Ant Colony Optimization with Multi-Pheromones for Solving Constraint Satisfaction Problems

Takuya Masukane
Department of Computer Science
Faculty of Engineering
Takushoku University
Email: r38470@st.takushoku-u.ac.jp

Kazunori Mizuno
Department of Computer Science
Faculty of Engineering
Takushoku University
Email: mizuno@cs.takushoku-u.ac.jp

***Abstract*—To solve large-scale constraint satisfaction problems, CSPs, ant colony optimization, ACO, based meta-heuristics has been effective. However, the naive ACO based method is sometimes inefficient because the method has only single pheromone trails. In this paper, we propose an ant colony optimization based meta-heuristics with multi pheromone trails in which artificial ants construct a candidate assignment by referring several pheromone trail graphs to solve CSP instances. We also implement the proposed model to some ACO based methods, demonstrating how our method is effective for solving graph coloring problems that is one of typical examples of CSPs.**

***Index Terms*—constraint satisfaction; search; meta heuristics; ant colony optimization; graph coloring; phase transition;**

I. Introduction

A constraint satisfaction problem (CSP) involves finding values for problem variables which are subject to given constraints specifying the acceptable combinations of values. Such combinatorial search problems are ubiquitous in artificial intelligence and pattern analysis, including scheduling, planning, resource allocation, or machine vision.

To solve large-scale CSPs that are NP-complete, meta-heuristics for stochastic search approaches has been recently made remarkable progress [1], [9], [10], [13]–[15]. Ant colony optimization (ACO), which is one of typical meta-heuristics, has been effective for many combinatorial search and optimization problems such as traveling salesperson problems [4]–[6], graph coloring problems [2], or vehicle routing problems [1], where the behavior of real ants is modeled as a metaphor, that is inspired by the collective behavior of ants in the real world in finding paths from the colony to food [6], [15]. The idea is to keep track, through *pheromone* laying, of promising areas of the search space, i.e., good assignments. This pheromone information is used to guide the search as heuristics.

The Ant System (AS), has been initially proposed as the algorithm based on ACO [6]. Although many algorithms using AS have been proposed, most of them have required a lot of search time due to reconstructing a candidate solution, or an assignment of values to variables, entirely at each cycle, or iteration. In particular, when CSPs are solved by using AS, much more search time is required because it is necessary for solving CSPs to find not a semi-optimal solution but a complete solution with no constraint violations. In contrast, the cunning Ant System (cAS) proposed by [17] can generate a candidate solution by borrowing a part of the candidate solution generated at the previous cycle. In [17], the performance of cAS has been evaluated by applying to the traveling salesperson problem (TSP), which is an example of combinatorial optimization problems. In [7], [12], novel approaches have been proposed based on cAS, evaluating their efficiency by applying to binary CSPs.

In AS and cAS, however, each ant constructs an assignment by referring single pheromone trail graph that is periodically updated based on the best constructed assignment with the lowest constraint violations at each cycle. Because low-quality parts are even included in the best assignment, it is possible to get stuck in locally optimal assignments in the next search steps due to the updated pheromone information.

In this paper, we propose an ACO model with multi pheromone trail graphs for solving CSPs. In our model, in addition to the usual pheromone trail graphs, another graph, which stores pheromone values obtained from the worst assignment with the most constraint violations, is provided. Each ant constructs a candidate assignment while referring to both of pheromone graphs. It is expected that diversity of search processes can maintained by adopting comparatively high-quality partial assignments, or building blocks, included in the worst assignment. We have also implemented the algorithms, where the proposed model is introduced to AS and cAS respectively and demonstrated that the proposed methods can be more effective than naive AS and cAS for some hard CSP instances around phase transition regions.

The CSP is well-known as an NP-complete problem, but actual problem instances with such computational complexity are found only in a locally limited region of the problem space. Recent studies have revealed that really hard problem instances tend to happen in situations very similar to physical phase transitions [3], [8], [11]. Problem instances within phase transition regions are very hard to solve for not only systematic approaches but also stochastic approaches. Hence, it is important for the studies of meta-heuristics to place their interests on how well they cope with such hard problem instances within phase transition regions.

978-1-5090-5732-0/16/$31.00 ©2016 IEEE

II. Problem definition and ant colony optimization

A. Graph coloring problem

Let us briefly give some definition and terminology about CSPs. A CSP is defined as a triple (X, D, C) such that

- $X = \{x_1, \ldots, x_n\}$ is a finite set of variables.
- D is a set of values to be assigned to variables.
- $C = \{c_1, \ldots, c_m\}$ is a set of constraints, each of which is a relation between some variables which restricts the set of values that can be assigned simultaneously to these variables.

In this paper, we employ the graph coloring problem with 3 colors available, 3COL, for evaluating the performance of the proposed method, which is one of the typical constraint satisfaction problems [10], [11], [16]. Arranging to the above definition of CSPs, X and C correspond to the set of vertices and edges, respectively, when a graph $G = (X, C)$ is defined. D (= $\{red, green, blue\}$) corresponds to the set of colors. An edge $c_k = (x_i, x_j) \in C$ stands for the constraint that prohibits assigning the same color to vertices, x_i and x_j.

Letting $|X| = n$ and $|C| = m$, we define the constraint density, d, as $d = m/n$, that corresponds to half of the average degree of a graph, to classify 3COL instances. Many research reports have observed phase transition phenomena in combinatorial search and optimization problems including CSPs and 3COLs: matter commonly undergoes drastic changes in its qualitative properties when certain order parameters pass through particular values. In CSPs and 3COL, the search cost follows an easy-hard-easy patterns as a function of constraint tightness or density. According to many research reports on 3COLs, instances really hard to solve, which are generated randomly, have tended to be concentrated around the region, $d = 2.3 \sim 2.4$ (i.e., the average degree of the graph is around 4.6), where phase transition phenomena occurs [8], [10], [11]. Based on the above, we randomly generate 3COL instances with $d = 2.0 \sim 3.0$ for evaluating the algorithms.

B. Ant colony optimization

Ant colony optimization, ACO, is a kind of swarm intelligence based meta-heuristics inspired by the behavior of real ants in finding paths from the colony to food [6]. Instead of real ants, artificial ants are introduced in ACO. Artificial ants act as reactive agents that do not use any kinds of central symbolic world model but exploit a stigmergetic communication mechanism. In particular, artificial ants lay pheromone trails on edges of the graph and choose their path with respect to probabilities that depend on the amount of previously left on edges. Pheromone trails are also progressively decreased, simulating some kind of evaporation.

The Ant System, AS, is the ACO based search algorithm proposed by [4], [5].

Figure 1 gives the algorithmic scheme of AS. In the procedure, 'Ant-Solver' [15], in Figure 1, every artificial ant constructs a complete assignment, or candidate solution, of values to all variables in the procedure 'ConstructAssignment'.

```
begin
  Initialization;
  Ant-Solver(maxcycle, nbAnts, α, β, ρ);
end

procedure Ant-Solver(maxcycle, nbAnts, α, β, ρ)
begin
  repeat
    for k in 1..nbAnts do
      A_k ← ConstructAssignment(τ, α, β, (X, D, C));
    end for
    UpdatePheromoneTrails(τ, ρ, {A_1, ..., A_nbAnts})
  until conf(A_i) = 0 for ∀i ∈ {1, ..., nbAnts}
        or maxcycle reached
end procedure

procedure ConstructAssignment(τ, α, β, P)
begin
  A ← φ;
  while |A| < |X| do
    x_j ← SelectVariable(A, P);
    v ← ChooseValue(A, x_j, τ, α, β, (X, D, C));
    A ← A ∪ {< x_j, v >};
  end while
  return A;
end procedure
```

Fig. 1. The Ant System algorithm.

Then, pheromone trails are updated according to the set of constructed assignments, or when all ants have constructed complete assignments.

We also define the graph structure on which artificial ants are going to lay pheromone trails, associated with any CSP in a generic way. The pheromone trail graph associated with a CSP, (X, D, C) introduced by section II-A, is defined as the undirected graph, $G = (V, E)$, such that

$$\begin{aligned} V &= \{< x_i, v > |\ x_i \in X \text{ and } v \in D\}, \\ E &= \{(< x_i, v >, < x_j, w >) \in V^2 \mid x_i \neq x_j\}. \end{aligned}$$

Artificial ants communicate in an indirected way, by laying pheromone trails on the graph edge. The amount of pheromone laying on an edge $(< x_i, v >, < x_j, w >)$ is noted by $\tau(< x_i, v >, < x_j, w >)$. This pheromone information is used to guide the search as heuristics for choosing values to be assigned to variables.

In the procedure 'Ant-Solver', the artificial ant starts with an empty assignment, say A, and then iteratively selects a variable to be assigned and chooses a vertex in the graph corresponding to a value assignment for the selected variable, until all variables have been assigned. In the construction process of the assignment A, the variable, say x_j, selects randomly one of all unassigned variables. Then, the value,

procedure ConstructAssignment(τ, α, β, (X, D, C))
begin
  $A_{c-ant} \leftarrow \phi$;
  $A_{d-ant} \leftarrow A_k$ at the previous cycle;
  BorrowBlocks(A_{c-ant}, A_{d-ant});
  while $|A_{c-ant}| < |X|$ **do**
    Select $x_j \in X$ randomly;
    Choose value, v, according to pheromone trails;
    $A_{c-ant} \leftarrow A_{c-ant} \cup \{< x_j, v >\}$;
  end while
  if conf(A_{c-ant}) $<$ conf(A_{d-ant}) **then**
    $A_{d-ant} \leftarrow A_{c-ant}$;
  end if
  return A_{d-ant};
end procedure

procedure BorrowBlocks(A_{c-ant}, A_{d-ant})
begin
  Fix the partial size, $N_x (\leq |X|)$, of the assignment;
  while $|A_{c-ant}| < N_x$ **do**
    Select variable, x_d;
    $A_{c-ant} \leftarrow A_{c-ant} \cup \{< x_d, v >\} (\in A_{d-ant})$;
    $X \leftarrow X \backslash \{x_d\}$;
  end while
end procedure

Fig. 2. The part of cAS based algorithm proposed in [12] .

say v, chooses one of values which can be assigned to x_j with the probability defined as

$$p_A(< x_j, v >) = \frac{[\tau_A(< x_j, v >)]^\alpha [\eta_A < x_j, v >)]^\beta}{\sum_{w \in D_j} [\tau_A(< x_j, w >)]^\alpha [\eta_A(< x_j, w >)]^\beta}, \quad (1)$$

$$\tau_A(< x_j, v >) = \sum_{< x_k, u > \in A} \tau(< x_k, u >, < x_j, v >),$$

$$\eta_A(< x_j, v >) = \frac{1}{1 + conf(\{< x_j, v >\} \cup A) - conf(A)},$$

where $conf(A)$ denotes the number of constraint violations of the assignment A. Parameters, α and β, in the equation (1) correspond to the pheromone factor weight and the quality factor weight respectively. When increasing the value of α, artificial ants are more sensitive to pheromone trails so that they converge more quickly, i.e., the search takes a serious view of convergence. On the contrary, when increasing the value of β, the search proceeds to the direction to attach importance to the solution quality rather than pheromone trails, which tends to choose the value with few constraint violations increased.

C. Ant colony optimization with cunning ants

The cunning ant system (cAS) is first proposed by [17] to improve the performance of ACO. In cAS, two kind of ants are provided, donor and cunning ants. The donor ant preserves a candidate assignment constructed at the previous cycle. The cunning ant can construct a candidate assignment by borrowing a part of the candidate assignment that the donor ant preserves, although each ant always starts with a empty assignment to construct a candidate assignment at each cycle in AS. The cAS thus can relatively reduce search time since each ant starts with a partial assignment. Besides, a final solution with no constraint violations can be found with higher probability because promising partial assignments can be inherited to the next cycle, as shown in the experimental results of [12]. Mizuno, *et. al.* have proposed the modified cAS to apply to binary CSPs [7], [12]. Fig. 2 gives the part of the algorithm proposed by [12], where the procedure 'ConstructAssignment()' is used by replacing as the same one within the procedure 'Ant-Solver()' in Fig. 1.

III. ACO META-HEURISTICS WITH NEGATIVE PHEROMONES

A. Basic strategies

In AS and cAS mentioned in the section II-B and II-C, pheromone are updated based on the candidate assignment the best ant created in each cycle. However, even the best candidate assignment locally contains low-quality parts, or partial assignments with some constraint violations. The low-quality parts may affects the later constructed assignments because each ant constructs a candidate by referring the updated pheromone trail graph. Therefore, we propose an ACO model that uses several pheromone trails. More specifically, we provides another pheromone trail graph that stores pheromone values updated by the candidate assignment the worst ant created, called the "negative" pheromones. By using the negative pheromones as well as usual pheromones, the proposed method tries to avoid making the wrong partial assignment.

Basic ideas of the proposed model is summarized as follows:

1) In addition to the usual pheromone trail graph that is updated based on the best candidate assignment, another graph which contains "negative" pheromones is provided.
2) Each ant constructs a candidate assignment by taking account of the two types of pheromones.
3) Two pheromone trail graphs are updated at each cycle based on the candidate assignments the best and the worst ant constructed, respectively.

Our idea is straightforwardly applicable to ACO based algorithms. We thus applies the idea to AS and cAS, called ASNEP (Ant System with NEgative Pheromones) and cASNEP (cunning Ant System with NEgative Pheromones), respectively.

B. The algorithm

The graph structure of negative pheromone trails is also same as $G = (V, E)$ of usual pheromone trail denoted in section II-B. The amount of pheromone laying on an edge $(< x_i, v >, < x_j, w >)$ is noted by $N\tau(< x_i, v >, < x_j, w >)$.

The algorithms of ASNEP and cASNEP are almost the same as AS and cAS, respectively, except that the procedure

procedure UpdateNegativePheromoneTrails
($N\tau$, ρ, $\{A_1, \ldots, A_{nbAnts}\}$)
begin
  for each edge (i,j) of the pheromone graph **do**
    $N\tau(i,j) \leftarrow (1-\rho) \times N\tau(i,j)$;
  end for
  for each assignment $A \in \{A_1, \ldots, A_{nbAnts}\}$ **do**
    if $conf(A) \geq conf(A_l)$, $\forall l \in 1..nbAnts$ **then**
      for each pair of vertices $(i,j) \in A \times A$ **do**
        $N\tau(i,j) \leftarrow N\tau(i,j) + \frac{1}{conf(A)}$;
  end for
  for each edge (i,j) of the pheromone graph **do**
    if $\tau(i,j) < \tau_{min}$ **then** $\tau(i,j) \leftarrow N\tau_{min}$;
    if $\tau(i,j) > \tau_{max}$ **then** $\tau(i,j) \leftarrow N\tau_{max}$;
  end for
end procedure

Fig. 3. The procedure 'UpdateNegativePheromoneTrails()' to implement to ASNEP and cASNEP.

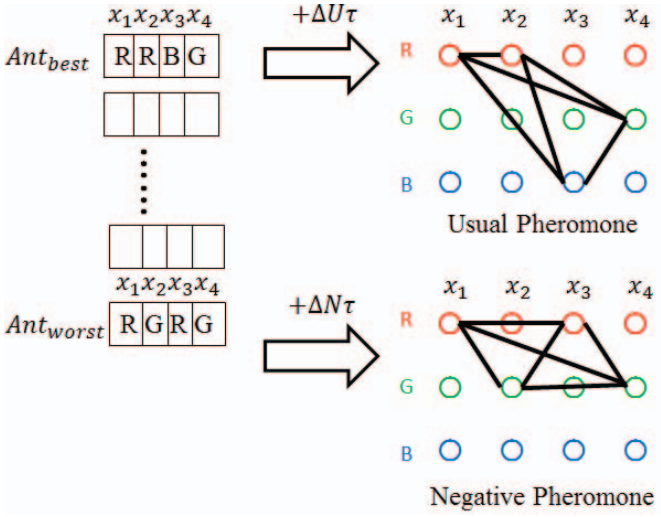

Fig. 4. Two pheromone trail graphs used in our model.

'UpdateNegativePheromoneTrails()' shown in Fig. 3 is executed after the procedure 'UpdatePheromoneTrails()' within the procedure 'Ant-Solver()' in Fig. 1. Fig. 4 gives an example of updating pheromone trails graphs in the proposed model, where both of usual and negative pheromone are updated according to each assignment constructed by the best and worst ant, respectively. In our model, the probability of assigning value v to the variable x_j is defined as

$$p_A(< x_j, v >) = \frac{\frac{[\tau_A(<x_j,v>)]^{\alpha_u}[\eta_A(x_j,v>)]^{\beta}}{[N\tau_A(<x_j,v>)]^{\alpha_n}}}{\sum_{w \in D} \frac{[\tau_A(<x_j,w>)]^{\alpha_u}[\eta_A(<x_j,w>)]^{\beta}}{[N\tau_A(<x_j,v>)]^{\alpha_n}}}, \quad (2)$$

$$N\tau_A(< x_j, v >) = \sum_{<x_k,u> \in A} N\tau(< x_k, u >, < x_j, v >),$$

where α_u and α_n are parameters corresponding to the usual and negative pheromone factor weight. As shown in equation (2), the more increased negative pheromone values seem to be lower probability of choosing the value to be assigned to the variable, enabling to avoid to construct a low-quality partial assignment.

IV. Experiments

A. Experimental settings

To evaluate the effectiveness of the proposed model, we attempt to conduct the experiments, comparing the methods, ASNEP and cASNEP, to which the proposed ACO model is implemented, with naive AS and cAS. We employ randomly generated 3COL instances with n =50, whose search space size is 3^{50} ($\simeq 10^{24}$). For 11 cases of constraint density, d = 2.0 $\sim$ 3.0 at the intervals of 0.1, we randomly generate 100 instances per case, i.e., a total of 1,100 instances.

Let us clarify the parameter setting of the methods. The number, $nbAnts$, of artificial ants is set to 50 and 'maxcycle' is set to 200, i.e., a total cost corresponds to 10,000. Weighted parameters $< \alpha, \beta >$ are fixed at $< 1, 5 >$, where $\alpha = \alpha_u = \alpha_n$. Tne pheromone trail evaporation rate, ρ, is set at 0.5% and 1%, respectively.

As for evaluation viewpoints, we compare the percentage of solved 3COL instances, i.e., the propotion of the number of instances for which the algorithm can find a solution with no constraint violations to all tried instances and average cycles required for solving on only solved instances. In each algorithm, 100 trials are executed for each instance, i.e., 10,000 trials at each d. The algorithms are implemented in Java language on a PC with 3.07GHz of Intel Core i7 880 and 4GBytes of RAMs.

B. Experimental results and discussions

Figs. 5 $\sim$ 8 give the results. Fig. 5 and Fig. 6 show the percentage of solved instances and average of total cycles when ρ = 0.5% and Fig. 7 and Fig. 8 show those when ρ = 1%.

As shown in these figures, phase transition phenomena occur clearly, where hard instances are almost concentrated around the region of d = 2.3. This result can be suitable to the results discussed in [3], [8], [10], [11]. The proposed methods, ASNEP and cASNEP, can be superior to naive AS and cAS from the viewpoint of the percentage of solved instances although the methods needs slightly more search costs. In particular, when ρ = 0.5, our proposed methods can remarkably overcome naive methods.

Also, to confirm applicability to larger-scale instances, we provide randomly generated 3COL instances with n = 100. As well as the above experiments, we randomly generate each 100 instances for 11 cases of d = 2.0 $\sim$ 3.0 at the intervals of 0.1. Fig. 9 and Fig. 10 give the result of the percentage of solved instances and average cycles when ρ is fixed at 0.5 for all methods. Although the performance

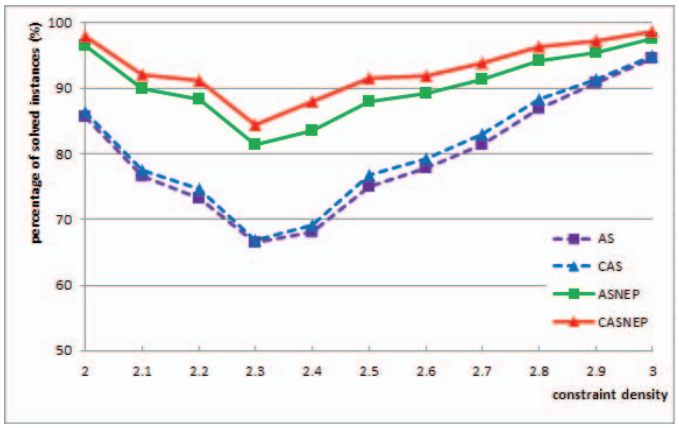

Fig. 5. Experimental result on the percentage of solved instances, where n = 50 and ρ = 0.5%.

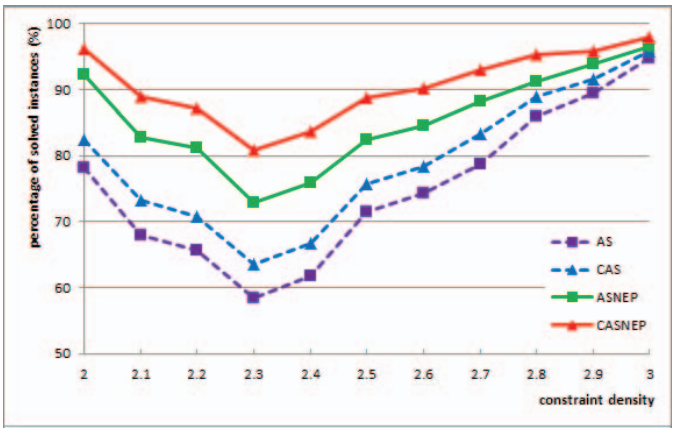

Fig. 7. Experimental result on the percentage of solved instances, where n = 50 and ρ = 1.0%.

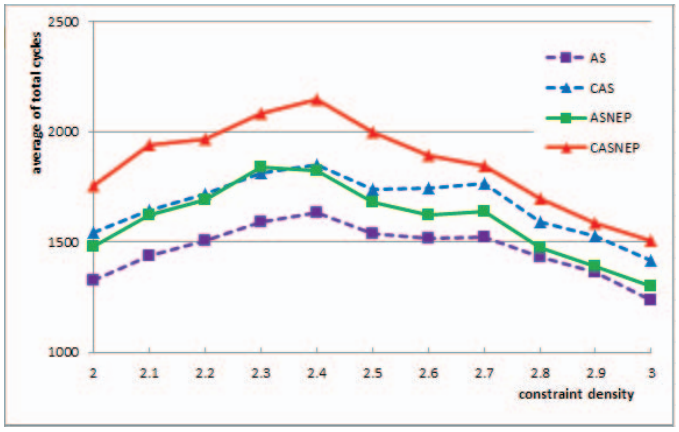

Fig. 6. Experimental result on the average of total cycles, where n = 50 and ρ = 0.5%.

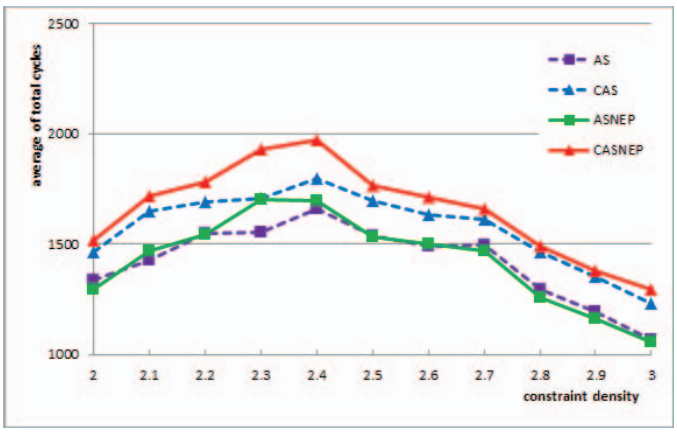

Fig. 8. Experimental result on the average of total cycles, where n = 50 and ρ = 1.0%.

of ASNEP seems to be almost similar to AS and cAS, the percentage of solved instances of cASNEP is superior to other methods, demonstrating that adopting negative pheromone can be effective for cAS. As for ASNEP, we cannot obtain stable results, but it should be significant and interested to make effectiveness of the variation of our method clear by conducting more experiments using various types of CSP instances.

V. Conclusion

We have proposed an ACO model that have multiple pheromone information. Our model described in this paper has the negative pheromone graph, which is updated based on the worst constructed assignment in addition to the usual pheromone trail graph. We have also implemented the proposed model to the ant system and the cunning ant system and demonstrated that our model can be effective on search efficiency by solving hard graph coloring problems.

Our future works should consist in conducting more experiments by comparing with other meta-heuristics, e.g., particle swarm intelligence, artificial bee colony, firefly algorithms, etc., applied to other types of CSPs such as binary constraint satisfaction problems and propositional satisfiability problems, clarifying applicable ranges of our method.

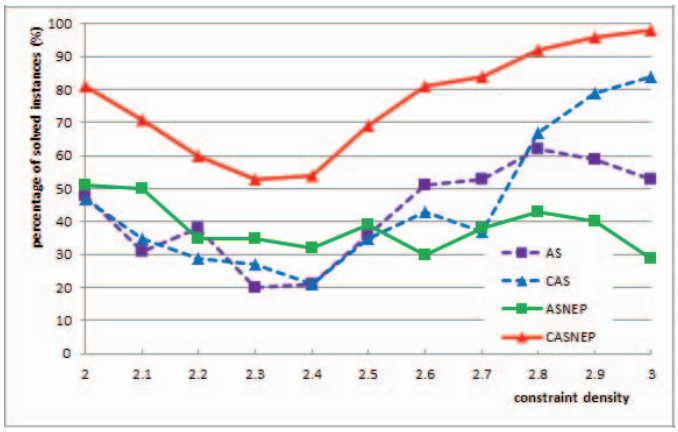

Fig. 9. Experimental result on the percentage of solved instances, where n = 100 and ρ = 0.5%.

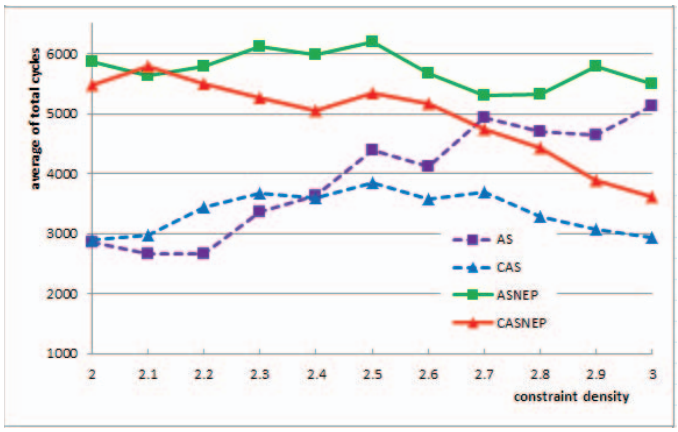

Fig. 10. Experimental result on the average of total cycles, where n = 100 and ρ = 0.5%.

ACKNOWLEDGMENT

This research was supported in part by Grant-in-Aid for Scientific Research (C), No. 25330267, Japan Society for the Promotion of Science, 2013-2016.

REFERENCES

[1] Bell, J. E. and McMullen, P. R.: Ant colony optimization techniques for the vehicle routing problem, *Advanced Engeneering Informatics*, Vol. 18(1), pp. 41–48 (2004).

[2] Bui, T. N., Nguyen, T. H., Petal, C. M., Phan, K. T.: An ant-based algorithm for coloring graphs, *Discrete Applied Mathematics*, Vol. 156, pp. 190–200 (2008).

[3] Cheeseman, P., Kanelfy, B., Walsh, T.: Where the really hard problems are, *Proc, IJCAI'91*, pp. 331–337 (1991).

[4] Dorigo, M., Di Caro, G.: The Ant Colony Optimization meta-heuristic, *New Ideas in Optimization*, pp. 11–32 (1996).

[5] Dorigo, M., *et. al.*: The Ant System: Optimization by a Colony of Cooperating Agents, *IEEE Transaction on Systems, Man, and Cybernetics* - Part B, Vol. 26, pp. 26–41 (1996).

[6] Dorigo, M. and Di Caro, G.: The Ant Colony Optimization Meta-Heuristics, *New Ideas in Optimization*, pp. 11–32 (1999).

[7] Hayakawa, D., Mizuno, K., Sasaki, H. and Nishihara, S.: Solving Constraint Satisfaction Problems by A Population Based Cunning Ant System, *The 2012 International Conf. on Technologies and Applications of Artificial Intelligence (TAAI2012)*, pp. 205–210 (2012).

[8] Hogg, T., Huberman, B. A., Williams, C. P.: Phase transition and search problem, *Artificial Intelligence*, Vol. 81, pp. 1–16 (1996).

[9] Minton, S., *et. al.*: Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems, *Artificial Intelligence*, Vol. 58, pp. 161–205 (1992).

[10] Mizuno, K. Nishihara, S., *et. al.*: Population migration: a meta-heuristics for stochastic approaches to constraint satisfaction problems, *Informatica*, Vol. 25, pp. 421–429 (2001).

[11] Mizuno, K. Nishihara, S.: Constructive generation of very hard 3-colorability instances, *Discrete Applied Mathematics*, Vol. 156(2), pp. 218–229 (2008).

[12] Mizuno, K., Hayakawa, D., Sasaki, H., and Nishihara, S.: Solving Constraint Satisfaction Problems by ACO with Cunning Ants, *The 2011 Conf. on Technologies and Applications of Artificial Intelligence (TAAI2011)* (2011).

[13] Morris, P.: The Breakout Method for Escaping From Local Minima, *Proc. AAAI'93*, pp. 40–45 (1993).

[14] Selman, B., Kautz, H., Cohen, B.: Noise Strategies for Improving Local Search, *Proc. AAAI'94*, pp 337–343 (1994).

[15] Solnon, C.: Ants can solve constraint satisfaction problems, *IEEE Transactions on Evolutionary Computation*, Vol. 6, pp. 347–357 (2002).

[16] Tayarani-N, M. H. and Prugel-Bennett, A.: Anatomy of the fitness landscape for dense graph-colouring problem, *Swarm and Evolutionary Computation*, Vol. 22, pp. 47–65 (2015).

[17] Tsutsui, S.: cAS: Ant Colony Optimization with Cunning Ants, *Proc. of the 9th Int. Conf. on Parallel Problem Solving from Nature (PPSN IX)*, pp. 162–171 (2006).