Y. Grushchak, Master student
O. Hrabar, PhD in Engr., As. Prof., research advisor
L. Mohelnytska, PhD in Phil., As. Prof., language advisor
Zhytomyr State Technological University

USING SIMULATED ANNEALING IN ANT COLONY OPTIMISATION ALGORITHMS

An algorithm of simulated annealing is a general algorithmic method of solving a global optimization task, especially discrete and combinatorial optimization, where procedure of searching of global solution imitates a physical process of annealing.

Basic principles of simulated annealing algorithm may be explained with help of the next physical analogy. Fig.1. shows a ball in the box. Inner surface of the box fits the shape of target function. When heavily shaking the box horizontally a ball can move from any point to any other. Gradually decreasing shaking power will reach the point enough for moving the ball from A point to B point but not enough for moving the ball from B to A. Keeping on decreasing shaking power to zero will cause keeping the ball in B point, which is the point of global minimum. In simulated annealing algorithms an analogue of shaking power is the probability of moving to a state with higher value of target function. During working of the algorithm the probability of moving to another state decreases according to the chosen rule.

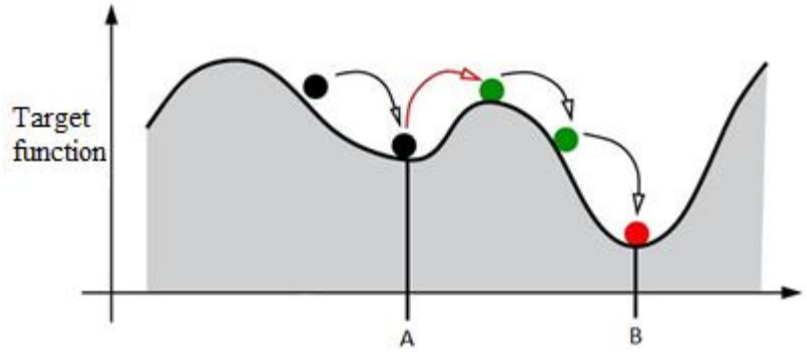

Fig.1. Principle of simulated annealing method

Algorithm of simulated annealing in the process of searching the best solution allows probable moving to the state with worse (in case of minimization — with higher) value of target function. This property allows coming out from the local optimums.

The algorithm doesn't guarantee function minimum to be found, however as a rule, the initial value improves. After combining simulated annealing method with ACO (ant colony optimization) algorithm we can use this property for improvement of certain parts of path found with help of ACO algorithm.

ACO algorithm is on of the effective polynomial algorithms for solving a travelling salesman's problem and for solving tasks of searching routes on graphs. The idea of this approach is based on the behavior of ants seeking a path between their colony and a source of food.

The algorithm is based on behavior of ant colony, i.e. marking of appropriate path with high quantity of pheromone. The work begins from placing ants in nodes of graph (cities). Then ants begin their movement. Its direction is determined by the probability method and the formula:

$$P_i = \frac{l_i^q \cdot f_i^p}{\sum_{k=0}^{N} l_k^q \cdot f_k^p}$$

where:

P_i — probability of using an “i” way,

— desirability of “i” way(typically $1/d_i$, where “d” is distance),

— amount of pheromone on “i” way,

q — a parameter to control the influence of ,

p — a parameter to control the influence of and

q+p=1

The result isn’t precise and may be even one of the worst. But thanks to probability of the solution, reiteration of the algorithm may return quite precise result.

In the natural world, ants of some species (initially) wander randomly, and upon finding food return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep travelling at random, but instead to follow the trail, returning and reinforcing it if they eventually find food.

Over time, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. A short path, by comparison, gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than longer ones. Pheromone evaporation also has the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained.

The overall result is that when one ant finds a good (i.e., short) path from the colony to a food source, other ants are more likely to follow that path, and positive feedback eventually leads to all the ants following a single path. The idea of the ant colony algorithm is to mimic this behavior with "simulated ants" walking around the graph representing the problem to solve.

What can be done for optimization? We can use parallelization. This approach lets increase significantly the speed of getting a final result. There are a few ways of parallelization of ant colony algorithm:

- Global single populated. It means there is one single population on the CPU main core and the calculation of target function is parallelized on several cores.

- Single populated. Path searching for each ant takes place on a separate core in the same way as counting of it’s length and refreshing of pheromones on it.

- Multi populated. There are multiple independent populations on each core and ants of each population search their ways. After each ant of population has founded the way, refreshing of pheromones begins on paths of this population.

Ants of the same core are generally called a colony. After each colony has finished their work it is possible to calculate the best solution of each colony and share information about the shortest path and amount of pheromone on graph edges between other colonies. In addition we can try to share this information not at each step but after a certain period of time.

Innovative is the fact that ants within every separate colony may have different amount of pheromones. For example, in the first colony the amount of pheromone that each ant has may be 1, in the second colony each ant may have 2 pheromones and so on. The purpose of the thesis is analyses of influence of this experiment on the algorithm. Also after some interval of iterations the genetic algorithms may be applied to our previous decision and we may begin to shuffle ants among colonies that have diverse amount of pheromones.

REFERENCES

1. 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

2. А.А. Кажаров , В.М.Курейчик Об одном «муравьином» алгоритме – [Електронний ресурс] – Режим доступу: www.raai.org/conference/cai-08/files/cai-08_paper_144.doc

3. Муравьиный алгоритм [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Муравьиный_алгоритм

4. Муравьиные алгоритмы / Хабрахабр [Електронний ресурс] – Режим доступу: https://habrahabr.ru/post/105302/

5. Введение в оптимизацию. Имитация отжига / Хабрахабр [Електронний ресурс] – Режим доступу: https://habrahabr.ru/post/209610/