Population migration: a meta-heuristics for stochastic approaches to constraint satisfaction problems

Kazunori Mizuno, Seiichi Nishihara, Hitoshi Kanoh and Isao Kishi
Institute of Information Sciences and Electronics, University of Tsukuba,
Tsukuba, Ibaraki 305-8573, Japan
mizuno@algor.is.tsukuba.ac.jp, http://www.npal.is.tsukuba.ac.jp/

Keywords: constraint satisfaction, search algorithms, stochastic search, meta-heuristics

Received: July 25, 2000

A meta-heuristics for escaping from local optima to solve constraint satisfaction problems is proposed, which enables self-adaptive dynamic control of the temperature to adjust the locality of stochastic search. In our method, several groups with different temperatures are prepared. To each group the same number of candidate solutions are initially allotted. Then, the main process is repeated until the procedure comes to a certain convergence. The main process is composed of two phases: stochastic searching and population tuning. As for the latter phase, after evaluating the adaptation value of every group, migration of some number of candidate solutions in groups with lower values to groups with higher values are induced. Population migration is a kind of parallel version of simulated annealing, where several temperatures are spatially distributed. Some experiments are performed to verify the efficiency of the method applied to constraint satisfaction problems. It is also demonstrated that population migration is exceptionally effective in the critical region where phase transitions occur.

1 Introduction

A constraint satisfaction problem (CSP) involves finding values for problem variables which are subject to constraints specifying the acceptable combinations of values. Such combinatorial search problems are ubiquitous in artificial intelligence and pattern analysis, including scheduling and planning problems. Most of the previous work on CSP algorithms has adopted a systematic backtracking-based approach in which a partial assignment to the variables is incrementally extended. However, this approach often needs too much time to find a solution on large-scale problems due to their exponential complexity. In contrast, a repair-based stochastic approach, which starts with a complete but inconsistent assignment and then repeats repairs of constraint violations until a consistent assignment is achieved[1], has recently made remarkable progress because this approach may sometimes solve large-scale problems in a practical time. However, this approach has a drawback of getting caught in locally optimal states that are not acceptable as solutions. Therefore, many techniques to escape from local optima have recently been proposed for stochastic approaches[2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]. We call such techniques meta-heuristics.

Simulated annealing (SA) has been studied as a kind of meta-heuristics which is widely applicable to stochastic approaches[2, 11, 12, 14]. SA involves a unique operation, after which it was named, that gradually reduces the value of a parameter, or a temperature, for determining a state transition probability. Accordingly, as the search proceeds, its focal area varies from global to local. The main drawback to SA, however, is its difficulty in deciding in advance the schedule of temperature reduction because this depends on each problem.

In this paper, we propose a new meta-heuristics with a self-adjustment mechanism which automatically, but implicitly, controls its temperature schedule for a given problem during the search. First, several groups preassigned with different temperatures are created, in each of which an equal number of candidate solutions are stored. Then, the main process is repeated until the system comes to a certain convergence. The main process is composed of two phases: searching and population tuning. As for the latter phase, after evaluating all the adaptation values of groups, migration operations are executed, in which a proper number of candidate solutions in groups with low adaptation values are moved into groups with high adaptation values taking account of how far those values differ from the average.

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 problems tend to happen in situations very similar to physical phase transitions. Hence it is important for the studies of meta-heuristics to place their interests on how well they cope with phase transitions. We show that our method is efficient especially for hard problems found in the region of phase transitions.

In section 2, after reviewing stochastic approaches to

CSPs and meta-heuristics for them, we explain the phase transition of graph-colouring problem, which has become one of the standard benchmark problems for testing CSP algorithms. In section 3, we give the basic idea of population migration and propose a concrete algorithm. In section 4, we show experimental results for our method from two viewpoints: its efficiency compared with other meta-heuristics and its effect on phase transitions.

2 Stochastic Approaches and Phase Transition

2.1 Stochastic Approaches

Many methods to solve CSPs have been proposed. Two approaches can be distinguished: systematic and stochastic. The former is the constructive approach to a solution based on tree search with backtracking. Many heuristics like forward checking, partial and full-looking ahead, back marking, arc and path-consistency have been developed. However, it is still difficult to solve really hard CSPs completely within tractable time. In contrast, the stochastic approach based on the repair-oriented search starting from an initial candidate solution (i.e., an inconsistent complete assignment of values to all variables) often gives a final or semi-optimal solution in a practical time. Hill-climbing is one of the standard search algorithms that navigates the search space while attempting to minimize the total constraint violations included in the present candidate solution. It is efficient when the landscape of search space is simple, i.e. single peaked or so, but, at times, hill-climbing tends to get caught in local optima that are not acceptable as solutions. Stochastic hill-climbing, abbreviated SHC[2], is a revocable hill-climbing that permits random shifts to directions with no improvement in some non-zero probability depending on a given temperature, which may help the search escape from local optima.

Up to now, many meta-heuristics exist that give ways to avoid local optima, like restarting with another candidate solution generated randomly[3, 6, 9], adjusting the evaluation function by increasing the weight of unsatisfied constraints[4, 5, 8, 13], introducing a state transition probability to determine the next state[2, 10], and simulated annealing (SA)[2, 11, 14].

SA, modeled after the annealing process of statistical mechanics, is a general-purpose stochastic technique that is effective in approximating global optima for many NP-hard combinatorial problems[2, 12]. Figure 1 shows the SA algorithm to which SHC is incorporated as the basic local search method. SA works on SHC as meta-heuristics by generating monotonically decreasing temperature values, T, which are iteratively used to control the transition probability in the SHC procedure. A temperature decrease corresponds to a narrowing of the search area of SHC from global to local.

As a fundamental result, it is known that SA certainly guides to a global optimum when the temperature is set initially to a large enough value and then reduced logarithmically. However, since logarithmic reduction is too slow for practical use, the decay rate $\gamma(0 < \gamma < 1)$ is generally used instead to control the temperature from T_max to T_min, as shown in Figure 1. Determining the best decay rate γ in advance is difficult because it depends on each problem instance. Further, there is always the risk that SA may freeze before it finds a global optimum when the starting state is not chosen appropriately.

```
procedure simulated annealing(){
  generate a candidate solution, s ;
  for ( T = T_max ; T ≥ T_min ; T := T × γ ) {
    SHC( T ) ;
  };
}

procedure SHC( T ) {
  for ( hc = 1 ; hc ≤ hc-max ; hc := hc + 1){
    calculate the constraint satisfaction ratio ;
    randomly select a variable v with constraint violations ;
    randomly select a value c for v ;
    assign c to v with probability p = 1 / ( 1 + exp( Δ / T ) ;
  };
}
```

Figure 1: The SA algorithm, in which Δ indicates how the number of constraint violations changes by replacing the value of v in s by c.

2.2 Phase Transition of Graph-Colouring Problems

As a well studied NP-complete problem, the graph-colouring problem has often been used to evaluate combinatorial algorithms empirically. We also employ the 3-colouring problem, GCP for short, to test the efficiency of the method that we propose in Section 3. An instance of GCP is defined as a triple (V, C, E), where $V = \{v_1, \cdots, v_n\}$ is a set of variables, $C = \{red, blue, green\}$ is a set of values (different colours) which should be assigned to each variable, and $E = \{e_1, \cdots, e_m\} \subseteq V \times V$ is a set of binary constraints. Notice that (V, E) corresponds to an undirected graph, where V is the set of nodes, and E is the set of edges. An edge $e = (v_p, v_q)$ in E stands for the constraint claiming variables v_p and v_q should not have the same value.

Several recent papers have observed phase transitions: matter commonly undergoes dramatic changes in its qualitative properties when certain parameters pass through particular values[15, 16, 17, 18, 19]. In GCPs also, the solution cost follows an easy-hard-easy pattern[17] as a function of the constraint density, d, which is the ratio of the number of constraints m to the number of variables n. Actually, when the density d is increased gradually, GCPs suddenly become hard to solve in the sense of the computational complexity in the region where d varies from 2 to 3[15]. These surprising phenomena are understood to

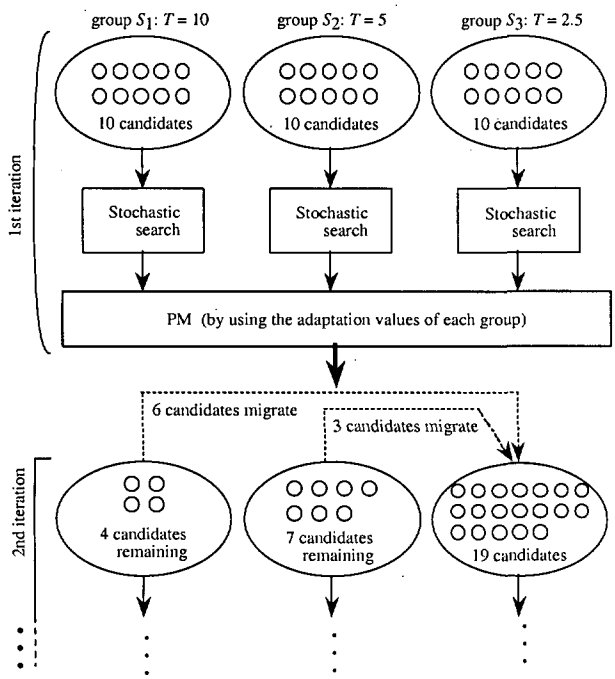


Figure 2: Population migration dynamically updates the allocation of candidates among group.

happen due to the competition between a decreasing number of solutions and an increasing number of prunings[16]. Mammen and Hogg have found another kind of easy-hard-easy pattern which is observed even when the number of solutions is held constant; the pattern in these cases appears to be due to changes in the size of the minimal unsolvable subproblems rather than the changing number of solutions[17].

For any systematic or stochastic method getting complete solutions for critically constrained problems, which are usually found around the transition point, would take an exponential order of computation time. Some known stochastic methods boosted by appropriate meta-heuristics like SA can often find optimal or semi-optimal solutions in an acceptable time. However, most of the meta-heuristics have not necessarily been proposed to cope with phase transitions.

3 Population Migration Strategy — A Meta-Heuristics

3.1 Basic Ideas

Instead of the temporal reduction of temperature in SA, in our method we prepare a set of initial temperature values and for all values a normal stochastic search is performed independently in parallel. We summarize our method in three points as follows:

(i) We determine a set of temperature values that do not change throughout the search process.

(ii) For each temperature, a finite set of candidate solutions is created initially and processed by some basic stochastic search algorithm, SHC in our case.

(iii) The size of each set, called a group, of candidate solutions is adjusted periodically taking its current adaptation value into account.

Periodical tuning of population distribution introduced in (iii) is the key operation and the reason for calling our method population migration, abbreviated as PM. Thus our method can be regarded as a parallel version of SA where the set of available temperatures is fixed in advance. Figure 2 illustrates the mechanism of population tuning. First, three groups, S_1, S_2 and S_3, allotted with the same number, say 10, of candidate solutions, called candidates shortly, are created. To each group a different temperature value is assigned: 10, 5 and 2.5, respectively. For each group, the stochastic search is performed for a predefined period of time, and then a new adaptation value is calculated. In our example, let us assume the updated values in the first iteration become $g(S_1) < g(S_2) < g(S_3)$ and $g(S_2) < \widetilde{g}$, where $g(x)$ is the adaptation value of group x, and $\widetilde{g}$ is the mean value of $g(S_1)$, $g(S_2)$ and $g(S_3)$. At this point PM is started to reorganize the allocation of candidates for the next iteration: as can be seen from Figure 2, a proper number of candidates are moved randomly from the groups with lower adaptation values to the groups with higher ones in proportion to the difference from the mean value $\widetilde{g}$. As a result, PM works as a meta-heuristics that enables implicit self-adaptive temperature scheduling or dynamic control of search ranges.

3.2 The Algorithm

Figure 3 shows an outline of the population migration algorithm, where the meta-heuristics PM is integrated with the stochastic search algorithm SHC. In the following we give supplementary explanations for the numbered statements in Figure 3, assuming GCP as the CSP to be solved.

(1) generate k groups, $S_1, \cdots, S_k$:

As defined in 2.2, let (V, C, E) an instance of GCP with $n = \mid V \mid$ and $m = \mid E \mid$. A candidate, s, is a complete set of assignments of randomly selected values in C for all variables. This statement generates k groups with different temperatures, to each of which the same number of random candidates are allotted.

(2) SHC(T_i):

T_i is the temperature assigned previously to group S_i, which is used by SHC in determining the sigmoidal probability function, shown in Figure 1, to enable stochastic moves to the next candidates. SHC is performed per candidate in S_i. Notice that the whole procedure of Figure 3 terminates whenever a final solution is found during the search of SHC.

(3) calculate the adaptation value $g(S_i)$:

Let $s = (c_1, \cdots, c_n)$ with $c_i \in C$ for $1 \leq i \leq n$ be a candidate. For each constraint $e = (v_p, v_q)$ in E, let $conf(e)$ be equal to 0 when $c_p \neq c_q$, indicating that e is satisfied, or 1 otherwise. Then the constraint satisfaction ratio f of s is given as

$$f(s) = 1 - \frac{\sum_{i=1}^{m} conf(e_i)}{m}. \tag{3.1}$$

Thus, the average ratio $\widetilde{f}$ for S_i is

$$\widetilde{f}(S_i) = \frac{\sum_{s \in S_i} f(s)}{\mid S_i \mid}, \qquad \text{for } 1 \leq i \leq k. \tag{3.2}$$

After the execution of SHC, the adaptation value for each group S_i is calculated as

$$g(S_i) = a \times \widetilde{f}(S_i) + b \times (\widetilde{f}(S_i) - \widetilde{f}_i), \qquad \text{for } 1 \leq i \leq k, \tag{3.3}$$

where a and b are non-negative constants and $\widetilde{f}_i$ is the average satisfaction ratio in the previous iteration of the outermost loop in Figure 3. The second term in (3.3) reflects how much the average ratio $\widetilde{f}$ of the i-th group has been improved by SHC in statement (2) in Figure 3.

(4) divide the groups into two classes, G_{high} and G_{low}:

Let $\widetilde{g}$ be the mean of all the adaptation values calculated as

$$\widetilde{g} = \frac{\sum_{i=1}^{k} g(S_i)}{k}. \tag{3.4}$$

By using $\widetilde{g}$, the groups $S_1, \cdots, S_k$ are classified into two classes as follows:

$$G_{high} = \{S \mid g(S) \geq \widetilde{g}\}, \tag{3.5}$$

$$G_{low} = \{S \mid g(S) < \widetilde{g}\}. \tag{3.6}$$

(5) migrate proper number of candidates :

Let μ be the sum of differences between $\widetilde{g}$ and $g(S)$ for $S \in G_{low}$:

$$\mu = \sum_{S \in G_{low}} (\widetilde{g} - g(S)) \quad \left(\equiv \sum_{S \in G_{high}} (g(S) - \widetilde{g}) \right). \tag{3.7}$$

Population migration is performed from groups in G_{low} to groups in G_{high}. The number of candidates to be removed from group S in G_{low} is determined as

$$\nu = \frac{\widetilde{g} - g(S)}{\mu} \times \mid S \mid, \tag{3.8}$$

except that at least one candidate must remain in S. Each removed candidate determined by (3.8) goes to one of the groups in G_{high}, say S, with probability

```
procedure population-migration(){
    generate k groups, S_1, ..., S_k ;                          (1)
    for ( j = 1 ; j ≤ max ; j := j + 1) {
        for ( i = 1; i ≤ k ; i := i + 1) {
            for (each candidate, s in S_i ) {
                SHC( T_i) ;                                     (2)
            };
            calculate the adaptation value g(S_i) ;             (3)
        };
        PM-meta-heuristics() ;
    };
}

procedure PM-meta-heuristics(){
    divide the groups into two classes, G_high and G_low ;      (4)
    for ( each S in G_low ){
        migrate proper number of candidates;                    (5)
    };
}
```

Figure 3: Population migration algorithm incorporating SHC as the basic local search algorithm.

$$\rho(S) = \frac{g(S) - \widetilde{g}}{\mu}, \tag{3.9}$$

which means that the higher adaptation value a group has, the more candidates the group tends to be allotted.

In the example of Figure 2, $G_{low} = \{S_1, S_2\}$, $G_{high} = \{S_3\}$, $\nu_1 = 6$, $\nu_2 = 3$ and $\rho(S_3) = 1.0$. As a result, population migration dynamically tries to keep an optimal allocation of the limited resources (or candidates) by recruiting promising groups.

4 Experiments

We evaluate the effectiveness of the population migration as a meta-heuristic from two major points of view: we compare its efficiency with SHC and SA, and we investigate in detail its behavior around the critical region where phase transitions may occur. Throughout the experiments, we use solvable GCPs (i.e., graphs colourable with 3 colours) which are generated randomly by using the procedure given in [1]. As to the calculation of the adaptation values of groups defined in equation(3.3), the ratio of coefficient b and a is set to 5 to boost rapid movement of population to the promising groups. All algorithms are implemented in the language C on an IBM Aptiva B75.

4.1 Comparison with SHC and SA

4.1.1 Comparison with SHC

SHC, a naive stochastic search method without heuristics, is adopted to evaluate the efficiency of PM. Fixing the number of nodes n and the number of edges m to 150 and 375 respectively, we generated 100 solvable GCPs. Thus the constraint density $d(= m/n)$ is equal to 2.5, around which it is known that GCPs tend to become hard to solve.

Table 1: Experimental results for SHC, parallel SHC, and the proposed method.

method	SHC with T =						parallel SHC	PM
	0.313	0.625	1.25	2.5	5	10		
%-solved	12	77	99	16	0	0	80	93
time	2.89	2.77	2.08	3.67	–	–	2.78	2.77
σ	1.38	1.10	0.78	1.23	–	–	1.24	1.13

%-solved : percentage of success(%)
time : mean solution time(min)
σ : standard deviation

In our implementation of the PM procedure of Figure 3 we prepared 5 groups with temperatures 10, 5, 2.5, 1.25 and 0.625 respectively and we allotted 20 random candidates are allotted to each group. The PM procedure is performed once per GCP. The maximum number of iterations of the outer loop, max in Figure 3, is set to 100. SHC (statement (2) in Figure 3) performs 100 hill-climbing operations (referred to as hc-steps hereafter) for each candidate as far as a final solution is not attained.

Simple SHC is performed for six different temperatures T: from 10 to 0.313 by halving the value. For each GCP, simple SHC is repeated 100 times with different starting candidates, where the upper limit of total hc-steps is fixed to 0.1×10^5 in each repetition, to make the amount of computation equal to that of PM.

We also tested a mixture of simple SHCs, called parallel SHC, in which 5 simple SHCs with different temperatures are performed in parallel practically under the same conditions as described above except that each simple SHC is repeated only 20 times at most.

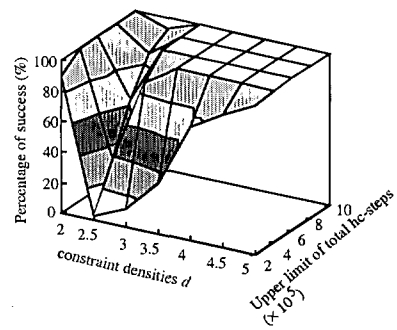
Table 1 summarizes the experimental results, where *%-solved* gives the percentage of solved GCPs, and *time* and σ shows the cpu time averaged over solved cases and its standard deviation.

In comparison with SHCs, PM is a robust method. In fact, simple SHCs do not complete successfully in most cases of temperature except a narrow range near $T = 1.25$, indicating that some mechanism (heuristic) of temperature control is necessary for SHC. Parallel SHC seems to give results comparable to PM from a computational time point of view. However, PM solves much more GCPs within the time limit. Thus, when compared to parallel SHC, PM solves difficult problems without increasing cpu time by help of the population migration.

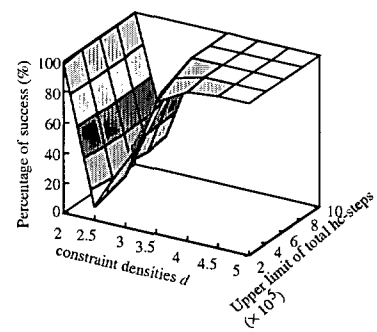
4.1.2 Comparison with SA

In order to compare PM with SA, we ran experimental simulations from two different viewpoints: the constraint density d and the number of variables n.

Let us clarify the PM and the SA used to solve GCPs. The PM is the same as the one used in 4.1.1 except that the



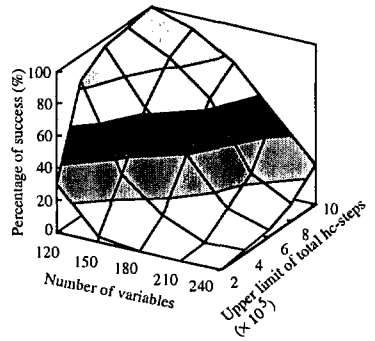
(a) The percentage of success for PM



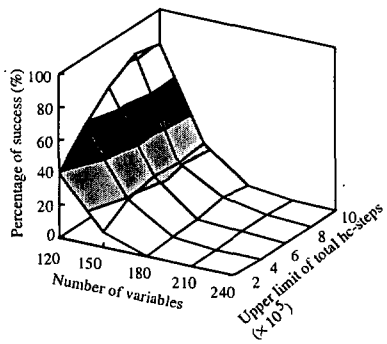
(b) The percentage of success for SA

Figure 4: Experimental results on the constraint densities d.

number max of iterations of the outer loop ranges from 20 to 100 at intervals of 20. Thus the available total computational cost ranges from 2×10^5 to 10×10^5 hc-steps since the hill-climbing effort in every iteration amounts to 0.1×10^5 hc-steps. The SA used in the experiments is based on the SA procedure in Figure 1, where we set $T\_max = 10.0$, $T\_min = 0.625$, $\gamma = 0.5$ and $hc\text{-}max$ in SHC equal to 1,000, in order to make the SA comparable to the PM above. As long as a final solution is not found the SA procedure is repeatedly restarted with a new initial candidate. In fact, we tested five different upper limits on the number of repetitions: 40 to 200 at intervals of 40, which corre-



(a) The percentage of success for PM



(b) The percentage of success for SA

Figure 5: Experimental results on the size of the search space.

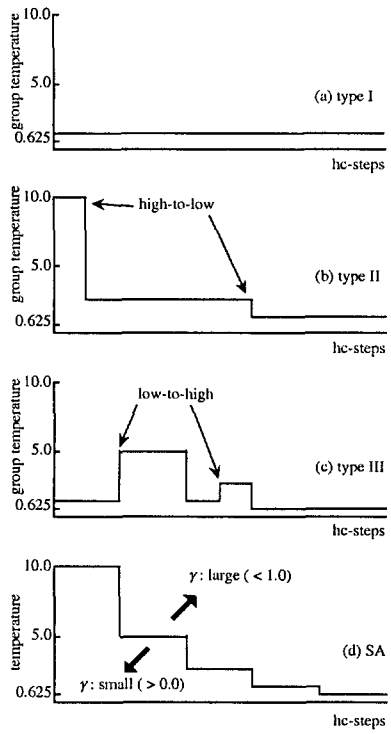


Figure 6: Migration patterns (a), (b) and (c), and SA temperature control (d).

spond to the 2×10^5 to 10×10^5 hc-steps of total computational costs described above.

In the first experiment, we tested seven different densities d varying from 2 to 5 at intervals of 0.5. For each density, we randomly generated 100 GCPs with a fixed number of variables, $n = 150$. Figure 4 gives the results. When the total computational cost limit is low, say less than 4×10^5, SA is slightly superior to PM. But, in the density range 2 to 3.5, where most GCPs are extraordinarily hard to solve, PM apparently gives superior results. In that range, SA fails to solve most of the GCPs and its performance is not improved even when the computational time is increased. In contrast, the percentage of success for PM remarkably increases as the available total hc-steps is increased.

In the second experiment we varied the number n of variables fixing the constraint density d to 2.5. We tested five cases of n: 120 to 240 at the intervals of 30. For each n, we generated 100 random GCPs. Figure 5 shows the results. We can clearly see that PM gives higher success ratios than SA everywhere except in the restricted case that both the problem size and the available computational cost are small. The size of the search space grows exponentially as n increases. As a result, the probability of success for SA declines rapidly and does not seem to improve even when the available computational cost is increased. In the case of PM, however, the percentage of success does decline slowly as n increases when the total computational cost is chosen to be large proportionally to the problem size.

To illustrate the wide applicability and reliability of PM as a meta-heuristics, we ran a set of supplementary experiments in which neural networks are used as the basic stochastic search technique to solve SAT problems. The results are shown in the Appendix.

4.2 Detailed Analysis of Temperature Control

4.2.1 Migration Patterns

We traced the behavior of all the candidates during the execution of the PM procedure. When the PM operation is performed in Figure 3, the population distribution of the groups is updated autonomously. Thus, for each candidate we get a transition pattern along which the candidate migrated among the five groups.

Observing these migration patterns, we found that they can be classified into three types as shown symbolically in Figure 6. Type I is the simplest where the candidate remains in its initial group. Type I corresponds to the simple SHC in which the initial temperature stays unchanged during the search. Type II is the pattern containing high-to-low migrations only: one or more migrations from groups with higher temperatures to groups with lower temperatures. Type II represents the temperature control similar to that of SA, whose typical pattern is given in Figure 6(d).

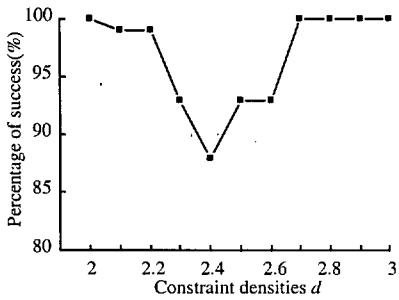


Figure 7: Experimental results from $d = 2$ to $d = 3$.

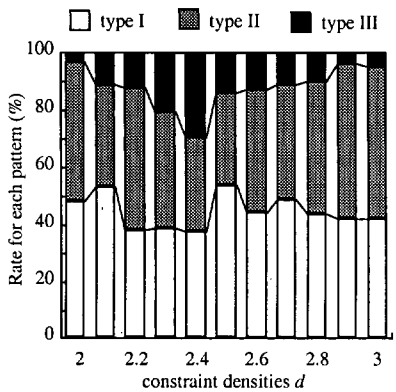


Figure 8: Distributions of temperature behavior in PM.

Various patterns with different decreasing speeds, which correspond to different values of γ in SA, are included in Type II. Therefore, PM is robust because it implicitly performs SA in various cases of γ in parallel. Type III is the pattern containing at least one low-to-high migration, which ensures that population migration enables dynamic self-adaptation control of temperature. In SA the temperature is controlled so that it monotonically decreases. Therefore, Type III is an especially interesting pattern specific to PM.

4.2.2 Discussions from the viewpoint of Phase Transition

The three types of migration patterns (Type I, II and III introduced in the previous section) are similar to the typical cases of temperature control realized by the three major search strategies that we are concerned with: simple SHC, SHC with SA, and SHC with PM, respectively.

To clarify how well these three major types affect the efficiency of the PM procedure, we ran further experiments. We fixed the number n of variables to 150 and varied the constraint density from 2 to 3 at small increments of 0.1. For each density, we tried to solve 100 solvable GCPs by the PM procedure under the same conditions as described in 4.1.1, with the maximum computational power limited to 10×10^5 hc-steps and 100 population migrations.

Experimental results are summarized in Figure 7 and Figure 8. Figure 7 shows the percentage of problems solved within 10×10^5 hc-steps. The percentage of success is low in the region around $d = 2.4$, where the phase transitions are expected to occur[15].

We traced back every candidate that led to a final solution and classified its migration pattern into the three types shown in Figure 6. Figure 8 shows the results. We see that the curve of Type III is quite similar to that of the success percentage in Figure 7. Actually, the percentage of Type III becomes the highest at $d = 2.4$, where hard problems are concentrated. Thus, it is expected that the Type III pattern, which is specific to the PM meta-heuristics, will be helpful to reduce the hardness of GCPs in the critical region.

5 Conclusions

We proposed a novel meta-heuristics named population migration (PM), which is applicable to stochastic search methods for constraint satisfaction problems including stochastic hill-climbing and neural networks.

It may be possible to view PM as a spatially parallel version of temperature control of SA in which temperature always decreases monotonically. The proposed meta-heuristics, however, enables a more sophisticated control of temperature since it implicitly conducts dynamic self-adaptive temperature control. Its effectiveness was verified by some experiments: (1) comparison between naive stochastic hill-climbing (SHC) and SHC assisted by PM, (2) comparison beween SA and PM applied to two basic methods: SHC and neural networks, (3) detailed investigation of the dynamic controllability of temperature from the viewpoint of computational complexity.

The last experiment is particularly interesting because efficiency of self-adaptive temperature control, which is specific to PM, is remarkable in the critical region where phase transitions occur.

Acknowledgment

This research was partially carried out while the second author was staying at the International Institute for Applied Systems Analysis (IIASA) in Laxenburg, Austria as a research scholar of RMP (Risk, Modeling and Policy) Project under the supervision of Dr. Marek Makowski.

References

[1] Minton, S., Johnston, M. D., Philips, A. B., and Laird, P. (1992) Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problem, *Artificial Intelligence*, Vol.58, pp.161–205.

[2] Ackley, D. H. (1987) *A Connectionist machine for Genetic Hillclimbing*, chapter 3, Kluwer Academic Publishers.

[3] Adorf, H. M. and Johnston, M. D. (1990) A Discrete Stochastic Neural Network Algorithm for Constraint Satisfaction Problems, *Proceedings of IJCNN'90*, pp. 917–924.

[4] Boyan, J. A. and Moore, A. W. (1998) Learning Evaluation Functions for Global Optimization and Boolean Satisfiability, *Proceedings of AAAI'98*, pp. 3–10.

[5] Davenport, A., Tsang, E., Wang, C. J., and Zhu, K. (1994) Genet, A Connectionist Architecture for Solving Constraint Satisfaction Problems by Iterative Improvement, *Proceedings of AAAI'94*, pp. 325–330.

[6] Frank, J. (1996) Weighting for Godot: Learning Heuristics for GSAT, *Proceedings of AAAI'96*, pp. 338–343.

[7] Frank, J., Cheeseman, P., and Stutz, J. (1997) When Gravity Fails: Local Search Topology, *Journal of Artificial Intelligence Research*, Vol.7, pp.249–281.

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

[9] Selman, B., Levesque, H., and Mitchell, D. (1992) A New Method for Solving Hard Satisfiability Problems, *Proceedings of AAAI'92*, pp. 440–446.

[10] Selman, B., Kautz, H., and Cohen, B. (1994) Noise Strategies for Improving Local Search, *Proceedings of AAAI'94*, pp. 337–343.

[11] Spears, W. M. (1996) A NN Algorithm for Boolean Satisfiability Problems, *Proceedings of ICNN'96*, pp. 1121–1126.

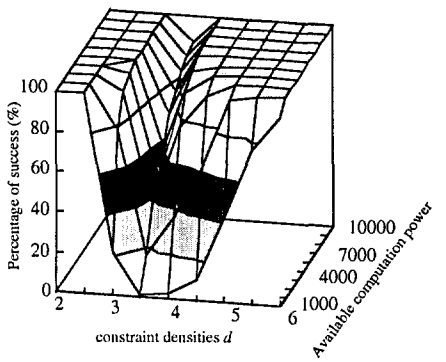
[12] Varanelli, J. M. and Cohoon, J. P. (1995) Population-Oriented Simulated Annealing: A Genetic / Thermodynamic Hybrid Approach to Optimization, *Proceedings of ICGA'95*, pp. 174–181.

[13] Wong, J. H. Y. and Leung, H. F. (1998) Extending genet to solve fuzzy constraint satisfaction problems, *Proceedings of AAAI'98*, pp. 380–385.

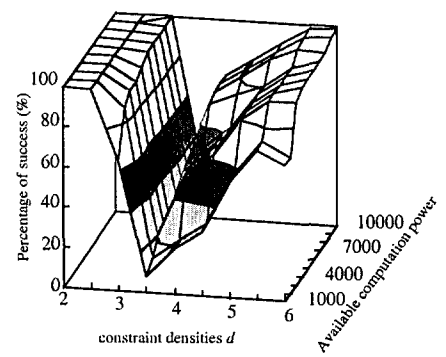
[14] Wah, B. W. and Wang, T. (1999) Simulated Annealing with Asymptotic Convergence for Nonlinear Constrained Global Optimization, *Proceedings of CP'99*, pp. 461–475.

[15] Hogg, T. and Williams, C. P. (1994) The hardest constraint problems: a double phase transition, *Artificial Intelligence*, Vol.69, pp.359–377.

[16] Hogg, T., Huberman, B. A., and Williams, C. P. (1996) Phase transition and search problem, *Artificial Intelligence*, Vol.81, pp.1–15.



(a) The percentage of success for PM



(b) The percentage of success for SA

Figure 9: Experimental results on the constraint densities d.

[17] Mammen, D. L. and Hogg, T. (1997) A New Look at the Easy-Hard-Easy Pattern of Combinatorial Search Difficulty, *Journal of Artificial Intelligence Research*, Vol.7, pp.47–66.

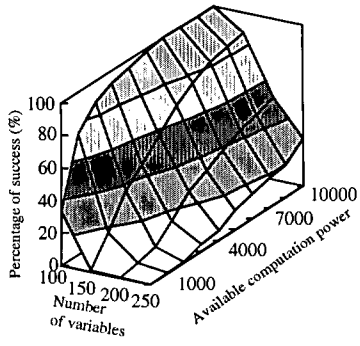
[18] Mitchell, D., Selman, B., and Levesque, H. (1992) Hard and Easy Distributions of SAT Problems, *Proceedings of AAAI'92*, pp. 459–465.

[19] Yokoo, M. (1997) Why Adding More Constraints Makes a Problem Easier for Hill-climbing Algorithms: Analyzing Landscape of CSPs, *Proceedings of the Third International Conference on Principles and Practice of Constraint Programming (CP'97)*, pp. 356–370.

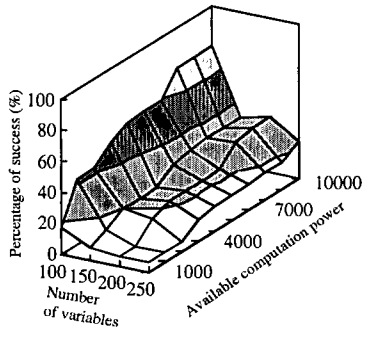
[20] Asahiro, Y., Iwama, K., and Miyano, E. (1996) Random Generation of Test Instances with Controlled Attributes, *DIMACS*, Vol. 26, pp. 377–393.

Appendix

In these supplementary experiments the basic stochastic search method to which the meta-heuristics SA and PM are applied is based on NN-SAT, the neural network proposed in [11] instead of SHC in Section 4. The standard CSP to



(a) The percentage of success for PM



(b) The percentage of success for SA

Figure 10: Experimental results on the size of the search space.

be tested is CNF 3-SAT, or 3-SAT shortly, instead of GCP in Section 4.

Let us clarify the parameters of PM and SA. The PM procedure is the one of Figure 3 except that instead of SHC, NN-SAT without temperature control is used as the basic stochastic search method. The number k of groups is set to five with temperatures fixed to 0.15, 0.08, 0.04, 0.02 and 0.01. To each group 8 candidates are alloted. The SA procedure we used is the same as NN-SAT[11], with the temperature T controlled like $T = T\_max \times \exp(-j/(restarts \times neurons))$ from $T\_max = 0.15$ to $T\_min = 0.01$. The parameters j, $restarts$ and $neurons$ indicate the j-th trial of search operation, the number of restarts with a new candidates, and the number of neurons, respectively. For a SAT problem, n is the number of variables in the propositional expression and d is the constraint density given by the number of disjunctive clauses divided by n. In each case 100 random 3-SAT problems were generated using the procedure in [20].

In the first experiment, fixing the number n of variables to 150, we tested nine cases of constraint density d: 2 to 6 at intervals of 0.5. Figure 9 shows the results. As was seen in Figure 4, PM again becomes apparently superior to SA as the available computational power increases.

In the second experiment, fixing $d = 3.5$, four values of n are tested: from 100 to 250 at intervals of 50. Figure 10 shows the results. We see the percentage of success of SA tends to decline more rapidly than that of PM as the problem size increases, which is similar to the result of Figure 5.

Let us notice finally that all three migration patterns shown in Figure 6 are also observed in these supplementary experiments.