Analysis of the Exploration-Exploitation Dilemma in Neutral Problems with Evolutionary Algorithms

Paolo Pagliuca paolo.pagliuca@istc.cnr.it
Institute of Cognitive Sciences and Technologies (ISTC)
National Research Council (CNR)
Via Giandomenico Romagnosi 18A, 00196, Rome, Italy

Corresponding Author: Paolo Pagliuca

Abstract

Finding a compromise between the exploration of new opportunities that could yield excellent performances and the exploitation of existing solutions through local refinements represents a major challenge in different sectors. In fact, although the search for improved solutions may be costly and time consuming in the short term, its effects could have a big impact in the long term. Conversely, exploitation is likely to be beneficial in the short term, but might have a catastrophic effect in the long term. With respect to evolutionary computation, several approaches attempt to address the issue from different perspectives. In this work, we analyze the exploration-exploitation dilemma in problems where neutrality — the condition according to which the search space consists of vast areas accessible through mutations that do not jeopardize the survival chances — is a distinctive feature. Specifically, we present the results achieved in two benchmark problems: (i) function optimization and (ii) 5-bit parity. Moreover, a novel method mixing exploration and exploitation, called SSSHC*, is introduced and compared with two other algorithms. The experiments reported in this paper indicate that the SSSHC* finds remarkably better solutions than other methods in the optimization of function tests and is competitive with respect to the parity problem. Overall, the outcomes show how exploration and exploitation can be effectively combined, but the strategy for doing so is task-dependent.

Keywords: Exploration, Exploitation, Neutrality, Evolutionary algorithms.

1. INTRODUCTION

The exploration-exploitation dilemma represents a paramount concept in decision-making [1–3]. Specifically, the problem consists in finding a trade-off between exploration (i.e., the search for new unseen possibilities) and exploitation (i.e., the refinement of existing solutions). While the latter might be advantageous in the short term, these modifications could have a detrimental effect in the long term, especially when external conditions change [4]. Consequently, balancing effectively exploration and exploitation represents a challenging task. Furthermore, discovering techniques

Citation: Paolo Pagliuca. Analysis of the Exploration-Exploitation Dilemma in Neutral Problems with Evolutionary Algorithms. Journal of Artificial Intelligence and Autonomous Intelligence. Research. 2024; 1(2):8. Pages: 110-121

that efficiently combine exploration and exploitation could be beneficial in widespread domains like optimization and machine learning [5, 6].

The exploration-exploitation dilemma has been largely studied in the field of evolutionary computation by using techniques like Reinforcement Learning (RL) and Evolutionary Algorithms (EAs). For example, the authors in [7], analyzed the capability of various learning policies to deal with the exploration-exploitation dilemma in a collective foraging task performed by a swarm of Khepera robots [8]. In [9], a novel adaptive framework called AdaZero was proposed to automatically determine whether to explore or to exploit. The authors demonstrated how AdaZero bests state-of-the-art RL algorithms in several Atari and MuJoCo [10], problems. The exploration-exploitation trade-off has been investigated also in maze navigation scenarios [11, 12] and mountain car problem [13].

Regarding EAs, different methods combining exploration and exploitation have been proposed in literature (for a review, see [14]) among which the family of Memetic Algorihtms (MAs) [15], is worth mentioning. Specifically, MAs combine exploration at a population level with a local refinement process of current solutions (i.e., exploitation). The aim is to enhance the solutions — termed also as genotypes — discovered so far, hence minimizing the premature convergence problem. Successful examples of MAs can be found in the context of multi-objective optimization [16], combinatorial optimization [17], hurdle problems [18], and swarm robotics [19]. In [20], a novel MA termed Stochastic Steady State with Hill-Climbing (SSSHC) bested a pure Evolutionary Algorithm, the Stochastic Steady State (SSS) [21], in several benchmark tasks under particular circumstances. These examples demonstrate the existence of conditions fostering the discovery of an effective exploration-exploitation trade-off.

In this article, we delve into the exploration-exploitation dilemma in two different domains: (i) a function optimization scenario and (ii) the 5-bit parity problem. Both tasks are characterized by high neutrality, the situation in which the search space consists of vast regions that can be accessed through mutations not jeopardizing the survival possibilities [22]. Moreover, we present a novel method, called SSSHC*, which represents a modification of the SSSHC algorithm introduced in [20]. The two techniques differ with regard to the local refinement process (i.e., exploitation), which consists of altering a single gene in the current solution. The idea is to leverage neutral theory [22], to effectively combine exploration and exploitation so as to tackle premature convergence, especially in the function optimization scenario (see the considerations reported in [20]). The comparison of the SSSHC*, SSSHC and SSS methods in the two experimental settings reveals that the SSSHC* outpeforms the other techniques in optimizing test functions. Conversely, SSSHC is superior to SSS and SSSHC* regarding the parity task. Therefore, the mix of exploration and exploitation is beneficial in neutral problems, but the effectiveness of the local refinement strategy is affected by the peculiarities of the considered task.

This paper provides the following contributions:

- a thorough investigation of the exploration-exploitation dilemma in neutral problems is presented;
- a novel algorithm, called SSSHC*, is introduced;
- an effective exploration-exploitation trade-off has been found in two benchmark domains;

- the nature of the exploitation phase is task-dependent.

The article is structured as follows: Section 2 describes the two experimental problems and the algorithms. Section 3 reports the analysis of the outcomes. The most relevant findings are discussed in Section 4. Lastly, final remarks are drawn in Section 5.

2. MATERIALS AND METHODS

In this section, we describe the experimental scenarios and the algorithms used to investigate the exploration-exploitation dilemma.

2.1 Function Optimization

Function optimization is a widely used domain to assess the properties of EAs and other optimization techniques [23]. The general formulation of the problem can be stated as follows: consider a vector $x = [x_1, ..., x_n]$ of length n and an objective function F. The goal is to solve the following minimization problem:

$$\min F(x)$$

In this study, four test functions have been considered: (i) the Griewank function [24], (ii) the Rastrigin function [25], (iii) the Rosenbrock function [26], and (iv) the Sphere function. The functions are mathematically defined as follows:

$$F_{griewank} = 1 + \frac{1}{4000} \sum_{i=1}^{n} {x_i}^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}})$$

$$F_{rastrigin} = \sum_{i=1}^{n} ({x_i}^2 - 10\cos(2\pi x_i) + 10)$$

$$F_{rosenbrock} = \sum_{i=1}^{n-1} (100({x_{i+1}} - {x_i}^2)^2 + (x_i - 1)^2)$$

$$F_{sphere} = \sum_{i=1}^{n} {x_i}^2$$

A two-dimensional (2D) plot of the test functions is shown in FIGURE 1. The green entry represents the optimum value for each function.

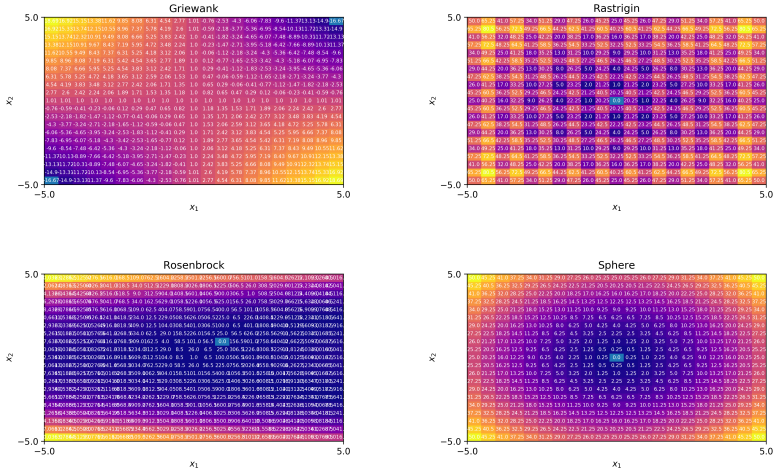

Figure 1: 2D plot of the considered test functions with a two-dimensional input $x = [x_1, x_2]$. The entry highlighted in green corresponds to the minimum value.

Table 1: Parameters used to synthesize solutions for the function optimization scenario.

Parameter	Value
$numReplications$	20
$numEvaluations$	10^6
$popSize$	20
$mutRate$	0.05
n	50

In this work, we delve into the case in which the considered test functions have to be optimized by taking into account vectors x (i.e., genotypes) of length $n = 50$, which constitutes a relatively difficult problem compared to the analysis reported in [20]. TABLE 1 contains the full list of parameters, which have been derived from [20].

2.2 5-Bit Parity

The second scenario involves the synthesis of a digital circuit able to produce an output equal to 1 when the 5-bit input sequence contains an even number of 1-bits. In particular, in this study we consider the evolution of digital circuits composed by many logic gates, each one performing a logic function (an example is displayed in FIGURE 2). Each gate takes two binary inputs and produces one binary output. As it can be observed in FIGURE 2, gates are connected through wires and the output function computed by the circuit depend on both the logic functions performed by the single gates and the way in which gates are connected. The logic functions performed by the circuit can be the logic AND, OR, NAND or NOR. This makes the problem relatively challenging [27].

Effective solutions are those maximizing the following fitness function:

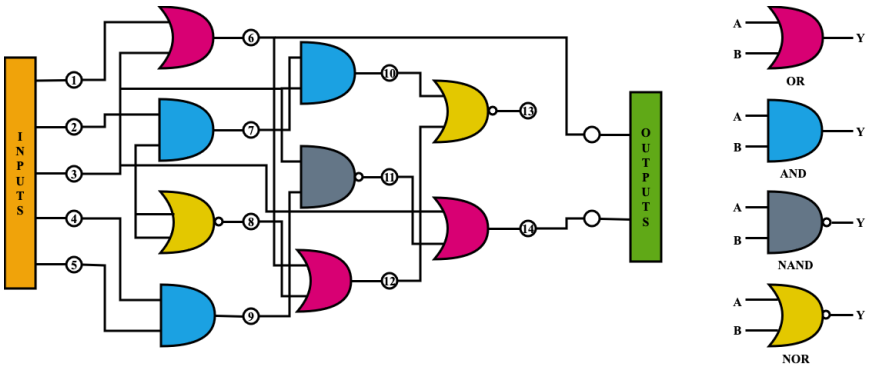

Figure 2: A digital circuit with 5 inputs and 2 outputs. The circuit contains 9 logic gates, each one computing a specific function, which are organized into 3 layers. The symbols representing the logic functions are provided in the right panel. Lines indicate the connections between the elements of the circuit.

Table 2: Parameters used to evolve solutions in the parity problem.

Parameter	Value
$numReplications$	20
$maxEvaluations$	10^8
$popSize$	20
$mutRate$	0.05
n	5

$$F_{parity} = 1 - \frac{1}{2^n} \sum_{i=1}^{2^n} |O_i - E_i|$$

where n is the size of the input bit-string to the digital circuit (in this case $n = 5$), i represents the generic input ($i \in [1, 2^n]$), O_i denotes the output of the circuit for input i, and E_i specifies the expected output. TABLE 2 reports the list of parameters used in this scenario. Differently from function optimization, in this case the evolving genotypes contain 406 genes (see the description reported in [20]).

2.3 Evolutionary Algorithms

As stated in Section 1, three algorithms have been compared. The first technique is the Stochastic Steady State (SSS) [21], an Evolutionary Algorithm (EA) working by evolving a population of solutions through a pure exploration, without any local refinement. It represents the baseline method of this study. Specifically, the algorithm considers an initial population of possible solutions, which are progressively modified based on their capability to cope with the given problem, which is defined as fitness or performance. The modifications of the solutions are realized through mutations only. The SSS has been already used to evolve solutions in both the function optimization and the 5-bit parity scenarios [20]. Moreover, it has been successfully employed to evolve solutions for

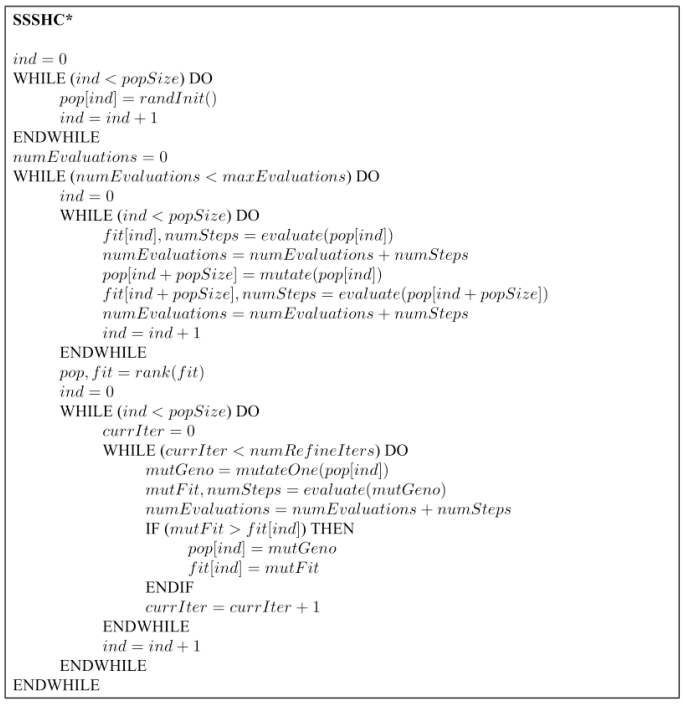

Figure 3: Pseudo-code of the SSSHC* method.

the double-pole balancing problem [21], and has been applied to other domains, like car racing simulators and collective foraging [28]. A complete description of the algorithm is reported in [20, 21].

The second technique is the Stochastic Steady State with Hill-Climbing (SSSHC) [20], a Memetic Algorithm (MA) that combines the SSS with a local refinement method seeking to enhance current solutions. Specifically, the SSSHC performs an iterative local search through mutations. However, there is no guarantee that this process is beneficial. In fact, especially in tasks characterized by stochasticity and/or the existence of agent-environment interactions, mutations are mainly maladaptive and represent pitfalls that the SSSHC can be unable to handle. A detailed description of the SSSHC can be found in [20].

The last technique is the SSSHC*, a variant of the SSSHC algorithm designed to enhance the quality of solutions in neutral problems. The SSSHC* is illustrated in FIGURE 3: it works by initializing a population of possible solutions with random values. Until the evolutionary process is not terminated, the SSSHC* attempts to improve the performance of current solutions through a mutation operator (exploration). Selected solutions undergo a local refinement process (exploitation) seeking to further enhance the fitness discovered so far. In particular, differently from the SSSHC, the local refinement process of the SSSHC* attempts to discover adaptive modifications by mutating one gene only (see the pseudo-code in FIGURE 3). The strategy should be beneficial in the context of function optimization, in which the SSSHC fails to improve the solutions discovered by the SSS (see the discussion reported in [20]).

With regard to the experiments described in this work, the length of the local refinement process of the SSSHC and SSSHC* algorithms has been set to $numRefineIters = 5$ in both scenarios.

The experiments have been performed with FARSA [29], a freely available tool already employed in a similar experimental setting [20], as well as to synthesize effective controllers in robotic scenarios [30–33].

3. RESULTS

This section reports the experimental outcomes. For the statistical analysis, we used the Mann-Whitney U test and we applied the Bonferroni correction.

3.1 Function Optimization

The performance analysis in the optimization of test functions clearly indicate that the SSSHC* significantly outperform both the SSS and the SSSHC ($p < 0.05$). Specifically, the SSSHC* obtains remarkably lower values with respect to all the considered test functions (see FIGURE 4 and TABLE 3): the ratio between the performance of the SSSHC* and those of the other techniques is an order of magnitude equal to 2 (Rastrigin and Rosenbrock functions) or 3 (Griewank and Sphere functions). This implies that the local refinement process of the SSSHC* is notably effective in this domain. As illustrated in FIGURE 4, the SSSHC* excels at finding quasi-optimal solutions. On the other hand, SSS and SSSHC algorithms fail to discover effective strategies and get stuck in local optima. Besides, the outcome is achieved independently of the number of replications (see standard deviations in TABLE 3 and FIGURE 4).

The SSSHC performs worse than the SSS in this domain ($p < 0.05$). The result could be partly related to the number of refinement iterations ($NumRefineIters = 5$), especially in the case of the Rosenbrock function (for a discussion about the relationship between performance and length of the exploitation phase, see [20]). In addition, optimizing test functions with sequences of length $n = 50$, as the ones considered in this work, is far from trivial.

Table 3: Average performance of the different algorithms on the optimization of test functions. Standard deviations are provided in squared brackets. Data achieved by replicating the experiment 20 times. Best outcomes are expressed in bold.

	SSS	SSSHC	SSSHC*
Griewank	0.2455 [0.0329]	0.3042 [0.0305]	**0.0027 [0.0032]**
Rastrigin	134.9092 [9.7396]	142.1595 [8.6134]	**3.8090 [0.0]**
Rosenbrock	2060.1830 [272.5708]	2766.7775 [300.3088]	**45.7004 [26.4894]**
Sphere	12.5683 [1.2789]	15.5306 [1.2895]	**0.0192 [0.0]**

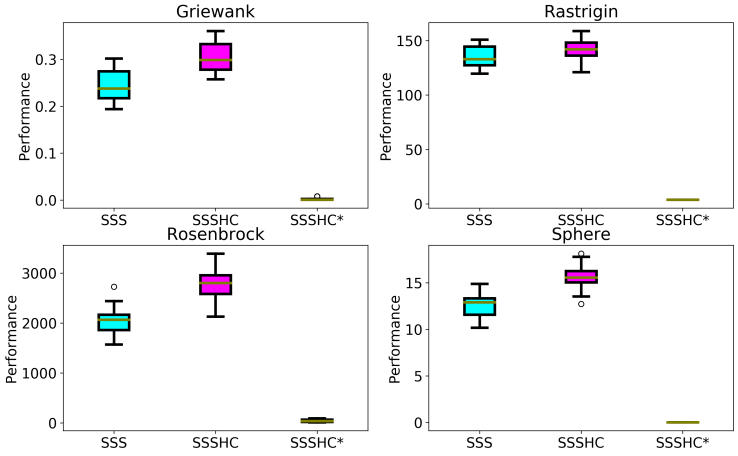

Figure 4: Fitness comparison in the function optimization scenario. Boxes contain data bounded in the range $[Q1, Q3]$ (i.e., first and third quartiles of data, respectively). The horiziontal lines denote the median values. The whiskers are drawn up to the largest data points falling within 1.5 times the inter-quartile range of data. Data derived from 20 replications.

3.2 5-Bit Parity

Regarding the 5-bit parity problem, the SSSHC* obtains similar performance to the SSS (see FIGURE 5 and TABLE 4), although slightly inferior ($p > 0.05$). Furthermore, the SSSHC* is outperformed by the SSSHC ($p < 0.05$). Differently from the previous scenario, the refinement process implemented by the SSSHC* fails to enhance current solutions. Conversely, the SSSHC manages to discover adaptive modifications leading to enhanced performances. In this domain, the refinement mechanism of the SSSHC is advantageous over the SSSHC*, which necessitates longer evolutionary process in order to reach higher performance levels. In addition, the SSSHC* is more sensitive to the variability of different replications: the standard deviation of the SSSHC* is remarkably higher than the other methods, as illustrated in FIGURE 5.

The performance of the SSSHC and SSS do not differ statistically ($p > 0.05$), although the former method finds out better solutions on average. The result is in line with those reported in [20], where SSS and SSSHC perform similarly when the length of the refinement process is in the range $[1-10]$.

Table 4: Average performance of the different algorithms on the 5-bit parity task. Standard deviations are provided in squared brackets. Data achieved by replicating the experiment 20 times. Best outcomes are expressed in bold.

	SSS	SSSHC	SSSHC*
5-bit parity	0.9156 [0.0560]	**0.9531 [0.0702]**	0.8672 [0.1126]

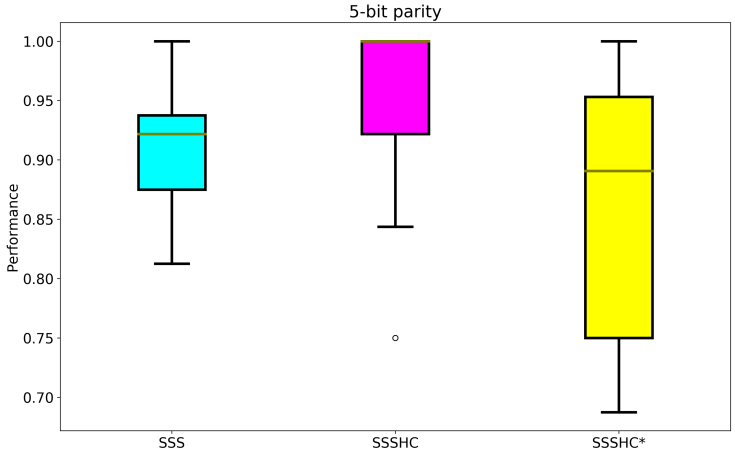

Figure 5: Fitness comparison in the 5-bit parity problem. Boxes contain data bounded in the range $[Q1, Q3]$ (i.e., first and third quartiles of data, respectively). The horiziontal lines denote the median values. The whiskers are drawn up to the largest data points falling within 1.5 times the inter-quartile range of data. Data derived from 20 replications.

4. DISCUSSION

The results presented in this work indicate that finding an effective trade-off between exploration and exploitation is far from trivial and depends on the considered task. In fact, with regard to the optimization of test functions, the local refinement process of the SSSHC* is highly effective at improving current solutions (see TABLE 3). In this domain, mutating one gene enables the algorithm to discover adaptive modifications ultimately leading to enhanced performances. On the other hand, the SSSHC works by mutating multiple genes simultaneously at each refinement iteration. Consequently, the beneficial effect of altering the value of some genes is neutralized by maladaptive modifications of other values. Further research should verify these claims with respect to other optimization functions (e.g., Ackley [34]).

Instead, concerning the parity problem, the refinement process of the SSSHC* fails in the discovery of effective solutions. This can be explained by considering that the size of the evolving genotype is considerably higher than that of the genotypes evolved for the function optimization scenario. Given the length of the exploitation, finding adaptive mutations in this case is remarkably more complex. Conversely, the simultaneous modification of different genes performed by the SSSHC enables to improve the performance and escape from local minima. Overall, the outcomes suggest a relationship between the genotype size and the way exploration and exploitation are balanced. Research on this topic goes beyond the scope of this paper and could be address in future works.

To sum up, the exploration-exploitation trade-off is beneficial in both tasks, but the way exploitation is performed is tightly linked to specific problem. Further studies should clarify to what extent the considerations reported here might be generalized to other domains.

5. CONCLUSIONS

The exploration-exploitation dilemma is a pivotal topic in the field of evolutionary computation as well as in many other domains. There exists a broad literature presenting approaches that seek to disentangle the subject by using Evolutionary Algorithms (EAs). In this respect, the application of memetic algorithms (MAs) shows promise in different scenarios.

In this work, we analyze the possibility to find effective trade-offs between exploration and exploitation in the context of neutral problems, i.e. tasks in which the search space is characterized by large areas that can be accessed through mutations not jeopardizing the survival chances. In particular, we considered two benchmark problems: (i) the optimization of test functions, and (ii) the 5-bit parity task. Moreover, we present a novel MA, called SSSHC*, which represents a modification of the SSSHC specifically tailored for neutral problems. The achieved outcomes reveal that an effective exploration-exploitation trade-off has been found in both domains, but the nature of exploitation depends on the specific problem: the SSSHC* allows to significantly improve the quality of solutions in the optimization function scenario, but has no positive effects regarding the parity problem. Instead, opposite results are obtained by the SSSHC. Overall, these findings demonstrate that the efficacy of exploitation is intertwined with the specificity of the task.

Notwithstanding the limitations of the current work, it contributes to open avenues on the comprehension of the exploration-exploitation dilemma and might pave the way to new approaches tackling this complex phenomenon. Future research directions could involve the study of mechanisms allowing to find an effective exploration-exploitation trade-off regardless of the specific nature of the considered problem, as the example reported in [35]. Moreover, future work should investigate the exploration-exploitation dilemma in highly dynamic environments. Lastly, the considered algorithms represent modifications of the SSS. Future studies could investigate the effectiveness of mixing exploitation with more sophisticated EAs, like the CMA-ES [36], and the OpenAI-ES [37], algorithms. Another interesting direction could involve the identification of mechanisms to adjust the exploration-exploitation trade-off online during the evolution, according to some performance criterion, in order to minimize the risk of premature convergence.

References

[1] Barbier-Chebbah A, Vestergaard CL, Masson JB. Approximate Information for Efficient Exploration-Exploitation Strategies. 2023. ArXiv preprint: https://arxiv.org/pdf/2307.01563

[2] Berger-Tal O, Nathan J, Meron E, Saltz D. The Exploration-Exploitation Dilemma: A Multidisciplinary Framework. PloS One. 2014;9:e95693.

[3] March JG. Exploration and Exploitation in Organizational Learning. Organ. Sci. 1991;2:71-87.

[4] Kwa HL, Leong Kit J, Bouffanais R. Balancing Collective Exploration and Exploitation in Multi-Agent and Multi-Robot Systems: A Review. Front. Robot. AI. 2022;8:771520.

[5] Hu J, Zhu K, Cheng S, Kovalchuk NM, Soulsby A, et al. Explainable AI Models for Predicting Drop Coalescence in Microfluidics Device. J. Chem. Eng. 2024;481:148465.

[6] Nathanael K, Cheng S, Kovalchuk NM, Arcucci R, Simmons MJ. Optimization of Microfluidic Synthesis of Silver Nanoparticles: A Generic Approach Using Machine Learning. Chem. Eng. Res. Des. 2023;193:65-74.

[7] Yogeswaran M, Ponnambalam SG. Reinforcement Learning: Exploration–Exploitation Dilemma in Multi-Agent Foraging Task. OPSEARCH. 2012;49:223-236.

[8] Mondada F, Franzi E, Ienne P. Mobile Robot Miniaturisation: A Tool for Investigation in Control Algorithms. In Experimental Robotics III: The 3rd International Symposium, Kyoto, Japan, 1993. Springer Berlin Heidelberg. 1994:501-513.

[9] Yan R, Gan Y, Wu Y, Liang L, Xing J, et al. The Exploration-Exploitation Dilemma Revisited: An Entropy Perspective. 2024. ArXiv preprint: https://arxiv.org/pdf/2408.09974

[10] Todorov E, Erez T, Tassa Y. Mujoco: A Physics Engine for Model-Based Control. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2012:5026-5033.

[11] Shen Y, Zeng C. An Adaptive Approach for the Exploration-Exploitation Dilemma in Non-stationary Environment. In 2008 International Conference on Computer Science and Software Engineering. IEEE. 2008;1:497-500.

[12] Zhang K, Pan W. The Two Facets of the Exploration-Exploitation Dilemma. In 2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology. 2006:371-380.

[13] Sledge IJ, Príncipe JC. Balancing Exploration and Exploitation in Reinforcement Learning Using a Value of Information Criterion. In 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). 2017:2816-2820.

[14] Črepinšek M, Liu SH, Mernik M. Exploration and Exploitation in Evolutionary Algorithms: A Survey. ACM Comput. Surv. (CSUR). 2013;45:1-33.

[15] Moscato P, Cotta C, Mendes A. Memetic Algorithms. Handbook of Approximation Algorithms and Metaheuristics. 2004.

[16] Knowles JD, Corne DW. M-PAES: A Memetic Algorithm for Multiobjective Optimization. In Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512). IEEE. 2000;1:325-332.

[17] Merz P. Memetic Algorithms and Fitness Landscapes in Combinatorial Optimization. In Handbook of Memetic Algorithms. Springer Berlin Heidelberg. 2012:95-119.

[18] Nguyen PT, Sudholt D. Memetic Algorithms Beat Evolutionary Algorithms on the Class of Hurdle Problems. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). 2018:1071-1078.

[19] Lin CC, Chen KC, Chuang WJ. Motion Planning Using a Memetic Evolution Algorithm for Swarm Robots. Int. J. Adv. Robot. Syst. 2012;9.

[20] Pagliuca P. Learning and Evolution: Factors Influencing an Effective Combination. AI. 2024;5:2393-2432.

[21] Pagliuca P, Milano N, Nolfi S. Maximizing Adaptive Power in Neuroevolution. PloS One. 2018;13:e0198788.

[22] Kimura M. The Neutral Theory of Molecular Evolution. Cambridge University Press (CUP). 1985.

[23] Dhawan D, Singh R. Performance Evaluation of Nature Inspired Meta-Heuristic Algorithms Using Rosenbrock, Rastrigin and Sphere Test Function for Optimization. Int. J. Recent Technol. Eng. 2019;8:1157-1163.

[24] Griewank AO. Generalized Descent for Global Optimization. J. Optim. Theory Appl. 1981;34:11-39.

[25] Rastrigin LA. Systems of Extremal Control. Nauka. 1974.

[26] Rosenbrock H. An Automatic Method for Finding the Greatest or Least Value of a Function. Comput. J. 1960;3:175-184.

[27] Miller JF. An Empirical Study of the Efficiency of Learning Boolean Functions Using a Cartesian Genetic Programming Approach. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). 1999;2:1135-1142.

[28] Pagliuca P, Nolfi S. Robust Optimization Through Neuroevolution. PloS One. 2019;14:e0213193.

[29] Massera G, Ferrauto T, Gigliotta O, Nolfi S. FARSA: An Open Software Tool for Embodied Cognitive Science. In ECAL. 2013:538-545.

[30] Aldana-Franco F, González FM, Nolfi S. Evolutionary Utility of Emerging Communication Systems and Singnal Complexity in Robotics. International Journal of Combinatorial Optimization Problems and Informatics (IJCOPI). 2024;15:15-27.

[31] Pagliuca P, Nolfi S. Integrating Learning by Experience and Demonstration in Autonomous Robots. Adapt. Behav. 2015;23:300-314.

[32] Pagliuca P, Vitanza A. The Role of N in the N-Mates Evaluation Method: A Quantitative Analysis. In Artificial Life Conference Proceedings (ALIFE 2024). MIT Press. 2024:812-814.

[33] Santucci VG, Baldassarre G and Mirolli M. Autonomous Selection of the “What” and the “How” of Learning: An Intrinsically Motivated System Tested With a Two Armed Robot. In 4th International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EpiRob). IEEE. 2014:434-439.

[34] Ackley D. A Connectionist Machine for Genetic Hillclimbing. Springer science & business media. 2012.

[35] Lin L, Gen M. Auto-Tuning Strategy for Evolutionary Algorithms: Balancing Between Exploration and Exploitation. Soft Comput. 2009;13:157-168.

[36] Hansen N, Ostermeier A. Completely Derandomized Self-Adaptation in Evolution Strategies. Evol. Comput. 2001;9:159-195.

[37] Salimans T, Ho J, Chen X, Sidor S, Sutskever I. Evolution Strategies as a Scalable Alternative to Reinforcement Learning. ArXiv preprint: https://arxiv.org/pdf/1703.03864.