A Hybrid Search Optimization Technique Based on Evolutionary Learning in Plants

Deblina Bhattacharjee and Anand Paul(✉)

Department of Computer Science and Engineering, Kyungpook National University, 80 Daehak-ro, Bukgu, Daegu 702701, South Korea
{deblina0210,anand}@knu.ac.kr

Abstract. In this article, we have proposed a search optimization algorithm based on the natural intelligence of biological plants, which has been modelled using a three tier architecture comprising Plant Growth Simulation Algorithm (PGSA), Evolutionary Learning and Reinforcement Learning in each tier respectively. The method combines the heuristic based PGSA along with Evolutionary Learning with an underlying Reinforcement Learning technique where natural selection is used as a feedback. This enables us to achieve a highly optimized algorithm for search that simulates the evolutionary techniques in nature. The proposed method reduces the feasible sets of growth points in each iteration, thereby reducing the required run times of load flow, objective function evaluation, thus reaching the goal state in minimum time and within the desired constraints.

Keywords: Plant growth simulation algorithm · Evolutionary learning · Reinforcement learning · Plant intelligence · Search optimization

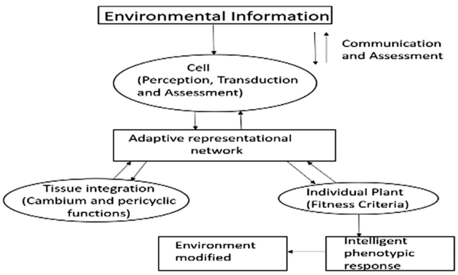
1 Introduction

In this paper, we have discussed the decision making process in plants guided by an underlying learning mechanism which is both evolutionary and adaptive to solve an important search optimization problem, unravelling a form of natural intelligence. Plants gather and continually update diverse information about their surroundings, combining this with internal information about their internal state and making decisions that reconcile their well-being with their environment. The life time goal of a plant is to maximize this fitness which it does by adapting to non-stationary environment by competing vigorously with the surrounding plants for resources, and as the individuals grow along with the competitors the resources get exhausted rapidly. Thus, a search for new resources must be immediately undertaken. This requires the plants to perceive an

This study was supported by the Brain Korea 21 Plus project (SW Human Resource Development Program for Supporting Smart Life) funded by Ministry of Education, School of Computer Science and Engineering, Kyungpook National University, Korea (21A20131600005) and the Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP). [No. 10041145, Self-Organized Software platform (SoSp) for Welfare Devices].

Y. Tan et al. (Eds.): ICSI 2016, Part I, LNCS 9712, pp. 271–279, 2016.
DOI: 10.1007/978-3-319-41000-5_27

information spectrum in a continuous flux and to also maintain a network that can manipulate their own information flow. Different models depicting plant learning has been earlier displayed by [1, 3]. Interestingly, to model the plant learning and cognition many have used the neural network model of Hopfield [2] giving enough proof that plants too learn like animals although they lack a neural structure. This neural like mechanism is brought about by the complex cellular and biochemical reactions leading to wave like signal transductions [7]. In all the models mentioned so far, the plant behavior has been described as either evolutionary (concerned with the very gradual change of the behavior of the entire plant population over a very long time) or adaptively learnt (concerned with a behavioral change in an individual plant in a short span of time). However, the basis to model the type of learning in such systems should be both evolutionary and adaptive learning, as the decisions that are learnt via experiences in individual plants over a short span is passed on across generations to make plants adapt naturally to such rewarding decisions. These decisions are then constantly modified and evaluated based on fitness functions in unfavorable environments. In plants, learning is mainly used for morphological functions like growth and survival. Thus, growth should occur to give rise to an optimal configuration for effective resource use. The selection of these growth points is done via a search technique in plants. In Fig. 1a we have outlined the existing model of plant intelligence and the centre of the plants decision making, i.e. the adaptive representational framework. In our proposed model in Fig. 1b we have shown a three tier architecture that simulates the inner mechanism of this adaptive representational framework by forming a hybrid learning model, applied and solved in context of shoot growth (a search optimization problem seen in plants).



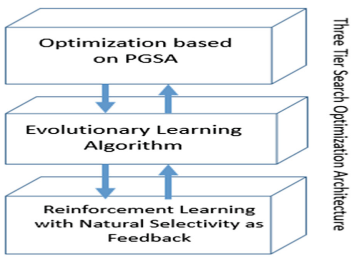


Fig. 1. a. An overview of the existing plant learning and decision making model. **b.** The proposed three tier architecture

PGSA has made huge progress of late to provide methods to find the global optimum of a function search space by overcoming the problem of falling into a local minimum [4, 10]. Initial works of Lindenmayer and Prusinkiewicz in the field of graphics, fractals and the generation of L-System Grammar [5], threw little light on the natural optimization models that simulate plant growth.e.g.: phototropism, gravitropism. Therefore, very little is known about the search optimization algorithms which simulate the natural selectivity based on evolution that is demonstrated by nature while giving globally optimal solutions, except for Genetic Algorithms. Genetic Algorithms have been long attached to evolutionary techniques due to its inherent genetic parameters which form an integral part in evolution. Although, Genetic Algorithms can be applied to unstructured and discrete optimization problems, it searches from a population of points where the objective function coefficient, mutation rate and crossover rate need to be mentioned [10]. Thus, there have been many refined hybrid versions of Genetic Algorithm till date to simulate the evolutionary learning mechanism in nature, in order to modify the weights associated in any learning network. PGSA on the other hand doesn't need any external parameters. Moreover, the objective function and constraint condition is treated separately [4, 10]. It has a search mechanism with ideal direction and randomness balancing properties which are determined by the morphactin concentration in plants and hence, it finds global optimal solution quickly. However, PGSA leads to a large number of growth points and its efficiency needs to be enhanced. To solve this problem, this article proposes a refined PGSA with Evolutionary Learning technique (where the weights are evolved using a hybrid version of Genetic Algorithm and a feedforward reinforcement learning method using natural selectivity as the reward), as a possible approach.

2 Methodology

In this section, we present a detailed overview of our proposed scheme. The proposed scheme as mentioned in Fig. 1(b) has a three tier architecture and is divided into three phases namely (1) PGSA (2) The Evolutionary Learning Algorithm (ELA) (3) The Reinforcement Learning Algorithm. The above Fig. 1a shows the overall learning and decision making model of the plant with an adaptive representational network where all the information processing is done [8]. This representational framework which is refined using our three tier architecture that comprises a hybrid learning network that we have used in context of shoot branching, is shown in detail in Fig. 2.

2.1 Tier 1 - Plant Growth Simulation Algorithm

Based on plant phototropism, the PGSA regards the feasible region of Integer programming as plant growth environment and evaluates the probability on different growth points according to the changes in the objective function [10]. It then grows towards the global optimal solution – light source. Biological experiments prove the following plant growth laws:

First, in the growth process of a plant, when it has more than one node starting at the root; the node with the higher morphactin concentration will have a higher growth probability to grow into a new branch. Second, the morphactin concentrations of the nodes in a plant are not predetermined or fixed but varies based on environmental information of the nodes that depend on their respective positions. While the new nodes appear, the morphactin concentrations of all plant nodes will be freshly allotted as per the new environment.

Mathematical Model for Plant Growth. According to [4, 9], in a plant system, if the length of a trunk is M and the number of growth points is I, then SM = (SM1, SM2,.., SMI), corresponding respectively to the morphactin concentration PM = (PM1,PM2, …PMI). All the growth points which are inferior to the root, that is, those which have a poorer growth function than the initial feasible solution, if there are G growth points such that G lesser than I, these growth points and their corresponding morphological concentration is SM = (SM1,SM2,..SMG) and PM = (PM1,PM2,..PMG). If the length of the branches are m (m < M) and the available growth point numbers are g, these growth points and their concentrations were SM = (Sm1, Sm2,…,Smg) and Pm = (Pm1,Pm2,…,Pmg). We then calculate the morphactin concentration of every point using (1) and (2);

$$P_{Mi} = \frac{f(a) - f(S_{Mi})}{\sum_{i=1}^{G}(f(a) - f(S_{Mi})) + \sum_{i=1}^{g}(f(a) - f(S_{Mi}))} \tag{1}$$

$$P_{mk} = \frac{f(a) - f(M_{mk})}{\sum_{i=1}^{G}(f(a) - f(S_{Mi})) + \sum_{i=1}^{g}(f(a) - f(S_{Mi}))} \tag{2}$$

Where, *a* is the root (initial feasible solution), f (…) is the environmental information of a growth point (objective function).

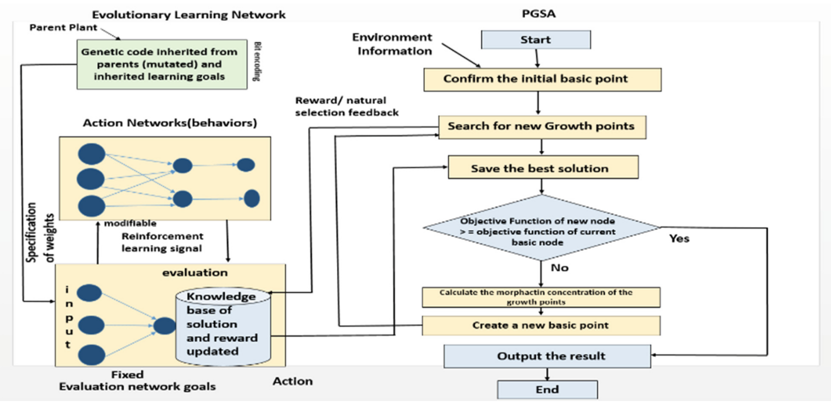


Fig. 2. The detailed model of the proposed three tier architecture based on PGSA and Evolutionary Learning to solve the search optimization problem of optimal shoot growth.

Add all the morphactin concentrations of growing points, we get the result as one. Now, a random number in state space [0, 1] is randomly located which grows out a new generation branch as shown in Fig. 3. This process is repeated until no new branches are generated in the state space. Our approach is to use the learning algorithm with natural selectivity as a feedback to select this random point in the state space thus finding the most optimum growth point in each iteration.

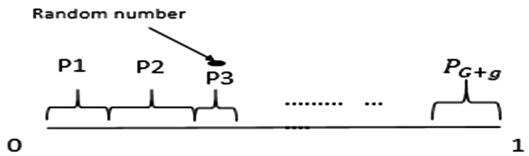


Fig. 3. Morphactin Concentration Space

2.2 Tier 2 - Evolutionary Learning Algorithm (ELA)

The PGSA takes in the sensor inputs and confirms the initial basic point (root). Thereafter, it iterates to search for new growth points as per the mathematical model discussed above. The control then goes to the learning network which takes the state from the recent environment as the input and instructs the PGSA to perform the new branch growth action from the naturally selected growth point based on the reward from the current environment. The problem of training the weights are solved by specifying not only the inherited behaviors but also the inherited goals that are used to facilitate learning. This is done by constructing a genetic code specifying two major components (1) a set of initial values for the weights of the action network that maps from sensory input to what action needs to be taken and (2) the evaluation network that maps from the inputs to a fitness value of the current decision. The weights in the learning network are trained using a modified genetic algorithm model with reinforcement. The initial weights of the action network are specified genetically. However, they are adjusted over time by a reinforcement learning mechanism that uses natural selectivity as its reward.

Input: Plant p , Seed s, EnvironmentalVariables I_t, GeneticCode gco
Output: Evaluation E_t, ReinforcementSignal R_t , NewAction X_t
if $age_s = 0$ **then**
$gco_s = gco_p$
$cgco_s = Crossover(gco_s, 1)$
$mgco_s = Mutate(cgco_s, mutation_{probabilty})$
$[W]_{mgco} = Translate(mgco, [E], [A])$
end if
if $age_s \neq 0$ **then**
$E_t = Evaluate(I_t)$
if $age_p = 0$ **then**
$X_t = Learn(Reinforce(I_t))$
Output X_t
else
$R_t = E_t - E_{t-1}$
$Reinforce(Update([A], X_{t-1}, I_{t-1})$
end if
end if
end

In the above ELA, a parent plant P's genetic code is copied to the genetic code of seed S at birth (t = 0), followed by crossover and mutation of the bit encoded genetic bits. The genetic code is now translated into weights for S's evaluation network and the initial weights for S's action network. At time t, when there is a living plant P and a new current input vector from the environment I_t, the plant starts at the evaluation network, propagating I_t through the evaluation network to produce a scalar evaluation E_t. If this is P's day of birth, the algorithm invokes the reinforcement algorithm to generate a new action (branch) based on I_t otherwise, a reinforcement signal R_t is produced by comparison with the previous evaluation. The reinforcement algorithm is invoked to update the action network with respect to the previous action and input.

Here a specific reinforcement function suggested by Sutton has been used for learning to proceed [6]. However, the algorithm has been modified as described in Sect. 2.3. Learning is mainly brought about by the inherited evaluation function which converts long time scale feedback (natural selection over lifetimes) into short time scale feedback (reinforcement signals over moments). Thus we see that randomization is regressed here.

2.3 Tier 3 - Reinforcement Learning Algorithm

In this Reinforcement algorithm, the supervised training like back propagation network is refined which provides a memory model for the plant. This makes the plant rehearse the same positively rewarding decision into its knowledge base as shown in Fig. 2.

Input: ReinforcementNetwork $[R]_n$ with input dimensionality n and output dimensionality m, and a ReinforcementFunction $f(R^n, R^n) \to r$.
Output: BinaryOutputVector *o*, ReinforcementNetwork $[R]_m$

Initialize $t = 0$, Receive vector $i_t \in R^n$.

repeat

r = *ComputeReinforcement f(*i_t, i_{t-1}*).*

GenerateOutputErrors( )

begin

if $r > 0$ **then**

$e_j = (o_j - s_j)\, s_j\, (1 - s_j)$

else

$e_j = (1 - o_j - s_j)\, s_j\, (1 - s_j)$

end if

end *Procedure{GenerateOutputErrors( )}*

BackpropogateErrors(e_j*)*

UpdateWeights( )

begin

if $r \geq 0$ **then**

$\eta = \eta_+$

else

$\eta = \eta_-$

end if

$\Delta w_{jk} = \eta e_k s_j,$

end *Procedure{UpdateWeights ( )}*

s_j = *ForwardPropogate*($[R]_n$)

GenerateTemporaryOutput(o^0)

Perform action associated with o.

t=t+1

end

---

3 Implementation and Analysis

For the artificial simulation of this learnt behavior of plants in context of search optimization, we considered a heterogeneously unfavorable environment, i.e. almost all input parameters are non-stationary and unknown to the plant. The simulations were performed in a rectangular cell of length $l = d\ x\ n$, n is the number of seeds that are subjected to branch out into shoots growing into plants according to the varying environmental cues and d is the separation between two adjacent seeds. Through the simulations, we calculated the shoot growth rate, the delay in finding the global solution, the direction of root growth for a resource optimal configuration and the energy cost or resource use that the plants foraged during the growth simulation. We have analyzed the results across a sample of 10 generations. Initially, the axiom growth point searches are fed to PGSA alone and then to our proposed scheme. In the Evolutionary Learning Algorithm (ELA), the population of growth points show regressive behavior for the initial few iterations. Thereafter, the algorithm shows a 3 point convergence of the population space, showing the fastest approach. For just the PGSA mechanism, the population seemed to behave in a much similar way as ELA except it was comparatively slower and resource costly. The comparison of the two models has been shown in Fig. 4(a) and (b). While Fig. 4(a), shows the net energy cost or total number of growth points in unfavorable environments for both the models, Fig. 4(b) shows the rate of shoot growth across 10 generations as studied. Clearly, the number of growth points or Energy Cost of applying the proposed model by the plant turned out to be optimal. Thus, the major problem of too many growth points in PGSA was solved by the proposed scheme. The comparative successes in growth point search in both the models has been analyzed and seen to be almost similar as seen in Fig. 5(a). Therefore, while the two models are successful in locating the optimal morphactin concentration in the shoot and in finding the optimal growth point, yet the proposed model is clearly more cost efficient and fast as compared to the existing model (Fig. 5a and b). Thus, this leads to optimal solutions of search problems within desired constraints.

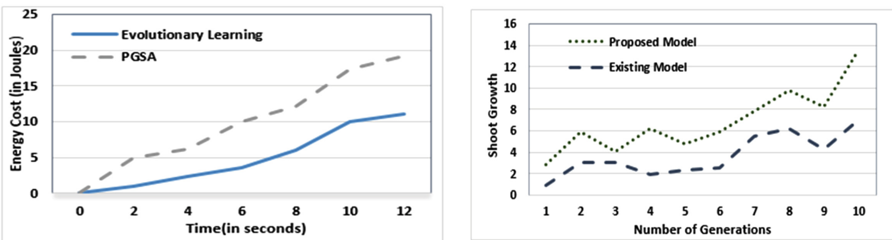


Fig. 4. a. Analysis of the 2 models 1: The proposed model labeled as Evolutionary Learning 2: Plant Growth Simulation Algorithm with respect to Energy Cost (No. of growth points) vs Time. **b.** Analysis of shoot growth over 10 generations using both the models. (Color figure online)

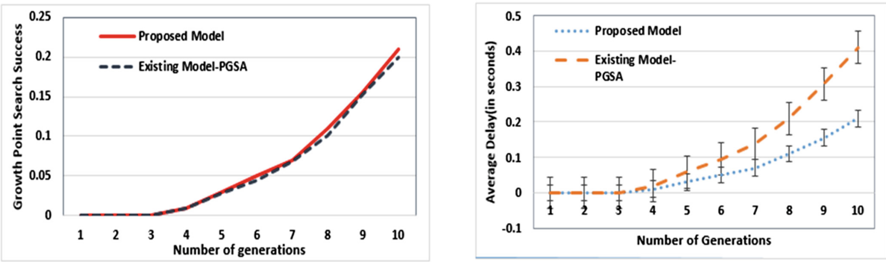


Fig. 5. **a.** The comparative analysis from the simulation of the two models with respect to Growth Point Search Success vs Number of generations. It shows that both the models of PGSA and Evolutionary Learning give optimal search solutions but PGSA is slow and energy costly. **b.** Analysis of Average Delay vs. Number of Generations in both the proposed and existing model. (Color figure online)

4 Conclusion

In this paper, a new bionic random search algorithm has been proposed that makes use of the objective function's value as an input to the learning model while simulating a plant's phototropism. The learner evaluates the various growth points and then directs the new branch to grow from the most optimal growth point, by giving a simulated natural selectivity feedback. Therefore, the proposed algorithm can exactly simulate the natural evolutionary process by combining it with learning algorithms to give fast globally optimal solutions of multimode functions. Therefore, the hybrid approach of learning (that evolves through generations) and PGSA, proves to be efficient and optimal to solve complex search optimization problems under desired constraints and time. Thus, this algorithm when applied to search optimization problems, can effectively reduce load factor, search space as well as help the system to evolve on its own via information processing, maintaining a memory base and ultimately learning through its experiences. The applications range from optimal network architecture, modelling of hydro-plants motivated from water absorption in trees to architectural designs of robust structures etc. The future scope of this study will incorporate the ELA scheme with respect to Baldwin Effect, Shielding Effect and Genetic Drift which were not covered in the scope of this paper.

References

1. Bose, I., Karmakar, R.: Simple models of plant learning and memory. Phys. Scr. **2003** (T106), 1–10 (2003)
2. Hopfield, J.J.: Neural Networks and Physical Systems with emergent collective computational abilities. Proc. Natl. Acad. Sci. U.S.A. **79**, 2554–2558 (1982)
3. Inoue, J., Chakrabarti, B.K.: Competition between ferro-retrieval and antiferro orders in Hopfield- like network model plant intelligence. Physica A **346**, 58–67 (2004)

4. Tong, L., Zhong-tuo, W.: Application of plant growth simulation algorithm on solving facility location problem. Syst. Eng. Theory Pract. **28**(12), 107–115 (2008)
5. Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants. Springer, New York (1990)
6. Sutton, R.S.: Temporal credit assignment in reinforcement learning. Ph.D. Thesis, Department of Computer and Information Science, University of Massachusetts (1984)
7. Trewavas, A.J.: Calcium makes waves. Plant Physiol. **120**, 1–6 (1999)
8. Trewavas, A.J.: Green plants as intelligent organisms. Trends Plant Sci. **10**(9), 413–419 (2005)
9. Xu, L., Tao, M., Ming, H.: A hybrid Algorithm based on genetic algorithm and plant growth simulation. In: International Conference on Measurement, Information and Control (MIC), vol. 1, pp. 445–448 (2012)
10. Jing, Y., Wang Zong F.: A refined plant growth simulation algorithm for distribution network configuration. In: IEEE International Conference on Intelligent Computing and Intelligent Systems, vol. 1, pp. 357–361 (2009)