stored in the memory.

Different operations, relations and operators are defined over the transitions of the GNs and over the same nets. A variety of different types of GN-extensions are defined and each of them is proved [1, 2] to be a conservative extension of the ordinary GNs.

Now, we will give the general algorithm for tokens transfer in the frames of a transition at time moment $t_1 = TIME$ (the current GN time-moment), as described in [2]. In the following section we will present our idea for modifying some of its steps, which is inspired by the ACO.

(A01) Sort the input and output places of the transitions by their priorities. The tokens from a given input place are divided into two groups. The first one contains those tokens that can be transferred to the transition output, the second contains the rest (the motivation for this will be clear from the next steps of the algorithm). Let the two parts be denoted by "$P_1(l)$" and "$P_2(l)$", respectively, where l is the corresponding place.

(A02) Sort the tokens from group P_1 of the input places (following the order from A01) by their priorities. Let the index matrix R correspond to the index matrix IM_P. Thus, the (u, v)-th element of R is

$$R_{u,v} = \begin{cases} 1, & \text{if the } (u,v)\text{-th predicate } r_{u,v} \text{ is true} \\ 0, & \text{if the } (u,v)\text{-th predicate } r_{u,v} \text{ is false or if the value is} \\ & \text{determined by A03.} \end{cases}$$

(A03) Assign a value 0 to all elements of R for which either

(a) the input place which corresponds to the respective predicate is empty (the part P_1 is empty); or

(b) the output place which corresponds to the respective predicate is full; or

(c) the current capacity of the arc between the corresponding input and output places is 0.

(A04) Calculate the values of the other elements of IM_P and assign the obtained values to the elements of R.

(A05) Calculate the values of the characteristic functions related to the corresponding output places in which tokens will enter. Assign these characteristics to the entering tokens.

(A06) Perform the following for each input place by the order of input place priorities:

a) select the tokens with the highest priority in this input place;

b) transfer the selected tokens to all output places, for which the corresponding predicate enables this (the tokens go to group P_2 of the output places).

(A07) Transfer the tokens with the highest priority, for which all calculated values of the predicates are equal to "$false$" to the group P_2 of the corresponding places. In this group, also transfer all tokens that cannot be transferred to the corresponding output places because these places have already been filled with tokens from other places with higher priorities.

(A08) Add t^0 to the current time, i.e., $TIME := TIME + t^0$.

(A09) Check whether the value of the current time is less than $t_1 + t_2$ (the time-components of the considered transition).

(A10) If the answer to the question in A09 is "yes", go to A02 (to update the tokens' order in the places).

(A11) If the answer to the question in A09 is "no", terminate the current functioning of the transition.

3 Main results

Up to now, GNs have been used for modelling, simulation, in certain cases management, optimization or machine learning of real processes. For example, there has been developed a GN model that makes decisions of the structure of a neural network that solves particular problems with predefined accuracy of the solution and duration of functioning [3]. However, as of today, no GN has been constructed in a way to optimize models that take place inside of it. An idea of such a GN is the Self-Modifying GN, but up to now no such net has been constructed and published. Now, using ideas from the ACO algorithm we will initiate the first step towards researching the possibility for construction of a particular GN that is capable of taking decisions for changes in some of its own parameters.

In other words, the basic idea of this work is to combine the notions of GNs and ACO in the opposite way of those utilized so far. As of today, the concept of GNs was used to describe different variants of the ACO algorithm [6]. Here we

follow the reverse approach, applying the principle of ants' movement to the tokens' movement throughout the net. To do so, we have to pay attention to the following considerations and interpretations of the elements of the ACO algorithm in terms of GNs.

- The ACO algorithm can be reduced to finding optimal paths through graphs. Hence, here we will utilize the fact that the GN has a graphic structure that may be interpreted as a graph.
- The artificial ants are interpreted as the tokens in the GN.
- The pheromone trails are used by the artificial ants in the ACO algorithm as communication medium: once the agents have found a solution they depose these traces, i.e. communicate their discovery with the agents-to-come. In terms of GNs, this information shall be given the form of a list of the net's places that have been visited.

The changes in the pheromone's intensity (increase due multiple ants using the track, or decrease due to evaporation) are modeled by changes in the characteristics of some appropriately chosen tokens. These changes will be an object of discussion in a next authors' research.

Let us have a GN that models a concrete process, of which we know:

- the separate stages as represented by the net's transitions,
- the carriers of dynamic behaviour, as represented by individual tokens, and
- the moments of the tokens' entering the GN.

If we possess all of this information about the process, we will be capable of constructing an adequate GN model of this process, while if a part of this information is missing, our GN model will not be complete but partial. Below, we will discuss how we may approach to replacing some of the missing data. We will show how we can generate appropriate values of some of the model's parameters, which will be derived by the modeled process itself, making the assumption that it functions in an optimal way.

For instance, one case of incomplete information of the modeled process is to assume that in the real process we miss the data about the durations of the transitions from one state to another, as well as the durations of the separate states. Another possible situation (when we happen to have more information) is if we know the durations of the separate sub-processes, but we do not know what characteristics we may assign to the net's tokens that describe the dynamics of the process. It is an even more interesting case when we possess part of this information, as well. For each of these three examples we may design a GN that reflects the relations between the separate parts of the modeled process. It is a priori clear that at least this knowledge ought to be in being.

The present article will deal only with the first of the so described scenarios.

Let us take a GN with 1 or more input places and 1 or more output places. Let us make the following assumptions:

- On each step every transition of the net is fired (gets activated) and its active state continues 1 time unit.

- All tokens are allowed to split.
- The tokens' memory is unlimited, i.e. all tokens may store an indefinite number of characteristics.
- Each token have the initial characteristic of the moment of time when it enters the GN.

In order to describe the first example we shall assume that the capacities of the places are equal to infinity. In this case, every token transfer from the input place to each of the output places of the respective transition. It is sufficient in this case to have exactly one token entering each input place, because otherwise the next-to-come will repeat the exact ways of splitting and the routes of the preceding tokens.

In each place, the tokens obtain as a new characteristic the place's identifier (the current place's identifier is added to the list of identifiers of all previous, already attended, places in the net).

The so-described GN precisely copycats the idea of an ACO procedure with a finite number of ants, each of which is here represented by a GN token. The token, which starts from the i-th input place and is the first to reach the j-th output place of the net, will possess as characteristics the shortest route between both of these places.

When describing the second example, we will have to assume that the capacities of the places are finite numbers, in particular 1. In this case, we are able to take into account the eventual instances of route clogging, and for this reason this case is more interesting than the first one. Now, we can have a new token entering each input place only when the previous token had already left the place.

In each place the tokens obtain as a current characteristic the place's identifier as well as the moment of entering. In the end, the final token characteristic will also include the calculated total time of token's movement throughout the net.

It is appropriate to have the process of tracking the tokens' movement described in the GN itself, i.e. to have the net self-controlling. For this purpose, we add to the given GN a new transition T (see Fig. 1) with only one place P that serves both for input and output place. Only one token α loops in the transition. The transitions T, the place P and the token α are assigned the minimal possible priorities among their likes. In this way, on each step of the net's functioning we provide for the token α to make its move after all other tokens in the net, and allow it to obtain as a current characteristic the current distribution of tokens per places.

Fig. 1.

After the end of the net's functioning, we will determine the shortest route with respect to either time, or length by:

- tracing the routes of the individual tokens,
- determining the lengths of the paths, and
- rendering account of the time spent by the tokens in the net (Case 2).

Behind the so constructed GN construction, another important aspect can be perceived, namely the criteria of intended optimization. Our experience with the classical ACO has led us to the understanding that it is the time of taking the route and the length of the path in the GN, as generated by the GN structure, that are most important criteria for optimization. Now it is clear that this statement is valid for the first of the discussed cases, but it is invalid in the second case, when the duration and the length of the path may be fully independent criteria and the optimization may be conducted per both of them, in parallel.

On the basis of the accumulated information, we may built a simulation model in which the tokens transfer from input to output places with probabilities corresponding to the profits laid on the respective routes.

Now we will discuss the possible applications of the so constructed GN.

As we already mentioned, there is a point in using it only in cases when we possess incomplete information of the modeled process. In the first case discussed above, we may complicate the research by determining the lengths of the paths from the i-th place, which is not an input place of the GN, up to the the j-th output place, and then we can apply the following procedure:

1. For each (say, t-th) transition, we determine the number of the output places, via which a token that has started from the i-th place which is an input place for this transition, will reach the j-th place which is output place for the whole net. Let this transition possess s_t output places and let their route lengths be, respectively, $p_1^t, p_2^t, ..., p_{s_t}^t$. Then we determine the number $a_t = \sum_{i=k}^{s_t} \frac{1}{p_k^t}$.

2. We determine the numbers $\alpha_k^t = \frac{p_k^t}{a_t}$ $(1 \leq k \leq s_t)$.

3. The predicate of the index matrix of transition t that corresponds to the fixed i-th place and the k-th output place be $P_{i,k} =$ "$r \in \left[\sum_{u=1}^{k-1} \frac{1}{p_u^t},\ \sum_{u=1}^{k} \frac{1}{p_u^t}\right]$", where r is a random number in the $[0, 1]$ interval.

Following this procedure, the token from the i-th place will advance to an output place with a probability that corresponds to the length of the route to the j-th output place of the net. Moreover, the shorter the path, the larger the probability for the token to move towards this very place. This ensures the optimal movement of the nets around the net.

In contrast with the first case, in the second case we assume that tokens enter the net in every time moment. Now, for t-th transition and for its k-th output place $(1 \leq k \leq s_t)$ we will obtain that the tokens (whose number is q_k), which have passed through it, will arrive in the net's j-th output place for time periods of $Q_{k,1}^t, Q_{k,2}^t, ..., Q_{k,q_k}^t$. These time periods can be different, because in the second case the tokens can spend time waiting in some places. All of these tokens will travel a

path of lenght p_k^t (as in the first case). Now, we can determine the average duration for tokens' tranfser: $D_k^t = \frac{1}{q_k}\sum_{l=1}^{q_k} Q_{k,l}^t$. By analogy with the first case, we can determine the numbers $\beta_k^t = \sum_{k=1}^{s_t} \frac{1}{D_k^t}$, that we can use instead of α_k^t, constructed above.

4 Conclusion

This paper contains the general idea and the first step towards optimization of the GN functioning by the ant colony optimization algorithm. A next authors' research will be especially devoted to the formal description and exploration of the rest two cases, as well as other situations that may occur in the GNs. It must be noted that using the above discussed ideas a self-organizing GN can be constructed, which makes references to one of the open problems in artificial intelligence, namely the problem with self-reference and self-modifying algorithms (see [8, 9]).

5 Acknowledgments

This work has been supported by the Bulgarian National Science Fund under grants No. DID-02-29 "Modelling Processes with Fixed Development Rules" and DTK-02-44 "Effective Monte Carlo Methods for Large-Scale Scientific Problems".

References

[1] Atanassov K., Generalized Nets, World Scientific, Singapore, New Jersey, London, 1991.

[2] Atanassov, K., On Generalized Nets Theory. Prof. M. Drinov Publishing House, Sofia, 2007.

[3] Atanassov, K., Sotirov S., Optimization of a Neural Network of Self-organizing Maps Type with Time-Limits by a Generalized Net. Advanced Studies on Contemporary Mathematics, Vol. 13, 2006, No. 2, 213-220.

[4] Dorigo, M., Gambardella, L.M., Ant Colony system: A Cooperative Learning Approach to the traveling salesman problem, IEEE Transactions on Evolutionary Computation, Vol. 1, 1997, 53-66.

[5] Dorigo, M., Stutzle, T., Ant Colony Optimization, MIT Press, 2004.

[6] Fidanova, S., Atanassov, K., Generalized Net Models of the Process of Ant Colony Optimization. Issues in Intuitionistic Fuzzy Sets and Generalized Nets, Vol. 7, 2008, 108-114.

[7] Fidanova, S., Marinov, P., Intuitionistic fuzzy estimation of the ant methodology, Int. J. of Cybernetics and Information Technology, Vol. 9, No. 2, 2009, 79-88.

[8] Marshall, J., Hofstadter, D., Beyond Copycat: Incorporating Self-Watching into a Computer Model of High-Level Perception and Analogy-Making, In M. Gasser (ed.), Online Proceedings of the 1996 Midwest Artificial Intelligence and Cognitive Science Conference, Indiana University, Bloomington.

[9] Turney, P., http://apperceptual.wordpress.com/2007/12/18/open-problems/.