Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Hormesis-Based Optimization (HBO) Algorithm: A Biologically Inspired Computational Approach

Amit Malik[1], Amita Rani[2]
[1]*Department of Computer Science and Engineering, SRM University, Delhi-NCR, Sonepat, Haryana, India.*
[2]*Department of Computer Science and Engineering, DCRUST, Murthal, Sonepat, Haryana, India.*

ARTICLE INFO

Received: 18 Dec 2024
Revised: 10 Feb 2025
Accepted: 28 Feb 2025

ABSTRACT

The Hormesis-Based Optimization (HBO) algorithm is a new method designed to achieve optimization while keeping computational cost and time low. It can be applied to areas such as workload management, resource planning, and task scheduling. This method works by converting the system's performance parameters into a single value termed as "stress". It is inspired by the phenomenon of hormesis, where small doses of stress strengthen biological systems, and ensures a dynamic distribution of stress to optimize system performance. The key strength of HBO lies in its ability to quickly and effectively decide the optimal adjustment of this 'stress' across various components, which is critical for achieving the best possible performance and can be a NP-hard problem in multi-constrained system. To test this principle, we applied the HBO algorithm to the system of machines, tasked with predictive-reactive dynamic job shop scheduling (DJSS), with the aim of reducing the overall job completion time and latency of the total number of jobs. The results show that the HBO algorithm outperformed not only conventional techniques like Genetic Algorithm (GA), Simulated Annealing (SA), and Tabu Search (TS), but also their adaptive methods such as Adaptive Genetic Algorithm (AGA), Adaptive Simulated Annealing (ASA), and Adaptive Tabu Search (ATS). Specifically, it improved the total job completion time (makespan) by an average of 4.15% and reduced latency by 4.79%, with time complexity of O(T·n·log〖(n)〗) , as compared to best performing ATS technique which has time complexity of O(n^2) for the worst case.

Keywords: Hormesis-Based Optimization, Dynamic Job Scheduling, Resource Allocation Efficiency, Biologically Inspired Algorithms, Computational Adaptability.

1. Introduction

Resource allocation and workload distribution are very critical for optimizing system performance in sectors like computing, manufacturing, and real-time systems, as they directly affect task efficiency [1, 2]. In various real time environments, the workloads can change unpredictably and efficient allocation helps systems adapt and avoid delays [3]. Any system, without optimization, may suffer inefficiency, increased costs, and poor adaptability [4]. Therefore, it is an area of research with great interest. In this study, we found that the traditional optimization methods, such as linear and integer programming, have been effective in simpler environments but struggles with growing decision spaces and real-time adaptability due to the "curse of dimensionality" [5]. The heuristic methods, on the other hand, are computationally efficient but can get stuck in local optima and are less effective in highly dynamic settings [6]. The machine learning techniques, including neural networks and deep learning, offer solutions but require large datasets, high computational power, and often lack transparency [7, 8 and 9]. Therefore we need methods which can balance the solution quality, speed, and computational cost.

The Hormesis-Based Optimization (HBO) algorithm is designed to address the challenges and needs discussed so far, such as, balancing performance, speed, and computational efficiency in changing environments [10]. We derive this idea from natural or biological hormesis, depicted in Figure 1, in which an appropriate amount of stress can help the biological system to improvise the performance [11, 12]. For example, the small stress from exercise may help the muscles grow stronger and adapt to increased demands, or exposure to low dose of induced stress may help our immune system to fight infections. To understand the working of HBO practically, we can say that it

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

works by carefully adjusting the workload on machines like we adjust the doses on biological systems to improve its performance and resilience [11, 12]. If a machine is running well, HBO may increase its workload slightly to push the overall system toward better performance. On the other hand, HBO improves the efficiency of overburdened machines by reducing their workload. The amount of workload to be increased or decreased is quantified as "stress" and is referred to as "doses" in this study. HBO works by converting the key performance parameters of a mechanical system into a unified value of "stress", which can be seen equivalent to natural or biological stress. This approach of using a single unified value also enables the algorithm to tackle the NP-hard challenge of multi-constrained optimization without exhaustive exploration of all possible solutions. Also, HBO uses a deterministic rule-based framework to decide the optimal value of "stress" to be adjusted among the components. These two qualities help HBO in avoiding the computational overhead associated with traditional optimization (e.g., exhaustive search) or machine learning methods (e.g., training on large datasets).

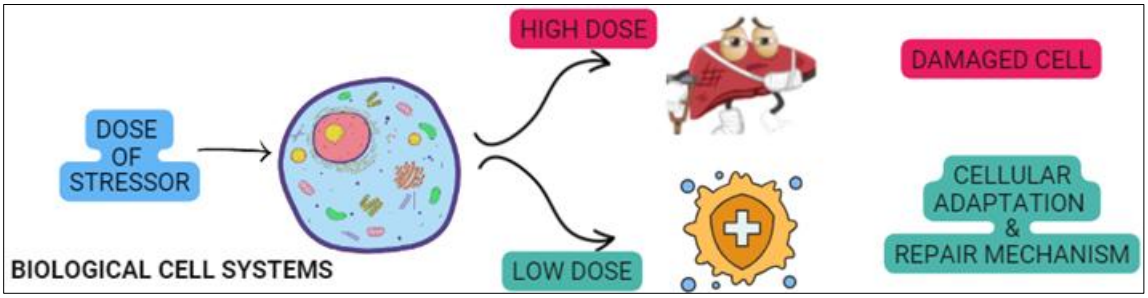

Figure 1: Hormesis Principle in Biological Systems

Next, we can summarize the key contributions of this work through following points:

1. Introduces a novel optimization algorithm applicable to a wide range of optimization problems, including resource allocation, workload distribution, and dynamic scheduling.
2. Reduces Computational complexity of a multi-constrained NP-hard problem into a single-objective, rule-based optimization framework with a time complexity of $(T \cdot n \cdot \log(n))$, making it suitable for dynamic and large-scale systems, while preserving the original problem's NP-hard nature.
3. Adapts the biphasic response principle of hormesis to balance workloads dynamically, ensuring system stability and efficiency by preventing overutilization and underutilization of resources.
4. Evaluate the effectiveness of the HBO algorithm on the Dynamic Job Shop Scheduling (DJSS) problem using a virtual machine setup and synthetic data.

Further, this paper is organized into following sections: Section 2 provides a review of related optimization methods, the basic concepts of hormesis principle, and the key observations which motivated us to develop the hormesis principle as optimization algorithm for computational problems. Section 3 explains the theoretical and mathematical aspects of HBO framework. Section 4 introduces the problem statement for implementing HBO algorithm in DJSS scenario, the pseudocode for implementation, the time complexity analysis, and about the datasets used to evaluate the HBO algorithm. Section 5 outlines the experimental setup, as well as the results and comparative analysis. Finally, Section 6 concludes the study and states the potential directions for future research.

2. Background

2.1 Related Work

This section reviews research on addressing the challenges in optimization methods, particularly focusing on mathematical models, heuristic techniques, machine learning (ML)-based models, and hybrid approaches. These challenges include computational intensity, rigidity, local optima issues, scalability, and adaptability of the existing approaches in dynamic environments [13, 14, and 15].

Mathematical models, such as linear and integer programming, have been commonly used in resource allocation due to their precision in solving well-defined problems [16, 17]. For example, Hou et al. [16] applied a discrete-

event mathematical model to optimize vehicle scheduling, showcasing its capability in structured environments. Similarly, Laisupannawong et al. [17] explored Mixed-Integer Linear Programming (MILP) in short-term scheduling for PCB manufacturing. However, these models struggle with real-time adaptability and scalability due to the curse of dimensionality, as demonstrated by [18 and 19]. These challenges make such models computationally intensive and inflexible in dynamic environments, rendering them less efficient for large-scale optimization.

Heuristic methods such as Genetic Algorithms (GA) [20], Particle Swarm Optimization (PSO) [21], and Tabu Search (TS) [22, 23] provide more computational efficiency compared to mathematical models, offering near-optimal solutions. For instance, Shao et al. [22] applied adaptive combinations of Tabu Search and Simulated Annealing for scheduling tasks, while Sujitjorn et al. [23] enhanced Tabu Search with adaptive mechanisms to improve convergence. However, Yang [24] and Ruan et al. [25] discussed, these methods still face challenges in multi-modal and dynamic optimization scenarios, particularly with local optima issues. Additionally, Ruan et al. [26] highlighted both the advantages and limitations of heuristic techniques in addressing dynamic changes, particularly when they are hybridized with ML techniques.

Machine learning models, particularly Neural Networks and Deep Learning, have become powerful tools for optimization [27, 28]. Sijabat and Parodos [27] reviewed the application of ML models for scheduling and routing, emphasizing their computational complexity. Wang et al. [28] and Jiang [29] also noted the black-box nature of deep learning models, which can complicate decision-making and reduce transparency in critical real-time systems. Although these models are highly effective in handling complex, high-dimensional problems, they require vast datasets and substantial computational resources, limiting their applicability in resource-constrained environments [30, 31].

Hybrid approaches attempt to overcome these limitations by combining the strengths of heuristic and ML-based models. Vega et al. [32] developed a hybrid framework combining regression analysis and metaheuristics, which balanced exploration and exploitation. Tao et al. [33] applied reinforcement learning with heuristic methods in distributed hybrid flowshop problems, showcasing potential benefits but also pointing out that such combinations can introduce additional computational overhead. The review by Ruan et al. [26] and other studies, such as [34, 35], also underlined the pros and cons of hybrid models, noting how these approaches improve solution quality but often add to the computational load, making them challenging for real-time applications.

In response to these challenges, bio-inspired algorithms have emerged as promising methods for dynamic optimization. Slime Mold Algorithm (SMO) [36], developed by Li et al., has been effectively applied in resource allocation tasks, showing a strong ability to adapt to dynamic workloads while avoiding local optima. Wang et al. [37] further demonstrated the success of SMO in load balancing within cloud computing environments. Likewise, Harris Hawk Optimization (HHO) [38], as introduced by Heidari et al., and later applied by Farjallah et al. [39] in job scheduling, demonstrated faster convergence and scalability in large-scale scheduling tasks compared to traditional heuristics. Another bio-inspired method, the Whale Optimization Algorithm (WOA) [40], has also been successfully integrated into job shop scheduling, providing a balance between computational complexity and system adaptability. Nadimi-Shahraki et al. [41] reviewed improvements to WOA, emphasizing its scalability and suitability for real-time dynamic environments.

The role of nature-inspired algorithms has also extended into various applications, such as the Internet of Things (IoT) and healthcare systems. Amiri et al. [42] conducted a systematic review of nature-inspired algorithms, highlighting their adaptability and efficiency in handling complex real-world problems, such as those found in IoT-based healthcare services. These algorithms are valued for their scalability and ability to manage optimization in resource-constrained environments [43, 44].

Although these bio-inspired methods provide adaptability and efficiency in complex systems, another category is rule-based deterministic approaches, which offer the advantage of predictability and lower computational costs. Ruan et al. [31] highlighted that rule-based strategies, while less flexible, provide robust, low-computation solutions in real-time applications. This makes them an attractive option where reliability and speed are paramount.

In conclusion, while traditional mathematical models, heuristic methods, and ML-based techniques have all contributed significantly to optimization research, their limitations in handling dynamic environments, computational intensity, and adaptability are clear as well. But, the Bio-inspired algorithms, such as SMO, HHO, and WOA, have successfully addressed many of these challenges by introducing adaptive optimization mechanisms.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Therefore, the Hormesis-Based Optimization (HBO) algorithm offers to solve optimization problems by combining the flexibility of bio-inspired methods with the reliability of rule-based approaches. This means that HBO can adapt to changing environments like bio-inspired algorithms but also maintains the predictability and low computational cost of rule-based systems. As a result, HBO is particularly effective for managing resources in systems where conditions are constantly changing, and quick decisions are needed. By using the principle of hormesis, which involves small adjustments to improve performance, HBO can handle complex tasks that other methods struggle with, ensuring steady and efficient operation even when conditions are unpredictable.

2.2 Basic Concepts of Hormesis

After reviewing existing optimization algorithms and their limitations, we introduce the fundamental concepts of hormesis, which inspired the development of the Hormesis-Based Optimization (HBO) algorithm.

In nature, certain chemical, physical, or biological factors may exist, which when introduced, can trigger some kind of responses in a biological system [11, 12]. These factors are referred to as stressors. Stressors can produce either positive or negative impacts on the system, depending on their intensity or dose. The effect of a stressor depends on its doses applied. At lower doses, stressors often activate beneficial response and at higher doses, the same stressors may become harmful and cause damage. This phenomenon, known as hormesis, exists in nature and is also utilized by scientists in experiments. For example, low doses of ionizing radiation enhance cellular repair mechanisms and immune function, while higher doses cause harmful effects in chemical hormesis [45]. Similarly, lifting extremely heavy weights can damage muscle tissue or even cause bone fractures. However, lifting smaller, manageable weights over time can strengthen muscles and improve overall fitness. Weights will be the stressor in this case and the amount of weights will be the doses.

When studied in controlled conditions, the results obtained in hormesis typically form a biphasic curve, which shows two phases: a beneficial effect at lower doses and a harmful effect as the dose increases beyond a specific threshold.

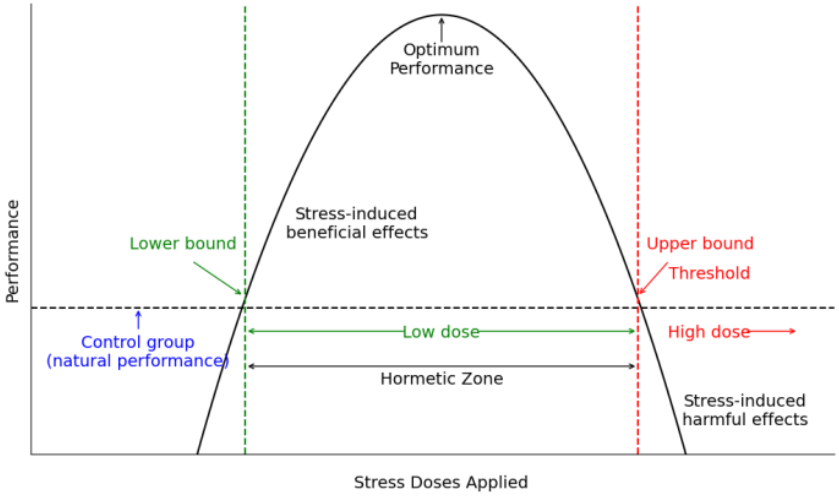

Figure 2: Visualization of Hormesis Effect and Biphasic Dose-Response Curve

Figure 2 illustrates the general concept of biphasic dose-response curve. At low doses the stressor induces beneficial effects, enhancing performance beyond the natural baseline established by a control group, which is marked as the "natural performance" or "Control group" on the graph. This region of beneficial response is termed as the "Hormetic Zone." As the dosage increases and crosses a lower threshold, it leads to optimum performance, which is the peak of the curve, signifying the highest performance achieved under stress-induced stimulation. Beyond this peak, as the dose continues to rise, it surpasses an upper threshold where the effects become detrimental, leading to a decline in performance. This high dose area is characterized by stress-induced harmful effects, marking the negative impact of excessive stress. This figure uses a parabolic curve to visually represent how varying levels of a stressor doses can differently impact performance. The green vertical line marks the lower

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

boundary of effective dosage, and the red line indicates the upper boundary beyond which the effects are adverse [45].

2.3 Motivation

Further, the following points highlight the intrinsic properties of hormesis principle [46, 47, and 48] which motivated us to develop it as an optimization algorithm:

i. Hormesis as a Mathematical Idea

The concept of the biphasic dose-response curve, explained in Section 2.2, is evident in various studies [49, 50 and 51] and has been generalized into mathematical equations. For instance:

- Chemical hormesis:

$$\boldsymbol{R(D) = \frac{aD}{1+bD}} \quad \textbf{...(2.1)}$$

Equation (2.1) captures initial stimulation and saturation at higher doses [49].

- Physical hormesis:

$$R(D) = R_{0+}aD - bD^2 \quad \textbf{...(2.2)}$$

Equation (2.2) explicitly models a decline at higher doses [50].

- Biological hormesis:

$$\frac{dH(t)}{dt} = \alpha U(t) - \beta H(t) \quad \textbf{...(2.3)}$$

Equation (2.3) emphasizes on time-dependent adaptation [51].

These equations demonstrate the sensitivity of the response $R(D)$ to changes in dose 'D'. Inspired by this behaviour, a method can be developed to continuously monitor system metrics and recalibrate adaptively, ensuring optimal performance while respecting the biphasic response.

ii. Homeostasis

Homeostasis, in hormesis, is the tendency of a system to maintain balance when under stress. There are various biological systems which naturally implements homeostasis through regulatory mechanisms and studies have, directly or indirectly, exploited the mathematical aspects of these mechanisms. For example, Equation (2.3) can be used to regulate this concept because it models adaptability. The parameters 'α' defines the system's ability to adapt and 'β' governs the rate of recovery, where, $U(t)$ represents the applied stress and $H(t)$ is response to the stressor. As an instance, this model can be used in computational systems to regulate the response to varying workloads.

iii. Overcompensation:

It is the ability of the system to adapt and become stronger to handle increased stress. The biological systems, when exposed to continuous optimal stress, attain a steady response state which can be derived from Equation (2.4) as:

$$H_{steady} = \frac{\alpha}{\beta} U(t) \quad \textbf{...(2.4)}$$

However, they just do not return to H_{steady} but also retains a residual effect giving them increased potential to handle the future stress. This can be expressed as in Equation (2.5):

$$H_{final} = H_{steady} + H_{residue} \quad \textbf{...(2.5)}$$

It means that $H_{final} > H_{steady}$. This concept can be applied to systems beyond biology, modelling them to adjust to changes and enhance performance over time.

iv. Threshold Dose and Hormetic zones

It is the point on the curve where the shift from beneficial to harmful effects occurs. Idea of using this threshold can be crucial for effective hormesis application. The threshold dose, $D_{threshold}$, is the point where $\frac{dH(t)}{dt} = 0$, separating beneficial and harmful effects. This mathematical insight inspires to define "hormetic

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

zones (Z)" as ranges of stress where response $R(D)$ from dose D remains optimal, and can be written as $Z \in Z_{min}, Z_{max}$.

v. Stressors

In biological systems, hormesis is triggered by stressors which may be natural or externally induced. These stressors act as signals stimulating adaptive responses. Similarly, in non-biological systems, certain parameters can serve as triggers to activate an optimization process. For instance, in computational environments, excessive workload can be treated as a stressor, initiating adaptive mechanisms in the system. By identifying and defining such stressors, we can control and optimize adaptation, in a non-biological system as well.

3. HBO Framework

From the discussions so far, we derive the fundamental steps of the HBO algorithm as a general problem-solving framework designed to address diverse optimization challenges by continuously observing and fine-tuning system performance through the allocation and balancing of stress across different components. The following explanation describes the process involved in HBO optimization further.

3.1 The Workflow

Imagine a system $'M'$ consisting of machines $\{M_1, M_2, \ldots, M_n\}$ each processing a subset from a set of tasks. The tasks may be assigned to the machines randomly or using a predefined strategy. The state of each machine is represented by metrics such as utilization, queue length, latency, and throughput, which are combined into a single value called stress. This stress value indicates whether a machine is overburdened, stable, or underutilized, reflecting its current workload and task-handling efficiency. Ideally, each machine operates within an optimal range of stress, between overburdened and underutilized states, where the machine's performance is considered stable. This optimal range mirrors the hormetic zone in the principle of hormesis. The goal is to ensure that all machines stay within these zones while optimizing the overall performance of the system.

To achieve this goal, stress is dynamically transferred to machines with lower stress levels. The amount of stress adjusted from an overburdened to an underutilized machine corresponds to the concept of a "dose" of stress in hormesis. For instance, if M_3 is overloaded, then its tasks are redistributed to less stressed machines in $'M'$. When these adjustments are made, the machines respond to the adjustments by producing modified response values due to the induced stress dose. This is analogous to a "dose-response" in hormesis and is referred to as the "stress metric" because the value produced signifies stress which is characterized by the combination of system parameters. These adjustments aim to improve the overall system performance by balancing workloads, preventing some machines from being overburdened while others remain underutilized.

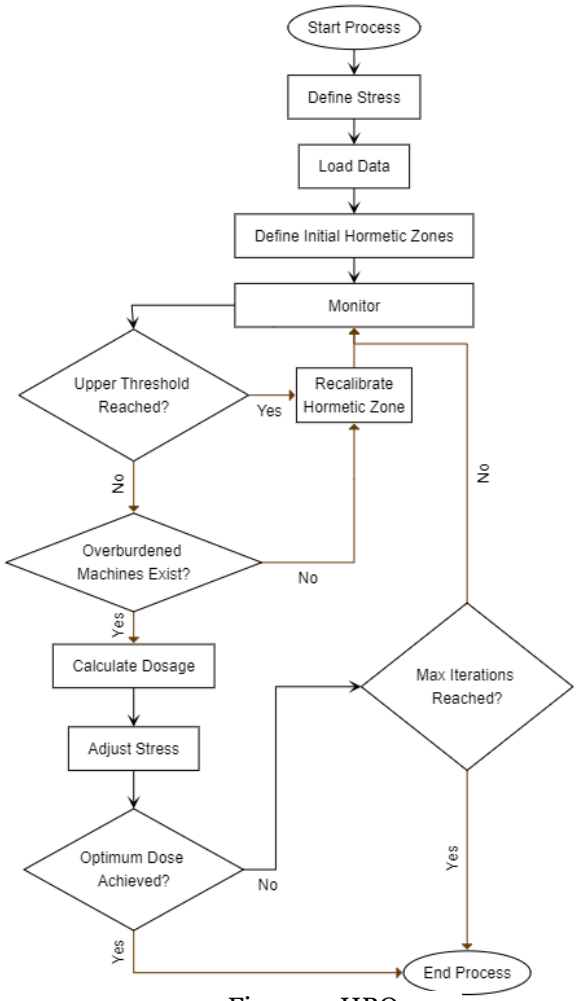

Figure 3: HBO

The algorithm begins by defining the system's stress metrics and loading the necessary historical or real-time performance data to establish baseline configurations.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Initial hormetic zones are defined for each machine, setting the lower and upper bounds for optimal stress levels. The system is continuously monitored to ensure performance stays within these predefined thresholds. If any machine exceeds the upper threshold or becomes overburdened, the algorithm recalibrates the hormetic zones and calculates the necessary dose of stress to transfer. The adjustment is made dynamically, redistributing tasks to underutilized machines to bring all components back into their optimal stress zones. These recalibrations continue iteratively, with the algorithm verifying whether the adjustments have stabilized performance and brought all machines within their hormetic zones. If the adjustments succeed or the maximum number of iterations is reached, the process concludes.

This approach ensures that the HBO algorithm dynamically adapts to fluctuating workloads while maintaining system performance within optimal stress ranges. The iterative recalibration of hormetic zones and dynamic stress adjustments forms the foundation for achieving efficient and balanced optimization in complex, dynamic environments. Next, we present the mathematical formulation of the steps outlined in this example. The equations provided are intended to illustrate the framework and may vary depending on the specific scenario or implementation of the algorithm.

3.2 Mathematical Foundations

The following steps outline the generic structure of the HBO framework, which can be adapted to various domains.

i.Initialization

- **a.** Define the system components and relevant performance metrics such as $M = \{M_1, M_2, \dots, M_n\}$ and $S = \{S_1, S_2, \dots, S_n\}$ where M_i represents components of a system (e.g. resources, subsystem, etc.) and S_i represents the metrics (e.g. workload, latency, utilization, etc.) used to evaluate the state of a component M_i.
- **b.** Prepare or load data necessary for establishing baseline performances (if necessary).

ii.Unified Metric Calculation (Define "stress metric")

A core principle of the HBO algorithm is the transformation of multiple system parameters into a unified metric $R(D)$, referred to as "stress metric". This approach reduces the computational complexity and serves as the basis for the algorithm's decision-making logic, enabling dynamic adjustments and optimizations. The method for combining multiple metrics into a unified metric can vary depending on the specific requirements of the application. For instance, as represented in Equation (3.1), a weighted sum approach can be used to aggregate system metrics, with weights reflecting the relative importance of each parameter.

$$R(D) = w_0 S_0 + w_1 S_1 + \cdots w_n S_n \quad \text{...(3.1)}$$

In this equation, S_i is the system parameter and w_i is its corresponding weight.

iii.Performance Threshold (Define Hormetic Zones)

Define the method for calculating "Hormetic Zones" for the system and find initial lower (Z_{min}) and upper (Z_{max}) bound for the hormetic zones. The approach for establishing these bounds may vary depending on the specific requirements of the system. For example, we can apply a percentile-based method to set the upper and lower bound values. These values will set the definition of the acceptable performance for system M and its components.

iv.Activation function (Define Stressor)

As already explained in section 3.2 stressor is an agent which acts as a trigger and activates the hormesis effect which helps system optimize its performance. Any condition, parameter value, or situation which may demand the optimization in the system can be used to define stressor function for that particular system. For instance, if the components of the system are experiencing increased workload beyond upper bound Z_{max}, then it can be treated as a stressor and can be used to activate the optimization mechanism of the algorithm. The trigger condition of the stressor ΔD, calculated in Equation 3.2, can be stated as Equation 3.3:

$$\Delta D = R(D) - Z_{max} \quad \text{...(3.2)}$$

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

$$if\ (\Delta D > 0) \quad \textbf{...(3.3)}$$

, i.e., if the current value of stress metric exceeds upper bound then activate homeostasis mechanism.

v.Adaptive mechanism (Homeostasis)

Homeostasis is the tendency of the system to maintain balance when it is experiencing stress. However, there is a limit of how much stress can be applied on the system at a given time so that it is able to maintain its balance. Once the activation function triggers the adaptive mechanism, the system may try to remain balanced by optimizing its performance. It can be achieved by adjusting the excessive stress to underutilized components of the system. This adjustment will be analogous to inducing stress dose in biological systems. The feature of homeostasis may be realized through following steps:

a. Recognizing the overloaded and underutilized system components through stress metric monitoring using the condition given by equation 3.4 and equation 3.5:

$$R(\mathrm{D}) > Z_{\max} \rightarrow \text{Overload} \quad \textbf{...(3.4)}$$

$$R(\mathrm{D}) > Z_{\min} \rightarrow \text{Underutilized} \quad \textbf{...(3.5)}$$

b. Define the stress dose for underutilized system components using the equation, such as,

$$D = \theta \cdot f_{\mathrm{w}} \cdot \Delta D \quad \textbf{...(3.6)}$$

, where,'θ' is the redistribution factor, which controls the proportion of excess stress that is reallocated to underutilized components to prevent system-wide imbalance.' f_w' is the weight function, defining the relative share of stress each underutilized component receives based on its current state and capacity to handle additional workload, and ΔD is the excessive stress calculated in equation 3.6 in the overloaded system components.

c. Adjust the load to the underutilized components of the system after defining the corresponding stress dose. This can be performed as given by equation 3.7:

$$R(D_i) \leftarrow R(D_{i-1}) + D_i \quad \textbf{...(3.7)}$$

d. Remove the excessive stress from the overloaded components of the system using the mechanism which can be modelled using the equation 3.8:

$$R(D_i) \leftarrow R(D_{i-1}) - k.\Delta D \quad \textbf{...(3.8)}$$

The adjustment factor $'k'$ here controls the magnitude of the correction needed to bring the component's metric $R(D)$ back to the target range $[Z_{min}, Z_{max}]$.

The equation 3.4 to 3.8 presents generalized way in which a system is following an adaptive method to maintain its performance under continuous and steady stress implementing the mechanism of homeostasis.

vi.Recalibrations (Overcompensation)

Overcompensation in hormesis lets a biological system to adapt so that it can improve the performance under the increased amount of stress for longer duration. It enhances the systems capability permanently. In non-biological systems, the overcompensation step may be implemented through recalibration of hormetic zones. As an example, equation 3.9 and 3.10 represents the procedure mathematically,

$$Z_{min}^{t+1} = \gamma Z_{min}^{t} + (1-\gamma) \cdot f_h(R(D)) \quad \textbf{...(3.9)}$$

$$Z_{max}^{t+1} = \gamma Z_{max}^{t} + (1-\gamma) \cdot f_h(R(D)) \quad \textbf{...(3.10)}$$

Here, $'\gamma'$ is the recalibration factor which governs that how much the previous threshold values influence the new threshold. Generally, $'\gamma'$ close to 1 means past values will dominate more and the change will be gradual and value close to 0 means dominance of recent values and faster change. 'f_h' is the function that determines how the stress metric influences the updated threshold.

vii.Convergence

The final step in the HBO framework is to determine convergence. This involves evaluating the stability of system metrics $R(D_i)$ and checking if they have stabilized within a predefined tolerance or if the maximum number of iterations is reached.

After the theoretical understanding of the HBO algorithm framework in this section, we validate its effectiveness in the next section.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 1: Parameter Symbol Description

Parameter Description	:	Parameter Symbol
Set of Machines	:	$M = \{M_1, M_2, \dots, M_m\}$
Set of Jobs	:	$J = \{J_1, J_2, \dots, J_n\}$
Processing time of job J_i	:	p_i
Unified Stress Metric	:	$R(D_j)$
Stress Metric for machine M_j at time t	:	$R(D_j, t)$
Integral Parameters set (μ_j) for $R(D_j)$ for machine M_j	:	$\mu_j = \{U_j, L_j, Q_j, T_j\}$ Machine Utilization (U_j) Latency (L_j) Queue Length (Q_j) Throughput $(T_j)\}$
Makespan machine M_j	:	C_j
Set of weights for Stress parameters	:	$\lambda = \{\lambda_U, \lambda_L, \lambda_Q, \lambda_T\}$
Lower & Upper Hormetic Bounds	:	$Z = \{Z_{min}, Z_{max}\}$
Hormetic Bounds for Machine M_j	:	$Z_j = [Z_{(j,\min)}, Z_{(j,\max)}]$
Set of overloaded Machines $[R(D_j) > Z_{(j,\max)}]$	:	$M_{overload}$
Set of underutilized Machines $[R(D_j) < Z_{(j,\min)}]$	:	$M_{underutil}$
Excess stress $(R(D_j) - Z_{(j,\max)})$	:	ΔD
Stress Dose (for redistribution to an underutilized machine M_j)	:	D_j
Adjustment avoidance threshold	:	τ
Time step	:	t
Recalibration factor	:	γ
Adjustment factor	:	k
Redistribution factor	:	θ

4. HBO Implementation

To evaluate the HBO algorithm in a real-time scheduling environment, we apply it to the Dynamic Job Shop Scheduling (DJSS) problem [52]. This section presents the problem formulation and details the algorithmic steps required for implementation. It also includes Table 1 which lists the symbols used in this work and their meaning.

4.1 Problem statement and Formulation

Dynamic Job Shop Scheduling (DJSS) involves assigning a set of job $J = \{J_1, J_2, \dots, J_n\}$ to a system of machines $M = \{M_1, M_2, \dots, M_m\}$ where each job $'J_i'$ has a processing time $'p_i'$. Unlike static scheduling, DJSS requires real-time adaptability as new jobs may arrive unpredictably, and machine workloads continuously change.

In this work, we treat the set of machines, capable of executing identical tasks, as a single system M. A set of jobs J is introduced into the system M the assignments $\{J_1, J_2, \dots, J_n\}$ to $\{M_1, M_2, \dots, M_m\}$ are determined dynamically using HBO algorithm. The scenario allows that the jobs can be partially processed on multiple machines, meaning that:

$$\sum_{j=1}^{m} x_{i,j} = \alpha_i\,,\ 0 \le \alpha_i \le 1, \forall\, i \in J \qquad \textbf{...(4.1)}$$

In Equation 4.1, $x_{i,j} \in \{0,1\}$ is a binary variable indicating whether job J_i is assigned to machine M_j. Also, each machine in the system M is characterised by stress metric which encapsulates parameters – utilization, queue length, task latency, and throughput. These stress components are combined into a single function in equation 4.2:

$$R(D_j) = \lambda_U U_j + \lambda_Q Q_j + \lambda_L L_j + \lambda_T T_j \qquad \textbf{...(4.2)}$$

where $\lambda_U, \lambda_Q, \lambda_L, \lambda_T$ are weight factors that determine the relative influence of each metric. The goal is to optimize the schedule in such a way that the system M is able to minimize the total completion time (makespan) and reduce the average waiting time (latency). The makespan of the system M for a given set of jobs J, given by equation 4.3 is:

$$C = \max_{j \in M} \sum_{i \in J} x_{i,j} \cdot p_i \qquad \textbf{...(4.3)}$$

The latency of such a system can be given as in equation 4.4:

$$L = \max_{j \in M} \left\{ \frac{1}{t_n - t_0} \int_{t_0}^{t_n} R(D_j, t)\, dt \right\} \qquad \textbf{...(4.4)}$$

where, $t_n - t_0$ is the total time elapsed and the integral computes the total stress (workload) by machine M_j. The machine with maximum

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

makespan and latency dictates the makespan and latency of the system M.
Instead of explicitly minimizing makespan and latency the HBO algorithm optimizes stress balancing across machines, which indirectly influences both performance metrics by keeping the performance within hormetic zones (Z_{min} , Z_{max}) given by equations (3.4) and (3.5). Hence, the objective function, Equation 4.5, can be written in terms of stress metric as:

$$min \sum_{j=1}^{m} |R(D_j, t) - R(D_j, t-1)| \quad \text{...(4.5)}$$

with the constraint of satisfying the Equation 4.6,

$$Z_{min} \leq R(D_i, t) \leq Z_{max}, \forall\, i \in M, \forall t \quad \text{...(4.6)}$$

This objective function ensures that the workload across all machines remains balanced by minimizing the deviation of machine stress from the optimal stress level $[Z_{min} , Z_{max}]$. By keeping each machine's stress within an optimal range, the system avoids overloading certain machines while others remain underutilized. The use of absolute values ensures that both overburdened and underutilized machines are corrected equally, preventing uneven adjustments. The indirect relationship between stress balancing and scheduling performance is expressed by Equation 4.7:

$$C \approx f(R(D)) \, , L \approx g(R(D)) \quad \text{...(4.7)}$$

where, $f(\cdot)$ and $g(\cdot)$ represent the system's emergent response to stress regulation.

Algorithm 1: Initialization and Setup

Input:
Machines $M = \{M_1, M_2 , \dots , M_m\}$
Jobs $J = \{J_1, J_2 , \dots , J_n\}$
Thresholds α, β for hormetic zone boundaries
Recalibration factor γ
Adjustment factor k
Redistribution factor θ
Adjustment avoidance threshold τ

Output:
Initial correlation-derived weights λ
Initial stress metrics $R(D_j)$
Initial hormetic zones $Z_j = [Z_{(j,\min)}, Z_{(j,\max)}]$

1. **for** each machine $M_j \in M$ **do**
2. Compute correlation-derived weights λ_j
3. Compute initial stress metric $R(D_j)$
4. Define initial hormetic zones: $Z_j = [Z_{(j,\min)}, Z_{(j,\max)}]$
5. **end for**
6. Initialize global parameters: γ, k, θ *and* τ
7. **Return** $\lambda, R(D_j)$ and Z_j

4.2 HBO pseudocode for DJSS

In this section we provide the explanation that how we implement the HBO framework tailored for the DJSS problem. The different algorithms presented here talks about the different sections of the implementation with the help of respective pseudocode.

Algorithm 1 is about the initial setup of the input parameters, defining and evaluating the unified stress metric, and defining the initial hormetic zones. The algorithm proceeds by computing weights (λ_j) for each system parameter using the Pearson-correlation method. The weights reflect the relative importance of parameters such as utilization, latency, queue length and throughput. These weights are computed on the basis of historical data collected for these parameters over multiple time-steps. Using equation 4.2, we then calculate the non-normalized stress metric $R'(D_j)$ for each machine as a unified value that aggregates multiple system parameters through a weighted sum as:

$$R'(D_j) = \lambda_U U_j + \lambda_Q Q_j + \lambda_L L_j + \lambda_T T_j \quad \text{...(4.8)}$$

However, to make sure that no single parameter in the stress metric dominates due to its large value, we apply min-max normalization approach, equation 4.9, to ensure consistent scaling across different metrics:

$$R(D_j) = 1 + 99 \cdot \frac{R'(D_j) - R'(D_j)_{min}}{R'(D_j)_{max} - R'(D_j)_{min}} \quad \text{...(4.9)}$$

here, $R'(D_j)_{min}$ and $R'(D_j)_{max}$ are the minimum and maximum stress metric across all machines in the system. Min-max normalization plays a crucial role in binding all machines together within a shared, system-wide reference scale. Since the stress metric for each machine is derived from multiple system parameters with varying ranges, normalizing these values ensures that no single metric disproportionately affects the overall stress assessment. This collective reference also allows stress

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

adjustments and redistribution to happen relative to system-wide stress thresholds, mimicking biological hormesis adaptation mechanisms. The unified stress metric $R(D_j)$ simplifies performance evaluation by reducing multidimensional parameters into a single value. In the next step, the algorithm defines the hormetic zone $[Z_{(j,\min)}, Z_{(j,\max)}]$ for each machine using the percentile method, such as,

Algorithm 2: Monitoring and Stress Evaluation

Input:
Machines M
Stress metric $R(D_j)$
Hormetic zone Z_j
Adjustment avoidance threshold τ

Output:
$M_{overload}$ & $M_{underutil}$ sets.

1. Initialize $\{M_{overload}, M_{underutil}\} = \emptyset$
2. **for** each machine $M_j \in M$ **do**
3. **if** $R(D_j) > Z_{(j,\max)}$ **then**
4. **if** $|R(D_j) - Z_{(j,\max)}| > \tau$ **then**
5. Add M_j to $M_{overload}$
6. **else if** $R(D_j) < Z_{(j,\min)}$ **then**
7. **if** $|R(D_j) - Z_{(j,\min)}| > \tau$ **then**
8. Add M_j to $M_{underutil}$
9. **end for**
10. **Return** $M_{overload}$ and $M_{underutil}$

$$Z_{(j,\min)} = percentile(\{R(D_j,t)|t = 1,2..n\}, \alpha) \quad \textbf{...(4.10)}$$

$$Z_{(j,\max)} = percentile(\{R(D_j,t)|t = 1,2..n\}, \beta) \quad \textbf{...(4.11)}$$

$\{R(D_j,t)|t = 1,2..n\}$ used in equation 4.10 and equation 4.11 represents a set of past stress values over $'n'$ time steps for machine M_j. 'α' and 'β' are the percentile thresholds. The use of percentile method on historic stress metric data ensures the hormetic boundaries evolve over time, preventing rigid or outdated stress limits. Finally, the algorithm initializes the parameters such as adjustment factor (k) which determines the magnitude of stress correction applied when machines deviate from their hormetic zones, while the redistribution factor (θ) proportionally allocates excess workload from overloaded machines to underutilized ones. The recalibration factor (γ) balances stability and adaptability by controlling how much influence previous hormetic zones have on recalibrated zones. The adjustment avoidance threshold (τ) which prevents the overloaded machines to adjust minor stress to other machines.

After the initial calculations in Algorithm 1, Algorithm 2 and Algorithm 3 provide the pseudocode for implementing the homeostasis mechanism in which we first categorize machines as overloaded, or underutilized. Then, adjust the appropriate stress from overloaded to underutilized machines so that system can maintain its performance within hormetic threshold under continuous stress. Algorithm 2 clearly states the conditions which compares the current stress metric $R(D_j)$ of a machine $M_j$with the upper and lower hormetic bound to categorize the machine as overloaded or underutilized, and adds them to set $M_{overload}$ or $M_{underutil}$ accordingly. If the condition in Equation (4.12) is true, then the machine $M_j$is added to set of overloaded machines.

$$R(D_j) > Z_{(j,\max)} \quad \textbf{...(4.12)}$$

$$R(D_j) < Z_{(j,\min)} \quad \textbf{...(4.13)}$$

If Equation (4.13) is true, then it is added to set of underutilized machine. Additionally, the adjustment avoidance threshold (τ) is used as an extra condition, as states in Equation (4.14) and (4.15) because it may not be effective to redistribute small amounts of stress and it may slow down the adaptation process in the system.

$$|R(D_j) - Z_{(j,\max)}| > \tau \quad \textbf{...(4.14)}$$

$$|R(D_j) - Z_{(j,\min)}| > \tau \quad \textbf{...(4.15)}$$

If the stress metric is such that $Z_{(j,\min)} \leq R(D_j) \leq Z_{(j,\max)}$ then the machine is judged as functioning optimally and does not adjust any extra load from overloaded machine. If there are no underutilized or overloaded machines, then, the overcompensation mechanism, provided in Algorithm 4, is activated. Meanwhile, Algorithm 3 implements the remaining aspects of homeostasis mechanism by adjusting the

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Algorithm 3: Stress Redistribution and Adjustment

Input:

Machines M, $M_{overload}$ and $M_{underutil}$
Stress metric $R(D_j)$
Adjustment factor k
Redistribution factor θ

Output:

Updated stress metric $R(D_j)$

1. **For each machine** $M_j \in M_{overload}$ **:**
2. Compute and reduce excess stress:
$R(D_j) \leftarrow R(D_{j-1}) - k.(R(D_{j-1}) - Z_{(j,\max)})$
3. Compute total excess stress
$\Delta D = \sum(R(D_j) - Z_{(j,\max)})$
4. **End for**
5. **For each machine** $M_i \in M_{underutil}$ **:**
6. Compute Stress Dose for redistribution, D_i:
$D_i = \theta \cdot \frac{(Z_{(i,\max)} - R(D_i))}{\sum(Z_{(i,\max)} - R(D_i))} \cdot \Delta D$
7. Apply **stress redistribution**:
$R(D_i) \leftarrow R(D_i) + D_i$
8. **End for**
9. **Return** updated $R(D_j)\ \forall\ M_j \in M$

workload from overloaded to underutilized machines. For each overloaded machine the algorithm first calculates and reduces the excess stress using Equation (4.16) given in step 2 of the algorithm.

$$R(D_j) \leftarrow R(D_{j-1}) - k.(R(D_{j-1}) - Z_{(j,\max)}) \quad \text{...(4.16)}$$

The mechanism implements the adjustment factor (k) in this step to control the amount of excess stress which is redistributed to the underutilized machines because uncontrolled and large amount of excess stress adjustment may result into oscillatory pattern in the system. In next step, it computes the excessive stress (ΔD), using Equation (4.17) as per the current boundaries of hormetic zones for the overloaded machines in the system.

$$\Delta D = \sum(R(D_j) - Z_{(j,\max)}) \quad \text{...(4.17)}$$

The algorithm then calculates the stress dose D_i which is the amount of stress to be redistributed to each underutilized machine in the set $M_{underutil}$. The stress dose for each underutilized machine, Equation (4.18), is proportional to the remaining capacity of the underutilized machine, which is the difference between the upper hormetic bound ($Z_{(i,\max)}$) and its current stress metric $R(D_i)$, included as step 6.

$$D_i = \theta \cdot \frac{(Z_{(i,\max)} - R(D_i))}{\sum(Z_{(i,\max)} - R(D_i))} \cdot \Delta D \quad \text{...(4.18)}$$

The redistribution factor (θ) in this step determines how much of the calculated excess stress (ΔD) is actually redistributed across the underutilized machines, ensuring a balanced redistribution of stress to the underutilized machines. Step 7 shows that how the dose D_i is added as additional stress to the underutilized machines as per Equation (4.19).

$$R(D_i) \leftarrow R(D_i) + D_i \quad \text{...(4.19)}$$

Hence, the mechanism provided in Algorithm 2 and Algorithm 3 tries to maintain balance in the system without hampering the efficiency of non-overloaded machines and optimizes the performance while adhering to the principles of homeostasis.

Algorithm 4 implements the overcompensation mechanism of hormesis by recalibrating the hormetic zones. Overcompensation represents the enhancement in the capability of the system to manage continuous stress. The algorithm adjusts the hormetic zones to better handle unattended excess stress (ΔD) through the Equations (4.20) and (4.21) outlined in steps 2 and 3 of the algorithm:

$$Z_{j,min}^{t+1} = \gamma Z_{j,min}^{t} + (1-\gamma) \cdot percentile(R(D_j, \alpha)) \quad \text{...(4.20)}$$

$$Z_{j,max}^{t+1} = \gamma Z_{j,max}^{t} + (1-\gamma) \cdot percentile(R(D_j, \beta)) \quad \text{...(4.21)}$$

Algorithm 4: Recalibration of Hormetic Zones

Input:

Machines M
Stress metric $R(D_j)$
Recalibration factor γ
Hormetic zone Z_j

Output:

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Updated hormetic zones Z_j

1. **For each machine** $M_j \in M$ **:**
2. Compute **new lower hormetic bound**:
   $Z_{j,min}^{t+1} = \gamma Z_{j,min}^{t} + (1-\gamma) \cdot percentile(R(D_j, \alpha))$
3. Compute **new upper hormetic bound**:
   $Z_{j,max}^{t+1} = \gamma Z_{j,max}^{t} + (1-\gamma) \cdot percentile(R(D_j, \beta))$
4. Update **hormetic zone bounds**:
   $Z_j = [Z_{(j,\min)}, Z_{(j,\max)}]$
5. **End for**
6. **Return** updated hormetic zones Z_j

Recalibration occurs under the following specific conditions in the scenario:

a. *Excess stress is not fully adjusted by underutilized machines:* This may happen if the capacity of underutilized machines is insufficient to absorb the excess load (ΔD), requiring recalibration to redefine stress thresholds and redistribute workloads more effectively.

b. *No more underutilized machines in the system:* If all machines are operating at or above their minimum hormetic thresholds $(Z_{(j,\min)})$, there are no eligible machines to redistribute excess stress, necessitating recalibration to expand the capacity of hormetic zones.

c. *No overloaded machines in the system*: If no machine exceeds its upper threshold ($Z_{(j,\max)}$), recalibration ensures the system is dynamically adjusted to maintain balance and optimize performance in the absence of stress triggers. These conditions are periodically evaluated by the algorithm, ensuring that the system makes continuous effort to achieve optimal performance.

The HBO algorithm operates iteratively, dynamically balancing system stress metrics (D_j) within predefined hormetic zones Z_j. Convergence occurs when all machines operate within their respective hormetic zones, ensuring balanced workload distribution. However, the algorithm terminates only when all jobs in the system are completed, marking the end of task scheduling and resource allocation.

4.3 Asymptotic Time complexity of HBO for DJSS

In this section, we analyse the time complexity of the HBO algorithm for DJSS scenario. We break down the complexity of different components and then derive the overall complexity. The following theorem proves the claim that the HBO algorithm worst time complexity is $O(T \cdot n \cdot \log(n))$, where, '$n$' be the number of machines and 'T' be the number of time steps or jobs and each machine have 'd' data points for intermediate adaptive calculations.

4.3.1 Lemmas for Individual Components

Lemma 1 (Data Loading and Pre-processing Complexity) *Let 'n' be the number of machines and 'T' be the number of time steps (jobs). The complexity of reading and pre-processing data is* $O(T \cdot n)$.

Proof: Each machine is associated with multiple parameters, such as utilization, throughput, queue length, and latency. The pre-processing step involves handling missing values, normalizing data, and computing statistical relationships among these parameters. Since 'n' machines' data is processed over 'T' time steps, the total complexity of this step is $O(T \cdot n)$.

Lemma 2 (Correlation and Composite Index Calculation Complexity) *Let 'p' be the number of parameters per machine. The computation of correlation values and the composite index for 'n'* machines *has complexity* $O(p^2 \cdot n)$.

Proof: The correlation matrix requires pairwise correlation calculations among 'p' parameters for 'n' machines. This results in $O(p^2 \cdot n)$ operations. Since 'p' is a small constant (e.g., $p = 4$), this term remains manageable.

Lemma 3 (Initial Hormetic Zone Calculation) *The computation of hormetic-zones for each machines involve percentile calculations, which require sorting. If there are 'n' machines and each machine have 'd' data points for percentile calculation, the worst-case time complexity is* $O(n \cdot d \cdot \log(d))$.

Proof: The process of determining hormetic zones, Equation (4.10) and Equation (4.11), involves computing percentiles, which requires sorting operations over 'd' data points. Since, sorting a list of size

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

$'d'$ data points incurs a computation cost of $O(d \cdot \log(d))$ and this operation is performed for each machine individually. Therefore, the overall complexity for the set of $'n'$ machines remains $O(n \cdot d \cdot \log(d))$.

Lemma 4 (Monitoring, Stress Redistribution and Adjustment) *The complexity of monitoring, redistribution and adjustment of stress across 'n' machines and 'T' number of time steps is* $O(T \cdot n)$.

Proof: It constitutes following steps:

Step 1: Identification of Overloaded and Underutilized Machines:
The algorithm determines which machines are overloaded or underutilized, using Equation (4.12) and (4.13). Since this requires scanning all $'n'$ machines, this step has a complexity of $O(n)$.

Step 2: Avoiding Redistribution for Minor Deviations:
The adjustment avoidance threshold, defined in Equation (4.14) and Equation (4.15), ensures that small fluctuations do not trigger unnecessary reallocations resulting in a time complexity of $O(n)$ for $'n'$ machines

Step 3: Redistribution of Excess Workload:
Once overloaded and underutilized machines are identified, the algorithm redistributes excess stress using Equation (4.16) and Equation (4.17). The redistribution factor in Equation (4.18), and Equation (4.19), ensures proportional stress adjustment to underutilized machines. This involves iterating over all $'n'$ machines again, leading to an additional time complexity of $O(n)$.

Hence, total time complexity across *'T'* time steps is*:* $m \cdot (O(n) + O(n) + O(n)) \equiv O(T \cdot n)$

Lemma 5 (Recalibration of Hormetic Zones) *The complexity of dynamic hormetic zone recalibrations, which involves recalculating stress thresholds periodically, is:* $O(\frac{T}{k} \cdot n \cdot \log(d))$*, where* $'k'$ *represents the adjustment period, meaning the hormetic zones are recalculated every time steps.*

Proof: The recalibration method in Equation (4.20) and Equation (4.21) uses percentile method which gives the time complexity of $O(d \cdot \log(d))$ (lemma 3). Over total $'T'$ time steps, hormetic zone recalibration occur once every $'k'$ number of step. Therefore, the total number of recalculations over $'T'$ time steps is $\frac{T}{k}$. Since recalculations happen for each of the $'n'$ machines, the total complexity becomes:

$$O(\frac{T}{k} \cdot n \cdot (d \cdot \log(d))) \equiv O(\frac{T}{k} \cdot n \cdot \log(d))$$

4.3.2 Overall Time Complexity

Theorem 4.1 *Let 'n' is the number of machines, T is the number of time steps (jobs), k is the adjustment period for recalibrations, and d is the number of data points used in percentile calculations of each machine. If the Hormesis-Based Optimization (HBO) algorithm computes stress metrics via correlation and normalization (Lemma 2), determines hormetic zones via percentile-based sorting (Lemma 3), performs task allocation and stress redistribution at each time step (Lemma 4), and periodically recalibrates stress thresholds (Lemma 5), then its total computational complexity over T iterations, in worst case, is:*

$$O(T \cdot n \cdot \log(d))$$

Proof: The total time complexity for the HBO algorithm for DJSS can be computed as the sum of the time complexities derived in Lemma 1 to Lemma 5, given by:

$$O(T \cdot n) + O(p^2 \cdot n) + O(n \cdot d \cdot \log(d)) + O(T \cdot n) + O(\frac{T}{k} \cdot n \cdot \log(d))$$

$\Rightarrow O(T \cdot n) + O(n \cdot d \cdot \log(d)) + O(\frac{T}{k} \cdot n \cdot \log(d))$ {$\because O(p^2 \cdot n)$ is ignored as per lemma 2}

$\Rightarrow O(T \cdot n) + O(n \cdot d \cdot \log(d)) + O(T \cdot n \cdot \log(d))$ {if $k = 1$ i.e. recalibration after every step (lemma 5)}

$\Rightarrow O(T \cdot n \cdot \log(d))$ {$\because$ T>>d and $k = 1$ in worst case}

□

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The time-complexity analysis establishes that HBO algorithm achieves worst case time complexity which is much faster than the exhaustive search based algorithm or the data intensive machine learning algorithms.

4.4 Datasets Used

This study uses synthetic data to evaluate the proposed model. It allows controlled testing under different workload conditions, ensures reproducibility, and enables scalability across various machine setups, providing a structured way to analyze the algorithm's effectiveness while mimicking realistic scheduling challenges [53]. The dataset includes historical performance data, real-time operational data, and job processing times for machines in a job scheduling system. The historical data captures machine performance trends like utilization, queue length, throughput, and latency, while the real-time data continuously updates as jobs are processed. The current real-time data becomes historical data for future scheduling decisions, ensuring continuous adaptation in task assignments.

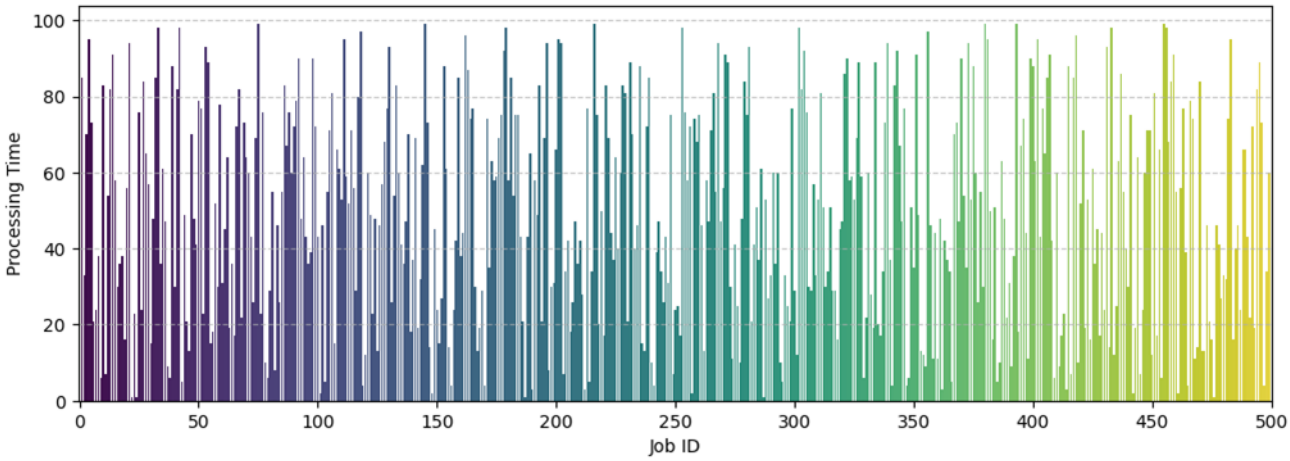

Figure 4: Processing Time per Job (500 Jobs)

Table 2: Parametric Values Used

Parameter (Applicable Algorithms)	:	Value or Range
Number of Machines	:	5 and 10
Adjustment Period (HBO)	:	6 to 10
Adjustment Factor (HBO)	:	1.3 to 1.9
Redistribution Factor (HBO)	:	1.2 to 1.5
Threshold (HBO)	:	0.3 to 0.6
Temperature (SA, ASA)	:	Initial: 1000, Minimum: 1
Cooling Rate (SA, ASA)	:	0.9
Tabu List Size (TS, ATS)	:	100
Max Iterations (TS, GA, ATS, AGA)	:	100
Population Size (GA, AGA)	:	100
Crossover Rate (GA, AGA)	:	0.8
Mutation Rate (GA, AGA)	:	0.1

Figure 4 shows the job processing times of 500 jobs which are used in executing the scenario. The processing times vary significantly across different jobs, ranging from short tasks requiring minimal machine time to longer tasks that create bottlenecks in the scheduling system. By incorporating this variability, the dataset reflects real-world scheduling scenarios where job durations are not uniform, influencing machine workload distribution.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

5. Results and Analysis

In this section, we present the results and analysis of the HBO algorithm's performance in optimizing the makespan and latency in DJSS reactive-predictive scenario [52]. We evaluate the effectiveness of HBO by comparing its ability with other well-known optimization methods. These include traditional approaches like Tabu Search (TS), which has outperformed many newer algorithms, along with Simulated Annealing (SA), Genetic Algorithm (GA), their adaptive versions ATS, ASA, AGA, and a baseline heuristic that uses a round-robin job scheduling mechanism. By selecting these methods, we aim to compare HBO against a diverse set of established optimization techniques, providing a clear understanding of its performance in complex scheduling and resource allocation tasks.

5.1 Simulation setup

The simulation setup includes two different scenarios, with 5 machines and 10 machines system, for 50, 100, 150, 250 and 500 jobs. It allows us to evaluate the scalability and adaptability of the HBO algorithm. By analysing systems of varying sizes, we can assess its performance in handling both smaller and larger job scheduling environments, ensuring its effectiveness across different levels of complexity and resource availability.

Each algorithm used in this study was configured following the best practices recognized in the domain, which ensures that the comparison framework is robust, aligning with guidelines that stress the importance of consistent benchmarking conditions for a precise evaluation of algorithmic performance [54].

Key parameters set for the simulation are outlined in the Table 2, which includes an adjustment period for HBO ranging from 6 to 10 iterations, adjustment factors between 1.3 and 1.9, redistribution factors from 1.2 to 1.5, and thresholds between 0.3 and 0.6. These parameter values were chosen based on empirical testing to determine the settings that maximize performance outcomes across different job counts and machine configurations.

For Simulated Annealing, we configured an initial temperature of 1000, cooling to 1, and a cooling rate of 0.9, adhering to the thermal reduction strategy used in various studies as a standard [22]. This approach ensures that the SA algorithm gradually transitions from exploration to exploitation, preventing premature convergence to suboptimal solutions. The tabu list size for Tabu Search was set to recommended value of 100 to effectively navigate the solution space [22], which is crucial for avoiding cycles and ensuring coverage of diverse solutions. Meanwhile, the Genetic Algorithm was parameterized with a population size of 100, a crossover rate of 0.8, and a mutation rate of 0.1, aligning with standard guidelines for achieving a balance between genetic diversity and convergence, optimizing the algorithm's ability to find optimal solutions efficiently [20].

The simulations were conducted using Python, allowing for a fair and accurate evaluation of their performance [55]. Throughout the simulation, detailed logs of machine utilization rates, processing times, and queue dynamics were maintained which are crucial for subsequent detailed analysis, providing a granular view of each algorithm's operational impact and efficiency.

5.2 Results Obtained

In this section, we detail the performance outcomes of our comparative study between the HBO algorithm and chosen conventional and adaptive scheduling methods. We present this data through a series of tables and charts, highlighting the specific improvements in makespan and latency that the HBO algorithm achieved under various job load and machine configurations. These results will help in understanding the practical impact of our approach on job scheduling efficiency.

Table 3 presents the makespan values for a 5-machine system across different job sizes, comparing HBO with heuristic methods (TS, GA, SA, ATS, AGA, ASA) and a baseline scheduling approach. The results show that HBO consistently achieves the lowest makespan across all job sizes, indicating its superior ability to optimize task scheduling. For small workloads (50 jobs), the improvement over TS is around 9.46% and 3.37% over ATS, while for larger workloads (500 jobs), HBO reduces makespan by 8.63% compared to TS and 8.23% compared to ATS. The largest reduction is against the SA which is over 22%. These reductions confirm that HBO effectively balances stress distribution, minimizing job completion time across machines.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 3: Makespan Values for 5 Machines

# Jobs	HBO	TS	ATS	GA	AGA	SA	ASA	Baseline
50.0	3666.68	4049.42	3794.59	4581.55	4129.05	4702.22	4366.45	4697.5
100.0	7635.63	7693.36	7790.35	9022.54	8930.88	9324.66	9379.97	9663.16
150.0	12045.93	12113.32	12164.31	14035.54	14017.05	14323.53	14506.24	14720.52
250.0	19586.66	20580.9	20399.97	22830.78	22931.09	23362.34	23703.13	23821.78
500.0	38917.02	42583.1	42410.17	45542.08	45572.94	46402.71	46724.3	46778.83

Table 4 shows the makespan values for a 10-machine system across different job sizes. In this case also, the HBO algorithm consistently outperforms all other methods, demonstrating its scalability and efficiency in handling larger machine configurations. For smaller workloads (50 jobs), HBO achieves a approximately 3.5% reduction in makespan compared to both TS and ATS, around 20% improvement over ASA and improvement as high as 61% against the baseline method. As the workload increases, HBO's advantage becomes more pronounced. For 500 jobs, HBO reduces makespan by 11.03% compared to TS, 10.53% as compared to ATS, 17.88% compared to GA, 18.34% compared to SA, and 19.40% over ASA. These results indicate that HBO effectively distributes stress across machines, preventing bottlenecks in high-demand scheduling environments.

Table 4: Makespan Values for 10 Machines

# Jobs	HBO	TS	ATS	GA	AGA	SA	ASA	Baseline
50.0	3567.77	3700.82	3697.1	4374.62	4361.02	4410.7	4475.04	9349.15
100.0	7106.57	7576.8	7448.29	8860.86	8919.9	9170.48	9341.43	9349.15
150.0	11343.46	11270.44	11422.94	13325.43	13197.96	13283.42	13668.14	13767.93
250.0	18214.6	18986.34	18898.82	21553.42	21729.53	21948.84	21976.08	22382.38
500.0	36132.68	40607.51	40391.63	44004.99	44065.23	44243.69	44829.75	45038.48

Table 5 presents the mean latency values for a 5-machine system across different job sizes. HBO consistently achieves lower latency compared to TS, GA, SA, ATS, AGA, ASA and the baseline. For 50 jobs, HBO reduces latency by 21.6% compared to TS, 16.28% compared to ATS and 32.8% compared to GA. As the workload increases, HBO maintains this advantage. For 500 jobs, it achieves a 8.5% reduction over TS, 8.14% compared to ATS and 14.4% over GA, demonstrating its ability to maintain lower delays across varying loads. The results indicate that HBO schedules tasks more effectively, preventing excessive queuing and maintaining system responsiveness.

Table 5: Mean Latency Values for 5 Machines

# Jobs	HBO	TS	ATS	GA	AGA	SA	ASA	Baseline
50.0	63.53	80.99	75.89	91.63	82.58	94.04	87.33	93.95
100.0	75.18	76.93	77.9	90.23	89.31	93.25	93.8	96.63
150.0	80.39	80.76	81.1	93.57	93.45	95.49	96.71	98.14
250.0	78.96	82.32	81.6	91.32	91.72	93.45	94.81	95.29
500.0	77.91	85.17	84.82	91.08	91.15	92.81	93.45	93.56

Table 6: Mean Latency Values for 10 Machines

# Jobs	HBO	TS	ATS	GA	AGA	SA	ASA	Baseline
50.0	72.56	74.02	73.94	87.49	87.22	88.21	89.5	93.49
100.0	72.81	75.77	74.48	88.61	89.2	91.7	93.41	93.49
150.0	76.48	75.14	76.15	88.84	87.99	88.56	91.12	91.79
250.0	74.06	75.95	75.6	86.21	86.92	87.8	87.9	89.53
500.0	72.57	81.22	80.78	88.01	88.13	88.49	89.66	90.08

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 6 shows the mean latency values for a 10-machine system across different job sizes. HBO continues to achieve lower latency all other methods compared. For 50 jobs, HBO reduces latency by 1.97% compared to TS, 1.84% compared to ATS and 17.0% compared to GA. As the number of jobs increases, HBO maintains its edge, with 10.65% less latency than TS, 10.16% less than ATS and 17.0% less than GA for 500 jobs. These results further demonstrate that HBO efficiently manages task delays, even in larger systems, keeping latency consistently lower compared to the other algorithms

Table 7 and Table 8 provide another perspective on the results by presenting the improvements in makespan and latency as percentage reductions, rather than absolute values, across all job set sizes. While Table 3 to Table 6 show the actual makespan and latency values, these tables highlight the relative performance gains of HBO over heuristic and adaptive methods, making it easier to quantify the extent of improvement across different configurations. The percentage values presented in these tables are an average of all the values achieved for different set of jobs i.e. 50, 100, 150, 250 and 500 jobs.

One additional insight that Table 7 and Table 8 offer is the variation in percentage improvements across different algorithms. For example, while GA and SA consistently show larger performance gaps when compared to HBO, TS remains more competitive, with relatively smaller improvements in both makespan and latency. Additionally, adaptive methods (AGA, ASA) close some of the gap with HBO, but the percentage improvements in Table 8 confirm that HBO still maintains a significant advantage with the speed much quicker than the other algorithms mentioned here, as explained in section 4.3. Notably, HBO's largest latency reduction occurs against ASA i.e. 27.25% for 5 machines and 18.93% for 10 machines.

Table 7: Percentage reduction in Makespan and Latency by HBO as compared to TS, GA and SA

Algorithm	*No. of Machines*	*Makespan Improvement (%)*	*Latency Reduction (%)*
TS	5	4.84	7.38
	10	4.85	3.45
GA	5	15.65	17.89
	10	17.30	16.09
SA	5	17.67	19.84
	10	18.31	17.12

Table 8: Percentage reduction in Makespan and Latency by HBO as compared to ATS, AGA and ASA

Algorithm	*No. of Machines*	*Makespan Improvement (%)*	*Latency Reduction (%)*
ATS	5	3.71	6.41
	10	4.59	3.18
AGA	5	13.89	16.26
	10	17.35	16.14
ASA	5	17.13	19.47
	10	19.54	18.37

5.3 Discussion

The following discussion talks about the implications of our findings, analyzing how the HBO algorithm enhances operational efficiencies and adapts to dynamic conditions. We explore the significance of adaptive hormetic zones and the pattern of biphasic responses within the broader context of optimization theory and practice. This section also compares our results to existing research, helping to frame future directions for this innovative scheduling approach.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

a) Adaptive Hormetic Zones of HBO

We analysed the dynamic adjustment of hormetic zones in the HBO algorithm over the execution of 500 time steps across 10 machines. Figure 5 provides a detailed visualization of how these hormetic bounds evolve over time, reflecting the algorithm's real-time adaptation to workload variations. Each plot represents a specific machine, capturing fluctuations in the upper and lower stress thresholds as HBO regulates task allocation. The observed trends indicate that HBO continuously refines these bounds, responding to changes in system conditions to maintain an optimal workload distribution. The horizontal gaps in the figure arise because hormetic zones are updated only when the recalibration conditions mentioned in section 4.2 are met and those conditions are checked at given intervals rather than continuously, meaning adjustments occur at specific time steps rather than in every instance. This interval-based recalibration ensures that the system adapts efficiently without excessive oscillations and providing the speed to algorithm without increasing the time complexity, as explained in Lemma 5 of section 4.3.1.

It can also be observed in Figure 5 that the stress across different machines become more similar over time by gradually aligning, indicating that HBO is dynamically adjusting scheduling to maintain balance as job demands fluctuate. Each machine's upper and lower bound values fluctuate initially, reflecting variations in workload distribution. However, as execution progresses, these fluctuations become more synchronized across machines, suggesting that the hormetic adjustment mechanism is stabilizing the system over time which satisfies the objective function, given by Equation 4.5 in conjunction with Equation 4.6, which advocates that the stress distribution should minimize with time during execution This pattern highlights HBO's ability to distribute tasks efficiently, preventing overloading of specific machines while ensuring optimal utilization across the system.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

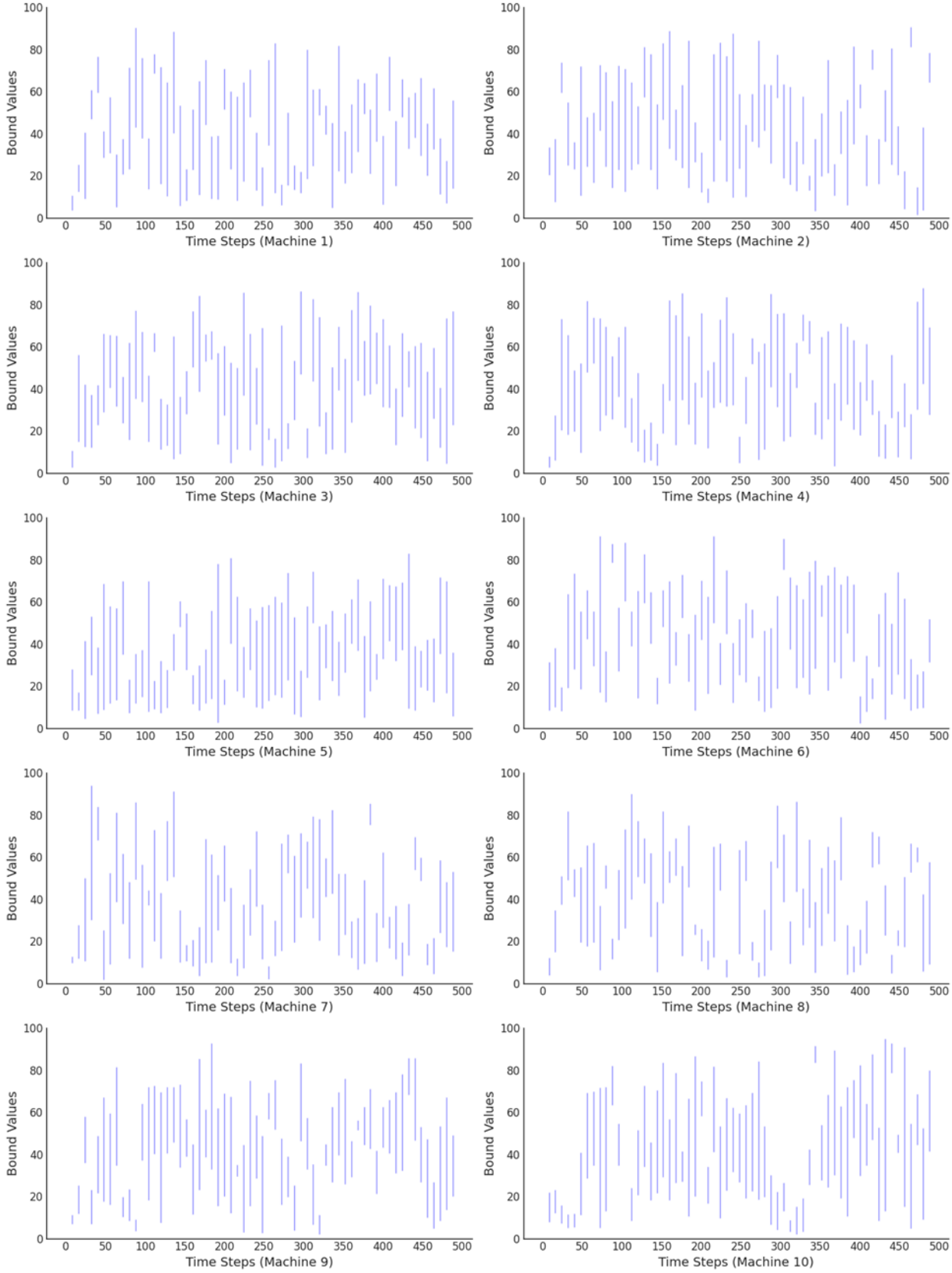

Figure 5: Variation of Hormetic Zones with Time-steps (10 Machine Scenario)

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

b) Biphasic Nature of HBO

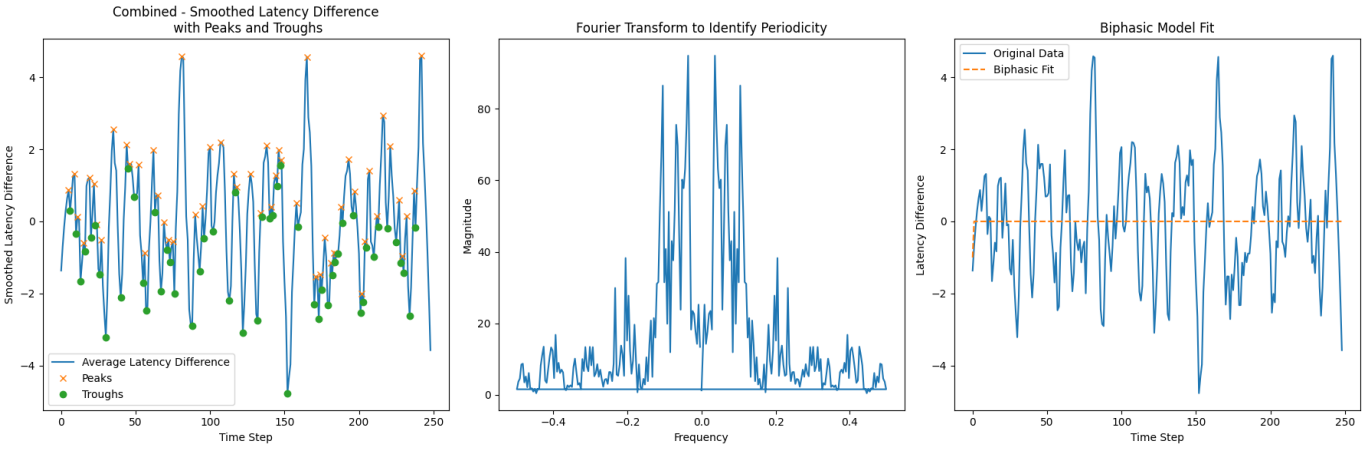

Figure 6: Evaluation of Biphasic Nature of the HBO Algorithm

To evaluate the biphasic nature of the dose-response curve in our study, we analysed latency differences of machines under varying operational stressors. First, we calculated the latency difference at successive time steps for each machine, capturing the immediate impact of job allocations, using the Equation (5.1),

$$\Delta D_t = R(D_t) - R(D_{t-1}) \quad \textbf{...(5.1)}$$

We then smoothed these differences using a moving average to highlight underlying patterns using the Equation (5.2):

$$(\overline{R(D_t)} = \frac{1}{n}\sum_{i=0}^{n-1} R(D_{t-i}) \quad \textbf{...(5.2)}$$

As shown in the first graph of Figure 6, peaks (x) and troughs (o) in the smoothed data were identified by evaluating the first derivative of $\overline{D_t}$ to detect points of rate change, indicative of oscillatory behaviour. We further analysed periodicity through Fourier Transform using the Equation (5.3),

$$F(\omega) = \int_{-\infty}^{\infty} \overline{R(D_t)} e^{-i\omega t} dt \quad \textbf{...(5.3)}$$

The purpose was to reveal the frequency components and periodic patterns in the latency fluctuations. The second graph in Figure 6 depicts this periodic pattern and confirms the regular adjustments and resilience in the system.

Subsequently, the data was fitted to a biphasic dose-response model using nonlinear regression, given by the following Equation (5.4),

$$L_{i,latency}(t) = \beta_0 + \beta_1 k(t) + \beta_2 k(t)^2 + \beta_3 e^{\beta_4 \theta(t)} \quad \textbf{...(5.4)}$$

In this equation, $k(t)$ represents the adjustment factor for task allocation, $\theta(t)$ encapsulates the redistribution factor for task balancing and $L_{i,latency}(t)$ is the performance metric. Also, β_0 is the baseline latency, β_1 adjusts latency linearly with task allocation intensity, β_2 captures non-linear effects of task adjustments, and β_3, β_4 modulate latency exponentially based on task redistribution.

This model allowed us to quantitatively describe the biphasic nature of the response with respect to both hormetic zone adjustments and stressor-induced changes in task dynamics which is shown in the third graph of Figure 6 illustrating the biphasic model fit over the smoothed latency difference data. This explanation also demonstrates how the model continuously monitors and adjusts workloads, resulting in improved functionality and maintaining system performance within optimal stress levels. Lastly, The biphasic model parameters obtained by applying the equations explained in this section, is

[6.57733699e-01, -1.36452318e+00, -7.05820785e-05, 1.21485163e-02, 1.00000003e+00]

Positive and negative coefficients in the model underscore an initial increase in latency followed by a decrease, illustrating the biphasic response.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Hence, we can say that the HBO algorithm follows the biphasic property found naturally in hormesis, which enables it to handle stress by first reacting to it and then adapting. This dual-phase approach is critical because it allows the system to initially respond to increase in stress by boosting performance, and then stabilize by adjusting operations to prevent overload, thus maintaining overall system stability and efficiency.

c) Performance Comparison

The line graphs shown, in Figure 7 and Figure 8, further illustrate the comparative performance of the HBO algorithm against other algorithms. The graphs plot makespan and mean latency as functions of the number of jobs for both 5-machine and 10-machine configurations. For the makespan versus number of jobs graphs, the HBO algorithm consistently shows a lower makespan across different job sizes, demonstrating its efficiency in managing and distributing tasks. The reduction in makespan is more prominent as the number of jobs increases, highlighting HBO's ability to handle larger

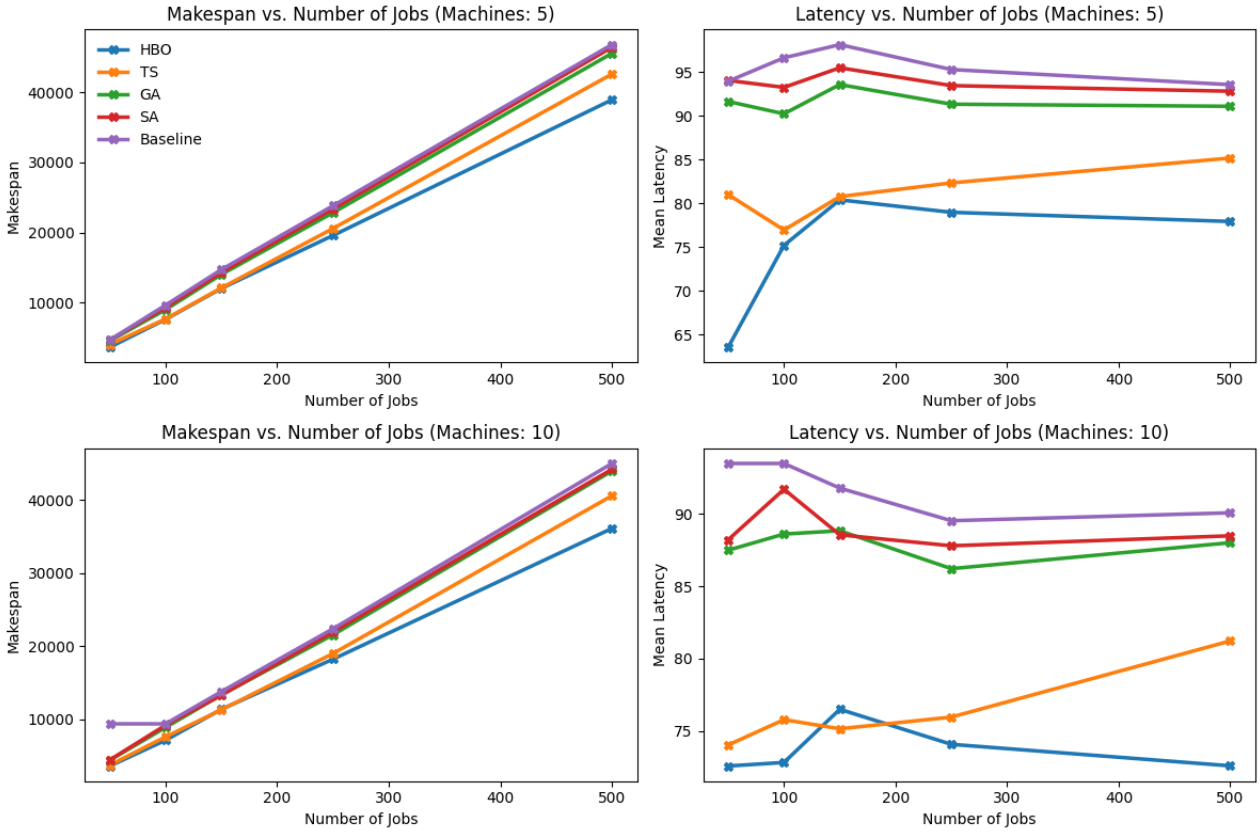

Figure 7: Makespan and Latency comparison (HBO, TS, GA and SA)

workloads more effectively compared to other algorithms. The latency versus number of jobs graphs reveals that HBO maintains a lower average latency, especially noticeable in scenarios with a high number of jobs. This indicates that HBO not only balances the workload efficiently but also ensures that the system responds promptly to task requirements, reducing delays. The comparative performance is particularly notable in the adaptive versions, where HBO outperforms ATS, AGA, and ASA in terms of both makespan and latency. The adaptive nature of these algorithms typically aims to improve performance dynamically, yet HBO demonstrates superior adaptability and efficiency in workload management. These findings confirm that HBO consistently outperforms both traditional and adaptive optimization algorithms, demonstrating its robustness and effectiveness in dynamic scheduling and resource allocation scenarios. The significant improvements in makespan and latency reduction underscore the potential of HBO in enhancing overall system performance.

Journal of Information Systems Engineering and Management
2025, 10(49s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

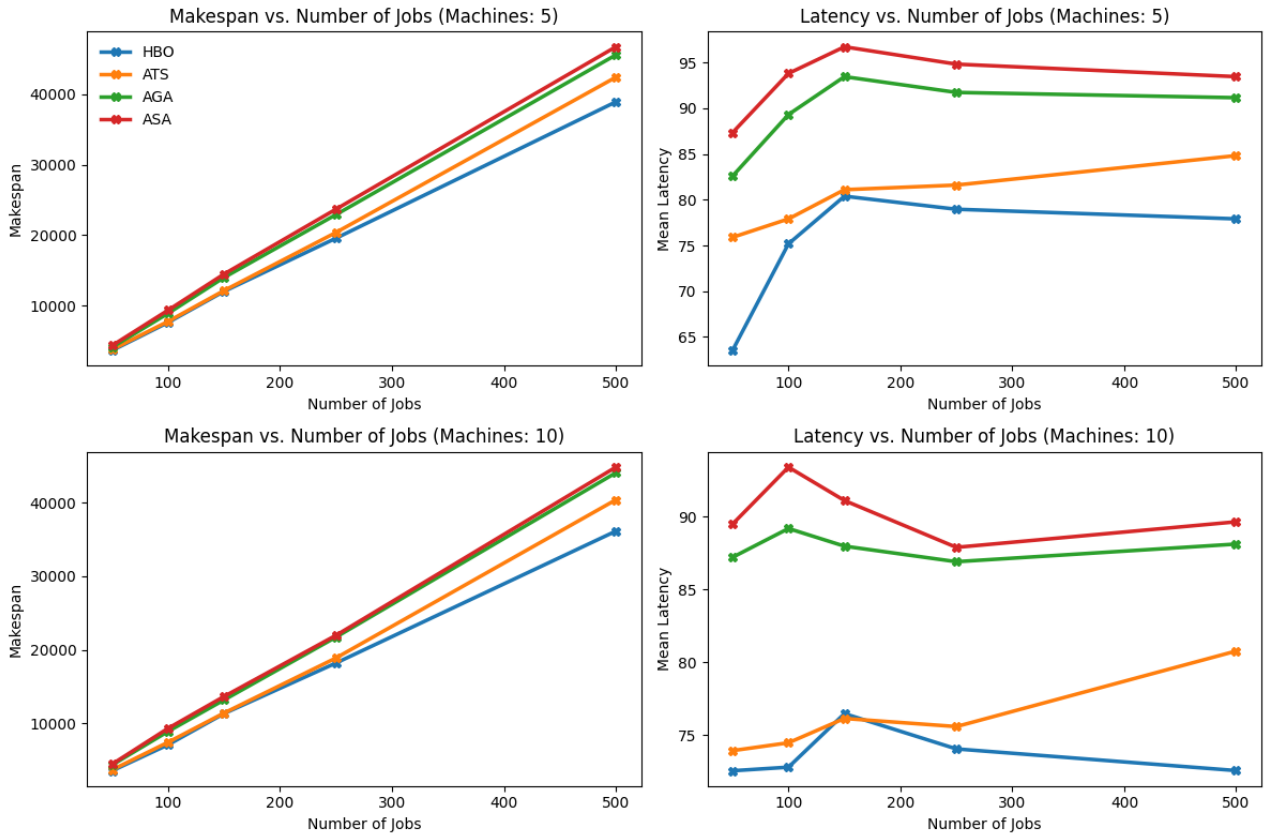

Figure 8: Makespan and Latency comparison (HBO, ATS, AGA and ASA

6. Conclusion & Future Scope

The Hormesis-Based Optimization (HBO) algorithm presented in this study uses the principles of hormesis as a framework to improve resource allocation and system performance dynamically. By maintaining stress levels within hormetic zones and leveraging adaptive responses, HBO ensures that system components operate efficiently and effectively. The evaluation results demonstrated that HBO significantly reduces makespan and latency compared to traditional and adaptive optimization algorithms. Specifically, HBO achieved an average makespan improvement of up to 18.31% and a latency reduction of up to 19.84% compared to conventional algorithms, and up to 20.27% improvement in makespan and 27.25% reduction in latency against adaptive variants. These substantial improvements underscore the potential of HBO in optimizing dynamic workload management and resource planning, paving the way for its application in various complex optimization scenarios. However, future research could extend the HBO algorithm to sectors like logistics and healthcare to assess its adaptability across diverse environments. Integrating machine learning for real-time prediction of optimal hormetic zones could enhance its responsiveness and efficiency. Exploring multi-objective optimization that balances efficiency with energy and cost considerations could lead to more sustainable practices. Long-term studies on the algorithm's impact on system maintenance and wear could provide insights into its sustainability and practical long-term benefits as well.

References:

[1] M. Del Gallo, G. Mazzuto, F. E. Ciarapica, and M. Bevilacqua, "Artificial Intelligence to Solve Production Scheduling Problems in Real Industrial Settings: Systematic Literature Review," Electronics, vol. 12, Nov. 2023.

[2] N. Jennings and R. Stadler, "Towards efficient execution of large-scale scientific applications and workflows on complex infrastructures," Journal of Cloud Computing, vol. 9, no. 16, Apr. 2020.

Research Article

[3] Gongada, T.N. et al. (2024) 'Optimizing Resource Allocation in cloud environments using fruit fly optimization and Convolutional Neural Networks', International Journal of Advanced Computer Science and Applications, 15(5).

[4] D. Lin, J. Zhao, F. Yu, W. Min, Y. Zhao and Y. L. Guan, "A Novel High-Precision and Low-Latency Abandoned Object Detection Method Under the Hybrid Cloud-Fog Computing Architecture," in IEEE Internet of Things Journal.

[5] V. P. Chassein, "Metaheuristics in Optimization: Algorithmic Perspective," INFORMS.

[6] D. T. Pham and S. O. Clark, "Intelligent Optimisation Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks," Springer London, 2017. doi: 10.1007/978-1-4471-0721-7.

[7] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016.

[8] H. Bouzary and F. F. Chen, "A classification-based approach for integrated service matching and composition in cloud manufacturing," Robotics and Computer-Integrated Manufacturing, vol. 66, Article ID 101989, 2020.

[9] V. K. Ojha, A. Abraham, and V. Snášel, "Metaheuristics in Neural Networks: A Comprehensive Overview," Neural Computing and Applications, vol. 31, no. 12, pp. 9053-9072, Dec. 2019.

[10] M. P. Mattson, "Energy Intake, Meal Frequency, and Health: A Neurobiological Perspective," The American Journal of Clinical Nutrition, vol. 101, no. 5, pp. 1237S-1243S, 2015. DOI: 10.3945/ajcn.114.100320.J. W. M. van der Meer and A. S. W. de Jong, "Mathematical Models of Hormesis in Toxicology," Environmental Toxicology and Chemistry, vol. 39, no. 5, pp. 1261-1270, May 2020. DOI: 10.1002/etc.4872.

[11] M. A. Murado and J. A. Vázquez, "The notion of hormesis and the dose–response theory: A unified approach," Journal of Applied Toxicology, vol. 35, no. 6, pp. 620-634, 2015. DOI: 10.1002/jat.3078.E. J. Calabrese and M. J. Baldwin, "Hormesis: The Dose-Response Revolution," Annual Review of Pharmacology and Toxicology, vol. 43, pp. 175-197, 2003. DOI: 10.1146/annurev.pharmtox.43.100901.140223.

[12] K. Koyama, "Exercise-induced oxidative stress: A tool for 'hormesis' and 'adaptive response'," Japan Physical Fitness and Sports Medicine, vol. 3, pp. 115-123, 2014. DOI: 10.7600/jpfsm.3.115.

[13] Singh, Harvinder, Anshu Bhasin, Parag Ravikant Kaveri, and Vinay Chavan. "Cloud resource management: comparative analysis and research issues." International Journal of Scientific & Technology Research 9, no. 06 (2020): 96-113.

[14] Ashawa, M., Douglas, O., Osamor, J. et al. "Improving cloud efficiency through optimized resource allocation technique for load balancing using LSTM machine learning algorithm". J Cloud Comp 11, 87 (2022). https://doi.org/10.1186/s13677-022-00362-x

[15] Radwan, K., Elhakeem, A. and Elbeltagi, E. (2024) 'Resource assignment optimization in design firms', Ain Shams Engineering Journal, 15(4), p. 102612. doi:10.1016/j.asej.2023.102612.

[16] Hou, Y., Mao, Y., Zhang, Y., Li, Q., and Ji, Y., "A Discrete-Event Mathematical Model for Resource Allocation Optimization: A Case Study of Vehicle Scheduling in a Signal-Free Intersection," MDPI Mathematics, vol. 10, no. 22, 2022.

[17] Laisupannawong, T.; Intiyot, B.; Jeenanunta, C. Mixed-Integer Linear Programming Model and Heuristic for Short Term Scheduling of Pressing Process in Multi-Layer Printed Circuit Board Manufacturing. Mathematics 2021, 9, 653.

[18] Z. Zhao, S. Liu, M. Zhou, X. Guo and J. Xue, "Iterated Greedy Algorithm for Solving a New Single Machine Scheduling Problem," 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC), Banff, AB, Canada, 2019, pp. 430-435, doi: 10.1109/ICNSC.2019.8743328.

[19] Wang, X., Hu, H., Liang, Y. et al. On the Mathematical Models and Applications of Swarm Intelligent Optimization Algorithms. Arch Computat Methods Eng 29, 3815–3842 (2022).

[20] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning," Addison-Wesley, Reading, MA, USA, 1989.

[21] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in Proc. IEEE Int. Conf. on Neural Networks, vol. 4, Perth, Australia, 1995, pp. 1942-1948.

[22] Shao, Xiaorui, Su Yeon Lee, and Chang Soo Kim. "A Novel and Effective University Course Scheduler Using Adaptive Parallel Tabu Search and Simulated Annealing." KSII Transactions on Internet & Information Systems 18, no. 4 (2024).
[23] S. Sujitjorn, J. Kluabwang, D. Puangdownreong, and N. Sarasiri, “Adaptive Tabu Search and Management Agent”, ECTI-EEC, vol. 8, no. 1, pp. 1–10, Dec. 2009.
[24] Yang, Xin-She. (2018). Nature-Inspired Algorithms. 297-321. 10.1002/9781119490616.ch14.
[25] G. Ruan, L. L. Minku, S. Menzel, B. Sendhoff and X. Yao, "Learning to Expand/Contract Pareto Sets in Dynamic Multi-Objective Optimization With a Changing Number of Objectives," in IEEE Transactions on Evolutionary Computation, doi: 10.1109/TEVC.2024.3375751.
[26] G. Ruan, H. Zhong, G. Zhang, Y. He, X. Wang and T. Pu, "Review of learning-assisted power system optimization," in CSEE Journal of Power and Energy Systems, vol. 7, no. 2, pp. 221-231, March 2021
[27] R. R. M. Sijabat and Z. K. Parodos, “Machine learning-based multi-objective optimization for dynamic scheduling and routing of heterogeneous instant delivery orders and scheduling strategies with real-time adaptation”, emod, vol. 16, no. 2, pp. 59–70, May 2022.
[28] S. Wang et al., "Machine/Deep Learning for Software Engineering: A Systematic Literature Review," in IEEE Transactions on Software Engineering, vol. 49, no. 3, pp. 1188-1231, 1 March 2023
[29] Jiang, W. Bike sharing usage prediction with deep learning: a survey. Neural Comput & Applic 34, 15369–15385 (2022).
[30] G. C. Ruan, H. W. Zhong, J. X. Wang, Q. Xia, and C. Q. Kang, “Neural network-based Lagrange multiplier selection for distributed demand response in smart grid,” Applied Energy, vol. 264, pp. 114636, Apr. 2020
[31] Xu, R., Cao, S., Kearns, S. K., Niechwiej-Szwedo, E., & Irving, E. (2024). Computational Cognitive Modeling of Pilot Performance in Pre-flight and Take-off Procedures. Journal of Aviation/Aerospace Education & Research, 33(4)
[32] Vega, E.; Soto, R.; Crawford, B.; Peña, J.; Castro, C. A Learning-Based Hybrid Framework for Dynamic Balancing of Exploration-Exploitation: Combining Regression Analysis and Metaheuristics. Mathematics 2021
[33] X. -r. Tao, Q. -k. Pan and L. Gao, "An Iterated Greedy Algorithm With Reinforcement Learning for Distributed Hybrid FlowShop Problems With Job Merging," in IEEE Transactions on Evolutionary Computation, doi: 10.1109/TEVC.2024
[34] C. Wu, Y. Zhou and J. Wu, "Two-Layer Data-Driven Robust Scheduling for Industrial Heat Loads," in Journal of Modern Power Systems and Clean Energy, doi: 10.35833/MPCE.2024.000105.
[35] Babatunde, Olubayo Moses, Josiah Lange Munda, and Yskandar Hamam. "A comprehensive state-of-the-art survey on hybrid renewable energy system operations and planning." IEEE Access 8 (2020)
[36] Li S, Chen H, Wang M, Heidari AA, Mirjalili S. Slime mould algorithm: A new method for stochastic optimization. Future Generation Computer Systems 2020; 111: 300–323.
[37] Wang, Zebin, Yu Li, Guodao Zhang, Xiaotian Pan, and Ji Li. "Multi-Objective Optimization of Solar Resource Allocation in Radial Distribution Systems Using a Refined Slime Mold Algorithm." Heliyon (2024).
[38] Heidari, Ali Asghar, Seyedali Mirjalili, Hossam Faris, Ibrahim Aljarah, Majdi Mafarja, and Huiling Chen. "Harris hawks optimization: Algorithm and applications." Future generation computer systems 97 (2019)
[39] F. Farjallah, H. E. Nouri and O. B. Driss, "Harris Hawks Optimization Algorithm for Dual-Resource Constrained Flexible Job Shop Scheduling Problem with Makespan Criterion," 2023 IEEE Afro-Mediterranean Conference on Artificial Intelligence (AMCAI), Hammamet, Tunisia, 2023, pp. 1-8
[40] Guanjun, M. E. N. G., H. U. A. N. G. Jiangtao, and W. E. I. Yabo. "Hybrid Beluga Whale Optimization Algorithm for Flexible Job Shop Scheduling Problem." Journal of Computer Engineering & Applications 60, no. 12 (2024).
[41] Nadimi-Shahraki, M., Zamani, H., Asghari Varzaneh, Z. et al. A Systematic Review of the Whale Optimization Algorithm: Theoretical Foundation, Improvements, and Hybridizations. Arch Computat Methods Eng 30, 4113–4159 (2023).
[42] Amiri, Zahra, Arash Heidari, Mohammad Zavvar, Nima Jafari Navimipour, and Mansour Esmaeilpour. "The applications of nature-inspired algorithms in Internet of Things-based healthcare service: A

systematic literature review." Transactions on Emerging Telecommunications Technologies 35, no. 6 (2024)

[43] Lei, Tingjun, Chaomin Luo, Simon X. Yang, Daniel W. Carruth, and Zhuming Bi. "Bio-inspired intelligence-based multi-agent navigation with safety-aware considerations." IEEE Transactions on Artificial Intelligence (2023).

[44] Zhao, Tianhao, Linjie Wu, Di Wu, Jianwei Li, and Zhihua Cui. "Multi-factor Evolution for Large-scale Multi-objective Cloud Task Scheduling." KSII Transactions on Internet & Information Systems 17, no. 4 (2023).

[45] M. A. Murado and J. A. Vázquez, "The notion of hormesis and the dose–response theory: A unified approach," Journal of Applied Toxicology, vol. 35, no. 6, pp. 620-634, 2015. DOI: 10.1002/jat.3078.

[46] E. J. Calabrese and M. J. Baldwin, "Hormesis: The Dose-Response Revolution," Annual Review of Pharmacology and Toxicology, vol. 43, pp. 175-197, 2003. DOI: 10.1146/annurev.pharmtox.43.100901.140223.

[47] M. P. Mattson, "Hormesis defined," Ageing Res. Rev., vol. 7, no. 1, pp. 1-7, Jan. 2008

[48] K. Koyama, "Exercise-induced oxidative stress: A tool for 'hormesis' and 'adaptive response'," Japan Physical Fitness and Sports Medicine, vol. 3, pp. 115-123, 2014. DOI: 10.7600/jpfsm.3.115.

[49] T. Yoshimasu, T. Ohashi, S. Oura, Y. Kokawa, M. Kawago, Y. Hirai, M. Miyasaka, H. Nishiguchi, S. Kawashima, Y. Yata, M. Honda, T. Fujimoto, and Y. Okamura, "A Theoretical Model for the Hormetic Dose-response Curve for Anticancer Agents," Anticancer Res., vol. 35, no. 11, pp. 5851-5855, Nov. 2015.

[50] K. Kino, "Calculations of the Radiation Dose for the Maximum Hormesis Effect," Radiation, vol. 4, no. 1, pp. 69-84, 2024.

[51] E. L. Kendig, H. H. Le, and S. M. Belcher, "Defining Hormesis: Evaluation of a Complex Concentration Response Phenomenon," Int. J. Toxicol., vol. 29, no. 3, pp. 235-246, 2010.

[52] Smith, A., Jones, D., and Roberts, S., "Dynamic Job Scheduling in Manufacturing and Service Systems," Journal of Manufacturing Systems, vol. 36, no. 4, pp. 216-229, 2020.

[53] V. Beiranvand, W. Hare, and Y. Lucet, "Best Practices for Comparing Optimization Algorithms," Optimization and Engineering, vol. 18, no. 4, pp. 815–848, Dec. 2017.

[54] Adams, R., "Using Synthetic Data in Simulation Studies," Journal of Simulation, vol. 14, no. 2, pp. 83-97, 2021.

[55] Lopez, M., "Advanced Data Analysis Techniques in Python," Data Science and Engineering, vol. 5, no. 1, pp. 34-45, 2022.