Article

Not peer-reviewed version

Energy Efficiency Evaluation of Artificial Intelligence Algorithms

Kalin Penev [*] , Alexander Gegov , Olufemi Isiaq , Raheleh Jafari

Posted Date: 27 June 2024

doi: 10.20944/preprints202406.1808.v2

Keywords: green computing; green and sustainable software; software energy efficiency; Artificial Intelligence; Free Search; Bremermann's Limit; Planck's constant; energy efficiency; sustainability.

Article

Energy Efficiency Evaluation of Artificial Intelligence Algorithms †

Kalin Penev [1],*, Alexander Gegov [2,3], Olufemi Isiaq [4] and Raheleh Jafari [5]

[1] Department of Science and Engineering, Solent University, Southampton, SO14 0YN, United Kingdom
[2] University of Portsmouth, School of Computing, Portsmouth PO1 3HE, United Kingdom; alexander.gegov@port.ac.uk
[3] Technical University of Sofia, English Faculty of Engineering, 1756 Sofia, Bulgaria
[4] Creative Computing Institute, University of the Arts London, London, WC1V 7EY, United Kingdom; f.isiaq@arts.ac.uk
[5] School of Design, University of Leeds, Leeds LS2 9JT, United Kingdom; r.jafari@leeds.ac.uk
* Correspondence: kalin.penev@solent.ac.uk; Tel.: +44-(0)-2382013640

† In memory of the 100th anniversary of the birth of Ivan Georgiev Chuoruolinkov, one of the great scientists of the 20th century.

Abstract: This article continues research efforts on performance and energy efficiency of intelligent algorithms and its role for green and sustainable computing. The introduction focusses on Bremermann's Limit and its effect on extensive approach for improvement of computers performance. The article aims to identify role of Intelligent Algorithms energy efficiency, how it differs from general software energy efficiency. An improved empirical investigation on heuristic methods for search and optimisation illustrates algorithms' energy efficiency. Experimental results and consideration of further work conclude the article.

Keywords: green computing; green and sustainable software; software energy efficiency; Artificial Intelligence; Free Search; Bremermann's limit; Planck's constant; energy efficiency; sustainability

1. Introduction

Inspired by environmental behaviour euphoria to green technologies logically converges to green computing where major player is software. According to the literature green computing aspects involve optimising hardware and software design, reducing power consumption, using renewable energy sources, improving software efficiency, virtualizing servers, and managing e-waste [1]. Green computing should enhance the performance and energy effectiveness of computational systems by optimising hardware and software. Recent trends and concerns on environment and society regarding well-being, reasonably targeted Responsible Artificial Intelligence (RAI) [2] which accelerated demand for energy efficient intelligent software. [3]

Although "Artificial Intelligence (AI) has the potential to drive towards a future in which all of humanity flourishes" [2] the energy consumption of Information Technologies (IT) including, portable devices, data centres and cloud servers, has been dramatically increasing every year [4], which is reflected on global carbon emissions as identified in published global energy reviews [5,6].

Issues with computing energy efficiency need more deep analysis. According to the publications in fundamental research "There is a maximum rate at which data processing can proceed. This maximum rate applies to all data pro-cessing systems, manmade as well as biological. This conjecture aims stimulate the discussion towards a tightening of concepts in artificial data processing and biological evolution alike. The conjecture is the following:

No data processing system whether artificial or living can process more than (2×10^{47}) bits per second per gram of its mass." [7]

Formulation of these computational limitations is based on fundamental physical principles: "The capacity of any closed information transmission or processing does not exceed mc^2/h bits per second, where m is the mass of the system, c the light velocity, h Planck's constant" [8].

More recent publications stated that: "suggested by H. J. Bremermann in 1962 limit has to be corrected to make it compatible with general relativity. As a result, Bremermann's limit, proportional to mass M of system, $Mc^2/h = \sim (M/gram)10^{47}$ bits per second, should be replaced by an absolute limit $(c^5/Gh)^{1/2} = \sim 10^{43}$ bits per second, where universal constants c, G, and h are the speed of light, the gravitational constant, and Planck's constant [9].

Presence of Bremermann's limit alerts that extensive improvement of computational performance will face insurmountable barrier. Initial indications that computers approaching this barrier are: - increasing need for cooling for extraction of the heat produced from computational systems; - need for extensive growth of electricity sources; - changes of the environmental heat pollution, and impact on the climate which seems irreversible.

A natural solution to cope with limitations could be adapting efficiency similarly to the natural systems and considering two aspects - hardware and software energy efficiency. As far as large computational systems are home of most of AI services including popular OpenAI it is essential to mention that according to [10–12], based on the LINPAC Benchmark [13] for the last 20 years Supercomputers hardware energy efficiency is improved more than 200 times. This article focuses on software energy efficiency where is difficult to see such categorical progress, and more specifically on Artificial Intelligence and intelligent algorithms, which are mostly dealing with high volume, largely uncertain, and time dependent data.

2. Survey of Related Literature

Hardware energy efficiency has been improved constantly since computers exists [14,15], however software developers exploit this improvement and concerns about the efficiency of software have appeared only more recently, illustrated by growing number of publications [16]. Software sustainability should be applied to software systems at whole, specific software products, online applications, and data processing amongst others, by minimising power consumption, and harmonising software life cycle processes addressing required human, economic, and energy resources [16].

Currently, due to a growing demand, large research efforts are directed towards reducing energy consumption of portable devices such as smartphones, tablets, and laptops, by improving the software quality using code refactoring which supports the restructuring of existing source codes to obtain more energy-efficient code [17–19].

Generalising assessment of the software as sustainable including efficiency, quality and other essential properties seems to be an advanced approach [20–22]. It is essential to encourage software practitioners considering sustainability during software design and development [20]. Facilitation of the software development process by providing systematic guidelines, technical dimensions of sustainability [22], a framework that allowing professionals to foresee how and how much impact the software has on sustainability and its dimensions [21]. Taking into account specific for Computational Intelligence and Artificial Intelligence properties proposed models and frameworks are considered and adapted within the study presented in this article.

Intelligent systems and Artificial Intelligence as an advanced type of software play a key role in computing, Cloud Services and Data processing, which affects many aspects of the live. According to recent research [23] current approaches to build AI-based systems target mainly accuracy and reliability which require large data volume, large Artificial Intelligence Models and resource consuming infrastructure. The same publications proposed a hypothesis, which states: "When modelling and developing green AI-based systems, the impact of architectural decisions on energy efficiency must be better understood, defined, reported, and managed in order to deliver AI-based systems with less demanding computational power needs" [23]. Based on past experience in developing intelligent software can be argued that intelligent software needs appropriate data abstractions, adoption of heuristic and metaheuristic algorithms and most importantly reduction of outdated limitations in software development [24,25] which could help adaptation of the intelligent software in resolving tasks using minimal computations resources, for minimal period of time in the range from simple tasks [24,26] to problems with high number of parameters [27,28].

Due to the growing number of services and applications, based om Machine Learning and Artificial Intelligence, research publications [23] proposed approaches for sustainable, green Artificial Intelligence based software systems architecture, which aims to "To provide data scientists and software engineers tool-supported, architecture-centric methods for the modelling and development of green AI-based systems" [23]. However proposed architecture centric approach misses entirely core properties of natural intelligent species such as cognition and adaptation, which will be discussed further in this article.

Comprehensive research on design patterns for Machine Learning applications [29] identifies and distinguishes 15 different patterns. For example, pattern 6 solution - "Support both real-time data processing and continuous reprocessing with a single stream processing engine." and pattern 7 solution – "Store data, which range from structured to unstructured, as "raw" as possible into a data storage." Although process of analysis of a range of paters and then identification and distinction of key design patterns all of them are limited to analytical programming and misses properties of natural intelligent species such as abstraction and intuition amongst others, which also will be discussed further in this article. Key aspect in evaluation and assessment of Artificial Intelligence systems sustainability and quality is efficiency is measurement and assessing energy efficiency of software products and components. Recently published research on software energy and resource measurements identifies and compares several existing approaches aiming to generalize, extract, and categorize a comprehensive Green Software Measurement Model. The aim is to allow the categorisation of existing measurement methods and the derivation of adapted methods for individual measurement use cases, such as for software types, hardware and software setups, and individual components of software systems. [30]

This model was adapted for empirical research and evaluation of experimental software presented in this article.

An aspect with need consideration is how level of intelligence and its sustainability depends on used language. Published comparative research identifies significant differences in the energy consumption when certain algorithm is different languages using different compilers [31]. This aspect needs more research not only on computer languages but also on human languages.

Serious concerns regarding the proclivities of current Artificial Intelligence, Machine Learning, and Deep Learning are exponential growing demand for data, training, and infrastructure, amongst others [32,33]. All these are in sharp contradiction with emerging regulations and requirements for efficiency, and sustainability [34,35] and to the laws of nature for natural selection [36].

A harmony with the laws of nature would facilitate sustainability of Artificial Intelligence Systems. Primary questions regarding data processing, which need answers for example are: - Which biological system memorises full data for example images in similar to full definition format? Which natural creatures communicate high volume of data similar to transmission of high-definition video? Etc.

Software energy efficiency and performance are in some extent contradictive but increasingly essential non-functional requirements in software engineering particularly when applied to high-performance computing (HPC) systems.

To achieve performance and energy efficiency goals, software developers require a deep understanding of both the problem at hand and the target computer architecture. Moreover, software developers have to consider a multitude of programming models and languages, tools, and heterogeneous architectures and systems, which increases the development complexity [37].

Artificial intelligence applications are amongst the number of software needing increasing high performance and energy efficiency, and only specialized developers master this necessary knowledge. Thus, methodologies and tools to assist both specialized and typical developers are of paramount importance when targeting high-performance computing systems [37]. Proposed approach for time and energy consumption measurement could be invaluable for evaluation of

Intelligent Computing and Artificial Intelligence software systems. Although not automated similar approach, for simultaneous measurement of time, and energy consumption of heuristic algorithms is applied in presented in this article experimental results.

In the field of Artificial Intelligence, Computational Intelligence, and software development in general, similar to the behaviour of biological species, it can be often observed how the same task can be resolved for different period of time using different volume of resources. Typical software examples are sorting algorithms [38] and in Computational Intelligence adaptive heuristic algorithms [39].

A good idea for improving intelligent systems is applying Genetic, Swarm, Evolutionary, Heuristic, Metaheuristics, and Adaptive algorithms. A detailed study on improving machine learning classifiers performance optimised by swarm intelligent algorithms compares more than 10 Metaheuristics applied to code smell detection [40]. Achieved results deserve attention. Summarised algorithms hillites significant progression of metaheuristics in recombination and calculation of distances to desired solutions. This summary [40] (pp 23-29) helps to identify easily common limitation of all used Metaheuristics.

According to [41] "The ultimate goal of Artificial Intelligence is to create technology that allows computational machines to function in a highly intelligent manner."

Formulation of a clear aim motivates researchers to develop new algorithms with superior performance and large-scaled datasets with high quality. However, it is still infeasible for Artificial Intelligence systems to cover possible potential situations when applied to real world applications. The core question is how to leverage the strengths of these uncertainties guaranteeing socially responsible behaviour of Artificial Intelligence algorithms [42]. In pursuing this goal appropriate interpretation is facing high diversity answering the question: - What is Artificial intelligence? [43], and this could be serious constraint achieving socially responsible behaviour of Artificial Intelligence algorithms. The question which this study rises is: - Can we classify Artificial Intelligence as socially responsible according to its energy efficiency and sustainability? The answer to this question needs comprehensive further research.

2. Materials, Tools and Methods

For the purpose of this study are selected 7 test problems. Criteria for tests selection are:

- must be scalable for multidimensional format.
- must be with heterogeneous landscape.

The chosen numerical tests are scalable and form different search spaces. All tests are transformed for maximisation and scaled to 100 parameters.

- Griewank test is global [43]. Optimal value is 0.
- Michalewicz test is global test with unknown optimum [44]. Optimal value depends on dimensions number.
- Norwegian test is global test with unknown optimum [45]. Optimal value depends on dimensions number.
- Rastrigin test is global [46]. Optimal value is 0.
- Rosenbrock test is smooth flat test with single solution [47]. Optimal value is 0.
- Schwefel test is global [48]. Optimal value is 0.
- Step test introduces plateaus to the topology and the search process cannot rely on local correlation [49]. Optimal value depends on the dimensions number and for variety of dimension is unknown.

For the purpose of this study 3 algorithms are selected:

- Particle Swarm Optimisation (PSO) [50] – representing the group of swarm algorithms applied to real coded tasks over continues space.
- Differential Evolution (DE) [51] - heuristic approach for optimising nonlinear and non-differentiable continuous space functions.
- Free Search (FS) [52] - adaptive heuristic algorithm for search and optimisation within continuous space.

All algorithms are implemented to operate on 10 solutions and limited to 100000 iterations, 320 sequential runs. The aim of the experiment is to measure time and energy required for completion of the defined number of iterations.

For these experiments, a computer system with the following components and settings is used - processor Intel XEON E5 1660 V2 overclocked at 4.750 GHz, running 1 core – 1 thread, Max TDP – 130 W, CPU water cooler, RAM at 2000 MHz, motherboard ASUS P9X79-E WS and solid-state disk - SanDisk Extreme SSD SATA III.

All experiments are completed individually running one single algorithm on a single test function at time.

Methodology

For this study is adopted Green Software Measurement Model proposed in the literature [30], which is adapted according to the nature of the study by control of the following parameters:

- Duration *min*
- Number of iterations *integer*
- Mean system power W
- System Energy Wh
- CPU usage %
- CPU power W
- CPU cores - 1 core – 1 thread

Application of full Green Software Measurement Model as proposed in [30] could be a subject of further research and availability of necessary resources.

Power is measured for entire system using digital Power Consumption Energy Meter [53]. Power for each algorithm applied to each test is calculating as a difference between the level during the algorithms' execution and level on standby - running OS components and tools for CPU parameters measurement – CPUZ [54] and CPU cores temperature measurement CoreTemp [55]. SPUZ and CoreTemp are ringing during all experiments in parallels to the algorithms.

Each experiment is limited to 100000 iterations, and the duration is measured by recording start time manually and end time of the experiment is stored as an attribute of the results file. This ensures the time recording has no effects on the performance of the algorithms or the system's configurations.

3. Results

Experimental results are presented in Table 1, where: Column Test indicates the test function. Column PSO indicates time of experiments achieved by Particle Swarm Optimisation. Column DE indicates time of experiments achieved by Differential Evolution. Column FS indicates time of experiments achieved by Free Search. Time is presented in format *hh:mm:ss*.

Table 1. Time for execution of 100-dimensional version of the tests.

Test	PSO	DE	FS
	Time	Time	Time
Griewank	01:45:00	00:41:00	00:14:00
Michalewicz	02:44:00	01:46:00	01:02:00
Norwegian	01:50:00	00:47:00	00:12:00
Rastrigin	01:46:00	00:45:00	00:11:00
Rosenbrock	01:39:00	00:40:00	00:05:00
Schwefel	02:44:00	01:03:00	00:27:00
Step	02:37:00	00:42:00	00:06:00

The Mean system power on standby for Task Monitor 0% workload measured on the socket is 166W. CPU power for 1 core – 1 thread measured by CoreTemp is 33.4W and 21.5W.

The Mean system power for Task Monitor 100% workload for all experiments presented in Table 1 is 185W with variation 3% over the time of execution. Reason for this variation may be due to variation in temperature and other environmental factors but, this could be a subject of further research.

CPU power for 1 core – 1 thread measured by CoreTemp is 42.8W and 30.4W.

Presented in Table 1 data indicates variation of time for execution per test and per algorithm.

Analysis of the experimental data suggest that based on the complexity of the search space time for evaluation per test vary. More complex search space needs more time. Based on capabilities of the search algorithms time for search also vary.

Presented on Figure 1 graphic indicates time per test and on Figure 2 per algorithm for completion of 100000 iterations on selected test. Particle Swarm Optimisation needs more time for all tests. Differential Evolution uses medium interval of time over all tests. Free Search completed faster all tests.

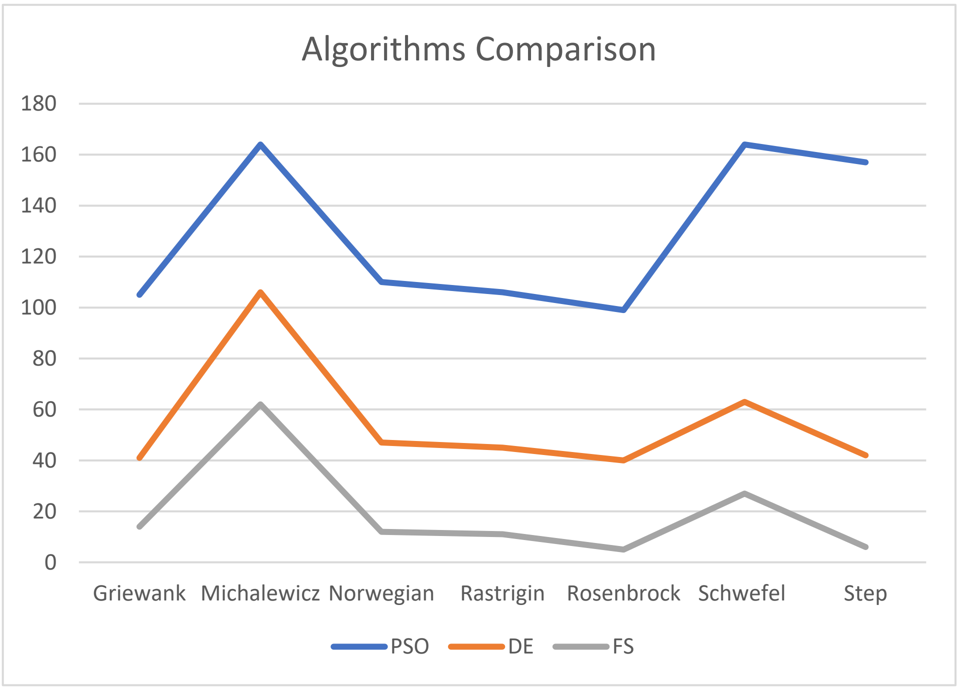

Figure 1. Tests time comparison per algorithm.

Results for Schwefel and Step test on Figure 2 suggest presence of possible specific factors which reflect on the time for exploration.

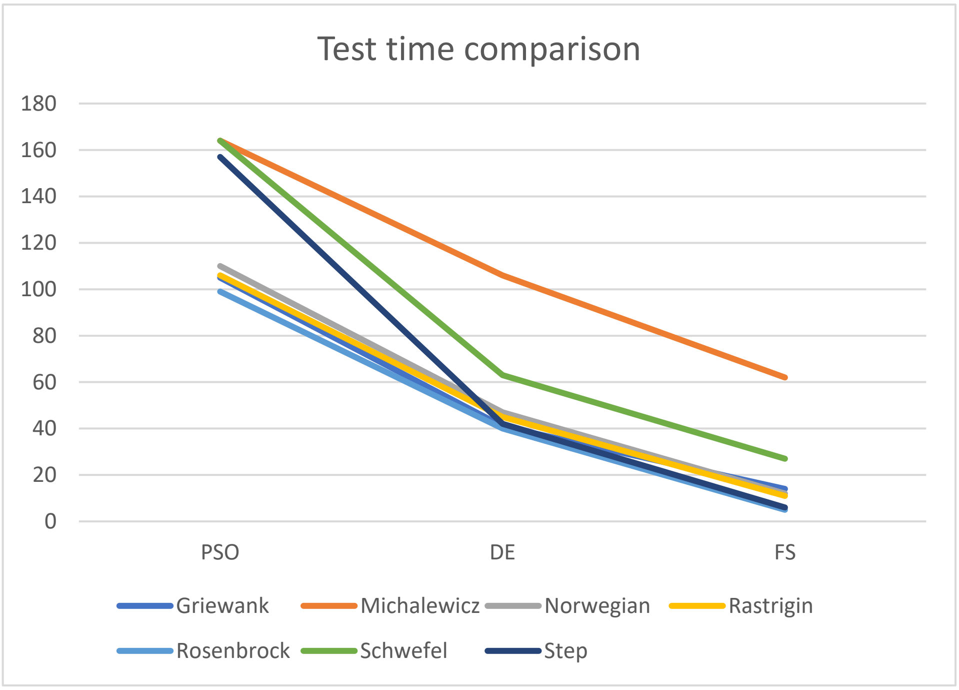

Figure 2. Algorithms time comparison per test.

Within the time of evaluation can be distinguished specific components such as:

- Time of objective function evaluation in other words this is time for apprehension of the space.
- Time for algorithm execution in other words this is time for interpretation and assessment of the search space.
- Time for algorithm decision making in other word this is time for selection and further action.

Energy use presented in Table 2 is based on algorithms energy consumption (difference between measured power for 100% workload during the experiments and 0% workload on standby), multiplied by period of time for completion of the experiment.

Table 2. Energy use for execution of 100-dimensional version of the tests.

Test	PSO	DE	FS
	Wh	Wh	Wh
Griewank	33.25	12.98	4.43
Michalewicz	51.93	33.57	19.63
Norwegian	34.83	14.88	3.80
Rastrigin	33.57	14.25	3.48
Rosenbrock	31.35	12.67	1.58
Schwefel	51.93	19.95	8.55
Step	49.72	13.30	1.90

Different algorithms can cope with the same task for different periods which lead to different energy use.

Analysis aiming to identify systematic relations between these components summarized of Table 3 identifies general qualitative difference only.

Presented within Table 3 relative difference between time in % in general suggest that DE is faster than PSO on all test, and FS is faster that RSO and DE. The difference highly varies and cannot be identified precise systematic relation per test and per algorithm. Precise quantitative analysis could be subject of further research.

Table 3. Relative time difference per test in %.

Test	DE/PSO	FS/PSO	FS/DE
	%	%	%
Griewank	39%	13%	34%
Michalewicz	65%	38%	58%
Norwegian	43%	11%	26%
Rastrigin	42%	10%	24%
Rosenbrock	40%	5%	13%
Schwefel	38%	16%	43%
Step	27%	4%	14%

4. Discussion

This section analyses the results, their interpretation from the perspective of previous studies and of the Role of Intelligent Algorithms Energy Efficiency.

Achieved results confirms and in certain extent clarifies published earlier results [27] on intelligent algorithms aimed to study computational limitations, energy consumptions and time. It can be argued that different efficiency is due to software design, implementation and execution which by its own is absolutely correct. However, it should be clarified how different software engineering techniques implement intelligent behaviour.

First little attention will be given to Epistemology [56,57]. Popular models Data-Information Knowledge (DIK) [58] and Data-Information-Knowledge-Wisdom (DIKW) [59] suggest a way in which intelligent beings and systems can cope with apprehension of the surrounding environment is generation and preprocessing data abstracting essential for certain purposes information which can be used and then abstracting from the information knowledge, which could be memorised for further use. This process first reduces the amount of data, which should be stored and second allow much faster processing of known cases and adapting to and processing unknown cases. Both contribute significantly to efficiency and sustainability of intelligent beings. Implementing this process as a software for Intelligent Computing can contribute to the sustainability of Artificial Intelligence Systems.

For amongst the many definitions of knowledge [60–62] probably the closest to understanding of software design and implementation is “Knowledge is the perception of the agreement or disagreement of two ideas” [60].

In the cognitive processes understanding the fundamental principles could help significantly to strengthen machine learning efficiency and sustainability of intelligent systems. According to the literature [63] “Knowledge of the external world can be obtained either by intuition or by abstraction”.

Where appropriate understanding of intuition and abstraction could support significant improvement of the process of learning and more precisely formulated in a process of building knowledge. A good interpretation of intuitive and abstractive cognition is formulated by William of Ockham [64].

“Intuitive cognition is defined as an act of apprehension in virtue of which the intellect can evidently judge that the apprehended object exists or does not exist, or that it has or does not have some particular quality or other condition; in short, an intuitive cognition is an act of immediate awareness in virtue of which an evident judgment of contingent fact can be made.

Abstractive cognition is defined as any act of cognition in virtue of which it cannot be evidently known whether the apprehended object exists or does not exist, and in virtue of which an evident contingent judgment cannot be made” [64].

Applying these concepts to the Blackbox model helps the adaptive heuristic algorithm called Free Search to perform fasted amongst heterogenous landscapes and tasks.

Future research could focus on new models and software implementations for machine training, intelligent computing, apprehension of the environment, building actionable knowledge and adaptive behaviour. For simple tasks computational intelligence con-tributes in competition between algorithms and systems. For large complex problems for example where the search space exceeds the range of 101000000 locations, and the time for exploration for space with this size could tend to infinite, intelligent, adaptive behaviour is essential for reducing the time and energy use of the process of search, and a good example is the result available in [28] achieved by Free Search applied to optimisation of a task with 100 000 parameters.

5. Conclusions

In summary this research completed an empirical evaluation and comparison of intelligent and adaptive algorithms time and energy consumption applied to heterogeneous numerical tests. It identified notable difference in energy efficiency and speed when different algorithms are applied to same tasks.

Experimental results illustrate strengths and limitations of used algorithms. Discussion on concepts and their relationship with computational intelligence and possible value for sustaining intelligent systems concludes the article.

Author Contributions: Conceptualization - Kalin Penev & Alexander Gegov; methodology & software & validation - Kalin Penev; formal analysis - Kalin Penev &Femi Isiaq; investigation & resources & data curation - Kalin Penev; writing—original draft preparation Kalin Penev; writing—review and editing - Kalin Penev & Alexander Gegov & Femi Isiaq & Raheleh Jafari. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no funding.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Paul S. G. et al., “A Comprehensive Review of Green Computing: Past, Present, and Future Research,” in IEEE Access, vol. 11, pp. 87445-87494, 2023, doi: 10.1109/ACCESS.2023.3304332.
2. Cheng, L., Varshney, K. R., & Liu, H. (2021). Socially responsible AI algorithms: Issues, purposes, and challenges. Journal of Artificial Intelligence Research, 71, 1137–1181.
3. Lee, S.U.; Fernando, N.; Lee, K.; Schneider, J. A survey of energy concerns for software engineering. Journal of Systems and Software 2024, 210, doi:10.1016/j.jss.2023.111944.
4. Naumann, S.; Dick, M.; Kern, E.; Johann, T. The GREENSOFT Model: A reference model for green and sustainable software and its engineering. Sustainable Computing: Informatics and Systems 2011, 1, 294, doi:10.1016/j.suscom.2011.06.004.
5. Raimi, D.; Zhu, Y.; Newell, R.G.; Prest, B.C. Global Energy Outlook 2024: Peaks or Plateaus? 2024, available at: https://www.rff.org/publications/reports/global-energy-outlook-2024/, Accessed 10.06.2024
6. IEA, GlobalEnergyReview2021, available at: https://www.iea.org/reports/global-energy-review-2021, Accessed 10.06.2024
7. Bremermann, H.J., “Optimization through Evolution and Recombination,” in Self-organizing Systems, M.C. Yovits, G.T. Jacobi and G.D. Goldstein, Eds. Washington, DC: Spartan Books, 1962, pp. 93-106;
8. Bremermann, H.J., 1967, January. Quantum noise and information. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability (Vol. 4, pp. 15-20). University of California Press Berkeley, CA.
9. Gorelik, G., 2009. Bremermann’s Limit and cGh-physics. arXiv preprint arXiv:0910.3424.
10. Top500A 2024, FRONTIER - HPE CRAY EX235A, AMD OPTIMIZED 3RD GENERATION EPYC 64C 2GHZ, AMD INSTINCT MI250X, SLINGSHOT-11, https://top500.org/system/180047/ Accessed 15.05.2024
11. Top500B, 2024, BLUEGENE/L - ESERVER BLUE GENE SOLUTION, https://top500.org/system/174210/, Accessed 15.05.2024
12. Top500C, 2024, JEDI - BULLSEQUANA XH3000, GRACE HOPPER SUPERCHIP 72C 3GHZ, NVIDIA GH200 SUPERCHIP, QUAD-RAIL NVIDIA INFINIBAND NDR200 https://top500.org/system/180269/ Accessed 15.05.2024
13. Dongarra, J., 2007, Frequently Asked Questions on the Linpack Benchmark and Top500, available at https://www.netlib.org/utk/people/JackDongarra/faq-linpack.html, Accessed 19.01.2024
14. JVA Initiative Committee and Iowa State University, 2011, ATANASOFF BERRY COMPUTER, available at https://jva.cs.iastate.edu/operation.php, Accessed 12.06.2024
15. Freiberger, Paul A. and Swaine, Michael R.. “Atanasoff-Berry Computer”. Encyclopedia Britannica, 20 Mar. 2023, https://www.britannica.com/technology/Atanasoff-Berry-Computer. Accessed 11 June 2024.

16. Calero, C.; Mancebo, J.; Garcia, F.; Moraga, M.A.; Berna, J.A.G.; Fernandez-Aleman, J.L.; Toval, A. 5Ws of green and sustainable software. Tinshhua Sci. Technol. 2020, 25, 401, doi:10.26599/tst.2019.9010006.
17. Gottschalk, M.; Jelschen, J.; Winter, A. Energy-Efficient Code by Refactoring. Available at: https://dl.gi.de/server/api/core/bitstreams/a42bfb61-43dd-4d5c-9b1d-2c8c2b4ca51f/content, Accessed 12.06.2024
18. Sanlıalp, ˙I.; Öztürk, M.M.; Yigit, T. Energy Efficiency Analysis of Code Refactoring Techniques for Green and Sustainable Software in Portable Devices. Electronics 2022, 11, 442. https://doi.org/10.3390/electronics11030442 Accessed 11.06.2024
19. Anwar, H.; Pfahl, D.; Srirama, S.N. Evaluating the impact of code smell refactoring on the energy consumption of Android applications. In Proceedings of the 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Kallithea, Greece, 28–30 August 2019.
20. Noman, H.; Mahoto, N.; Bhatti, S.; Rajab, A.; Shaikh, A. Towards sustainable software systems: A software sustainability analysis framework. Information and Software Technology 2024, 169, doi:10.1016/j.infsof.2024.107411.
21. Heldal, R.; Nguyen, N.; Moreira, A.; Lago, P.; Duboc, L.; Betz, S.; Coroamă, V.C.; Penzenstadler, B.; Porras, J.; Capilla, R.; Brooks, I.; Oyedeji, S.; Venters, C.C. Sustainability competencies and skills in software engineering: An industry perspective. Journal of Systems and Software 2024, 211, doi:10.1016/j.jss.2024.111978.
22. Venters, C.C.; Capilla, R.; Nakagawa, E.Y.; Betz, S.; Penzenstadler, B.; Crick, T.; Brooks, I. Sustainable software engineering: Reflections on advances in research and practice. Information and Software Technology 2023, 164, doi:10.1016/j.infsof.2023.107316.
23. Martínez-Fernández, S.; Franch, X.; Durán, F. Towards green AI-based software systems: an architecture-centric approach (GAISSA). 2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) 2023, doi:10.1109/seaa60479.2023.00071.
24. Penev, K.; Littlefair, G. Free Search—a comparative analysis. Information Sciences 2005, 172, 173, doi:10.1016/j.ins.2004.09.001.
25. Penev, K. and Gegov, A., 2008. Free Search of real value or how to make computers think. St. Qu.
26. Vasileva, V. and Penev, K., 2012, September. Free search of global value. In 2012 6th IEEE International Conference Intelligent Systems (pp. 425-430). IEEE.
27. Penev, K., Free Search in Multidimensional Space M, 2018, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Lirkov, I. & Margenov, S. (eds.). Switzerland: Springer Nature, Vol. 10665. p. 399-407 9 p.
28. Penev, K., 2022, An optimal value for 100 000-dimensional Michalewicz test. Available at: https://pure.solent.ac.uk/files/33733992/100_000_dimensional_Michalewicz_test_2.pdf, Accessed 12.06.2024
29. Washizaki, H.; Khomh, F.; Gueheneuc, Y.; Takeuchi, H.; Natori, N.; Doi, T.; Okuda, S. Software-Engineering Design Patterns for Machine Learning Applications. Computer 2022, 55, 30, doi:10.1109/mc.2021.3137227.
30. Guldner, A.; Bender, R.; Calero, C.; Fernando, G.S.; Funke, M.; Gröger, J.; Hilty, L.M.; Hörnschemeyer, J.; Hoffmann, G.; Junger, D.; Kennes, T.; Kreten, S.; Lago, P.; Mai, F.; Malavolta, I.; Murach, J.; Obergöker, K.; Schmidt, B.; Tarara, A.; De Veaugh-Geiss, J.P.; Weber, S.; Westing, M.; Wohlgemuth, V.; Naumann, S. Development and evaluation of a reference measurement model for assessing the resource and energy efficiency of software products and components—Green Software Measurement Model (GSMM). Future Generation Computer Systems 2024, 155, 402, doi:10.1016/j.future.2024.01.033.
31. Koedijk, L.; Oprescu, A. Finding Significant Differences in the Energy Consumption when Comparing Programming Languages and Programs. *2022 International Conference on ICT for Sustainability (ICT4S)* **2022**, doi:10.1109/ict4s55073.2022.00012.
32. Wu, C.; Raghavendra, R.; Gupta, U.; Acun, B.; Ardalani, N.; Maeng, K.; Chang, G.; Behram, F.A.; Huang, J.; Bai, C.; Gschwind, M.; Gupta, A.; Ott, M.; Melnikov, A.; Candido, S.; Brooks, D.; Chauhan, G.; Lee, B.; Lee, H.S.; Akyildiz, B.; Balandat, M.; Spisak, J.; Jain, R.; Rabbat, M.; Hazelwood, K.; Ai, F. Sustainable AI: Environmental Implications, Challenges and Opportunities. Available at: https://www.semanticscholar.org/reader/2c6df83795cd5baf3b8c6e2639b85e2df0cee1d0, Accessed 13.06.2024
33. Patterson, D.; Gonzalez, J.; Le, Q.; Liang, C.; Munguia, L.; Rothchild, D.; So, D.; Texier, M.; Dean, J. Carbon Emissions and Large Neural Network Training. arXiv preprint arXiv:2104.10350, 2021.
34. EU, "Regulation (EU) 2020/852 of the european parliament and of the council of 18 june 2020 on the establishment of a framework to facilitate sustainable investment, and amending regulation (EU) 2019/2088 (Text with EEA relevance)," Official Journal of the European Communities, vol. 198, pp. 13–43, 2020. Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32020R0852&from=EN, Accessed 13.06.2024
35. EU, AI Act, 2024. Available at: https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai, Accessed 13.06.2024

36. Darwin, C., On the Origin of Species by Means of Natural Selection. New York: D. Appleton and Company, 443 & 445 Broadway. MDCCCLXI. Available at: https://darwin-online.org.uk/converted/pdf/1861_OriginNY_F382.pdf, Accessed 13.06.2024
37. Pinto, P.; Bispo, J.; Cardoso, J.M.P.; Barbosa, J.G.; Gadioli, D.; Palermo, G.; Martinovic, J.; Golasowski, M.; Slaninova, K.; Cmar, R.; Silvano, C. Pegasus: Performance Engineering for Software Applications Targeting HPC Systems. IIEEE Trans. Software Eng. 2022, 48, 732, doi:10.1109/tse.2020.3001257.
38. GG, 2024, Sorting Algorithms, available at: https://www.geeksforgeeks.org/sorting-algorithms/, Accessed 13.06.2024
39. Penev, K. (2018). Free Search in Multidimensional Space M. In: Lirkov, I., Margenov, S. (eds) Large-Scale Scientific Computing. LSSC 2017. Lecture Notes in Computer Science(), vol 10665. Springer, Cham. https://doi.org/10.1007/978-3-319-73441-5_43
40. Jain, S.; Saha, A. Improving and comparing performance of machine learning classifiers optimized by swarm intelligent algorithms for code smell detection. *Science of Computer Programming* **2024**, *237*, doi:10.1016/j.scico.2024.103140.
41. Deng, L. Artificial Intelligence in the Rising Wave of Deep Learning: The Historical Path and Future Outlook [Perspectives]. IEEE Signal Process. Mag. 2018, 35, 180, doi:10.1109/msp.2017.2762725.
42. Hutter, M.; Legg, S. A. 2007. A Collection of Definitions of Intelligence. In Proceedings of the 2007 conference on Advances in Artificial General Intelligence: Concepts, Architectures and Algorithms: Proceedings of the AGI Workshop 2006. IOS Press, NLD, 17–24. arXiv:0706.3639
43. Griewank, A. O., 1981, “Generalized Decent for Global Optimization.” Journal of Optimization Theory and Applications, 34, pp. 11-39.
44. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Berlin, Heidelberg, New York: Springer-Verlag, 1992.
45. Penev, K. (2015). Free Search in Multidimensional Space II. In: Dimov, I., Fidanova, S., Lirkov, I. (eds) Numerical Methods and Applications. NMA 2014. Lecture Notes in Computer Science(), vol 8962. Springer, Cham. https://doi.org/10.1007/978-3-319-15585-2_12
46. Mühlenbein, H., Schomisch, D., and Born,J., 1991. “The Parallel Genetic Algorithm as Function Optimizer”. Parallel Computing, 17, pages 619–632.
47. Rosenbrock H.H., 1960, An automate method for finding the greatest or least value of a function. Comput. J. 3 (1960), pp.175-184.
48. Schwefel, H. P., 1981, Numerical Optimization of Computer Models. John Wiley & Sons, English translation of Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie, 1977.
49. De Jung K. A., 1975, An Analysis of the Behaviour of a Class of Genetic Adaptive Systems, PhD Thesis, University of Michigan.
50. Eberhart, R., and Kennedy J., 1995, Particle Swarm Optimisation, Proceedings of the 1995 IEEE International Conference on Neural Networks., vol. 4, p.p. (1942-1948). IEEE Press.
51. Storn, R., Price, K. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. Journal of Global Optimization 11, 341–359 (1997). https://doi.org/10.1023/A:1008202821328
52. Penev, K. (2008). Adaptive Heuristic Applied to Large Constraint Optimisation Problem. In: Lirkov, I., Margenov, S., Waśniewski, J. (eds) Large-Scale Scientific Computing. LSSC 2007. Lecture Notes in Computer Science, vol 4818. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-78827-0_68
53. GMM-DDS108 (KWE-PM01) Digital Power Consumption Energy Meter UK Plug Socket, 2024, Available at: https://testmeter.sg/webshaper/pcm/files/Data%20Sheet/GMM-DDS108-KWE-PM01-UK.pdf, Accessed 17.06.2024
54. CUPID, 2024, CPU-Z for Windows® x86/x64, available at: https://www.cpuid.com/softwares/cpu-z.html, Accessed 17.06.2024
55. CoreTemp, 2024, Core Temp 1.18.1, available at: https://www.alcpu.com/CoreTemp/, Accessed 17.06.2024
56. Stroll, Avrum and Martinich, A.P.. “epistemology”. Encyclopedia Britannica, 19 Apr. 2024, available at: https://www.britannica.com/topic/epistemology. Accessed 18.06.2024
57. Steup, Matthias and Ram Neta, “Epistemology”, The Stanford Encyclopedia of Philosophy (Spring 2024 Edition), Edward N. Zalta & Uri Nodelman (eds.), available at: URL = https://plato.stanford.edu/archives/spr2024/entries/epistemology/. Accessed 18.06.2024.
58. Zins, C. Conceptual approaches for defining data, information, and knowledge. J. Am. Soc. Inf. Sci. 2007, 58, 479, doi:10.1002/asi.20508.
59. Rowley, J. (2007). The wisdom hierarchy: representations of the DIKW hierarchy. Journal of Information Science, 33(2), 163-180. https://doi.org/10.1177/0165551506070706
60. Locke, J., 1689, An Essay Concerning Human Understanding, 1689, ISBN 0-87220-217-8
61. Nonaka, I and Takeuchi, H., 1995, The Knowledge – Creating Company How Japanese Companies Create the Dynamics of Information, ISBN 0-19-509269-4

doi:10.20944/preprints202406.1808.v2

62. Davenport, T., & Prusak, L., 2000, Working Knowledge: how organisations manage what they know, ISBN 0-87584-655-6
63. Tommaso Campanella (1568-1639), Encyclopaedia of Philosophy, Macmillan, New York, Vol.2, p. 12.
64. William of Ockham (1285 – 1349), Encyclopaedia of Philosophy, Macmillan, New York, Vol.8, p. 308.