Ant Colony Optimization

1st Brian Kelein Ngwoh Visas
Electronics Engineering
Hamm-Lippstadt University of Applied Sciences
Lippstadt, Germany
brian-kelein-ngwoh.visas@stud.hshl.de

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

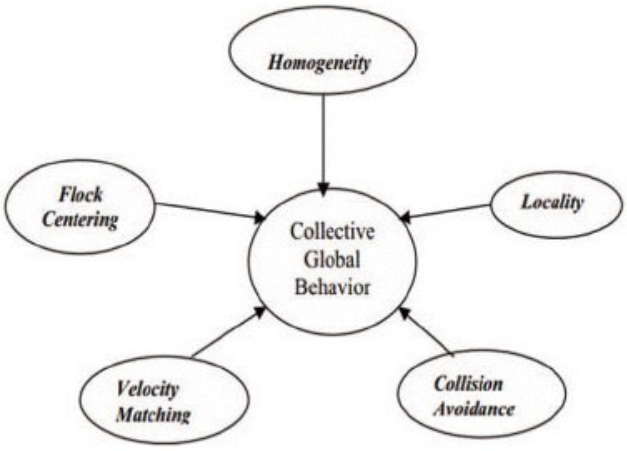
***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

A. Definition of the Algorithm

B. Swarm Intelligence and Biological background

The collective behavior and social movements of a group of creatures (like bees, birds, fishes, ants, wasps and etc.) is very significance, but personal behavior of a single creature often is simple. Because of the impressive applicability of the natural swarm systems, researchers have done numerous studies on such behaviours of social creatures. These systems proposed based on the concepts of Artificial Intelligence (AI), in the late-80s, by the computer scientists. In 1989, the first time, the term of "Swarm Intelligence" was applied by G. Beni and J. Wang in the global optimization framework as a set of algorithms for controlling robotic swarm. SI techniques have wide application domains in science and engineering. Therefore, SI can be considered as a new branch of AI. So that, it is used to modeling the collective behavior of social swarms in nature like bird flocks, honey bees, ant colonies, and etc. Fig. 1 shows an overview of the main properties of the collective behavior. [1]



[1]

Fig. 1. The main Properties of modelled collective behaviour

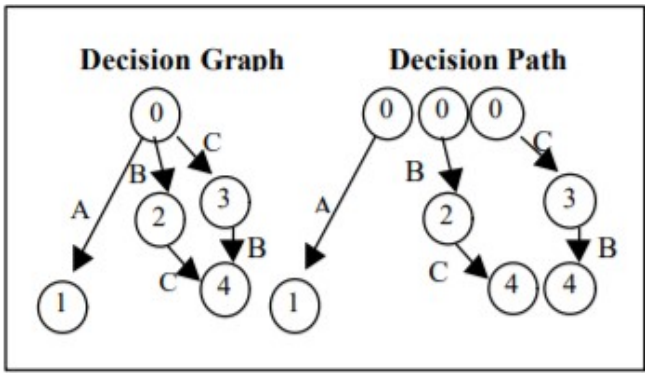
II. Dynamic Ant Algorithm

A. Decision Path

This is a record of the decisions, which were made for a design. Each node of a path represents a design problem state. The first node of the decision path is the starting point of the design process. From the first node a designer makes a decision in selecting a choice from all of the possible ways. After that the designer will reach to another state (node) in the design problem. This process is repeated until the designer gets a complete design or gets to the dead end.

B. Decision graph

This is a structure that represents a decision space for a design problem. Each node represents a design problem state. Each arc represents a choice that must have been made to get to the next node or the next state. It is the same as decision tree but in this case the different sequence of decision can get to the same state or the same node. Or each node can have more than one in degree. There is one root node, which represents the initial state of the design problem. A path from the root node to a leaf node is called a decision path.



[2]

Fig. 2. Decision Tree for Graph and Path

C. Construction of a decision path

This is a process that a sequence of selection is made. This process begins from the first node, which is an initial design state.Then the possible choices are listed and each possible

choice is assigned with a heuristic weight and a pheromone weight as in the Ant algorithm. A choice is selected according to the state transition rule, which will be described next, and then the algorithm gets to the new state of the design problem, which is explained in [2]. This process continues until the design is completed. While constructing the decision path the decision graph is also updated.

D. State Transition Rule

This is the rule that will be applied in order to advance from one design step to the next step by selecting a choice from the possible weighted choice list. If all of the choices were selected (by previous ants), then a choice will be randomly selected by the biases from pheromone level. This is diverted from another ACO, in ACO the selection will be based on a linear combination of heuristic weights and pheromone levels but in Dynamic Ant the heuristic weights are used until all choices are selected then the pheromone levels are considered as seen in [2] . The advantage of our method is that the determinations of heuristic weights are independent to pheromone levels.

E. Path Exploration

is a process to initialize new paths from the interesting path. The process begins by selecting a random exploration point in the path. Then a path from the root to the exploration point is copied for a new ant as an initial path.Now we introduce the steps of the Dynamic Ant Algorithm. We call this Dynamic Ant because some of techniques were borrowed from Dynamic Niche Sharing as explained by [2] .

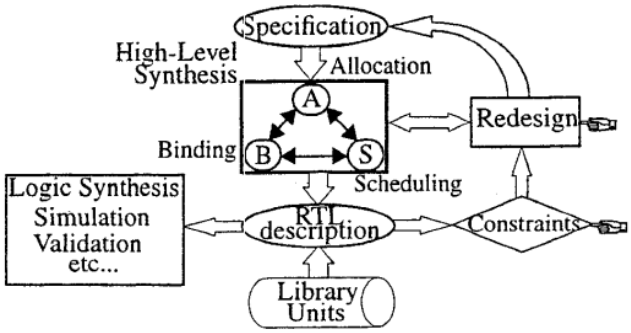
III. High Level Synthesis in Relation to Ant Colony Optimization(ACO

A. High Level Synthesis Flow

HLS Flow simply defines the process flow from design, right up to the level of synthesis. Here the design is implemented an abstract level in the form of an algorithm. Furthermore a tool is then used to generate a circuit at Register Transfer Level(RTL) as described by [3] But if the circuit then happens to be in a Hardware Description Language then it can be simulated and verified.

B. High Level Synthesis Formulation

Let v and w be arithmetic or logical operations and (v, w) be a precedence relation. This means that the output of v must be completed before w can start. Then the problem is choosing hardware h_v, h_w to execute$v, w$and scheduling each into a clock step s so that: $(t_w > t_v + l_V + M)\forall(w, v) \in P$ while minimizing a cost function $C(a, p, d)$ Here t_x is the initiation time of operation $xandl_vh$ is the latency (propagation delay) of operation v, executed on hardware unit h. P is the set of data precedence and M is a timing margin that accounts for interconnect, mux and register delays. In the cost function, a refers to the total area, p to the power consumption and d to the total latency of the algorithm.



[3]

Fig. 3. HLS Flow

Design	Op.	Prec.	Paths	Space
EWF	34	47	57	10^{70}-10^{90}
ROTOR	28	37	28	10^{60}-10^{80}

[3]

Fig. 4. Solution space for Benchmarks

C. High Level Synthesis Complexity

IV. HACO-F: An Accelerating High Level Synthesis Based Floating Ant Colony Optimization on FPGA

A. HACO-F Design based on High Level Synthesis

The HACO-F is designed on HLS by integrating the Ant System (AS) algorithm, as the computation scale is larger than those of other ACOs. The HACO-F design is composed of two aspects. One is data optimization design, which can be employed to redefine the precision of variables for further reduction of resource usage. The other is loop optimization design, which can be used to make the AS algorithm paralleled and integrated in FPGA. The notations of HACO-F are explained in in the table below [4]

B. HACO-F Data Optimization Design

Data optimization is to redefine the precision of the variables used in the algorithm to further reduce the resource usage. In the process of data optimization, it mainly focuses on the boundaries of input variables, which may differ in different applications. According to the boundaries of input variables, it is easy to redefine the valid bits of these input variables. Especially for float type variables, we redefine the precision by the following transition principle. For the precision insensitive variables, 6bits in decimal (20bits in binary) is defined, and the total bits should be defined smaller than 32bits (the same with float type). Otherwise there would be little optimization or even worse. For some precision sensitive variables, the total bits may be up to 64bits (the same with double type). In

Notations	Specification
d	city location data
n	city quantity
m	ant quantity
NC	trip quantity
IN	pheromone initial value
seed	random function seed
ρ	volatilization coefficient
Q	constant coefficient for the calculating of $\Delta\tau$
η	heuristic information , the visibility of the road
tabu	visited cities list of ants
allowed	unvisited cites list of ants
L	tour length of all the ants
τ	road pheromone
$\Delta\tau$	update pheromone
p	transition probability
besttour	best tour map
tourlength	best tour length
α	parameter of τ
β	parameter of η
i	index variable of n
j	index variable of n
s	index variable of n
k	index variable of m

[4]

Fig. 5. Notations of HACO-F

particular, we may approximate the percentage values with 6bits in binary. Other variables, valid bits are defined based on the relations such as the operations relations and valid bits dependency relations.

C. HACO-F Loop Optimization

V. Ant Colony Optimization Algorithm For Fuzzy Controller And Design Of Its FPGA

A. Basic Concepts Of Fuzzy Controller and Ant Colony Optimization

B. The Ith Rule

Let i be an index for a fuzzy rule number, The ith rule denotes:

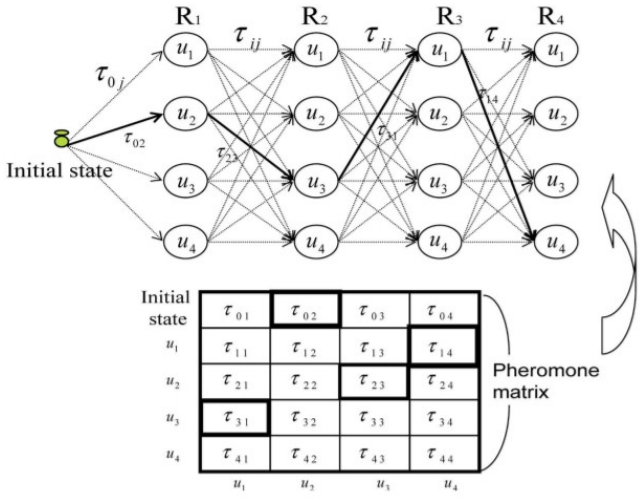
R_i : If $x_1(k)$ is A_{i1} And $\ldots$, And $x_n(k)$ is A_{in}

Then $u(k)$ is $a_i(k)$ (1)

C. Fuzzy Controller Design By Ant Colony Optimization

D. Hardware Implementation Of Ant Colony Optimization in ACO-FC

The ACO algorithm in the proposed ACO-FC is hardware implemented by FPGA. To test the design ACO chip performance, the designed FC and the controlled plant are simulated by a PC. Online control configuration is considered. Like a practical plant online control, performance evaluation data are available only when the control starts. That is, only one ant can be evaluated at a time. Thus, selection and evaluation functions of each ant in the proposed ACO chip are implemented sequentially. The input from PC to FPGA is the signal F,



[5]

Fig. 6. FC constructed by an Ant trip and the corresponding pheromone matrix

denoting performance measure and represented by eight bits. The FPGA output is the signal rule$[k]$ $[m]$, which sends information on the selected consequent action for fuzzy rule m by the kth ant. The architecture of FPGA-implemented ACO is shown in Fig. 2. It consists of five modules. The summation, random number generator, and consequent selection modules are used to select the FC and are activated for each new ant path building process. The best ant selection module is also activated for each new ant path building process, but it works after controller evaluation in PC. Pheromone matrix update is performed in the pheromone matrix module and is activated only once in each iteration. That is, it is activated only after all ants have built their paths. For illustration, assume that the candidate action number N is 16. The signal rule[k][m] is represented by four bits to represent one of the 16 candidate actions. Detailed descriptions of the five modules are as follows.

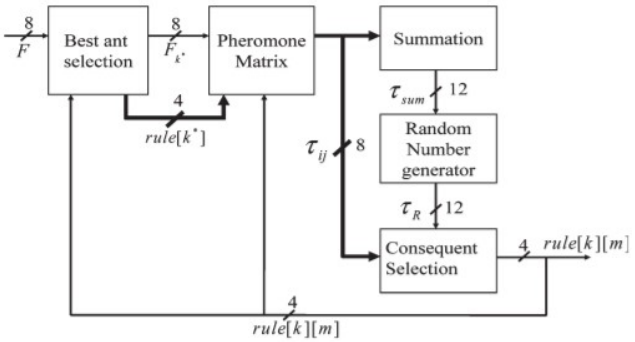
VI. Conclusion

References

[1] A. Mohammadi and S. H. Zahiri, "Analysis of swarm intelligence and evolutionary computation techniques in iir digital filters design," in *2016 1st Conference on Swarm Intelligence and Evolutionary Computation (CSIEC)*, 2016, pp. 64–69.

[2] R. Keinprasit and P. Chongstitvatana, "High-level synthesis by dynamic ant," *International journal of intelligent systems*, vol. 19, no. 1-2, pp. 25–38, 2004.

[3] E. Torbey and J. Knight, "High-level synthesis of digital circuits using genetic algorithms," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, 1998, pp. 224–229.



[5]

Fig. 7. Architecture of Hardware-Implemented ACO

[4] S. Zhang, Z. Huang, W. Wang, R. Tian, and J. He, "Haco-f: An accelerating hls-based floating-point ant colony optimization algorithm on fpga." *International Journal of Performability Engineering*, vol. 13, no. 6, 2017.

[5] C.-F. Juang, C.-M. Lu, C. Lo, and C.-Y. Wang, "Ant colony optimization algorithm for fuzzy controller design and its fpga implementation," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 3, pp. 1453–1462, 2008.