Evolutionary Algorithm with Phenotype Diversity for Virtual Network Embedding

Tatsuya Otoshi[¶], Masayuki Murata[†]
[¶] Graduate School of Economics, Osaka University
1-7 Machikaneyama-Cho, Toyonaka, Osaka 565-0043, Japan
[†]Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka 565-0871, Japan

***Abstract*—With the diversification of applications using the Internet, network virtualization technologies that flexibly allocate network resources are attracting attention. In network virtualization, virtual network embedding is important to properly map the requirements of the virtual network to the physical network. However, it takes time to calculate the solution by optimization, and recalculation of the embedding becomes a problem when the environment of the virtual and physical networks changes. Therefore, a method of having multiple solution candidates in advance and switching the solution depending on the situation is considered, but the design and updating of the solution candidates themselves remain an issue. Such a relationship between solution candidates and solution selection is similar to the relationship between genotype and phenotype in biological evolution, and it is a shortcut to get hints from evolution. In biological evolution, the phenotype searches for short-term practical solutions while the genotype continues to search for optimal solutions. By introducing this mechanism into the network, it is possible to select a quasi-optimal solution in a fluctuating environment while continuing the search for a better solution candidate itself. In this paper, we propose a dynamic virtual network embedding method in which the solution candidates themselves can be dynamically updated based on the evolution of genotype and phenotype. In this method, candidate solutions are encoded as genotypes, and phenotypes are decoded by attractor selection using noise-induced fluctuations. Through evaluation, we show that the attractor selection by individuals leads to the discovery of appropriate solutions faster than when using neural networks.**

***Index Terms*—Novelty Search, Attractor Selection, Virtual Network Embedding, Local Competition, Evolutionary Algorithm**

I. Introduction

With the diversification of Internet-based applications, network virtualization technologies that flexibly allocate network resources are attracting attention [1]. By mapping a virtual network, which reflects the different network resource demands of different applications, onto the physical network, various per-application requirements can be met on a single network. This is called virtual network embedding [2] and is formulated as an optimization problem, which generally cannot be solved in a realistic time. For this reason, approaches that seek approximate solutions by various heuristics are often used [3].

Even after the virtual network has been embedded, changes in application conditions or changes in the environment on the physical network may result in the mismatch of embedding. In recent years, the technology of dynamically constructing slices in response to changes in functional requirements has been attracting attention [4], [5], and the demand for dynamically changing the virtual network embedding has been increasing as background technology. Therefore, dynamic virtual network embedding is necessary to continuously monitor the status of the virtual network and the physical network to reconfigure the virtual network [6]. In this case, the time allowed for re-calculation of the virtual network embedding is short, making time-consuming approaches such as optimization difficult. In addition, the time required to switch between embeddings is not negligible, and therefore, it is necessary to prepare for the switchover in advance or to make continuous changes.

One way to reduce the time required for recalculation is to have multiple candidate solutions in advance and switch between them depending on the situation. For example, a method of continuing service even in the event of a failure of a physical resource by switching to a backup virtual network prepared in advance on another physical resource is being considered [7]. A method to dynamically embed a virtual network by combining switching among prepared solution candidates and searching with small noise has also been proposed [8]. However, in these methods, it is difficult to know what kind of solution candidates to keep. In particular, when large environmental changes occur, any solution candidate may no longer be appropriate, and updating the solution candidates themselves becomes an issue.

The process of selecting among solution candidates and updating the solution candidates themselves is similar to the relationship between phenotype and genotype in biological evolution. Biological systems are known to exhibit multiple different phenotypes depending on the environment, even for the same gene, and this property is called phenotypic plasticity. Genotypic evolution causes changes in this phenotypic plasticity [9], which in turn changes the phenotypic response to the environment. There is also a known Baldwin effect in which phenotypic plasticity causes evolution to promote learning in individuals [10]. Biological evolution has successfully adapted to environmental variation through the mutual influence of genotype and phenotype. However, in conventional genetic algorithms, genotype and phenotype are identical, and phenotypic plasticity has not been exploited [11]. Therefore, we expect that mimicking this evolutionary mechanism will be promising for selecting solutions from candidates while updating the candidates of the virtual network.

In this paper, we propose a dynamic virtual network embedding method in which the solution candidates themselves can be dynamically updated based on the evolution of genotype and phenotype. The method is based on novelty search [12], which is often used in the field of robot control. In novelty search, individuals achieve phenotypic plasticity using a neural network, and the structure of the neural network is often used as the genotype. In this case, the phenotypic plasticity is solely determined by the genotype, so there is no search for a new solution in the generation. However, in virtual network embedding, the environment changes dynamically according to the communication situation, so short-term search using noise [8] is effective. In addition, it is difficult to set backups in advance because the solution candidates are non-explicit in neural networks. Therefore, in our method, the candidate solutions are encoded as genotypes, and the phenotypes are decoded by attractor selection using noise-induced fluctuations. Through the evaluation, we showed that the individual's attractor selection leads to the discovery of an appropriate solution faster than when using a conventional neural network.

The remainder of this paper is organized as follows: Section II introduces the dynamic virtual network embedding assumed in this paper. Section III describes the proposed method, which combines the evolution of genotypes by novelty search with the selection and exploration of phenotypes by attractor selection; Section IV evaluates the virtual network embedding by simulation. In Section V, we summarize this paper and discuss future work.

II. Dynamic Virtual Network Embedding

First, we explain the dynamic virtual network embedding envisioned by the proposed method. In conventional virtual network embedding, it is assumed that after the virtual network embedding is configured upon arrival of a request, the same configuration is basically continued to be used. However, in reality, as the environment changes, resource requests also change dynamically, and the virtual network needs to be reconfigured to keep up with the changes in requests [8]. Reconfiguration of virtual networks requires changing the configuration of network devices and waiting for the changes to be reflected in the overall network, which can cause delays in response and lead to network instability due to frequent changes. Therefore, multiple virtual networks are built in advance, and the virtual network to be used can be changed dynamically according to the situation, thereby solving the problem of lag in configuration changes [7]. However, it is necessary to cover the solution space with a constant number of virtual networks because the solution space for virtual network embedding is huge, whereas the virtual networks that are generally prepared in advance are constant. Even if the switching of virtual networks can be made fast, changing an existing virtual network requires reconfiguration of the network as in the past, so it is necessary to limit the number of changes to a few at a time for the virtual network being prepared. Therefore, by constructing candidate virtual networks to be prepared in an evolutionary manner, we aim to ensure diversity with a constant number of candidates while keeping up with environmental changes through gradual changes due to mutations.

In addition, computation time is a particular issue for dynamic virtual network embedding. In general, the objective of virtual network embedding is to embed as many virtual networks as possible on limited physical resources, considering resource arbitration among multiple virtual networks. However, such mediation among multiple virtual networks is impractical for dynamic virtual network embedding because the computation time increases as the number of virtual networks increases. Therefore, in this paper, embedding is performed for each single virtual network.

In this case, if there is no combination of physical resources available for embedding at the time the request for embedding arrives, the request is rejected. Therefore, it is necessary to perform embedding at the stage of accepting each request, leaving surplus resources for future requests. Thus, in this paper, the resource mediation among virtual networks is tackled by maximizing the surplus physical resources at each single virtual network embedding.

III. Novelty Search with Attractor Selection for Virtual Network Embedding

In a dynamic environment, the embedding is recalculated according to changes in resource demand and physical resources, but at this time, the conditions of the acceptable solution do not change significantly, and there are parts of the solution that can be reused. In such a case, it is desirable to have a list of candidate acceptable solutions and dynamically select the appropriate solution for the fluctuating environment. However, it is also necessary to update the candidate solutions themselves to cope with large fluctuations. A familiar example that implements such a complex operation of selecting solution candidates, selecting solutions according to the situation, and updating the solution candidates themselves is the mechanism of biological evolution. In biological evolution, the genotype defines the phenotypic response to the environment, and changes in the genotype through evolution allow the organism to respond to large environmental changes.

In this section, we introduce a method for solving dynamic virtual network embedding by evolving genotypes through novelty search and exploring phenotypes through attractor selection.

A. Overview of the Proposed Method

As an individual behavior, we consider the selection of a phenotype from multiple phenotypes by attractor selection, using feedback from the environment. In other words, the attractor structure (attractor set) is used as the genotype, and the solution suitable for the environment is selected from the attractor set by attractor selection. Figure 1 shows the overview of our method.

The following procedure generates an individual with an attractor set that is more adaptable to the environment and searches for a solution that is suitable for the environment.

1) Initialize the random genotype (attractor structure) of each individual (green part in the figure).
2) Each individual selects a solution suitable for the environment from a set of attractors by attractor selection with activity in simulation of virtual network embedding(blue part in the figure).
3) Evaluate novelty and fitness for each individual choice, then select surviving individuals (yellow part in the figure).
4) Move to the next generation and repeat the procedure from Step 2.

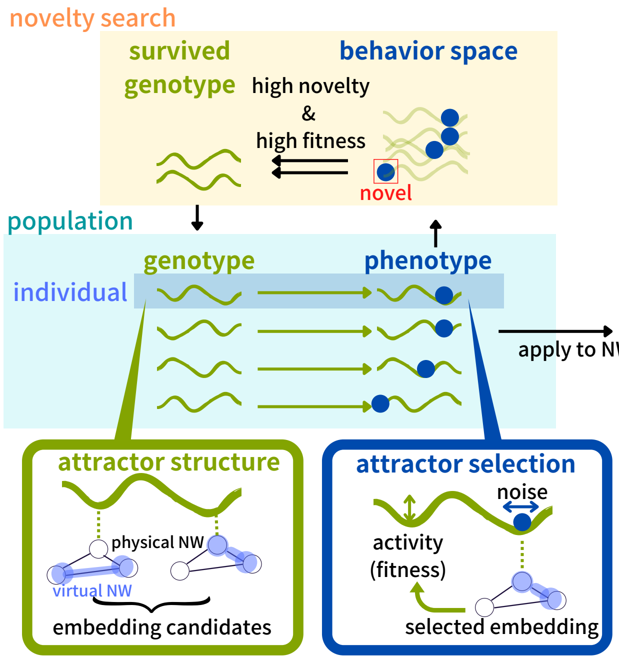


Fig. 1. overview of novelty search with attractor selection

B. Genotype and Phenotype with Attractor Selection

First, we define the genotype encoding and phenotype decoding with attractor selection.

a) Genotype: In attractor selection, the structure of the attractor is specified by the structure of the Hopfield network (green line in Fig. 1), which influences the selection. In novelty search, the coupling structure of a neural network is often used as the genotype of an individual, and it seems natural to use the coupling structure of the Hopfield network as the genotype of an individual here as well.

In other words, the weight of the coupling between nodes i,j of the Hopfield network is w_{ij} (where $w_{ij} = w_{ji}$ and $w_{ii} = 0$), and the series $\boldsymbol{w} = w_{01}, w_{02}, \cdots, w_{n-1,n}$ as the genotype.

Mutation of the genotype is equivalent to adding a normal random number to w_{ij}.

b) Phenotype: Once the structure of the Hopfield network is determined by the genotype, attractor selection updates the state $\boldsymbol{x_t}$ (blue circle in Fig. 1) at each time according to the following equation.

$$\frac{d\boldsymbol{x_t}}{dt} = \alpha_t F_{\boldsymbol{w}}(\boldsymbol{x_t}) + \boldsymbol{\eta_t} \tag{1}$$

where α_t is the activity representing the goodness of the state, $F_{\boldsymbol{w}}$ is the Hopfield network, and $\boldsymbol{\eta_t}$ is the noise term.

The state $\boldsymbol{x_t}$ determines the output of the individual at each time and incorporates feedback from the environment as activity α. Here, the time series of behavior $\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots$ becomes the phenotype. In the case of virtual network embedding, whether or not to embed virtual node i in physical node j is represented by x_{ij}^{node}, and whether or not to embed virtual link l in physical path p is represented by x_{lp}^{link} as values of 0 and 1. $\boldsymbol{x_t}$ is a vector of them: $\boldsymbol{x_t} = (x_{00}^{node}, x_{01}^{node}, \cdots, x_{00}^{link}, x_{01}^{link}, \cdots)$.

In the case of a neural network as an individual, the selection is critically determined by the weights from the genotype and the input from the environment. In the case of attractor selection, however, different phenotypes may be obtained even for the same genotype and environment due to the noise term.

Reinforcement learning [13] is also a typical example of feedback-based action decision making, but reinforcement learning assumes that the choices themselves are fixed. In reinforcement learning, the reward is estimated by accumulating feedback for each situation and each choice, but if the choice itself changes, it is necessary to accumulate feedback anew. Attractor selection, on the other hand, does not require the accumulation of feedback, since activity is calculated only based on the goodness of current choice.

C. Population update with Novelty Search

Unlike conventional genetic algorithms, novelty search evaluates not only fitness but also novelty in terms of individual survival. In the following, genetic selection using fitness and novelty will be explained.

a) Fitness: In attractor selection, the selection is done in such a way that the goodness of the state is used for updating the state so that it moves to a better state. In novelty search, the goodness of the solution also can be used to update the generation, considering not only a novelty but also fitness as in conventional genetic algorithms.

In the case of virtual network embedding, the goodness of the state corresponds to the objective function of the virtual network embedding, for example, maximization of surplus resources to accommodate future request under penalizing resource excess at each node and link as follows.

$$\begin{aligned}
F(\boldsymbol{x_t}) &= R(\boldsymbol{x_t}) - \lambda P(\boldsymbol{x_t}) && (2)\\
R(\boldsymbol{x_t}) &= \sum_j (r_j^{node} - y_j^{node}) + \sum_l (r_l^{link} - y_l^{link}) && (3)\\
P(\boldsymbol{x_t}) &= \sum_j [p_j^{node} - y_j^{node}]^+ + \sum_l [p_l^{link} - y_l^{link}]^+ && (4)\\
y_j^{node} &= \sum_i x_{ij}^{node} d_i^{node} && (5)\\
y_l^{link} &= \sum_{p \ni l} x_{lp}^{link} d_p^{link} && (6)\\
& && (7)
\end{aligned}$$

Here, $R(\boldsymbol{x})$ represents the total amount of surplus resources, $P(\boldsymbol{x})$ represents the penalty term for exceeding resources, y_j^{node}, y_l^{link} are the allocated demands of physical link and node, d_i^{node}, d_p^{link} are the demands of virtual link and node, and r_j^{node}, r_l^{link} are the resources of physical link and node. The λ represents the weight of the penalty term and should be set to a sufficiently large value. In our simulation, we set $\lambda = 500$, which is large enough to activate the penalty compared with the surplus resources.

We also use F as the activity in the attractor selection. Although activity and fitness both represent the goodness of a state or solution, they are used at different times. Activity is used to select the behavior of individuals within a generation, while fitness is used to select the surviving individuals across generations.

b) Fitness with Novelty: When fitness is taken into account in intergenerational renewal, one individual with high fitness will affect the entire population, resulting in the extirpation of other individuals and a decrease in diversity. In the literature [12], a method is proposed to limit the competition by a fitness to a local range (local competition) to avoid the influence of the fitness of some individuals spreading to the whole population.

By using the distance space of behavior as the distance space for local competition as well as the distance space for novelty calculation, the distance calculation used for novelty calculation can be used for local competition. Let $\mathscr{O}_i$, the neighborhood of individual i, be the top k individual with the closest distance from i. The relative competitiveness of the neighborhood of individual i is determined by the following formula.

$$F_i(\boldsymbol{x_t^i}) = \frac{F(\boldsymbol{x_t^i}) - \min_{j \in \mathscr{O}_i} F(\boldsymbol{x_t^j})}{\max_{j \in \mathscr{O}_i} F(\boldsymbol{x_t^j}) - \min_{j \in \mathscr{O}_i} F(\boldsymbol{x_t^j})} \tag{8}$$

Let $\boldsymbol{x_t^i}$ denote the solution at time t for individual i. If k is the total number of individuals, it is equal to the usual fitness.

Each individual is evaluated for survival by the weighted sum of its competitiveness in this neighborhood and its novelty. However, since $F_i(\boldsymbol{x_t^i})$ is normalized to 0-1, while novelty is not, it is not appropriate to take the weighted sum as is. Therefore, following the literature [14], we normalize the novelty to the maximum and minimum values and then take the weighted sum.

$$E(\boldsymbol{x_t^i}) = (1-w)\bar{\rho}(\boldsymbol{x_t^i}) + wF^i(\boldsymbol{x_t^i}) \tag{9}$$

Let $\bar{\rho}$ denote the standardized individual novelty and w denote the fitness weight. When w is 1, a pure novelty search is performed, and when w is 0, only the competitiveness in the neighborhood is used to select the surviving solution. When $0 < w < 1$, simultaneous optimization of novelty and competitiveness in the neighborhood is performed. In the literature [14], w was varied from 0 to 1 in increments of 0.1, and individuals with high fitness were obtained in the range of 0.2 to 0.6 (especially for $w = 0.5, 0.2$).

c) Behavior: In novelty search, the phenotype of individuals is represented as behavior, and the distance of behavior between individuals is represented as a novelty, and the search for a variety of individuals is carried out by maximizing this distance over generations [15].

The behavior can be a summary of a series of outputs, such as the final output of each individual, or it can be the series itself.

In the case of attractor selection, individuals search for a solution within a generation, and when an appropriate solution is found, they converge to a specific attractor, so it is natural to use the state at the time of convergence as behavior.

In other words, using the state $\boldsymbol{x_f^i}$ at the time of convergence of individual i, we can obtain behavior $\boldsymbol{b_i}$ is determined as follows.

$$\boldsymbol{b_i} = \boldsymbol{x_f^i} \tag{10}$$

Then, the distance (novelty) between individual behaviors is defined as follows.

$$\rho(\boldsymbol{b_i}) = \sum_j dist(\boldsymbol{b_i}, \boldsymbol{b_j}) \tag{11}$$

Euclidean distance is used as the distance function.

D. Characteristics of attractor selection in novelty search

In attractor selection, behavior can be naturally defined as the final output. The evaluation of an individual when it moves from one generation to the next is determined by considering what kind of behavior the individual takes. In a neural network, the convergence of behavior does not occur unless the neural network learns it as a rule in itself, but attractor selection converges if it finds a good attractor, using activity as feedback. Therefore, when an individual makes an attractor selection, it has the advantage that the converged attractor can be used to evaluate the individual naturally.

In attractor selection, the goodness of the solution can be evaluated within a generation. In the existing literature, fitness is considered at each generation as well as novelty, so when crossing generations, it is necessary to introduce their weighted sums, etc. A method that combines reinforcement learning and novelty search, [16], has also been considered, but again, the reward is considered as fitness for each generation, and the weighted sum with novelty is used to update the policy for behavioral decisions across generations. In attractor selection, activity gives the goodness of state as well as fitness, and selection is made to increase fitness even within a single generation. This makes it possible to search from both directions by clarifying the roles of the search, such as the search emphasizing diversity in genotypic changes between generations and the search emphasizing adaptation to the environment in phenotypic changes within a generation.

On the other hand, since attractor selection operates non-deterministically rather than deterministically to the input like a neural network, it may require a larger population to capture the statistical effect.

IV. Evaluation

A. Simulation Setting

To check the operation in a fluctuating environment, we randomly change the resource requirements of the virtual network every 40 generations. Initially, each individual is assumed to have a randomly generated hop field weight. The number of individuals is 100, and an individual embeds a 5-node virtual network into a 30-node physical network by attractor selection. In each generation, state updates are performed for 20 time slots, and the fitness and behavior of each individual are calculated by the attractor finally selected. The size of the population that defines the neighborhood of an individual is set to $k = 15$, and the weights of fitness and novelty are set to $w = 0.5$.

To confirm the effect of attractor selection by individuals, we compared the case where the neural network determines the phenotype from the information of the environment (*neural network* case), the case where the individual randomly selects one of the multiple solutions as the genotype and uses it as the phenotype(*random selection* case), and the case where the solution is directly encoded into the genotype(*gene* case). The inputs to the neural network are the resource requirements of the virtual network and the residual resources of the physical network, which provide more information than the attractor selection activity as input. At this time, the weights of the neural network correspond to the genotypes.

As described in Section II, the proposed method is assumed to operate independently for each virtual network to reduce the computation time even when multiple virtual networks operate on the physical network. In that case, resource arbitration among virtual networks is performed by maximizing each virtual network's surplus resources on the physical network, thereby leaving resources for other virtual networks. Therefore, whether proper embedding is achieved is evaluated by the value of fitness, which reflects the size of the surplus resources. Because we are interested in performance under normal conditions, the evaluation will always generate embedding requests where embedding exists that satisfies the resource constraints.

1) Metrics of Divergence: Since the diversity of solutions is thought to play a role in the ease of searching for solutions, in addition to evaluating the solutions found, we evaluate the diversity of solutions that the population has.

In the literature [17], the Jensen-Shannon divergence between the distribution of solutions and the uniform distribution is used as the diversity of solutions.

The Jensen-Shannon divergence is a measure of the modified symmetry of the Kalbach-Liebler divergence for two distributions P, Q, and is calculated as follows

$$D_{JS}(P,Q) = \frac{D_{KL}(P,M) + D_{KJ}(Q,M)}{2} \tag{12}$$

Here, D_{KL} is the Kullback-Leibler divergence and $M = \frac{P+Q}{2}$ is the mixture distribution of the two distributions under consideration. This is symmetric in $D_{JS}(P,Q) = D_{JS}(Q,P)$.

In the literature [17], P is the distribution of the solution, Q is the uniform distribution, and $D_{JS}(P,Q)$ is used to measure how far the solution is from the uniform distribution. In practice, P divides the solution space into n subspaces and represents the distribution in each subspace. In other words, it is defined by the following equation using the number of solutions c_i contained in each subspace.

$$P = \left(\frac{c_i}{C}, \cdots, \frac{c_n}{C}\right) \tag{13}$$

Here, $C = \sum_i c_i$ is the number of whole solutions.

The Kullback-Leibler divergence with uniform distribution is equivalent to the sign reversal of entropy as follows.

$$\begin{aligned} D_{KL}(P,Q) &= -\sum_x P(x)\log\frac{1}{n} + \sum_x P(x)\log P(x) \\ &= -\log\frac{1}{n} - H \end{aligned} \tag{14}$$

where the second term $H = -\sum_x P(x)\log P(x)$ is the entropy, and The first term is a constant. Therefore, there is no essential difference from using entropy to calculate the diversity. Therefore, we use entropy as a measure of diversity.

2) Alternative Individual Types: To confirm the effect of attractor selection, we compared the case where the neural network determines the phenotype from the information of the environment, the case where the individual randomly selects one of the multiple solutions it has as the genotype, the case the neural network determines the phenotype, and the case where the solution is directly encoded into the genotype. The inputs to the neural network are the resource requirements of the virtual network and the residual resources of the physical network, which provide more information than the attractor selection activity as input.

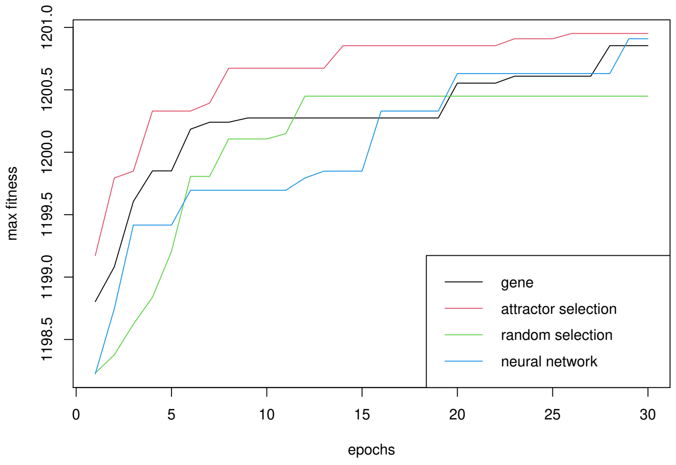
B. Result

Figure 2 shows the maximum fitness obtained by each generation after environmental changes and the entropy of the phenotype for each generation.”gene” refers to the case where the solution is directly encoded in the genotype of the individual, ”attractor selection” refers to the case where the individual performs attractor selection, ”random selection” refers to the case where the individual randomly selects one of the multiple solutions, and ”neural network” represents the case where the phenotype is determined by a neural network using environmental information as input [14].

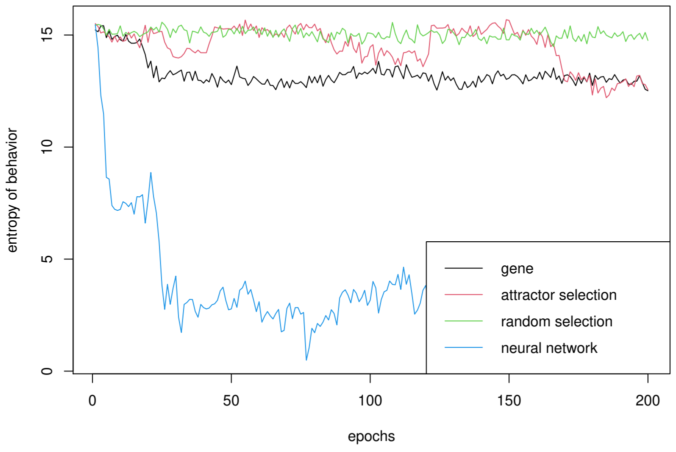
From the figure, it can be seen that the solution with higher fitness is found in the earlier generation when the individual uses attractor selection. This can be attributed to the fact that attractor selection allows individuals to search for solutions without changing generations. Except for ”genes”, one genotype contains multiple phenotypes, so some people may think that fitness is improved by the adaptation of one of the phenotypes, but as it turns out, this is not the case. However, this is not the case. It is not simply a matter of having a variety of phenotypes, but also how you change your phenotype in response to the environment that affects your fitness.

In addition, in ”attractor selection” and ”random selection”, there is a possibility that multiple phenotypes may appear from a single genotype, which is expected to increase the diversity of solutions. The ”attractor selection” and ”random selection” methods have higher entropy and produce more diverse solutions than the direct encoding of solutions into genotypes. However, ”random selection” does not improve fitness much because the solution itself is diverse but not necessarily selected according to the environment. For this reason, it is important to select a solution based on the situation, as in the case of attractor selection, to find an appropriate solution quickly.

In a neural network, the phenotype is also calculated according to the environmental situation, but the appearance of the phenotype is determined by the weights of the neural network, i.e. the genotype. For this reason, the search for a new solution is not performed within a generation, which does not lead to a speed-up in the search for a solution. In addition, even if the weights of the neural network are partially changed, the results may not change significantly, resulting in low entropy of the phenotype and low diversity of the population. Therefore, intra-generational search such as attractor selection is important for speeding up the search for solutions, and having multiple candidate solutions and switching between them is important for improving the diversity.



(a) Timeseries of maximum fitness



(b) Timeseries of entropy of phenotype

Fig. 2. Evolution with different types of individual

1) Effect of Population Size: To investigate the evolutionary transition when the number of individuals is increased, we examined the transition of the maximum fitness in the population for each generation when the number of individuals is set to 100, 500, and 1000. Figure 3 shows the evolution of the maximum fitness obtained up to generation 0 when the environmental change occurred. A single population was subjected to 20 environmental changes, and the figure shows the average maximum fitness for all environments.

From the figure, it can be seen that as the number of individuals increases, there is a tendency to find a better solution in fewer generations. This can be attributed to the fact that as the number of individuals increases, the variety of solutions that the population can include increases, and the diversity of the population increases.

On the other hand, when we compare the case where the number of individuals increases from 100 to 500 and the case where the number of individuals increases from 500 to 1000, the increase in maximum fitness is larger for the case from 100 to 500. This may be because the novelty search maintains the solution distance, allowing individuals of about 500 to cover a wide range of the solution space.

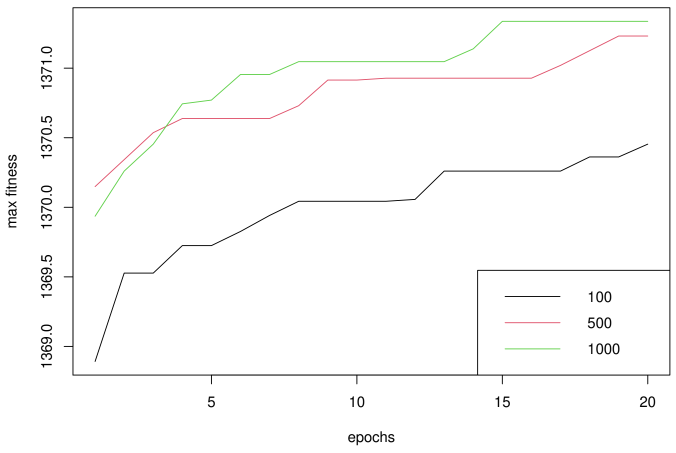


Fig. 3. Timeseries of maximum fitness with different Population sizes

V. Conclusion

In this paper, we proposed a dynamic virtual network embedding method that can rapidly adapt to environmental changes by searching in both genotype and phenotype spaces. In this method, candidate solutions are encoded as genotypes, and individuals stochastically decode their phenotypes according to their environment through attractor selection. In such phenotypic plasticity, the genotype is evolved by novelty search to ensure diversity, thereby promoting adaptation to the environment in both phenotype and genotype. The results of simulation evaluation showed that individuals can find better solutions faster than the conventional novelty search by using attractor selection.

Future work includes discussing the adaptive limits of the proposed method when the way the environment changes is altered.

Acknowledgment

This work was supported by MIC under a grant entitled ”R&D of ICT Priority Technology (JPMI00316)”.

References

[1] I. Alam, K. Sharif, F. Li, Z. Latif, M. M. Karim, S. Biswas, B. Nour, and Y. Wang, “A survey of network virtualization techniques for internet of things using SDN and NFV,” *ACM Computing Surveys (CSUR)*, vol. 53, no. 2, pp. 1–40, 2020.

[2] H. Cao, S. Wu, Y. Hu, Y. Liu, and L. Yang, “A survey of embedding algorithm for virtual network embedding,” *China Communications*, vol. 16, no. 12, pp. 1–33, 2019.

[3] H. Cao, H. Hu, Z. Qu, and L. Yang, “Heuristic solutions of virtual network embedding: A survey,” *China Communications*, vol. 15, no. 3, pp. 186–219, 2018.

[4] N. Kumar and A. Ahmad, “Machine learning-based qos and traffic-aware prediction-assisted dynamic network slicing,” *International Journal of Communication Networks and Distributed Systems*, vol. 28, no. 1, pp. 27–42, 2022.

[5] S. Jošilo and G. Dán, “Joint wireless and edge computing resource management with dynamic network slice selection,” *arXiv preprint arXiv:2001.07964*, 2020.

[6] C. K. Dehury and P. K. Sahoo, “DYVINE: Fitness-based dynamic virtual network embedding in cloud computing,” *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 5, pp. 1029–1045, 2019.

[7] S. Li, M. Y. Saidi, and K. Chen, “Survivable services oriented protection level-aware virtual network embedding,” *Computer Communications*, vol. 152, pp. 34–45, 2020.

[8] K. Inoue, S. Arakawa, S. Imai, T. Katagiri, and M. Murata, “Noise-induced vne method for software-defined infrastructure with uncertain delay behaviors,” *Computer Networks*, vol. 145, pp. 118–127, 2018.

[9] N. A. Levis and D. W. Pfennig, “Plasticity-led evolution: A survey of developmental mechanisms and empirical tests,” *Evolution & Development*, vol. 22, no. 1-2, pp. 71–87, 2020.

[10] G. G. Simpson, “The baldwin effect,” *Evolution*, vol. 7, no. 2, pp. 110–117, 1953.

[11] P. Zhang, H. Yao, M. Li, and Y. Liu, “Virtual network embedding based on modified genetic algorithm,” *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, pp. 481–492, 2019.

[12] J. Lehman and K. O. Stanley, “Evolving a diversity of virtual creatures through novelty search and local competition,” in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, 2011, pp. 211–218.

[13] Z. Yan, J. Ge, Y. Wu, L. Li, and T. Li, “Automatic virtual network embedding: A deep reinforcement learning approach with graph convolutional networks,” *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1040–1057, 2020.

[14] G. Cuccu and F. Gomez, “When novelty is not enough,” in *European Conference on the Applications of Evolutionary Computation*. Springer, 2011, pp. 234–243.

[15] S. Risi, S. D. Vanderbleek, C. E. Hughes, and K. O. Stanley, “How novelty search escapes the deceptive trap of learning to learn,” in *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, 2009, pp. 153–160.

[16] R. Ramamurthy, R. Sifa, M. Lübbering, and C. Bauckhage, “Novelty-guided reinforcement learning via encoded behaviors,” in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[17] S. Doncieux, G. Paolo, A. Laflaquière, and A. Coninx, “Novelty search makes evolvability inevitable,” in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, 2020, pp. 85–93.