Solving Constraint Satisfaction Problems by Cunning Ants with multi-Pheromones

Takuya Masukane and Kazunori Mizuno

***Abstract*—To solve large-scale constraint satisfaction problems, CSPs, ant colony optimization, ACO, based meta-heuristics has been effective. Many methods based on ACO have been proposed including the cunning Ant System, cAS. However, some of these methods cannot be stable to solve CSPs. In this paper, we propose an ant colony optimization based meta-heuristics with multi pheromone trails. Artificial ants construct candidate assignments by referring several pheromone trail graphs to solve CSP instances. We also implement the proposed model to cAS method and demonstrate how our method is effective for solving large scale and hard graph coloring problems that are one of typical examples of CSPs.**

***Index Terms*—Ant colony optimization, constraint satisfaction, graph coloring, meta heuristics.**

I. Introduction

A constraint satisfaction problem (CSP) is a problem to find a combination of values satisfying all given constrains. Artificial intelligence and pattern analysis, including planning or scheduling, can be an application of such CSP.

To solve large-scale CSPs with an exhaustive search, it takes impractical costs. Therefore, we must cut down wasted search costs. That's why meta-heuristics, which are stochastic search approaches, has been remarkable for the technique to solve CSPs [1]-[6]. Ant colony optimization (ACO), genetic algorithm (GA) and simulated annealing (SA), which are typical meta-heuristics, are proved their effectiveness for CSPs by experiments.

Ant colony optimization (ACO) is one of typical metaheuristics which model real ants' behavior to find path from their colony to their food [6], [7]. ACO has been effective for many combinatorial search and optimization problems such as traveling salesperson problems (TSP) [7]-[8], graph coloring problems (COL) [9], or vehicle routing problems (VRP) [1].

The Ant System (AS) is the initial algorithm based on ACO [7]. Although many algorithms using ACO have been proposed, most of them take a lot of search time because of reconstruction of candidate solutions at each cycle or iteration. In contrast, the cunning Ant System (cAS) proposed by [10] can generate a candidate solution by borrowing a part of the candidate solution generated at the previous cycle. The performance of cAS has been evaluated by applying to the TSP in [10]. Furthermore, novel approaches based on cAS have been proposed and evaluating their efficiency by applying to binary CSPs in [11], [12].

Manuscript received May 15, 2018; revised July 20, 2018.

The authors are with Takushoku University, Hachioji, Tokyo 193-0985 Japan (e-mail: phsl.masukane@gmail.com, mizuno@cs.takushoku-u.ac.jp).

In the AS and the cAS, each ant constructs an assignment by referring a single pheromone trail graph that is periodically updated based on the best constructed assignment with the lowest constraint violations at each cycle. At the early search, however, the pheromone trail graph has less bias and less effectiveness for the search. Therefore, if we make the term where there is less bias on the pheromone trail graph shorter, the search would be more efficient.

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 graph, 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 the early search term which the pheromone trail graph is not effective can be decreased due to referring to another pheromone information stored bad assignments. We have also implemented the algorithms, where the proposed model is introduced to the cAS and demonstrated that the proposed method can be more effective than the naive 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 [13]-[15]. 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.

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

$X = \{x_1, ..., x_n\}$ is a finite set of variables.

D is a set of values to be assigned to variables.

$C = \{c_1, ..., c_m\}$ 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 [3], [15], [16]. Arranging to the above definition of CSPs, X and C correspond to the set of

doi: 10.18178/ijmlc.2018.8.4.712

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.

```
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

procedure UpdatePheromoneTrails
                    (τ, ρ, {A_1, ..., A_nbAnts})
begin
  for each edge (i, j) of the pheromone graph do
    τ(i, j) ← (1 − ρ) × τ(i, j);
  end for
  for each assignment A ∈ {A_1, ..., A_nbAnts} do
    if conf(A) ≤ conf(A_l), ∀l ∈ 1..nbAnts then
      for each pair of vertices (i, j) ∈ A × A do
        τ(i, j) ← τ(i, j) + 1/conf(A);
  end for
  for each edge (i, j) of the pheromone graph do
    if τ(i, j) < τ_min then τ(i, j) ← τ_min;
    if τ(i, j) > τ_max then τ(i, j) ← τ_max;
  end for
end procedure
```

Fig. 1. The part of the ant system algorithm.

In this paper, we define the constraint density, d, as $d=m/n$, where $m=|C|$ and $n=|X|$, respectively. Many research reports have observed phase transition phenomena in combinatorial search and optimization problems including CSPs and 3COLs. In CSPs and 3COLs, the search cost follows an easy-hard-easy pattern 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, $d=2.3$-2.4 (i.e., the average degree of the graph is around 4.6), where phase transition phenomena occur [3], [14], [15]. Based on the above, we randomly generate 3COL instances with $d=2.0$-3.0 to evaluate the proposed algorithms.

B. Ant Colony Optimization

Ant colony optimization (ACO) is one of swarm intelligence based meta-heuristics which model real ants' behavior to find path from their colony to their food [6], [7]. Artificial ants, which are used instead of real 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. In addition, pheromone trails are progressively decreased, simulating some kind of evaporation.

The Ant System (AS) is the ACO based search algorithm proposed by [8]. Fig. 1 gives the algorithm of the AS. In the procedure, 'Ant-Solver' [6], in Fig. 1, every artificial ant constructs a complete assignment, or candidate solution, of values to all variables in the procedure 'ConstructAssignment.' Then, pheromone trails are updated according to the set of constructed assignments, or when all ants have constructed complete assignments in the procedure 'UpdatePheromoneTrails.'

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

$$V=\{<x_i,v>|\ x_i\in X \text{ and } v\in D\},$$

$$E=\{(<x_i,v>,<x_j,w>)\in V^2\mid x_i\neq x_j\}.$$

Artificial ants communicate each other by referring to pheromone on the graph edge. The amount of pheromone on an edge $(<x_i,v>,<x_j,w>)$ is noted by $\tau(<x_i,v>,<x_j,w>)$. This pheromone information leads ants' assignment to be converged.

At first of the algorithm, the artificial ant has an assignment, say A, which is empty. Then, it iteratively selects a variable to be assigned and chooses a value to assign to the selected variable until all variables have been assigned. In the constructing the assignment A, the variable, say x_j, is selected randomly from all unassigned variables. Then, the value, say v, chooses one of values which can be assigned to x_j with the probability defined as

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

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

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

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.

After each ant constructs the candidate assignment, the pheromone trail graph is updated in procedure 'UpdatePheromoneTrails' at each cycle.

C. Ant Colony Optimization with Cunning Ants

The cunning ant system (cAS), which is proposed by [10] to improve the performance of ACO, two kinds of ants, donor and cunning ants, are provided. 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. In the AS, each ant always starts with an empty assignment at each cycle. For the borrowing, however, the cAS 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*. has proposed the modified cAS to apply to binary CSPs [11], [12]. Fig. 2 gives the part of the algorithm proposed by [12].

procedure ConstructAssignment$(\tau, \alpha, \beta, (X, D, C))$
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;
    $A_{c-ant} \leftarrow A_{c-ant} \cup \{< x_d, v >\} (\in A_{d-ant})$;
    $X \leftarrow X \setminus \{x_d\}$;
  end while
end procedure

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

III. ACO with Negative Pheromones

A. Basic Strategies

In the AS and the cAS mentioned in the section II-B and II-C, a candidate assignment construction depends on the pheromone trail graph and constraint violations. Each amount of pheromone on the pheromone trails is commonly initialized to the same value. As the search progresses, the variation of the amount of pheromone becomes larger. Conversely, at the early search, the pheromone trail graph has less bias. Because of less bias, the probability to choose the value is almost the same. Therefore, pheromones should have little effect for the search at the early search and the search does not proceed quite easily. Accordingly, to make the early search more efficient should improve efficiency of the entire search.

In this paper, we propose an ACO model that uses several pheromone trails as the heuristics for assigning values [17]. More specifically, we provide another pheromone trail graph that stores pheromone values updated by the candidate assignment which the *worst* ant created. *Worst* denotes to have an assignment with the most constraint violations. We call the proposed pheromones "negative" pheromones. By using the negative pheromones as well as usual pheromones, the proposed method tries to avoid making the wrong partial assignment. For this avoidance, the proposed method promises less search times than the naive method.

Basic ideas of the proposed model are 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 apply the idea to the cAS, called the cASNEP (cunning Ant System with NEgative Pheromones).

B. The Algorithm

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

procedure UpdateNegativePheromoneTrails
  $(N\tau, \rho, \{A_1, ..., A_{nbAnts}\})$
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 $A \in \{A_1, ..., A_{nbAnts}\}$ **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**
        $N\tau(i, j) \leftarrow N\tau(i, j) + conf(A)$;
  end for
  for each edge (i, j) of the pheromone graph **do**
    if $N\tau(i, j) < N\tau_{min}$ **then** $N\tau(i, j) \leftarrow N\tau_{min}$;
    if $N\tau(i, j) > N\tau_{max}$ **then** $N\tau(i, j) \leftarrow N\tau_{max}$;
  end for
end procedure

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

The algorithm of the cASNEP is almost same as the cAS, except that the procedure 'UpdateNegativePheromoneTrails()' shown in Fig. 3 is executed after the procedure 'UpdatePheromoneTrails()'. In our model, the probability of assigning value v to the variable x_j is defined as

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

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

where α_u and α_n are parameters correspond 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 constructing a low-quality partial assignment.

IV. Experiments

A. Experimental Settings

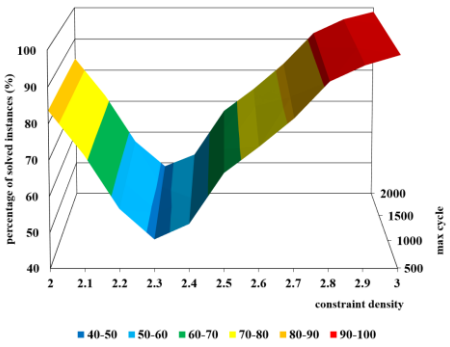
To evaluate the effectiveness of the proposed model, we attempt to conduct the experiments, comparing the methods, cASNEP, which is the proposed ACO model, with the naive cAS. We employ randomly generated 3COL instances with $n=100$, whose search space size is 3^{100} ($\simeq 10^{47}$). For 11 cases of constraint density, $d=2.0$-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 of artificial ants, *nbAnts*, is set to 100 and 'maxcycle' is set to 2000, i.e., a total cost corresponds to 200,000. Weighted parameters $<\alpha,\beta>$ are fixed at $<5,10>$, where $\alpha=\alpha_u=\alpha_n$. The pheromone trail evaporation rate, ρ, is set at 1%. These parameters are set according to pre-experimentation.

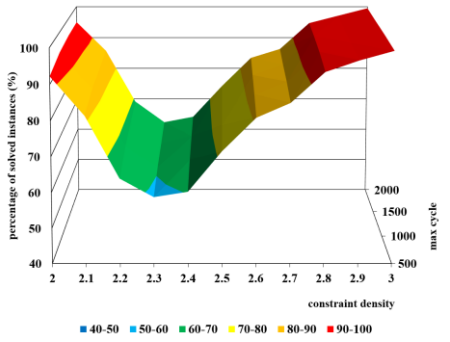
We evaluate cAS and cASNEP algorithm from two viewpoints: the "percentage of solved 3COL instances" and the "average of the generated candidate assignment." The "percentage of solved instances" denotes the proportion of the number of instances for which the algorithm can find a solution with no constraint violations to all tried instances. The "average of the generated candidate assignment" denotes the mean value of the number of assignments generated until a solution with no constraint violations is found. In each algorithm, 10 trials are executed for each instance, i.e., 1,000 trials at each *d*. The algorithms are implemented in language Java on a PC with 3.07GHz of Intel Core i7 880 and 4GBytes of RAMs.

B. Experimental Result and Discussions

Fig. 4 and Fig. 5 give the results of the experiments. Fig. 4(a) and Fig. 4(b) give the percentage of solved instances in the cAS and the cASNEP algorithms, respectively. Fig. 5(a) and Fig. 5(b) give the average of generated candidate assignment in the cAS and the cASNEP algorithms, respectively. In Fig. 4 and Fig. 5, the horizontal axis is the constraint density, *d*, and the depth axis is the cost to take for solving instances, respectively. The vertical axis in Fig. 4 and Fig. 5 corresponds the percentage of solved instances against tried instances and the average number of the assignment to solve instances, respectively. As shown in Fig. 4, 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], [13]-[15]. In Fig. 4, the proposed method, cASNEP, can be superior to the naive cAS from the viewpoint of the percentage of solved instances.

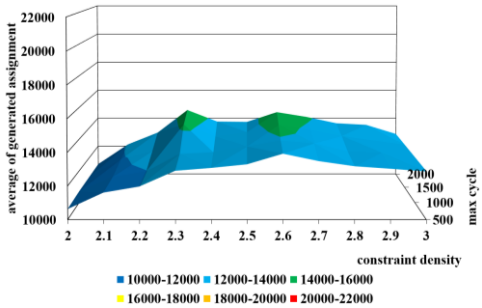


(a) cAS

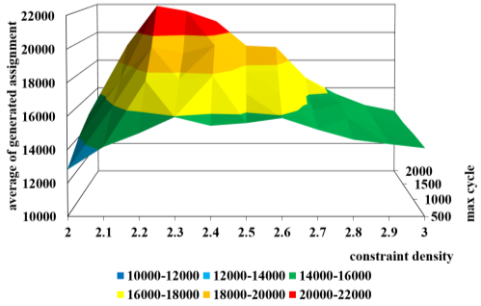


(b) cASNEP

Fig. 4. Experimental result on the percentage of solved instances.



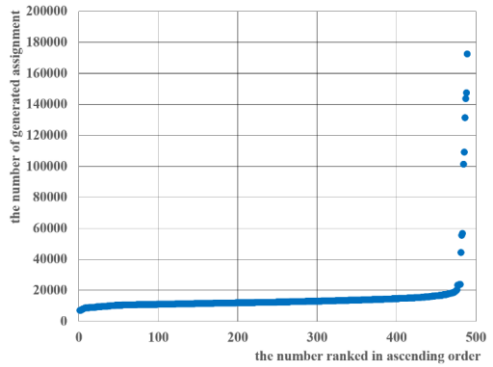
(a) cAS



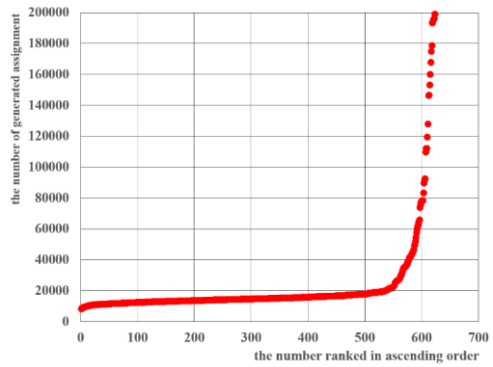
(b) cASNEP

Fig. 5. Experimental result on the average of generated assignments.

In particular, our proposed method can remarkably overcome the naive method around the region of $d = 2.3$, i.e., phase transition regions. However, as shown in Fig. 5, the cASNEP solves instances with more assignment construction averagely than the cAS in the most tried cases. Therefore, applying negative pheromone seems to be an inefficient method from the viewpoint of the search cost. Then, we observe the individual number of generated assignment. Fig. 6 shows the number of candidate solutions while the cAS and the cASNEP solved instances in ascending order, where $d = 2.3$. As shown in Fig. 6, the cAS and the cASNEP solve most instances with 20,000 assignments constructions. Therefore, both of the cAS and the cASNEP solve instances stably within 20,000 assignment constructions. In particular, the cAS algorithm solves 475 instances and the cASNEP algorithm solves 535 instances within 20,000 assignment constructions (i.e. the cASNEP solves more 60 instances which equal to 6% of the all tried instances than the cAS). This fact shows that applying negative pheromones should lead to increased stability. Also, about instances solved 20,000 costs, the number of instances solved by the cASNEP is more than the number of instances solved by the cAS. Owing to this, the average of generated assignment rises by applying negative pheromones in Fig. 5. Therefore, our proposed method should find the assignment with no consraint violation more stably than the naive method.



(a) cAS



(b) cASNEP

Fig. 6. Experimental result on the number of generated assignments, where $d = 2.3$.

C. Further Discussion

In Sec. IV-B, we confirmed the effectiveness of the proposed method. In this section, to evaluate the proposed method's reliability, we also perform some experiments from two viewpoints: the variation of ρ and the size of instances. We use the only instances with $d = 2.3$, which are in the phase transition region, for each experiment here.

1) The variation of ρ: We generated 100 instances with $n = 100$ and $d = 2.3$ randomly. Then, we evaluate the cAS and the cASNEP from the viewpoint of the percentage of the solved instances for 6 cases of the pheromone trail evaporation rate, $\rho =$ 0.1, 0.2, 0.5, 1.0, 2.0, 5.0%. Fig. 7 gives the experimental result. In the naive cAS method, the percentage of solved instances changes sensitively against the variation of ρ. However, in most of the case of ρ, the proposed cASNEP can solve the instances with the high percentage. In particular, the robustness against the variation of the ρ can be seen in $\rho =$ 0.2-2.0% clearly. In the cAS method, the range of the percentage of the solved instances between $\rho =$ 0.2% and 2.0% is about 40 points. However, in our proposed cASNEP method, the range is only about 20 points. Therefore, our proposed cASNEP method can be more convenient method for the ease of the parameter tuning than the naive cAS method.

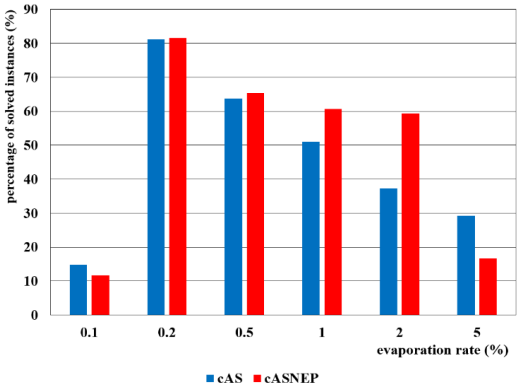


Fig. 7. Experimental result for the variation of ρ.

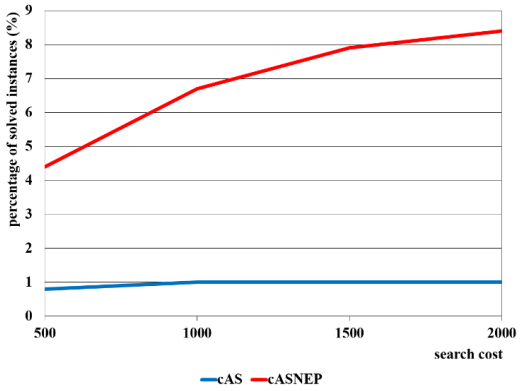


Fig. 8. Experimental result where $n = 200$.

2) The larger instances: We generated 100 instances with $n = 200$ and $d = 2.3$, where the search space size is 3^{200} ($\simeq 10^{95}$), randomly. Therefore, the search space size of $n = 200$ is 10^{48} times of the search space size of $n = 100$. Then, we evaluate the cAS and the cASNEP from the viewpoint of the percentage of the solved instances. Fig. 8 gives the experimental result. In comparison to the experimental result in Sec. IV-B (Fig. 4), because of difficulty of the instances which $n = 200$, the percentage of solved instances of both of the cAS method and the cASNEP method is lower in Fig. 8. Besides, in the naive cAS method, there is less growth of the percentage of the solved instances

as the cost for the search increases. However, in the proposed cASNEP method, the percentage of the solved instances rises from maxcycle = 500 to maxcycle = 2000 slightly. When we give more search cost to the cASNEP method, the percentage of the solved instances may grow much higher. Therefore, our proposed cASNEP method can have more adaptability to the large instances than the cAS method.

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 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 making experiments by comparing with other meta-heuristics, e.g., particle swarm intelligence, artificial bee colony, firefly algorithms, etc., applying to other types of CSPs such as binary constraint satisfaction problems and propositional satisfiability problems.

References

[1] J. E. Bell and P. R. McMullen, "Ant colony optimization techniques for the vehicle routing problem," *Advanced Engineering Informatics*, vol. 18, nol. 1, pp. 41-48, 2004.

[2] S. Minton *et al.*, "Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems," *Artificial Intelligence*, vol. 58, pp. 161-205, 1992.

[3] K. Mizuno, S. Nishihara, H. Kanoh, and I. Kishi, "Population migration: A meta-heuristics for stochastic approaches to constraint satisfaction problems," *Informatica*, vol. 25, pp. 421-429, 2001.

[4] P. Morris, "The breakout method for escaping from local minima," in *Proc. AAAI '93*, pp. 40-45, 1993.

[5] B. Selman, H. Kautz, and B. Cohen, "Noise strategies for improving local search," in *Proc. AAAI'94*, pp. 337–343, 1994.

[6] C. Solnon, "Ants can solve constraint satisfaction problems," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 347-357, 2002.

[7] M. Dorigo and G. D. Caro, "The ant colony optimization meta-heuristics," *New Ideas in Optimization*, pp. 11-32, 1999.

[8] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: Optimization by a colony of cooperating agents," *IEEE Transaction on Systems, Man, and Cybernetics - Part B*, vol. 26, pp. 29-41, 1996.

[9] T. N., Bui, T. H. Nguyen, C. M. Petal, and K. T. Phan, "An ant-based algorithm for coloring graphs," *Discrete Applied Mathematics*, vol. 156, pp. 190-200, 2008.

[10] S. Tsutsui, "cAS: Ant colony optimization with cunning ants," in *Proc. the 9th Int. Conf. on Parallel Problem Solving from Nature (PPSN IX)*, 2006, pp. 162–171.

[11] D. Hayakawa, K. Mizuno, H. Sasaki,and S. Nishihara, "Solving constraint satisfaction problems by a population based cunning ant system," in *Proc. the 2012 International Conf. on Technologies and Applications of Artificial Intelligence*, 2012, pp. 205-210.

[12] K. Mizuno, D. Hayakawa H. Sasaki, and S. Nishihara, "Solving constraint satisfaction problems by ACO with cunning ants," in *Proc. The 2011 Conf. on Technologies and Applications of Artificial Intelligence*, 2011.

[13] P. Cheeseman, B. Kanelfy, and T. Walsh, "Where the really hard problems are," in *Proc. IJCAI '91*, 1991, pp. 331-337.

[14] T. Hogg, B. A. Huberman, and C. P. Williams, "Phase transition and the search problem," *Artificial Intelligence*, vol. 81, pp. 1-15, 1996.

[15] K. Mizuno and S. Nishihara, "Constructive generation of very hard 3-colorability instances," *Discrete Applied Mathematics*, vol. 156, no. 2, pp. 218-229, 2008.

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

[17] T. Masukane and K. Mizuno, "Ant colony optimization with multi-pheromones for solving constraint satisfaction problems," in *Proc. the 2016 International Conf. on Technologies and Applications of Artificial Intelligence*, 2016, pp. 110-115.

Takuya Masukane graduated from the Department of Computer Science, Faculty of Engineering, Takushoku University, Japan, in 2016. He is currently enrolled in the graduate school of Takushoku University. He belongs to Mizuno Laboratory in Takushoku University. His area of research is constraint satisfaction problems and ant colony optimization.

Kazunori Mizuno received the B.Sc, M.Eng and Ph.D in engineering from University of Tsukuba, Japan, in 1996, 1998 and 2001, respectively. Since 2017, he is a professor at the Department of Computer Science, Takushoku University, Japan. His research interests include knowledge processing, combinatorial search algorithms, constraint satisfaction, evolutionary algorithms and multi-agent simulation.