論 文

Towards the Maintenance of Population Diversity: A Hybrid Probabilistic Model Building Genetic Network Programming

Xianneng Li — Graduate School of Information, Production and Systems, Waseda University
sennou@asagi.waseda.jp

Shingo Mabu — (affiliation as previous author)
mabu@aoni.waseda.jp

Kotaro Hirasawa — (affiliation as previous author)
hirasawa@waseda.jp

keywords: probabilistic model building evolutionary algorithm (PMBEA), estimation of distribution algorithm (EDA), genetic network programming (GNP), probabilistic model building genetic network programming (PMBGNP), diversity maintenance.

Summary

Some researchers have investigated that the diversity loss will significantly decrease the performance of Probabilistic Model Building Genetic Algorithm (PMBGA), especially under large search space, leading to the premature convergence and local optimum. However, few work has been done on the diversity maintenance in the Probabilistic Model Building Evolutionary Algorithms (PMBEAs) with more complex chromosome structures, such as tree structure based Probabilistic Model Building Genetic Programming (PMBGP) and graph structure based Probabilistic Model Building Genetic Network Programming (PMBGNP). For the PMBEAs with more complex chromosome structures, the required sample size is usually much larger than that of binary structure based PMBGA. Therefore, these algorithms usually become much more sensitive to the population diversity. In order to obtain enough population diversity, the large population size is needed, which is not the best way. In this paper, the maintenance of the population diversity is studied in PMBGNP, which is a kind of PMBEA, but has its unique characteristics because of its directed graph structure.

This paper proposed a hybrid PMBGNP algorithm to maintain the population diversity to avoid the premature convergence and local optimum, and presented a theoretical analysis of the diversity loss in PMBGA, PMBGP and PMBGNP. Two techniques have been proposed for the diversity maintenance when the population size is set at not large values, which are multiple probability vectors and genetic operators. The proposed algorithm is applied and evaluated in a kind of autonomous robot, Khepera robot. The simulation study demonstrates that the proposed hybrid PMBGNP is often able to achieve a better performance than the conventional algorithms.

1. Introduction

Probabilistic Model Building Evolutionary Algorithm (PMBEA) is a new class of evolutionary algorithms, where a probabilistic model is built based on the better individuals of the current population and used to generate the new population. There is a wide range of such work in Genetic Algorithm (GA), which is usually called PMBGA [Baluja 94, Mühlenbein 96, Harik 98, Larrañaga 02, Pelikan 02a], and work in Genetic Programming (GP), denoted as PMBGP [Salustowicz 97, Sastry 03, Yanai 03, Shan 06, Hasegawa 08]. However, very few work has been done on extending it to the graph-based evolutionary algorithms. In the previous research by the authors', a graph-based PMBEA named Probabilistic Model Building Genetic Network Programming (PMBGNP) [Li 09, Li 10a, Li 10b] has been proposed, where the directed graph based structure of Genetic Network Programming [Katagiri 00, Hirasawa 01, Eguchi 06, Mabu 07] is used to represent its chromosome.

PMBGNP could inherit the advantages of GNP which can deal with dynamic environments effectively and efficiently due to the directed graph based network structures, and the characteristics of PMBEA because of the probabilistic modeling of promising solutions. PMBGNP allows us to take into account the dependencies between the internal network structures of GNP to prevent the breakage of building blocks, therefore shows more suitability

for the decomposable problems.

Most of the current PMBEAs suffer from the problem that the diversity of the genetic information will be significantly decreased in the generated population when the population size is not large enough, leading to the local convergence. The diversity loss in PMBGA has been proven in [Shapiro 06]. Therefore, when PMBGA is used to solve problems, its population size should be set at an enough size in order to ensure the enough diversity for the global optimum [Pelikan 02b]. Some studies also investigated that applying mutation operator [Handa 07] or niching operator [Sastry 05] to PMBGA could maintain the population diversity.

On the other hand, for the PMBEAs with more complex structure representations, the problem of the diversity loss is much more essential than that of PMBGA, since the required sample size in such PMBEAs is usually much larger than that of PMBGA (see section 3). On the other hand, in order to solve the problems consisting of large search space, the required population size should be set at huge values to ensure the enough diversity. Therefore, the study on the diversity maintenance becomes much essential in the PMBEAs with more complex structure representations.

In the research on PMBGP, Probabilistic Incremental Program Evolution (PIPE) uses a mutation operator to explore the search space [Salustowicz 97], while Estimation of Distribution Programming (EDP) adjusts the calculated probabilistic model by Laplace Correlation to avoid the premature convergence [Yanai 03]. However, most of these algorithms are testified in GP's benchmark problems with not so large search space, such as symbolic regression problems and boolean function problems, and the importance of the diversity maintenance is not clarified clearly in PMBGP. As a PMBEA with graph-based structure representations, PMBGNP holds the same problem of the diversity loss as PMBGP, and even more serious. The reason is that, the graph structure and population size of GNP are usually set at small values to solve problems, which is one of the advantages of GNP [Mabu 07], therefore the sample space becomes much smaller than that of PMBGP, which probably causes the diversity loss.

This paper focuses on studying the diversity maintenance of PMBGNP to make it work in the problems with larger search space than that of classical benchmark problems. This paper presents a short survey of the current study of PMBEAs in terms of different kinds of chromosome representations, and theoretically analyzes the diversity loss of different types of PMBEAs. Also, this paper proposes two techniques to maintain the population diversity of PMBGNP in terms of improving the exploration ability, which are multiple probability vectors and genetic operators. The proposed algorithm is denoted as hybrid PMBGNP and is evaluated in a real world problem, controlling a kind of autonomous robot, Khepera robot [Michel 96, Cyberbotics]. The simulation study shows the proposed algorithm could significantly maintain the diversity of PMBGNP to solve the problems.

2. Literature review

2·1 PMBGA

The idea of probabilistic modeling building evolutionary algorithm (PMBEA) was first introduced in the field of binary GA in [Baluja 94], and has attracted much attention in the last decade. It consists of three types of algorithms: no interactions, pairwise interactions and multivariate interactions [Pelikan 02b], to study the building blocks of different complexity. The class of GA based PMBEA is usually called as PMBGA [Mühlenbein 96, Harik 98, Larrañaga 02, Pelikan 02a]. PMBGA identifies and recombines important building blocks through estimating the distribution of promising individuals. Algorithm 1 shows the basic pseudo-code of PMBGA, which is the same as most PMBEAs.

PMBGA studies the GA's binary string structure by calcuating the frequencies of 0's or 1's in each gene loci. Generally, marginal probability is used to represent the probablistic model of PMBGA without interactions, while the joint probability is used to represent the probabilistic model of PMBGA with interactions. Numerous algorithms have been proposed to draw the success of this topic, in both theory and application.

Algorithm 1 Algorithm of PMBEAs

1: $t \leftarrow 0$
randomly generate an initial population $S(t)$
2: evaluate the fitness of the initial population
3: execute selection operator to select a set of promising individuals $B(t)$
4: construct a probabilistic model P^t from $B(t)$
5: generate a new population $S(t+1)$ using the probabilistic model P^t
6: evaluate the fitness of the new population $S(t+1)$
set $t \leftarrow t+1$
7: if the termination criteria are not met, go to 3

2·2 PMBGP

Some research has introduced PMBEA into GP, which uses tree structures to represent its chromosome. The first

algorithm of PMBGP is PIPE [Salustowicz 97], where a prototype tree is constructed and maintained to evolve GP individuals. PIPE corresponds to the PMBGA without interactions, such as PBIL [Baluja 94] and UMDA [Mühlenbein 96]. EDP was proposed to extend PMBGP to pairwise interactions, where Bayesian network is used to identify the second order building blocks [Yanai 03]. eCGP [Sastry 03] and POLE [Hasegawa 08] extend PMBGP to multivariate interactions, therefore more complex building blocks in GP could be identified and recombined.

Since PMBGP uses tree structures to represent its chromosome, its probabilistic model is generally constructed by calculating the frequencies of functions in each node, which is different from that of PMBGA.

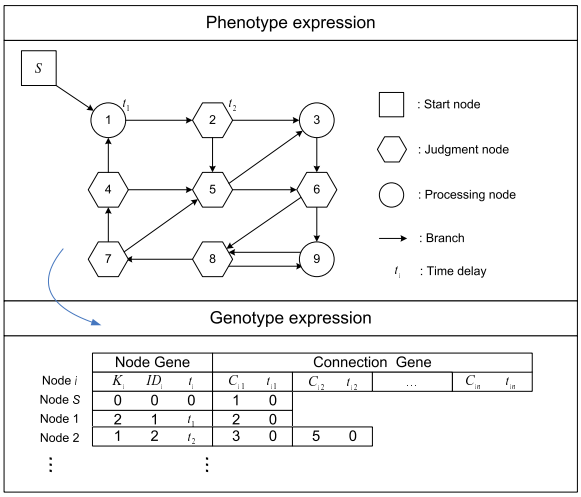

Fig. 1 The directed graph structure.

2·3 PMBGNP

In the previous research, the authors have first extended the idea of PMBEA to a graph based evolutionary algorithm - Genetic Network Programming (GNP) - and called the proposed algorithm as PMBGNP [Li 09]. GNP [Katagiri 00, Hirasawa 01, Eguchi 06, Mabu 07] is an extension of GA [Holland 75, Goldberg 89] and GP [Koza 92, Koza 94] in terms of using network structures to represent its chromosome. Multiple nodes and their branches are used to construct the directed graph structures to improve the search efficiency and expression ability. Many studies have investigated that GNP can outperform the conventional algorithms to solve the complex problems in dynamic environments, such as Multi-agent Systems [Murata 04, Eguchi 06, Mabu 07], Data Mining [Shimada 06], Elevator System Control [Hirasawa 08] and Financial Engineering [Chen 09], etc. PMBGNP is evaluated by applying to a real world based data mining application and the comparative analysis shows its superiority comparing with the conventional GNP [Li 10a, Li 10b].

Although there is few work on studying the graph structure representation based PMBEA, the previous research has shown the superiority of graph based evolutionary algorithms for some problems in terms of stronger expression and evolution ability than that of conventional GP [Teller 95, Miller 00, Hirasawa 01]. On the other hand, GNP is mainly designed for solving some complex problems in dynamic environments, where the required search space is generally large. Therefore, PMBGNP could be considered having two characteristics, that is, the extension of PMBEA to graph structure based evolutionary algorithms and applicability of PMBEA to dynamic environments rather than static environments.

§ 1 Directed graph structure

PMBGNP uses the directed graph structure of GNP to represent its chromosome, which can be illustrated by the phenotype and genotype expression. Phenotype shows the directed graph structure, while genotype demonstrates the encoding of GNP. As shown in Figure 1, let *i* represent a node number of GNP. K_i defines the type of node *i* such as start node, judgment node and processing node. ID_i identifies the node function, such as judgment function and processing function. C_{in} denotes the node which is directly connected from the n_{th} branch of node i. t_i and t_{in} are the delay time, which are the time required to execute node i and time to transit from node i to node C_{in}, respectively.

Generally, one start node, a fixed number of judgment nodes and processing nodes composes the structure of GNP. The start node is only used to decide the first node to be transited, while the judgment and processing nodes save some functions corresponding to the concrete problem. For agent control, each judgment node works as ”if-then” type decision making functions to judge the environment to make a decision, while processing nodes preserve the action functions to determine the agent’s action. In this structure, each judgment node consists of multiple branches connecting to different nodes, where the next node to be transited is determined by the judging result of the environment. Processing node only has one branch, since the processing functions only determine the agent’s actions. Therefore, the agent will be controlled by transiting the nodes until the task is solved. Generally, when solving problems, the number of judgment and processing nodes, the number of branches in judgment nodes and the time delays are predefined.

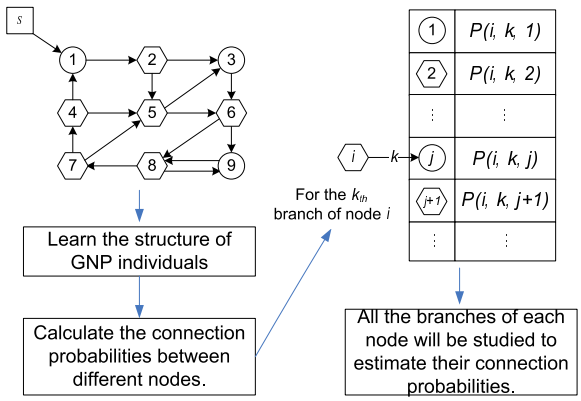

Fig. 2 The probabilistic model construction in PMBGNP.

§2 Probabilistic model construction

The flowchart of PMBGNP is the same as most PMBEAs as shown in Algorithm 1. The model construction of PMBGNP is inspired by PMBEA with no interactions. However, due to the graph structure of GNP, the connection probabilities between different nodes are considered to construct the probabilistic model in PMBGNP. Therefore, it could be classified as a kind of PMBEAs with pairwise interactions as shown in Figure 2. In the t_{th} generation, P^t denotes the probabilistic model, and $P^t(i,k,j)$ represents the connection probability from the k_{th} branch of node i to node j.

These connection probabilities are calculated by considering the connection information and transition information between different nodes. The reason why we consider these two factors is as follows. In GNP, the directed graph structure is based on the connections of different nodes by branches. Therefore, the connection information is the most important information which deserves the calculation of probabilities. On the other hand, in GNP, usually all the nodes are not used in the transition to solve the problems in one individual. The node transition is made by selecting the necessary nodes. In an example shown in Figure 3, the nodes are transited like $1 \to 2 \to 5 \to 6 \to 9$ to solve the problems, which means the information on the connections of the transited nodes is useful for solving the problems.

Therefore, the connection probabilities of the t_{th} generation is calculated as follows.

$$P^t(i,k,j) = \frac{\sum_{n \in N} [\delta_n^t(i,k,j) + \eta \sigma_n^t(i,k,j)]}{\sum_{\widehat{j} \in A(i,k)} \sum_{n \in N} [\delta_n^t(i,k,\widehat{j}) + \eta \sigma_n^t(i,k,\widehat{j})]}, \quad (1)$$

where,

N: set of suffixes of promising individuals.

$A(i,k)$: set of suffixes of connected nodes from the k_{th} branch of node i.

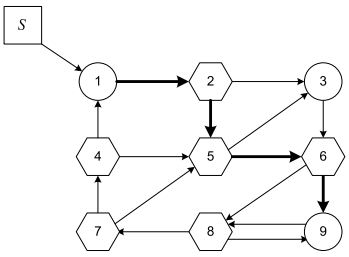

Fig. 3 An example of node transition.

$\delta_n^t(i,k,j)$: value defined by

$$\delta_n^t(i,k,j) = \begin{cases} 1 & \text{if the } k_{th} \text{ branch of node } i \text{ of individual } n \text{ is connected to node } j \text{ in } t_{th} \text{ generation,} \\ 0 & \text{otherwise.} \end{cases}$$

$\sigma_n^t(i,k,j)$: value defined by

$\sigma_n^t(i,k,j) = \ell$ if the transaction from the k_{th} branch of node i to node j of individual n occurs ℓ times in the t_{th} generation.

η: coefficient.

Besides, the following exponential smoothing method is considered to update the current probabilistic model considering the previous generation's connection probabilities.

$$P(i,k,j) \leftarrow (1-\alpha)P(i,k,j) + \alpha P^t(i,k,j). \quad (2)$$

Here, coefficient $\alpha \in (0,1)$ can be considered as a smoothing rate.

When sampling the model, the probability distribution $P(ind)$ of an individual is given by

$$P(ind) = \prod_{i \in N_{GNP}} \prod_{k \in A(i)} P[i,k,ind_{(i,k)}], \quad (3)$$

where,

N_{GNP}: set of suffixes of nodes in GNP individual.

$A(i)$: set of suffixes of branches of node i.

$ind_{(i,k)}$: the node which is connected from the k_{th} branch of node i in individual ind.

$P[i,k,ind_{(i,k)}]$: probability that the k_{th} branch of node i is connected to node $ind_{(i,k)}$.

3. Diversity loss

The diversity loss of PMBGA has been clarified by Shapiro in [Shapiro 06] using the trace of empirical co-variance matrix. However, there is few work on analyzing this issue in PMBGP. This section discuss the significance of the diversity loss in PMBGP and PMBGNP, and the authors argue that the diversity loss in PMBGP and PMBGNP is more essential than that of PMBGA with binary structures,

since the search space in each chromosome of GP or GNP is larger than that of GA.

Suppose that the set of suffixes of genes in each PMBGA individual is N_{GA}, while the set of suffixes of nodes in each PMBGP individual is also N_{GP}*1. In GNP, the number of nodes and the number of branches are generally predefined and fixed. Suppose that the set of suffixes of nodes in each PMBGNP individual is N_{GNP} and the set of suffixes of branches in each PMBGNP individual is N_{bra}. The size of the search space of these three algorithms can be calculated by

$$\Psi(GA) = 2^{|N_{GA}|}, \tag{4}$$

$$\Psi(GP) = \phi^{|N_{GP}|}, \tag{5}$$

$$\Psi(GNP) = (|N_{GNP}| - 1)^{|N_{bra}|}. \tag{6}$$

Here, ϕ represents the number of functions in each node of GP*2. There is no self-loop in GNP network structure, since it will cause the infinite loops of GNP programs. Therefore, for each branch of the node in GNP, the candidate nodes to be connected are all the nodes except itself, where $|N_{GNP}| - 1$ in Eq. (6) represents the number of candidate nodes to be connected. If variables in the search space are treated equally, the probability for each individual to be sampled can be calculated by

$$P_{GA}(ind) = \frac{1}{2^{|N_{GA}|}}, \tag{7}$$

$$P_{GP}(ind) = \frac{1}{\phi^{|N_{GP}|}}, \tag{8}$$

$$P_{GNP}(ind) = \frac{1}{(|N_{GNP}| - 1)^{|N_{bra}|}}. \tag{9}$$

The diversity loss D_{GA}, D_{GP} and D_{GNP} of PMBGA, PMBGP and PMBGNP could be calculated, respectively as follows.

[Theorem 1] If the number of individuals in a population is N, the diversity loss D_{GA} of PMBGA could be represented by the probability that an individual is not sampled in the search space, which is

$$D_{GA} = [1 - P_{GA}(ind)]^N. \tag{10}$$

《**Proof**》 When generating one individual, the probability that individual ind is not sampled to the population equals to $1 - P_{GA}(ind)$. Therefore, when generating N individuals, the probability that individual ind is not sampled to the population equals to $[1 - P_{GA}(ind)]^N$. Then, the diversity loss of an individual in GA can be obtained by Eq. (10). □

∗1 Generally, in order to avoid the bloating of GP, the structure of GP is designed with some constraints, such as the maximum number of nodes or the maximum tree depth. Here, the maximum number of nodes is selected as the constraint.

∗2 Although GP consists of function nodes and terminal nodes, this paper treats them equally.

[Theorem 2] If the number of individuals in a population is N, the diversity loss D_{GP} of PMBGP could be represented by the probability that an individual is not sampled in the search space, which is

$$D_{GP} = [1 - P_{GP}(ind)]^N. \tag{11}$$

[Theorem 3] If the number of individuals in a population is N, the diversity loss D_{GNP} of PMBGNP could be represented by the probability that an individual is not sampled in the search space, which is

$$D_{GNP} = [1 - P_{GNP}(ind)]^N. \tag{12}$$

The proof of **Theorem 2** and **Theorem 3** is similar to **Theorem 1**, therefore it would not be shown in this paper.

From **Theorem 1**, **Theorem 2** and **Theorem 3**, we could find that, the smaller $P(ind) \in \{P_{GA}(ind), P_{GP}(ind), P_{GNP}(ind)\}$ is, the more serious the diversity loss is. Moreover, there is an inverse relationship between $P(ind)$ and the size of the search space $\Psi \in \{\Psi(GA), \Psi(GP), \Psi(GNP)\}$. Therefore, when the search space becomes larger, the diversity loss is more serious. From Eq. (4)-(6), we can find generally the search space of GP and GNP is much larger than that of GA, since ϕ and $|N_{GNP}|$ is much larger than 2.

On the other hand, we could find when the population size N is small, the diversity loss $D \in \{D_{GA}, D_{GP}, D_{GNP}\}$ will approach near to 1, which means the diversity loss is very serious. Therefore, in the previous research of PMBEAs, the population size N is generally set at large values to obtain the enough population diversity. However, when solving the problems consisting of very large search space, the required population size should be set at huge values, which is almost impossible. GNP is generally designed to solve the complex problems in dynamic environments. Therefore, maintaining the population diversity is much essential in PMBGNP.

4. The proposed method

This section focuses on proposing an extension of PMBGNP to maintain its population diversity and to escape from the local optimum of final solutions. Two novel techniques, which are multiple probability vectors and genetic operators, have been proposed to maintain the population diversity of PMBGNP in terms of improving the exploration ability. Since the proposed algorithm is inspired by genetic operators of the conventional evolutionary algorithms, the proposed algorithm is named hybrid PMBGNP (hPMBGNP).

The motivation that makes hPMBGNP innovative is that: firstly, it evolves multiple populations by constructing multiple probability vectors. Secondly, it also applies genetic operators like crossover and mutation to the multiple probability vectors, where the exploration of the search space and population diversity could be improved. One should note that the proposed crossover and mutation are applied to change the probability vectors, not the individual structure which is used in conventional evolutionary algorithms.

In hPMBGNP, there are several populations, i.e., $|R|$. Each population consists of a number of individuals, i.e., M. All the individuals will be initialized by random and evaluated by a predefined fitness function. With their fitness values, the set of promising individuals would be selected. For each population, hPMBGNP constructs a probabilistic model as described in section 2. The probabilistic models are represented by connection probability vectors, therefore, hPMBGNP consists of $|R|$ probability vectors, whose r_{th} one is denoted as P_r $(r \in R)$. Genetic operators such as crossover and mutation are applied to probability vectors P_r to produce new probability vectors $P_r^{'}$.

The new $|R|$ populations will be generated by sampling the probability vectors $P_r^{'}$. In conventional GNP, crossover and mutation are directly used to generate the new population, as a result, the strongly related sub-structures of GNP sometimes will be broken down to produce uninteresting individuals, while the probability model is carried out by learning the structure of promising individuals to guide the evolution of hPMBGNP. Therefore, hPMBGNP inherits the characteristics of PMBEAs that the building blocks could be recognized and represented implicitly in the probabilistic model, then the generated population becomes capable of avoiding the breakage of building blocks. On the other hand, crossover and mutation are applied to the constructed model, which means maintaining the population diversity leads to that PMBGNP can handle the problems consisting of the large search space. The details of the probabilistic model construction and genetic operators in hPMBGNP will be introduced next.

4·1 Probabilistic model construction

The probabilistic model of population $r \in R$ is represented as probability vector P_r. P_r^t denotes the probability vector P_r in the t_{th} generation, and $P_r^t(i,k,j)$ represents the connection probability from the k_{th} branch of node i to node j. The mathematical formulas to calculate the probability vectors in hPMBGNP are the same as the ones used in PMBGNP in section 2. The different point is that in hPMBGNP, the probabilistic model consists of multiple probability vectors, while PMBGNP consists of a single probability vector. Therefore, the probabilistic model could be denoted as follows.

$$P = \{P_r | r \in R\}.$$
$$P_r = \{P_r(i,k,j) | i \in N_{GNP}; k \in A(i); j \in A(i,k)\}.$$

4·2 Genetic operators

Crossover and mutation are designed to produce the new probabilistic model $P^{'}$. The role of genetic operators of the probabilistic model is to explore the probability vectors. In each generation, the constructed multiple probability vector P_r is replaced with the new one generated by crossover and mutation. Tournament selection is used in hPMBGNP to select probability vectors for crossover and mutation. Crossover and mutation operators are carried out subject to the following condition.

$$\sum_{j \in A(i,k)} P_r(i,k,j) = 1 \tag{13}$$

for all $i \in N_{GNP}$ and all $k \in A(i)$.

§1 Crossover

Crossover is executed between two probability vectors and produces two new probability vectors. Crossover operator exchanges all the probabilities of the selected branches as shown in Algorithm 2. Figure 4 shows an example on how crossover works in hPMBGNP.

Algorithm 2 Crossover of hPMBGNP

1: $m, n \in R$
Select two probability vectors P_m and P_n from P.
2: Each branch (i,k) is selected as a crossover branch with the probability of p_c.
3: Two probability vectors exchange the probabilities of the corresponding crossover branches, i.e., $P_m(i,k,j)$ and $P_n(i,k,j)$ are exchanged.

§2 Mutation

Mutation is executed in one probability vector to produce a new one. The probabilities of the selected mutation branches are changed randomly by mutation operator, where it should satisfy the condition in Eq. (13). The mutation in hPMBGNP is designed as shown in Algorithm 3.

4·3 Diversity maintenance of hPMBGNP

In conventional PMBEAs, once the probability[*3] in the probabilistic model is equal to zero, the corresponding

*3 In PMBGA with no interactions, the probability is represented by $P_{gene}(0)$ or $P_{gene}(1)$, and that of PMBGP is represented by $P_{node}(function)$. In PMBGNP, $P(i,k,j)$ represents the probability.

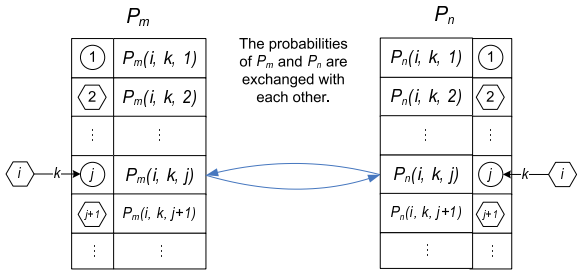

Fig. 4 Two probability vectors P_m and P_n are selected by tournament selection. In this example, branch (i,k) is selected as a crossover branch and their probabilities of P_m and P_n are exchanged with each other to generate the new probability vectors P'_m and P'_n.

Algorithm 3 Mutation of hPMBGNP

1: Select one probability vector P_r from P.
2: Each branch (i,k) is selected as a mutation branch with the probability of p_m.
3: For each node $j \in A(i,k)$, randomly generate a positive value $m(j)$.
4: Generate new probability vector $P_r^{'}$ using the following Equation.

$$P_r^{'}(i,k,j) = \frac{m(j)}{\sum_{\widehat{j} \in A(i,k)} m(\widehat{j})}$$

variable will never be sampled in the future generations. This subsection discuss the diversity maintenance of the proposed algorithm comparing with the standard PMBGNP.

Although crossover cannot preserve the population diversity, but it could explore the search space in another way. For example, even though a probability in probability vector P_r is equal to zero, it could be changed to a positive value by exchanging with another probability vector by crossover, which could explore the search space to avoid the local optimum.

Here, we discuss the diversity maintenance by the mutation in the proposed algorithm.

[Definition 1] $o(z)$ represents the number of probabilities equal to zero in branch z.

[Definition 2] S is the set of suffixes of branches that are not selected as mutation branch. It is very easy to know that $|S| = |N_{bra}|(1-p_m)$.

[Theorem 4] For the r_{th} population, the diversity maintenance rate is defined by Eq. (14) and Eq. (15).

$$M(r) = \frac{DM(r)}{(|N_{GNP}|-1)^{|N_{bra}|}}, \tag{14}$$

where,

$$DM(r) = (|N_{GNP}|-1)^{|N_{bra}|p_m} \prod_{z \in S} [|N_{GNP}| - 1 - o(z)] - \prod_{z \in N_{bra}} [|N_{GNP}| - 1 - o(z)]. \tag{15}$$

《**Proof**》 As described previously, when the probabilities in the probabilistic model are equal to zero, the corresponding variables will never be sampled in the future generations. For branch z, the number of probabilities that is not equal to zero is $[|N_{GNP}| - 1 - o(z)]$. Therefore, the size of the search space of the standard PMBGNP is

$$\prod_{z \in N_{bra}} [|N_{GNP}| - 1 - o(z)]. \tag{16}$$

When mutation is applied to the proposed algorithm, $|N_{bra}|p_m$ branches will be selected as mutation branches. For all mutation branches, their probabilities are always positive values by Algorithm 3. On the other hand, for the branches in set S, their probabilities are still possible to be zero like standard PMBGNP. Therefore, the search space could be calculated by

$$(|N_{GNP}|-1)^{|N_{bra}|p_m} \prod_{z \in S} [|N_{GNP}| - 1 - o(z)]. \tag{17}$$

It is easy to know the size of the search space is increased when mutation is done by comparing Eq. (16) and Eq. (17), and $DM(r)$ denotes the increased size of the search space as shown in Eq. (15). The total search space of PMBGNP can expressed by Eq. (6), therefore the diversity maintenance rate can be calculated by Eq. (14). □

It is easy to analyze that $DM(r)$ of Eq. (15) is larger than zero, which means the mutation could maintain the population diversity of PMBGNP. Moreover, when the value of $o(z)$ becomes large, $DM(r)$ will also become large when mutation rate is high, therefore the mutation could ensure better diversity maintenance when the population diversity is significantly lost.

5. Simulations

The proposed algorithm is evaluated by controlling the movement of Khepera robot and a comparative study among standard GNP, PMBGNP and hPMBGNP is carried out in this section.

5·1 Settings of the simulations

§1 Settings of the robot

Khepera robot is a small (5.5cm) differential wheeled mobile robot, which includes 8 infrared sensors allowing it to detect the proximity of objects in front of it, behind it, and to the right and left sides of it by reflexion. Each sensor returns a value ranging from 0 to 1023*4. Two motors

*4 0 means that no object is perceived, while 1023 means that an object is very close to the sensor, almost touching the sensor.

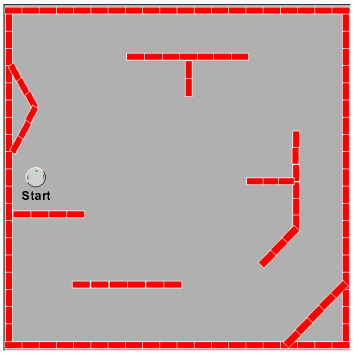

Fig. 5 Simulation environment.

corresponding to the left wheel and right wheel can take speed values ranging from -10 to +10. The robot movements are controlled by the speed of the two wheels.

§2 Wall following problem

The proposed algorithm is evaluated by solving the wall following problem of Khepera Robot [Mabu 06]. Figure 5 shows the environment used in this simulation. The size of the simulated environment is 1 m × 1 m, where there are obstacles (walls) in it. The task ends when the time step reaches predefined steps (1000 in this paper). The wall following behavior of the robot generated by GNP is evolved by evolution. Reward and fitness are designed based on [Nordin 98]. The aim of the fitness evaluation is to evolve the robot for moving along the wall as fast as and as straight as possible. Therefore, the reward and fitness is calculated as the following.

$$Reward = \frac{v_R + v_L}{20}(1 - \sqrt{\frac{|v_R - v_L|}{20}})C, \tag{18}$$

$$Fitness = \frac{(\sum_{step=1}^{1000} Reward)}{1000}, \tag{19}$$

where,

v_R, v_L: the speed of right and left wheels,

$$C = \begin{cases} 1 & \text{all the sensor values are less than 1000, and at least one of them is more than 100,} \\ 0 & \text{otherwise.} \end{cases}$$

§3 Settings of network structures

The node functions used for the Khepera robot are shown in Table 1. Each judgment node simulates the corresponding infrared sensor of the Khepera robot, and returns a value probing the position of the robot. In this paper, the number of branches of judgment nodes is set at 2, which means each judgment node returns a value ranging from 0 to 1023, and comparing the returned value with its threshold value (1000 is used in this paper) the judgment node determines which branch should be selected, as a result, which node to visit next. Each processing node determines the speed of the left or right wheel. The time delay of judgment nodes is set at 1 time unit, that of node transition is set at 0 time unit and that of processing node is set at 5 time units. The robot will take one step of action when 10 or more time units are used. For example, after executing four judgments and one processing, if another one processing is executed, the total time units become 14, which lead to the end of one time step. On the other hand, the simulation ends when the end condition is satisfied, that is, the time step exceeds 1000, which means the task will end when the robot moves 1000 time steps.

§4 Parameter settings

The parameter settings are shown in Table 2. In order to study the proposed algorithm, the number of populations $|R|$ is set at six, not single one as the standard PMBGNP, and the total number of individuals is set at a small value to evaluate the motivation of this work. The crossover and mutation probabilities in GNP and hPMBGNP are set appropriately through the simulations, i.e., $p_c = 0.1$ and $p_m = 0.01$ are used for both GNP and hPMBGNP in this paper. Each probability vector is constructed for its corresponding population, therefore total six probability vectors exist in hPMBGNP. Two of them are evolved by crossover and the rest four are evolved by mutation.

The number of judgment nodes is set at 40, which means each judgment function has 5 corresponding judgment nodes. Meanwhile, 20 processing nodes exist in an individual, which means each processing function has 2 corresponding processing nodes. The total number of branches $|N_{bra}|$ in an individual is $|N_{bra}| = 40 \times 2 + 20 = 100$, therefore the size of the search space of PMBGNP in the simulations is calculated by

$$\Psi(GNP) = (|N_{GNP}| - 1)^{|N_{bra}|} = 59^{100}. \tag{20}$$

In each generation, GNP directly preserves the best individual to the next generation, and the remaining individuals are generated by crossover and mutation (120 by crossover and 179 by mutation). For PMBGNP, 299 individuals are generated by sampling the probabilistic model, and combined with the elite individual to form the next new population. In the proposed algorithm, the process of evolving each population is the same as PMBGNP and finally 294 new individuals and 6 elite individuals corresponding to 6 populations are combined to form the new population.

5·2 Simulation results

§1 Simulation 1

In simulation 1, the start position of the robot is fixed at the left side of the environment every generation, as shown

Table 1 Node functions used for Khepera robot.

Node	Function
$J_1, J_2, ..., J_8$	Judge the value of the sensor 1, 2, ..., 8
$P_1(-10), P_1(-5), P_1(0), P_1(5), P_1(10)$	Determine the speed of the right wheel to -10, -5, 0, 5 or 10
$P_2(-10), P_2(-5), P_2(0), P_2(5), P_2(10)$	Determine the speed of the left wheel to -10, -5, 0, 5 or 10

Table 2 Parameter settings.

Parameter	GNP	PMBGNP	hPMBGNP
Number of populations $\|R\|$	1	1	6
Number of individuals per population M	–	–	50
Total number of individuals N	300	300	300
– Crossover	120	–	–
– Mutation	179	–	–
– Elite	1	1	6
Number of probability vectors	–	1	6
– Crossover	–	–	2
– Mutation	–	–	4
Number of promising individuals	–	150	25
Crossover probability p_c	0.1	–	0.1
Mutation probability p_m	0.01	–	0.01
Number of judgment nodes	40 (5 for each judgment function)		
Number of processing nodes	20 (2 for each processing function)		
η, α	–	0.01, 0.1	

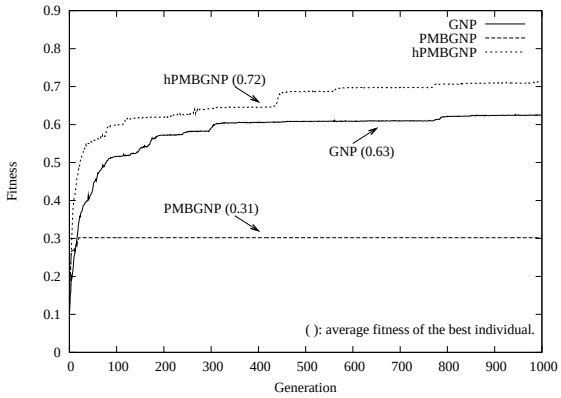

Fig. 6 Simulation results of GNP, PMBGNP and hPMBGNP in Simulation 1.

in Figure 5. Figure 6 shows the average fitness curves of the best individuals in each generation over 30 independent simulations.

In an early generations, PMBGNP and hPMBGNP show better fitness than GNP because the probabilistic models have higher evolution ability to find better solutions. However, when the generation goes on, PMBGNP suffer serious diversity loss due to the sample size is much smaller than the required one. Moreover, since PMBGNP does not have any mechanism to preserve the diversity, therefore it quickly converges to a local optimum.

GNP shows better fitness values than PMBGNP, because even if poor individuals are generated in an initial generation, crossover and mutation can explore the search space in each generation, which avoids the premature convergence.

The proposed hPMBGNP shows the best fitness among the three algorithms. Comparing with PMBGNP, the simulation result confirms that the proposed algorithm can maintain the population diversity that avoids the local convergence. On the other hand, the result shows that hPMBGNP has faster convergence than GNP in an early generations, and achieves better fitness value than GNP in the last generation.

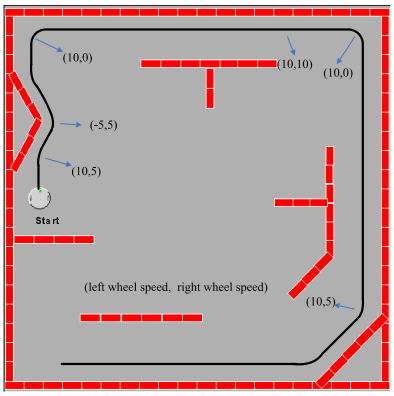

Fig. 7 The successful track of robot by hPMBGNP.

Table 3 Result of t-test between GNP and hPMBGNP in Simulation 1.

	GNP	hPMBGNP
Mean	0.63	0.72
Standard deviation	0.088	0.058
T-test (p value)	0.0074	–

Figure 7 shows a successful track of the best solution controlled by hPMBGNP. The figure shows that the proposed algorithm can solve the wall following problem well. The robot can move straight along the wall and avoid the obstacles. In some trials, GNP can also obtain good tracks. But in other trials, good results cannot be obtained, thus GNP achieves lower average fitness value than hPMBGNP. On the other hand, PMBGNP cannot solve the problem due to the premature convergence.

Table 3 shows the results of t-test (one side) of the mean fitness values between GNP and hPMBGNP. The p value shows that there is a significant difference between GNP and hPMBGNP.

§ 2 Simulation 2

In this simulation, the start position is randomly changed every generation, thus if there is no wall around the robot, it must search for the walls and move along them. The motivation of simulation 2 is to study if the evolution process can learn and evolve building blocks of GNP individuals which can adapt to more general situations, when the start position is randomly set every generation. There-

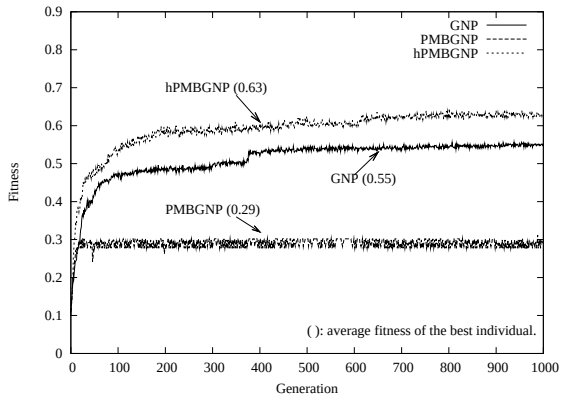

Fig. 8 Simulation results of GNP, PMBGNP and hPMBGNP in Simulation 2.

Table 4 Result of t-test between GNP and hPMBGNP in Simulation 2.

	GNP	hPMBGNP
Mean	0.55	0.63
Standard deviation	0.127	0.102
T-test (p value)	0.0011	–

fore, when the evolution process starts, the identification and recombination ability of building blocks among GNP, PMBGNP and hPMBGNP were compared.

Figure 8 shows the average fitness curves of the best individuals in each generation over 30 independent simulations. hPMBGNP also shows the best fitness values among the three algorithms, which means that, even if the simulation situation changes, the adaptable building blocks can be identified by the proposed algorithm and efficient recombination of building blocks make it possible to achieve better performances.

Table 4 shows the results of t-test (one side) of the mean fitness values between GNP and hPMBGNP in simulation 2, where there is a significant difference between GNP and hPMBGNP.

§3 Generalization ability

To testify the generalization ability of the proposed algorithm, the robot is evaluated in an testing environment as shown in Figure 9. The best solutions[*5] of GNP and hPMBGNP obtained from simulation 2 are used to control the robot. The average fitness values of each algorithm are calculated based on 1000 independent trials, in which the start position of the robot is set randomly. Table 5 shows the performance of GNP and hPMBGNP in the testing environment. The average fitness value and standard deviation[*6] in Table 5 show that the proposed algorithm

∗5 30 independent runs of simulation 2 can obtain 30 best solutions.
∗6 For each best solution, the average fitness value and standard deviation can be calculated from 1000 independent trials. The final average fitness value and standard deviation can be obtained from the 30 average fitness values and standard deviations.

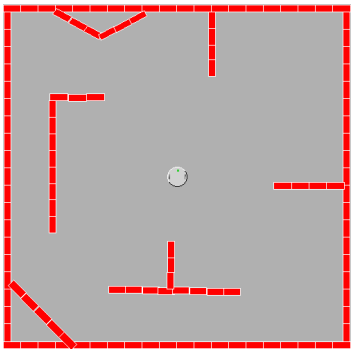

Fig. 9 Testing environment.

achieves the best average fitness value, and the t-test result shows that there are significant differences between GNP and hPMBGNP.

Table 5 Result of t-test between GNP and hPMBGNP in the testing environment.

	GNP	hPMBGNP
Mean	0.36	0.45
Standard deviation	0.115	0.080
T-test (p value)	2.83×10^{-5}	–

5·3 *Effect of multiple probability vectors*

In this subsection, the number of populations $|R|$ and the number of individuals in each population M are set at various values to study the effects of multiple probability vectors. Most simulation settings are the same as Table 2, while different settings of $|R|$ and M are used.

The total number of individuals N equals to 300. Since there are so many combinations of $|R|$ and M, we simply define that each population has the same number of individuals, which means $M = \frac{N}{|R|}$. Therefore, we tested four simulations, where the settings of $|R|$ are like $|R| = 4$, $|R| = 5$, $|R| = 6$ and $|R| = 7$[*7]. Table 6 shows the detailed parameter settings of the 4 simulations. Figure 10 shows the effects of different $|R|$ on the best fitness curves.

For each simulation, at least 2 populations are needed for crossover, and mutation is applied to the left populations. Since crossover has little exploration ability, a small number of populations $|R|$ will cause small mutation effects, which can not guarantee enough exploration ability to obtain the best performance. Therefore, in the case of $|R| = 4$, the simulation achieves the worst performance. On the other hand, too large $|R|$ will make too small population size M, which cannot guarantee enough sample size to estimate an accurate probabilistic model. We can find from Figure 10 that $|R| = 6$ is an appropriate value among the four simulations.

∗7 In the simulation of $|R| = 7$, 6 populations consist of 43 individ-

Table 6 Parameter settings of 4 simulations.

Simulation	$\|R\|=4$	$\|R\|=5$	$\|R\|=6$	$\|R\|=7$
Number of individuals per population M	75	60	50	42, 43
Total number of individuals N	300			
Number of probability vectors	4	5	6	7
– Crossover	2	2	2	2
– Mutation	2	3	4	5
Number of promising individuals	35	30	25	20

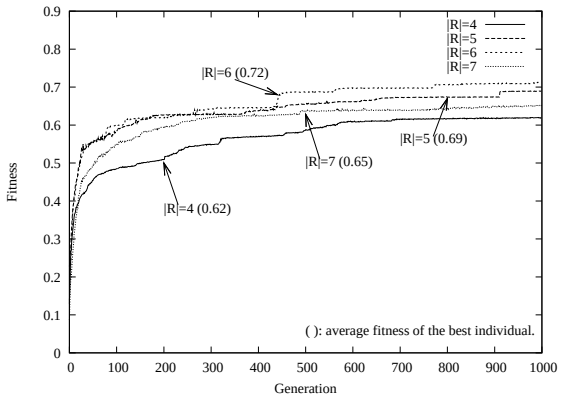

Fig. 10 Effects of different values of $|R|$.

5·4 *Diversity maintenance comparison between PMBGNP and hPMBGNP*

To evaluate the diversity maintenance, we compare the change of $o(z)$ in PMBGNP and hPMBGNP, where $o(z)$ represents the number of probabilities equal to zero in branch z as defined in **[Definition 1]**.

In PMBGNP, once the probability in the probabilistic model is equal to zero, the corresponding variable will never be sampled in the future generations, while hPMBGNP can explore its search space by mutation to overcome this problem. We consider all the probabilities in the probabilistic models, where the following value are used.

[Definition 3] $SUM(o)$ represents the total number of probabilities equal to zero in the probabilistic model, and we can easily know

$$SUM(o) = \sum_{z \in N_{bra}} o(z). \tag{21}$$

The value $SUM(o)$ can also represent the convergence of the probabilistic model.

However, one should note that exponential smoothing method in Eq. (2) causes the probabilities never equal to zero theoretically, once they are not equal to zero in the first generation. The probabilities tend to decrease with respect to the smoothing rate α, if they are smaller than the other probabilities of the same branch. Therefore, in order to calculate the value of $SUM(o)$ to compare the

uals and 1 population consists of 42 individuals.

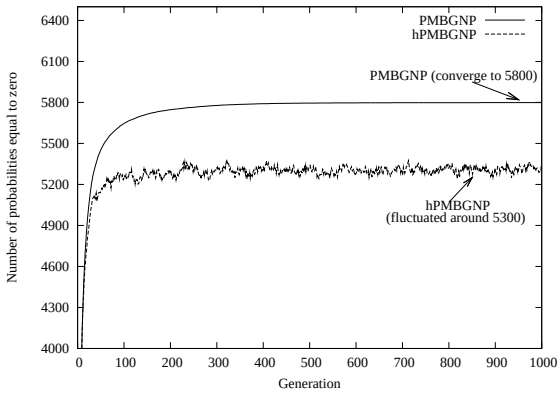

Fig. 11 Comparison of the diversity between PMBGNP and hPMBGNP.

diversity maintenance between PMBGNP and hPMBGNP, the following assumption is used.

[Assumption 1] In the probabilistic model, a probability $P(i,k,j)$ is regarded as zero, if it satisfies

$$P(i,k,j) < 1.0 \times 10^{-6}. \tag{22}$$

Although the probabilities smaller than 1.0×10^{-6} are not actually equal to zero, these probabilities are too small to be sampled. Therefore, **[Assumption 1]** is not contrary to the probabilistic model, but makes it possible for $SUM(o)$ to converge.

We use simulation 1 to compare the diversity maintenance between PMBGNP and hPMBGNP. We can find from Table 2 there are $|N_{GNP}| - 1 = 59$ probabilities in each branch, and each individual consists of $|N_{bra}| = 100$ branches. Therefore, the total number of probabilities in the probabilistic model N_{prob} can be calculated by

$$N_{prob} = |N_{bra}| \times (|N_{GNP}| - 1)$$

$$= 100 \times 59 = 5900. \tag{23}$$

Therefore, the probabilistic model of PMBGNP consists of $N_{prob} = 5900$ probabilities. On the other hand, since hPMBGNP consists of $|R| = 6$ probabilistic models, we can obtain total $6 \times N_{prob}$ probabilities. In order to make a fair comparison, after counting the number of probabilities equal to zero $SUM(o)$ for each algorithm, we compare $SUM_{\text{PMBGNP}}(o)$ with $\frac{SUM_{\text{hPMBGNP}}(o)}{6}$ in the simulations.

Figure 11 shows the comparison of the diversity between PMBGNP and hPMBGNP. In the figure, the y-axis of PMBGNP is calculated by $SUM_{\text{PMBGNP}}(o)$, while that of hPMBGNP is by $\frac{SUM_{\text{hPMBGNP}}(o)}{6}$. From this simulation, we can find that the probabilistic model of PMBGNP converges during the evolution process.

[Theorem 5] A probabilistic model of PMBGNP converges, if it satisfies

$$SUM(o) = N_{prob} - |N_{bra}|. \quad (24)$$

《**Proof**》 If the probabilistic model of PMBGNP converges, the probabilistic model will always produce the same individual, which means every branch is sampled to connect to a specific node with probability 1. In this case, if branch z is sampled to connect to a specific node with probability 1, it means that the number of probabilities equal to zero in branch z is

$$o(z) = (|N_{GNP}| - 1) - 1 = |N_{GNP}| - 2. \quad (25)$$

Then, the total number of probabilities equal to zero in the probabilistic model is

$$SUM(o) = |N_{bra}| \times o(z) = |N_{bra}| \times (|N_{GNP}| - 2)$$
$$= |N_{bra}| \times (|N_{GNP}| - 1) - |N_{bra}|. \quad (26)$$

By applying Eq. (23) to Eq. (26), **[Theorem 5]** is proven.

□

Based on **[Theorem 5]**, we can easily find the probabilistic model of PMBGNP has converged in this simulation, since

$$SUM_{\text{PMBGNP}}(o) = N_{prob} - |N_{bra}|$$
$$= 5900 - 100 = 5800. \quad (27)$$

On the other, mutation can avoid the convergence of the probabilistic model of hPMBGNP. From Figure 11, we can find $\frac{SUM_{\text{hPMBGNP}}(o)}{6}$ is fluctuated around 5300, and never converge. Therefore, hPMBGNP can maintain the diversity to find the optimal solution gradually.

6. Conclusions and future work

This paper theoretically analyzed the significance of the diversity loss in general PMBEAs, including PMBGA, PMBGP and PMBGNP. The analysis shows that this issue is especially serious in PMBEAs with more complex structures, such as PMBGP and PMBGNP. Based on the analysis, this paper proposed a hybrid PMBGNP to maintain the population diversity of PMBGNP. The proposed algorithm is an extension of PMBGNP which is a graph structure based PMBEA proposed recently. In the proposed algorithm, two techniques, named multiple probability vectors and genetic operators, have been proposed to maintain the population diversity and to make PMBGNP capable of handling the problems with large search space. This paper theoretically analyzed the diversity maintenance of the proposed algorithm by applying it to control the movement of Khepera robot. The experimental results show the superiority of the proposed algorithm, comparing with the conventional GNP and standard PMBGNP.

There are mainly two advantages of the proposed algorithm. Firstly, comparing with the conventional GNP, the proposed algorithm inherits the characteristics of PMBEAs avoiding the frequent breakage of building blocks to achieve better performances. On the other hand, comparing with the standard PMBGNP, the proposed algorithm can maintain the population diversity to avoid the premature convergence of local optima.

PMBGNP has much potential to be expanded. The previous research of GNP has showed its superiority to handle the dynamic environments, comparing with the classical evolutionary algorithms, such as GP and EP. It is possible that PMBGNP would inherit such advantages that outperforms the algorithms of PMBGP. In the future, we will emphasize the research of PMBGNP by comparing with the classical algorithms of PMBGP. On the other hand, the current PMBGNP is only designed based on the pairwise interactions, therefore, PMBGNP with multivariate interactions will be studied in the future.

Acknowledgments

We would like to thank Prof. Hitoshi Iba and the anonymous reviewers for their helpful comments.

♢ References ♢

[Baluja 94] Baluja, S.: Population-based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Leaning, Tech. Report. No. CMU-CS-94-163, Carnegie Mellon University (1994)

[Chen 09] Chen, Y., Ohkawa, E., Mabu, S., Shimada, K., and Hirasawa, K.: A Portfolio Optimization Model using Genetic Network Programming with Control Nodes, *Expert Sytems with Applications*, Vol. 36, pp. 10735–10745 (2009)

[Cyberbotics] Cyberbotics, : Webots: http://www.cyberbotics.com/

[Eguchi 06] Eguchi, T., Hirasawa, K., Hu, J., and Ota, N.: A Study of Evolutionary Multiagent Models Based on Symbiosis, *IEEE Trans. on Systems, Man and Cybernetics, Part B*, Vol. 36, No. 1, pp. 179–193 (2006)

[Goldberg 89] Goldberg, D. E.: *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison-Wesley (1989)

[Handa 07] Handa, H.: The Effectiveness of Mutation Operation in the case of Estimation of Distribution Algorithms, *BioSystems*, Vol. 87, pp. 243–251 (2007)

[Harik 98] Harik, G. R., Lobo, F. G., and Goldberg, D. E.: The Compact Genetic Algorithm, *In Proc. of the IEEE Conference on Evolutionary Computation*, pp. 523–528 (1998)

[Hasegawa 08] Hasegawa, Y. and Iba, H.: A Bayesian Network Approach to Program Generation, *IEEE Trans. on Evolutionary Computation*, Vol. 12, No. 6, pp. 750–764 (2008)

[Hirasawa 01] Hirasawa, K., Okubo, M., Katagiri, H., Hu, J., and Murata, J.: Comparison between Genetic Network Programming (GNP) and Genetic Programming (GP), *In Proc. of the IEEE Congress on Evolutionary Computation*, pp. 1276–1282 (2001)

[Hirasawa 08] Hirasawa, K., Eguchi, T., Zhou, J., Yu, L., and Markon, S.: A Double-Deck Elevator Group Supervisory Control System Using Genetic Network Programming, *IEEE Trans. on Systems, Man and Cybernetics, Part C*, Vol. 38, No. 4, pp. 535–550 (2008)

[Holland 75] Holland, J. H.: *Adaptation in Natural and Artificial Systems*, University of Michigan Press (1975)

[Katagiri 00] Katagiri, H., Hirasawa, K., and Hu, J.: Genetic Network Programming -Application to Intelligent Agents, *In Proc. of the IEEE Int'l Conf. on Systems, Man and Cybernetics 2000*, pp. 3829–3834 (2000)

[Koza 92] Koza, J. R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press (1992)

[Koza 94] Koza, J. R.: *Genetic Programming II: Automatic Discovery of Reusable Programs*, MIT Press (1994)

[Larrañaga 02] Larrañaga, P. and Lozano, J. A.: *Estimation of Distribution Algorithms, A New Tool for Evolutionary Computation*, Kluwer Academic Publishers (2002)

[Li 09] Li, X., Mabu, S., Zhou, H., Shimada, K., and Hirasawa, K.: Genetic Network Programming with Estimation of Distribution Algorithms and its Application to Association Rule Mining for Traffic Prediction, *In Proc. of the ICROS-SICE Int'l Conf.*, pp. 3457–3462 (2009)

[Li 10a] Li, X., Mabu, S., Zhou, H., Shimada, K., and Hirasawa, K.: Genetic Network Programming with Estimation of Distribution Algorithms for Class Association Rule Mining in Traffic Prediction, *In Proc. of the IEEE Congress on Evolutionary Computation*, pp. 2673–2680 (2010)

[Li 10b] Li, X., Mabu, S., Zhou, H., Shimada, K., and Hirasawa, K.: Genetic Network Programming with Estimation of Distribution Algorithms for Class Association Rule Mining in Traffic Prediction, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol. 14, No. 5, pp. 497–509 (2010)

[Mabu 06] Mabu, S., Hatakeyama, H., Thu, M. T., Hirasawa, K., and Hu, J.: Genetic Network Programming with Reinforcement Learning and Its Application to Making Mobile Robot Behavior, *IEEJ Trans. on Electronics, Information and Systems*, Vol. 126, No. 8, pp. 1009–1015 (2006)

[Mabu 07] Mabu, S., Hirasawa, K., and Hu, J.: A Graph-Based Evolutionary Algorithm: Genetic Network Programming (GNP) and Its Extension Using Reinforcement Learning, *Evolutionary Computation*, Vol. 15, No. 3, pp. 369–398 (2007)

[Michel 96] Michel, O.: Khepera Simulator Package Version 2.0: Freeware Mobile Robot Simulator Written at the University of Nice Sophia-Antipolis by Olivier Michel (1996), Downloadable from the World Wide Web at http://diwww.epfl.ch/lami/team/michel/khepsim/

[Miller 00] Miller, J. F. and Thomson, P.: Cartesian Genetic Programming, *In Proc. of the 3rd European Conference on Genetic Programming*, pp. 121–132 (2000)

[Mühlenbein 96] Mühlenbein, H. and Paaß, G.: From Recombination of Genes to the Estimation of Distributions I. Binary Parameters, *In Proc. of the 4th Conf. on Parallel Problem Solving from Nature*, pp. 178–187 (1996)

[Murata 04] Murata, T. and Nakamura, T.: Multi-Agent Cooperation Using Genetic Network Programming with Automatically Defined Groups, *In Proc. of GECCO 2004*, pp. 712–714 (2004)

[Nordin 98] Nordin, P., Banzhaf, W., and Brameier, M.: Evolution of a World Model for a Miniature Robot Using Genetic Programming, *Robotics and Autonomous Systems*, Vol. 25, pp. 105–116 (1998)

[Pelikan 02a] Pelikan, M., Goldberg, D. E., and Cantu-Paz, E.: Linkage Problem, Distribution Estimation, and Bayesian Networks, *Evolutionary Computation*, Vol. 8, No. 3, pp. 311–341 (2002)

[Pelikan 02b] Pelikan, M., Goldberg, D. E., and Lobo, F. G.: A Survey of Optimization by Building and Using Probabilistic Models, *Computational Optimization and Applications, Kluwer Academic Publishers*, Vol. 21, pp. 5–20 (2002)

[Salustowicz 97] Salustowicz, R. P. and Schmidhuber, J.: Probabilistic Incremental Program Evolution, *Evolutionary Computation*, Vol. 5, No. 2, pp. 123–141 (1997)

[Sastry 03] Sastry, K. and Goldberg, D. E.: Probabilistic Model Building and Competent Genetic Programming, *Genetic Programming Theory and Practice*, Vol. 13, pp. 205–220 (2003), In R. L. Riolo and B. Worzel, editors

[Sastry 05] Sastry, K., Abbass, H. A., Goldberg, D. E., and Johnson, D. D.: Substructural Niching in Estimation of Distribution Algorithms, *In Proc. of GECCO 2005*, pp. 671–678 (2005)

[Shan 06] Shan, Y., McKay, R. I., Essam, D., and Abbass, H. A.: A Survey of Probabilistic Model Building Genetic Programming, *Studies in Computational Intelligence (SCI)*, Vol. 33, pp. 121–160 (2006)

[Shapiro 06] Shapiro, J. L.: Diversity Loss in General Estimation of Distribution Algorithms, *In Proc. of the 9th Conf. on Parallel Problem Solving from Nature*, pp. 92–101 (2006)

[Shimada 06] Shimada, K., Hirasawa, K., and Hu, J.: Genetic Network Programming with Acquisition Mechanisms of Association Rules, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol. 10, No. 1, pp. 102–111 (2006)

[Teller 95] Teller, A. and Veloso, M.: PADO: Learning tree structured algorithms for orchestration into an object recognition system, Tech. Report. No. CMU-CS-95-101, Carnegie Mellon University (1995)

[Yanai 03] Yanai, K. and Iba, H.: Estimation of Distribution Programming based on Bayesian Network, *In Proc. of the IEEE Congress on Evolutionary Computation*, pp. 1618–1625 (2003)

〔担当委員：伊庭 斉志〕

Received August 6, 2010.

Author's Profile

LI, Xianneng (Student Member)

Received the B.E. degree from Nanjing University, China, in 2008 and the M.E. degree from Waseda University, Japan in 2010. Currently, he is a Ph.D. candidate at Graduate School of Information, Production and Systems, Waseda University, Japan. His research interest includes evolutionary computation, data mining and robotics. He is a student member of IEEE, IEEE Computational Intelligence Society and the Japanese Society for Evolutionary Computation.

Author's Profile

MABU, Shingo (Member)

Received the B.E. and M.E. degrees in Electrical Engineering from Kyushu University, Japan, in 2001 and 2003, respectively, and Ph.D degree from Waseda University, Japan, in 2006. In 2006, he was a Visiting Lecturer at Waseda University. Since 2007, he has been an Assistant Professor at the Graduate School of Information, Production and Systems, Waseda University, Japan. Dr. Mabu is a member of IEEE, ACM and Japanese Society for Evolutionary Computation.

Author's Profile

HIRASAWA, Kotaro

Received the B.E. and M.E. degrees from Kyushu University, Japan, in 1964 and 1966, respectively. From 1966 to 1992, he worked at Hitachi Ltd. as a Vice President of the Hitachi Research Laboratory. From December 1992 to August 2002, he was a Professor at the Graduate School of Information Science and Electrical Engineering of Kyushu University. Since September 2002, he has been a Professor at the Graduate School of Information, Production and Systems, Waseda University. Dr. Hirasawa is a member of the Society of Instrument and Control Engineers, the Institute of Electrical Engineers of Japan, IEEE and ACM.