On stochastic evolving algorithms

Iztok Fister Jr.
iztok.fister1@um.si
University of Maribor
Maribor, Slovenia

Iztok Fister
iztok.fister@um.si
University of Maribor
Maribor, Slovenia

ABSTRACT

Abstract: This pioneering study tries to break the wall of the question, how to develop algorithms capable of self-improving. The foundation of this work is the extended model of the human mind, while the information flow between components is inspired by molecular genetics. As a result, the proposed evolving algorithms consist of complex rules and stochastic processing, while the preliminary results also revealed enormous potential for the future.

KEYWORDS

evolving algorithms, cognitive model, stochastic algorithms

ACM Reference Format:
Iztok Fister Jr. and Iztok Fister. 2022. On stochastic evolving algorithms. In *Genetic and Evolutionary Computation Conference Companion (GECCO '22 Companion), July 9–13, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3520304.3528780

1 INTRODUCTION

Human intelligence is not an innate characteristic of the brain as proposed by classical symbolic Artificial Intelligence (AI) [Hiesinger 2021]. Indeed, intelligence is shown in the interaction of human with their environment.The main critic of the classical symbolic AI Brooks [1986] asserted that the premises, on which this AI was founded were false and therefore alternative approaches needed to be searched for.

The Evolutionary Algorithms (AEs) [Eiben and Smith 2015b] that has emerged a decade later also did not meet all the demands of the AI domains. On the one hand, an estimation of individuals is limited to the current generation and not to the quality of those during a long-term run, while on the other, the behavior of the individual is not evaluated directly in an environment [Eiben and Smith 2015a]. Last but not least, each algorithm running on a digital computer is rigid in the sense that it does not allow any evolvability [Roitblat 2020].

Nowadays, some programming languages offer an innovative approach to repairing the program code during the running of the code itself. The process is called hot code reloading or hot code swapping [Appavoo et al. 2003]. In practice, this means that we change a part of a module with new or replaced/updated code, and the code is then immediately reloaded. Interestingly, only a small number of programming languages support hot code swapping natively, as for example: Lisp, Smalltalk, Erlang, and Elixir. Some of the other languages do not support this feature natively, but offer specific libraries capable of ensuring this functionality.

In order to accommodate evolving algorithms enabling self-improvement, we searched for an appropriate computational model. A standard cognitive model of the fixed structure of the mind (also called cognitive architecture) has been proposed by Laird et al. [2017], and slightly extended version of this was chosen to serve as our computational model. According to our extended model (Fig. 1, cognitive architecture consist of Short-Term Memory (STM), and procedural, declarative, and goal-oriented Long-Term Memory (LTM). Thus, the procedural LTM refers to the production model consisting of numerous IF. . .THEN rules, the declarative LTM to connectionist model (also called Neural Network (NN)), while the goal-oriented to a sequence of actions that is the result of planning. The STM serves as the working memory devoted for perceiving senses from environment, and for launching the appropriate mental motor activities leading to actions, with which human change their environment. Obviously, the appropriate actions are selected on

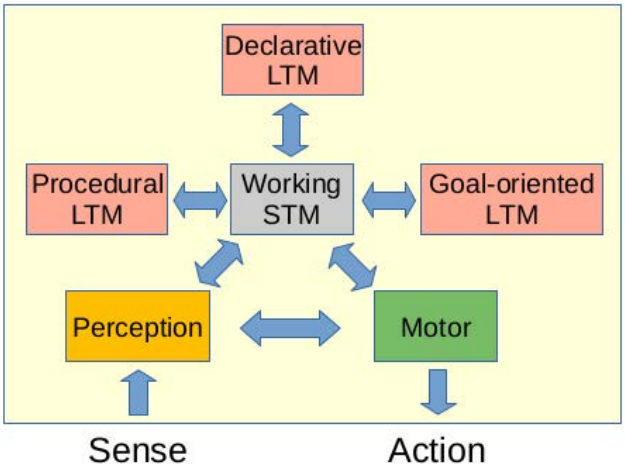

Figure 1: Extended model of the mind.

basis of decision-making process in which both types of LTMs are included. When the result of decision-making is a plan describing how to achieved the specific goal, the plan persists in the goal-oriented LTM until this is achieved or it is unreachable.

The purpose of the present study is to outline an architecture of the evolving stochastic algorithms based on the extended model of the mind. Obviously, the cognitive model is only one side of the coin. The another side of this presents the answer to the question how information flow among the mentioned components of the model. We found the answer on the foundations of the molecular genetics. The results of synthesis both domains lead us to the concept of an

GECCO '22 Companion, July 9–13, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9268-6/22/07.
https://doi.org/10.1145/3520304.3528780

evolving algorithms enabling self-improvement. In this preliminary work, the concept was applied for global optimization, while the results showed its potential for future development.

2 RESEARCH METHODS

Special-purposed algorithms performed well when applied to well-structured known problems [Roitblat 2020]. For instance, EAs present strong tools for optimization over a known search space. Unfortunately, they suffer from an inability to invent something from a new perspective, as for example: to design unforeseen structures, to formulate new scientific paradigms, or to create new representations [Roitblat 2020]. In general, the algorithms are deficient in self-improvement.

We have searched an inspiration for self-improvement algorithms in operations of human brain that enable mind. The brain consists of a huge number of nerve cells called neurons and enables the mind to function. Neurons are electrical excitable cells that communicate among each other using specialized connections called synapses. On the molecular level, the hereditary information for all living beings is made of DeoxyriboNucleic Acid (DNA) [Alberts et al. 2002]. The molecule is placed inside the nucleus, and share information about how to react in certain situation. The DNA information are transferred outside the nucleus into the cell by messenger RiboNucleic Acid (RNA), and influence the process of synthesizing the proteins that perform most jobs in the cells.

In general, there are two processes supporting transmission of information from the nucleus to the cells: transcription and translation. Transcription involves copying a segment of DNA into a messenger RNA. Translation, on the other hand, ensures that the encoded information for synthesizing the proteins is decoded using the universal genetic code. Actually, information in molecular genetics passes from genes to proteins in the following form, baeck1996evolutionary:

$$\text{DNA} \rightarrow \text{RNA} \rightarrow \text{Protein}.$$

As each human call, neurons also contain the nucleus with the same heredity information written in DNA. In neuron cells, there is not a plan for how to connect neurons to each other. Actually, algorithmic information for developing brain is contained in genes (regions of DNA). This means that neuron's connections can be reconfigured under the new algorithmic information from DNA. In this sense, genes allow growth and guide self-assembling, thus enabling brain to evolve [Hiesinger 2021].

The concept of self-assembling brain can be used also by development of evolving algorithms. Indeed, the concept of those algorithms can be gathered in the following information flow:

$$\text{Complex rules} \rightarrow \text{Encoding} \rightarrow \text{Stochastic processing},$$

where 'Complex rules' refer to decision-making (inspired by DNA), 'Encoding' to the encoded sequence of algorithmic information (inspired by RNA) for evolving the 'Stochastic processing' goal-oriented component (inspired by Protein). Here too, processes of transcription and translation ensure that the clues from the decision-making component are appropriately encoded in messages, which are then decoded into an evolving goal-oriented component (i.e., stochastic algorithm). This information flow directly complies with extended model of mind illustrated in Fig. 1.

In more general form, the evolving algorithms are defined as:

Evolving algorithm=Complex rules+Stochastic processing

A scheme of the evolving algorithms is illustrated in Fig. 2, from which it can be seen that. just as a DNA influences the connecting

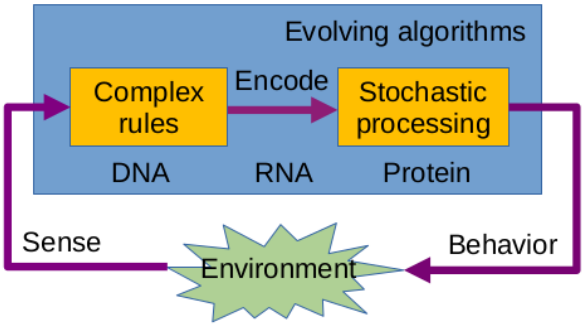

Figure 2: Scheme of the evolving algorithm.

of the synapses of neurons according to the new algorithmic information, so a stochastic algorithm also improves itself according to the new information obtained from the Complex rules.

Obviously, the stochastic processing goal-oriented component can be implemented using different stochastic methods, e.g., EAs, when IF...THEN rules are closer to the problem solving by the procedural LTM, or Self-Organizing Map (SOM) networks, when the problem is more suitable for solving by the declarative LTM subsystem. The decision-making component can also be implemented by various mathods, e.g., the First-Order Logic (FOL) rules, when a symbolic logic is applied, or Reinforcement Learning (RL), when long-term rewarding for solution quality is preferred.

3 TOWARDS STOCHASTIC EVOLVING ALGORITHM

In this preliminary study, the concept of the proposed evolving algorithms was applied to the global optimization problem, where three benchmarks issued for the CEC Competition on Real-Parameter Optimization in the last decade were observed (i.e., CEC'13, CEC'14, and CEC'17). Consequently, three of the favourites expected to achieve excellent results for some of the benchmarks are taken into consideration here (i.e., Shade using linear population size reduction (lShade) [Tanabe and Fukunaga 2014], Improved lShade (iL-Shade) [Brest et al. 2016], and Single Objective Real-Parameter Optimization (jSO) [Brest et al. 2017]). The goal of our study was to find such an evolving algorithm that would be able to achieve the best results across all three benchmarks.

In summary, there a set of three algorithms *procLTM* each with its own strategy for exploring the search space, which can be described formally as:

$$procLTM = \{\text{lShade,iL-Shade,jSO}\}.$$

Each of the strategies is hard-coded into a particular executable program and stored into procedural LTM. Because all of the programs implement adaptive algorithms searching for the best parameter setting is not necessary.

In terms of evolving algorithms, each of the strategies represent goal-oriented component (i.e., programs) that need to be loaded

into the working STM using the hot code swapping technique. As a decision-making component, the RL technique is applied that selects the stochastic DE program according to the decoding scheme illustrated in Table 1. As can be seen from the table, the coding

Table 1: Evolving algorithm's coding scheme.

Code	Strategy
A	lShade
B	iL-Shade
C	jSO
P	PARA n

scheme makes possible the encoding of the particular strategies (code A-C) and adaption of the stochastic algorithm (code P).

In the proposed evolving algorithm named RLDE, two optimization processes are simultaneously launched:

- A control process representing decision-making by the problem solving at higher level (Algorithm 1),
- An optimization process implementing the goal-oriented Shade algorithm at lower level (Algorithm 2).

Both algorithms interact with each other by posting and receiving messages. As can be seen from Algorithm 1, the complex rules component is a simple implementation of RL that rewards the better strategies on the basis of a Q-value function [Sutton and Barto 2018]. The task of the component is to determine action with ϵ-greedy selection, and to encode it into a variable *stream*, with which it affects the evolving stochastic component. After completion

Algorithm 1 Pseudo-code of the decision-making component.

```
1:  procedure COMPLEX_RULES(serverID, S)
2:      Init(∀s ∈ S⁺ ∧ a ∈ A(s) : Q(s, a) = 0; N(a) = 0);
3:      do
4:          a = ϵ-greedy(Q(S, A));
5:          stream = Encode(a);
6:          Post(serverID, stream);
7:          Receive(serverID, R, S′);
8:          Q(S, A) = Q(S, A) + α(R + γ · max_a Q(S′, a) − Q(S, A))
9:          S = S′
10:     while true
11: end procedure
```

of stochastic processing, the learned action-value function Q is updated. We should notice that the state S actually addresses the current DE population.

The main part of Algorithm 2 is creating the context for the problem solving. This consists of a description of the benchmark functions (i.e., *problem*) and internal data structures needed for operation of the Shade algorithms. The algorithm contains two **for** statements: The outer one controls the termination condition expressed as the maximum number of generations, while the inner one directs the modification of each particular individual in particular generations. The task of the stochastic algorithm is to manage the population of individuals from the population S according to clues received by the encoded *stream* ('Decode' function). If changing the strategy is demanded, the new program is loaded from the procedural LTM ('Load' function). Using hot code swapping, the function 'Load' ensures that only one copy of the program is located in the working STM. Then, the program is adapted according to the 'PARA n' clues in the 'Evolve' function, where the parameter n either includes or excludes some term in the DE mutation strategy. As a result, the basic program loaded from the procedural LTM can be evolved during the evolution cycle. Let us notice, the function 'Execute' implements the adaptation of the DE mutation strategy, the fitness function evaluation, and the DE one-to-one selection operator. The result of the function is double: the reward and the new state s'.

Algorithm 2 Pseudo-code of the goal-oriented component.

```
1:  procedure RLDE(serverID, problem)
2:      context = CreateContext(problem);
3:      processID=Create(serverID, problem);
4:      cur_strategy = NULL;
5:      for all t ∈ T do
6:          for all s ∈ S do
7:              Receive(processID, stream);
8:              strategy = Decode(stream);
9:              if strategy ≠ cur_strategy then
10:                 prog = Load(LTM, strategy);
11:                 param = Evolve(strategy, stream);
12:                 cur_strategy = strategy;
13:             end if
14:             ⟨r, s′⟩ = Execute(context, param, s);
15:             Post(serverID, r, s′);
16:         end for
17:     end for
18:     PrintSolution(context);
19:     Delete(processID);
20:     DeleteContext(context);
21: end procedure
```

4 EXPERIMENTS AND RESULTS

The following hypothesis guided our experimental work: "The evolving algorithm for global optimization is able to achieve the best results by solving all three benchmarks issued for CEC Special Session/Competition on Real-Parameter Optimization during the last decade.". In line with this, three RLDE versions were developed that were distinguished from each other by the initialization context, i.e., the RLDE-0 initialization adaptation scheme from the lShade, the RLDE-1 from the iL-Shade, and the RLDE-2 from the jSO algorithms. Detailed information about the initialization schemes is beyond the scope of this paper.

To prove the posited hypothesis, the results of the aforementioned versions of the RLDE algorithm were compared with the following original, self-adaptive, and Shade-based algorithms: original DE [Storn and Price 1997], Self-adaptive DE [Qin and Suganthan 2005], self-adaptive jDE [Brest et al. 2006], original Shade [Tanabe and Fukunaga 2013], Shade using linear population size reduction (lShade) [Tanabe and Fukunaga 2014], improved lShade (iL-Shade) [Brest et al. 2016], Single Objective real-parameter optimization (jSO) [Brest et al. 2017], and lShade with Real-based Selection Pressure (lShade-RSP) [Stanovov et al. 2018]. In summary, there are eleven DE variants in the study.

Characteristics of the benchmarks are presented in Table 1, where N denotes the number of the functions in benchmark suite. Let us mention that only functions of dimensions $D = 10$ were

Table 2: Characteristics of the CEC benchmark suites.

Type	CEC'13	CEC'14	CEC'17
Unimodal	5	3	3
Multi-modal	15	13	7
Composition	8	8	10
Hybrid	n/a	6	10
N	28	39	30

taken into consideration. The quality of the results was estimated by means of Friedman's non-parametric tests, where the results achieved according to the minimum, maximum, average, median and standard deviation values obtained by optimization of each of the N functions were assembled into classifiers of size $N \times 5$. However, the average results are obtained after 51 independent runs. The results of the Friedman's tests are ranks denoting the strength of a given algorithm compared to the others. In our study, the higher the rank, the better the algorithm.

The results of the comparative study obtained by optimizing all three benchmarks containing the functions of dimension $D = 10$ are illustrated in Fig. 3 as a box-plot diagram. As can be seen

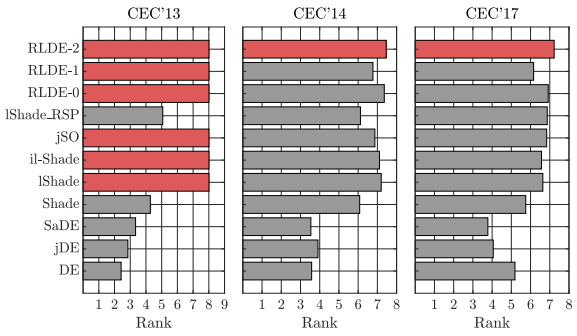

Figure 3: Results of Friedman's tests for $D = 10$.

from the figure, the RLDE-2 using the jSO initialization scheme outperforms the results of the other algorithms for all benchmarks. Thus, the hypothesis set at the beginning of the section is validated.

5 CONCLUSION

The aim of this preliminary study was to introduce the concept of evolving algorithms that could be capable of self-improving. The concept is founded on an extended model of the human mind taken from cognitive psychology. It consists of two components, i.e., complex rules and stochastic processing. Both components are connected via decoded messages similar to the information processing in molecular genetics. In the sense of the cognitive model, the first component refers to the decision-making processing of the human mind, while the second is purely goal-oriented.

In our illustrative example, we proposed an evolving algorithm for global optimization, where the decision-making component is implemented using the RL that choices between three variants of lShade algorithms using ϵ-greedy selection. The results of the proposed RLDE variants were promising when compared with the results of the other eight DE algorithms.

Future research could be extended to address the feature selection problem, where the search space is also capable of exploring three strategies: wrapper, filter, and embedded. However, even more promising is the possibility of applying the evolving algorithm to problems that are closer to being solved with the declarative LTM subsystem of the human mind, like SOM, LSTM, and so forth.

ACKNOWLEDGEMENTS

Iztok Fister Jr. is grateful the Slovenian Research Agency for the financial support under Research Core Funding No. P2-0057. Iztok Fister thanks the Slovenian Research Agency for the financial support under Research Core Funding No. P2-0042 - Digital twin.

REFERENCES

B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J.D. Watson. 2002. *Molecular Biology of the Cell* (4th ed.). Garland.

Jonathan Appavoo, Kevin Hui, Craig AN Soules, Robert W Wisniewski, Dilma M Da Silva, Orran Krieger, Marc A Auslander, DJ Edelsohn, Benjamin Gamsa, Gregory R Ganger, et al. 2003. Enabling autonomic behavior in systems software with hot swapping. *IBM systems journal* 42, 1 (2003), 60–76.

Janez Brest, Sašo Greiner, Borko Bošković, Marjan Mernik, and Viljem Žumer. 2006. Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *Trans. Evol. Comp* 10, 6 (dec 2006), 646–657. https://doi.org/10.1109/TEVC.2006.872133

Janez Brest, Mirjam Sepesy Maučec, and Borko Bošković. 2016. IL-SHADE: Improved L-SHADE algorithm for single objective real-parameter optimization. *2016 IEEE Congress on Evolutionary Computation, CEC 2016* (2016), 1188–1195. https://doi.org/10.1109/CEC.2016.7743922

Janez Brest, Mirjam Sepesy Maučec, and Borko Bošković. 2017. Single Objective Real-Parameter Optimization: Algorithm jSO. In *IEEE Congress on Evolutionary Computation (CEC)*. IEEE Computational Intelligence Society Institute of Electrical and Electronics Engineers, 1311–1318.

Rodney A. Brooks. 1986. A robust layered control system for a mobile robot. *IEEE J. Robotics Autom.* 2 (1986), 14–23.

A. E. Eiben and James E. Smith. 2015a. From evolutionary computation to the evolution of things. *Nature* 521 (2015), 476–482. Issue 7553. https://doi.org/10.1038/nature14544

A. E. Eiben and James E. Smith. 2015b. *Introduction to Evolutionary Computing* (2nd ed.). Springer Publishing Company, Incorporated.

P.R. Hiesinger. 2021. *The Self-Assembling Brain: How Neural Networks Grow Smarter*. Princeton University Press. https://books.google.si/books?id=FkgHEAAAQBAJ

John E. Laird, Christian Lebiere, and Paul S. Rosenbloom. 2017. A Standard Model of the Mind: Toward a Common Computational Framework across Artificial Intelligence, Cognitive Science, Neuroscience, and Robotics. *AI Magazine* 38, 4 (Dec. 2017), 13. https://doi.org/10.1609/aimag.v38i4.2744

A.K. Qin and P.N. Suganthan. 2005. Self-adaptive differential evolution algorithm for numerical optimization. In *2005 IEEE Congress on Evolutionary Computation*, Vol. 2. 1785–1791. https://doi.org/10.1109/CEC.2005.1554904

Herbert L. Roitblat. 2020. *Algorithms Are Not Enough: Creating General Artificial Intelligence*. The MIT Press, Cambridge.

Vladimir Stanovov, Shakhnaz Akhmedova, and Eugene Semenkin. 2018. LSHADE Algorithm with Rank-Based Selective Pressure Strategy for Solving CEC 2017 Benchmark Problems. *2018 IEEE Congress on Evolutionary Computation, CEC 2018 - Proceedings* (2018). https://doi.org/10.1109/CEC.2018.8477977

Rainer Storn and Kenneth Price. 1997. Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J. of Global Optimization* 11, 4 (dec 1997), 341–359. https://doi.org/10.1023/A:1008202821328

Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction* (second ed.). The MIT Press. http://incompleteideas.net/book/the-book-2nd.html

Ryoji Tanabe and Alex Fukunaga. 2013. Evaluating the performance of SHADE on CEC 2013 benchmark problems. *2013 IEEE Congress on Evolutionary Computation, CEC 2013* 1 (2013), 1952–1959. https://doi.org/10.1109/CEC.2013.6557798

Ryoji Tanabe and Alex S Fukunaga. 2014. Improving the search performance of SHADE using linear population size reduction. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2014, Beijing, China, July 6-11, 2014*. 1658–1665. https://doi.org/10.1109/CEC.2014.6900380