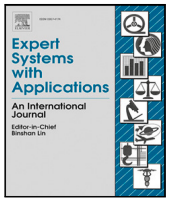
Expert Systems With Applications 237 (2024) 121327

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Variable Neighborhood Search for precedence-constrained tasks optimization on heterogeneous systems

Alejandro Humberto García Ruiz, Aurelio Alejandro Santiago Pineda *,
José Antonio Castán Rocha, Salvador Ibarra Martínez, Jesús David Terán Villanueva

Autonomous University of Tamaulipas, Faculty of Engineering Tampico, Centro Universitario Sur, Tampico, 89109, Tamaulipas, Mexico

ARTICLE INFO

Keywords:
Parallel applications
Task scheduling
Heterogeneous computing
Directed acyclic graph
Variable Neighborhood Search
Composite Local Search

ABSTRACT

Supercomputing power is one of the fundamental pillars of the digital society, which depends on the accurate scheduling of parallel applications in High-Performance Computing (HPC) centers to minimize computing times. However, precedence-constraint task scheduling is a well-known NP-Hard optimization problem, and no optimal polynomial-time algorithm exists to solve it. Therefore, as accurate as possible to the optimal values, heuristic algorithms are of relevant interest. Our new scheduling proposal name is EFT-GVNS, which stands for Earliest Finish Time - General Variable Neighborhood Search. EFT-GVNS uses a Composite Local Search (CLS), making our proposal more efficient than traditional GVNS. EFT-GVNS accuracy against four high-performance algorithms from the state-of-the-art (EDA, EFT-ILS, GRASP-CPA, MPQGA) and one reference algorithm in the literature (HEFT) is studied. Experimental results over four real-world applications (Fpppp, LIGO, Robot, Sparse) and 14 synthetic instances from the literature show that EFT-GVNS outperforms in terms of the median achieved results, with a global improvement of 37.6%, 27.4%, 17.8%, 6.1%, 2.2% to HEFT, EDA, EFT-ILS, GRASP-CPA, and MPQGA, respectively. EFT-GVNS achieves all the 14 optimal values of the synthetic benchmark.

1. Introduction

Supercomputers, also known as High-Performance Computing (HPC) systems, consist of many individual Central Processing Units (CPUs) or machines; those machines are, in most cases, heterogeneous in their processing capabilities. HPC systems execute on-demand intensive cloud and scientific applications. Therefore those applications are usually parallel, consisting of individual precedence-constrained tasks to take advantage of several processing units. Many scheduling problems are known as NP-complete (Ullman, 1975), and task scheduling for parallel systems is not an exception (Sinnen, 2007). Hence, their optimization version belongs to the class NP-Hard the most difficult in computational complexity. Several variants of task scheduling for parallel systems appear in the literature, dynamic scheduling (Nayak, Padhy, & Panigrahi, 2012), task failure (Shu, Cai, & Xiong, 2021), Internet of the things (IoT) tasks scheduling in cloud-fog (Abd Elaziz, Abualigah, & Attiya, 2021), distributed cyber–physical systems (Yi, Xu, Yan, & Huang, 2020), cloud computing (Arunarani, Manjula, & Sugumaran, 2019), edge computing (Li, Tang, Tang, & Luo, 2019), energy-aware (Wang et al., 2013), energy idle aware (Santiago et al., 2021), among others. HPC has proven to be very useful for modeling and simulation of large-scale numerical models (Niculescu, 2020), thus becoming an indispensable pillar on which researchers have focused their efforts for the development of innovative scientific advances, which they integrate HPC to make it possible to face the challenges to which industry and society daily treat. The use of HPC has become an essential aspect of optimizing different applications in multiple areas, for which, in addition to HPC, other areas of computing are integrated, such as AI. In Yi and Loia (2019), the authors present 12 research papers that have integrated HPC with AI in different applications. In this way, HPC, together with AI, has allowed the generation of applications with a high level of efficiency for decision-making in different fields of application. On the other hand, in recent years, the generation of data through different sources of information has increased considerably in such a way that for its treatment, the use of Big Data, which, together with HPC, has proven to be able to provide good results. Many sources can generate much information today, for example, vehicle communications, the information generated by social networks, and the Internet of Things, among others, thus becoming one of the leading applications in which HPC has begun to facilitate data processing (Wu,

* Corresponding author.
E-mail addresses: ahgarcia@docentes.uat.edu.mx (A.H. García Ruiz), aurelio.santiago@uat.edu.mx (A.A. Santiago Pineda), jacastan@docentes.uat.edu.mx (J.A. Castán Rocha), sibarram@docentes.uat.edu.mx (S. Ibarra Martínez), jdteran@docentes.uat.edu.mx (J.D. Terán Villanueva).

https://doi.org/10.1016/j.eswa.2023.121327
Received 21 November 2022; Received in revised form 7 July 2023; Accepted 24 August 2023
Available online 3 September 2023

Table 1
Research on the literature review section, × for no, ✓for yes.

Work	Scheduling variant	Approach	Heterogeneous System	Multi-objective or Multi-criteria
Cho et al. (2020)	Precedence real-time tasks	Heuristic	×	✓
Li, Wang, Cañizares Abreu, Zhao, and Bonilla Pineda (2021)	Cloud computing	Metaheuristic	✓	✓
Mohammad Hasani Zade, Mansouri, and Javidi (2021)	Cloud computing	Metaheuristic	✓	✓
Zhang, Zhou, and Salah (2020)	Cloud computing	Heuristic	✓	×
Ait Aba, Zaourar, and Munier (2020)	Precedence tasks	Linear programming/heuristic	✓	✓
Yi et al. (2020)	Cyber physical system	Metaheuristic	✓	✓
Kołodziej and Khan (2012)	Independent tasks	Metaheuristic	✓	✓
Zhang et al. (2015)	Precedence tasks	Heuristic	✓	✓
Zhang, Li, Li, and Li (2017)	Precedence tasks	Metaheuristic	✓	✓
Chen, Han, Wu, and Gan (2022)	Precedence tasks	Heuristic	✓	✓
Zhang, Tong, Li, and Xu (2021)	Precedence tasks	Metaheuristic	✓	✓
Santiago et al. (2021)	Precedence tasks	Heuristic	✓	✓
Gu and Budati (2020)	Cloud computing	Metaheuristic	✓	✓
Xie, Xiao, Peng, and Li (2022)	Precedence tasks	Survey	✓	✓
Pirozmand et al. (2021)	Cloud computing	Metaheuristic	✓	✓
Peng, Li, Chen, and Li (2022)	Precedence tasks	Heuristic	✓	✓
Mack, Arda, Ogras, and Akoglu (2022)	Precedence tasks	Heuristic	✓	✓
Valentini et al. (2013)	Precedence tasks	Survey	✓	✓
Kocot, Czarnul, and Proficz (2023)	Precedence tasks	Survey	✓	✓
Imene, Sihem, Okba, and Mohamed (2022)	Cloud computing	Metaheuristic	✓	✓
Mandal and Acharyya (2015)	Cloud computing	Metaheuristic	×	×
Emami (2022)	Cloud computing	Metaheuristic	✓	✓
Priyadarshini et al. (2022)	Cloud computing	Metaheuristic	✓	✓
Ghafari and Mansouri (2022)	Cloud computing	Metaheuristic	✓	✓
Hamed, Elnahary, Alsubaei, and El-Sayed (2023)	Cloud computing	Metaheuristic	✓	×
NoorianTalouki, Hosseini Shirvani, and Motameni (2022)	Cloud computing	Heuristic	✓	×
Murad, Muzahid, Azmi, Hoque, and Kowsher (2022)	Cloud computing	Survey	✓	✓
Houssein, Gad, Wazery, and Suganthan (2021)	Cloud computing	Survey	✓	✓
Menaka and Sendhil Kumar (2022)	Cloud computing	Survey	✓	✓
Hai et al. (2023)	Cloud computing	Heuristic	✓	×
Elcock and Edward (2023)	Precedence tasks	Metaheuristic	✓	×
Lin, Li, and Tian (2022)	Precedence tasks	Machine Learning	✓	×
Velarde Martinez (2020)	Precedence tasks	Metaheuristic	✓	×
Guo, Zhou, and Gu (2022)	Precedence tasks	Heuristic	✓	×
Memeti and Pllana (2021)	Not specified	Machine learning	✓	✓
Orr and Sinnen (2021)	Precedence tasks	Exact method	✓	×
Pinto and Nagano (2022)	Job Shop	Survey	✓	✓
Heydari and Aazami (2018)	Job Shop	Linear programming	×	✓
This work	Precedence tasks	Metaheuristic	✓	×

Xiang, Ge, & Muller, 2018). Another example is in Niculescu (2020), wherein the medical data analysis, HPC, can be applied. In Healthcare, some other applications have also developed in integration with areas such as electronics. For example, Alsina-Pages, Navarro, Alias, and Hervas (2017) presents the development of an electronic device based on HPC, making it possible to monitor the patient's condition remotely by detecting acoustic variations. In the same way, in the Healthcare area, machine learning applications have been developed in integration with HPC to generate applications oriented to COVID-19, the cause of one of the global pandemics with the most significant impact in recent years (Majeed & Lee, 2021). However, besides Healthcare, there are other areas where we can find the use of HPC applications. The finance area was one of the main forerunners that used HPC. A wide variety of applications have developed in this area. In Zenios (1999), the authors presented a brief summarize the origin of HPC applications and what the authors expect in a few years with the progress of this. Hand in hand with finance, recently, researchers have focused on the so-called High-Performance Business Computing, which consists of the application of HPC to solve business problems, being integrated today in diverse fields of application and what day-to-day it continues to spread widely (Schryen, Kliewer, & Fink, 2020). Another example of an HPC application in business is Jelovac, Ljubojević, and Ljubojević (2022). In addition to the above applications, HPC has been used in other fields of application, for example, Education (Chen, Impagliazzo, & Shen, 2020), Biomedical (Bastrakov et al., 2013), Remote Sensing (Lee, Gasster, Plaza, Chang, & Huang, 2011), Vehicle Systems (Kodiyalam, Yang, Gu, & Tho, 2004), Biology (Sanbonmatsu & Tung, 2007), Smart Grid (Green, Wang, & Alam, 2013), Manufacturing (Correa-Baena et al., 2018), among others. Based on the applications cited in this paper, HPC is useful in diverse fields and areas. However, the growing demand for HPC use makes it essential for scientists to find ways to reduce the computational cost of running these applications (Dan Mironescu & Vintan, 2017). The above is one of the main reasons why developing a task-scheduling algorithm with high performance continues to be a critical priority for efficient HPC applications. This paper studies the same scheduling model as in Santiago et al. (2020a), where a Directed Acyclic Graph (DAG) $G = (T, C)$ represents the precedence constraints $(t_l, t_i) \in C$, for every t_i task, and their respective edge weights the communication time between tasks (nodes), while every m_j machine's respective computing times per ith-task $p_{i,j}$ are stored in a matrix P. For tasks assigned to the same machine, communication is null. The scheduling problem consists of finding a machine/task assignment that minimizes the final computing time of the parallel application, known as makespan. The studied problem is relevant for their practical purposes in HPC systems and as a research of accurate and efficient novel methods to optimize NP-Hard problems. Recently

Table 2
Time and memory complexity for the six algorithms, time complexity without objective function evaluation. $|M|$ is the number of machines, $|T|$ is the number of tasks, $|P|$ is the population size, and k the number of neighborhoods.

	EFT-ILS	GRASP-CPA
Time	$O(\|T\| \cdot \|M\|)$	$O(\|T\| \cdot \|M\|)$
Memory	$O(\|T\|)$	$O(\|T\|)$
	EDA	MPQGA
Time	$O(\|T\| \cdot \|P\|)$	$O(\|T\| \cdot \|P\|)$
Memory	$O(\|T\| \cdot \|P\|)$	$O(\|T\| \cdot \|P\|)$
	HEFT	EFT-GVNS
Time	$O(\|T\| \cdot \|M\|)$	$O(k\|T\|^2)$
Memory	$O(\|T\|)$	$O(\|T\|)$

Table 3
Parameter settings for the six algorithms.

	EFT-ILS	GRASP-CPA
GRASP α:	–	1.0 and 0.9
Communication:	–	single-point cx, $p_r = 1.0$
Perturbation:	$P_m = 0.05$	$P_m = 0.05$
ILS max iterations:	stop criterion	5/stop criterion
	EDA	MPQGA
Selection:	roulette wheel	roulette wheel
Elitism size:	10%	10%
Crossover:	–	single-point cx, $p_r = 0.7$
Mutation:	–	$p_m = 0.35$
Population Size:	30	$\|T\| \cdot 10$
Tabu time:	–	$\|T\| \cdot 10$
Max generations:	stop criterion	stop criterion
	HEFT	EFT-GVNS
Initial tasks' priority:	b-level	b-level
EFT α:	–	0.6
Max iterations:	unlimited	stop criterion

Table 4
Friedman's average rankings of the algorithms over the set of real-world applications.

Algorithm	Ranking
EFT-GVNS	1.28
MPQGA	1.96
GRASP-CPA	3.62
EFT-ILS	3.80
EDA	5.02
HEFT	5.31

Local Search (Hoos & Stützle, 2004) has been used successfully to schedule parallel applications in heterogeneous systems (Pecero et al., 2012; Santiago et al., 2021; Xing, Zhang, Li, Gong, Yang, & Wang, 2022). However, Local Search in isolation is limited to reaching a local optimum and cannot escape it. Thus Local Searches are usually a part of more robust search methods for scheduling parallel applications as in Pineda, Pecero, Huacuja, Barbosa, and Bouvry (2013a), Qin, Pi, and Shao (2022), Santiago et al. (2020a). Variable Neighborhood Search (Hansen, Mladenović, Todosijević, & Hanafi, 2017) (VNS) is a Metaheuristic framework designed to exploit the search capabilities of different neighborhoods, as in different Local Searches. In this work, we explore the synergy between different local improvement mechanisms and compare our results with high-performance algorithms from the state-of-the-art. The contributions of this work are:

- A General Variable Neighborhood Search (GVNS), a variant of the classical VNS. Our GVNS proposal uses a Composite Neighborhood instead of the classical approach of Neighborhood Descending.
- An experimental comparison over more than 400 DAG scheduling instances on four high-performance algorithms from the literature: Estimation Distribution Algorithm (EDA), Earliest Finish Time - Iterated Local Search (EFT-ILS), Greedy Randomized Search Procedure - Cellular Processing Algorithm (GRASP-CPA), and the Multiple Priority Queues Genetic Algorithm (MPQGA). In addition, we add to the comparison the Heterogeneous Earliest Finish Time (HEFT) algorithm.
- A median values global improvement of 37.6%, 27.4%, 17.8%, 6.1%, 2.2% to HEFT, EDA, EFT-ILS, GRASP-CPA, and MPQGA, respectively
- Achieving all the optimal values from a set of 14 scheduling instances from the literature in polynomial time (without an exact method).

The structure of the paper is as follows. First this introduction (Section 1). The formal mathematical description of our studied problem is present in Section 2. Next, we summarize the most relevant works in the literature on scheduling parallel applications in Section 3, paying particular attention to heterogeneous systems. Section 4 is devoted to the detailed description of our proposed algorithm and discusses the advantages of using a Composite Local Search. Section 5 describes the experiments performed in this work, and the results obtained are in Section 6. A parameter sensitivity study is performed over EFT-GVNS in Section 7. Finally, we give our main conclusions and lines of future work in Section 8.

2. Problem description

This work represents parallel computer applications as a Directed Acyclic Graph (DAG) $G = (T, C)$ where T is the set of tasks $\{t_1, t_2, t_3, \dots, t_n\}$ of a parallel application, the set of edges $(t_i, t_j) \in C$ represent their precedences and their respective weights represent the communication costs between tasks, for tasks executed in the same machine/system the communication cost between them, is considered zero (no network bandwidth usage), positions $P_{i,j}$ from a matrix P defines the execution times of the t_i tasks in the machines $m_j \in M$ from the High-Performance Computing (HPC) system. The goal is to find an allocation $\forall t_i \in T$ on the available machines (not executing another task simultaneously) without violating the precedence constraints and communication times to minimize the parallel application computing time, known as makespan. The Eqs. (1), (2), (3), and (4) outline our studied scheduling problem. The variables rt_i, st_i, and ft_i represent the ready-to-execute time, starting time, and finish time of the i-th task, respectively, and w is the beginning of the first time window where it is possible to execute the t_i task in their allocated k machine.

$$rt_i = \max\{ft_j + C_{j,i} \ \forall (t_j, t_i) \in C\} \tag{1}$$

$$st_i = \max\left(rt_i, \min_{w \geq rt_i}\{ft_j \leq w \veebar w \leq st_j - P_{i,k} \forall t_j \in m_k\}\right) \tag{2}$$

$$ft_i = st_i + P_{i,k} \tag{3}$$

$$Makespan = \max_{\forall t_i \in T}\{tf_i\} \tag{4}$$

However, no simple, straightforward calculation exists to compute the mentioned makespan. The issue lies in the possible different orders to execute the tasks. Previously executed tasks lock time windows to execute others. The combinatory of possible orders equals $n!$ (permutations of the tasks) for the case of graphs without precedences. Even with the tasks precedences from a DAG, many permutations of tasks are still feasible, making it computationally expensive to evaluate them in the objective function. Therefore many authors follow the list scheduling approach (order of tasks execution), giving priorities of execution to the set of tasks. Two popular algorithms for task priorities are the bottom-level (b-level) and top-level (t-level) (Santiago et al., 2021). This work follows the principle of task priorities (order). Then given the order of tasks execution, we can use the same objective function as in Santiago et al. (2020a) with a complexity of $O(|T| \cdot |C|)$.

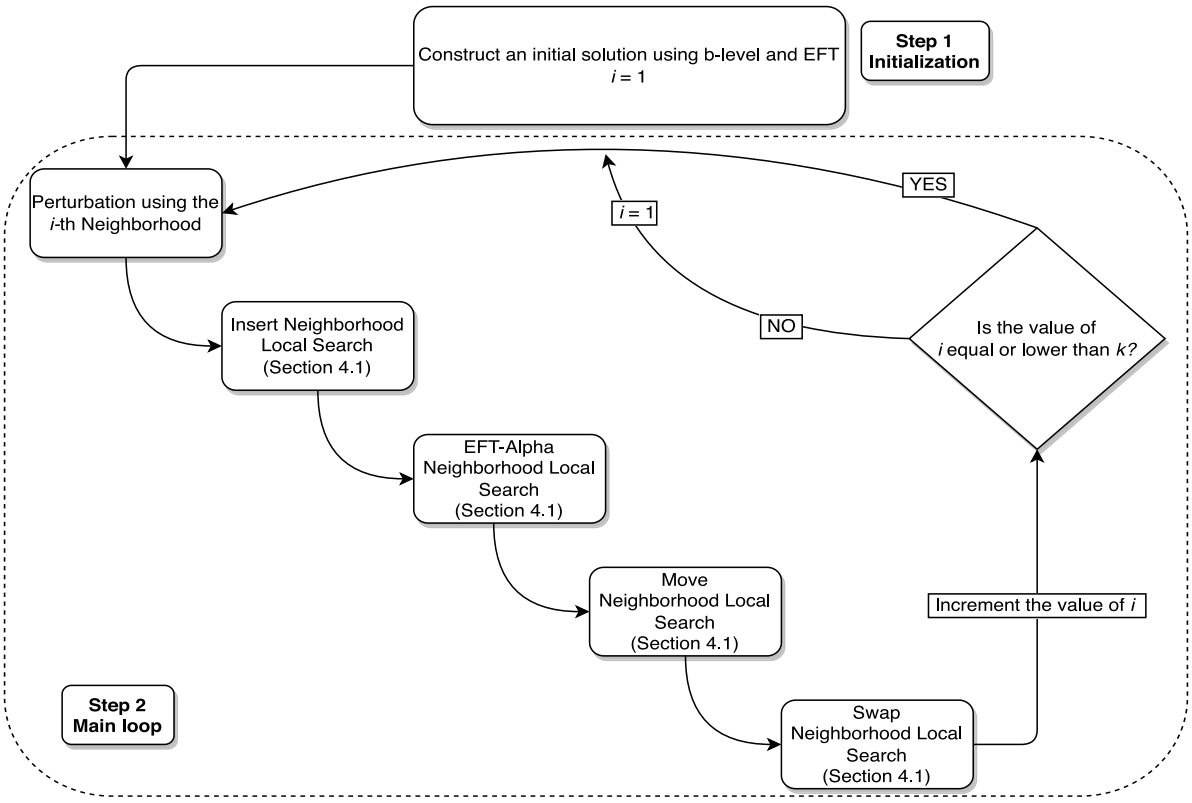


Fig. 1. EFT-GVNS flowchart, main loop iterated until the stop criterion is reach.

3. Literature review

This section surveyed some of the novel and relevant works in the literature related to our studied scheduling problem. Lately has been a high interest in focusing on different objectives/preferences of scheduling parallel applications on HPC systems more remarkable than the original makespan minimization problem. For example, some emerging variants of the scheduling problem are: minimizing the number of processors (Cho et al., 2020), quality of service (Li et al., 2021), security aware (Mohammad Hasani Zade et al., 2021), deadline constraints (Zhang et al., 2020), communication delays (Ait Aba et al., 2020), distributed cyber–physical systems (Yi et al., 2020), among others. In Kołodziej and Khan (2012), the authors treat the scheduling as a bi-objective problem, minimizing the makespan while minimizing the sum of finalization times of all the tasks in the batch (flowtime).

Other examples are in the works Zhang et al. (2017, 2015) where authors focus on low power consumption using the Dynamic Voltage Scaling (DVS) over the CPUs while at the same time focusing on maximizing the system reliability. The authors define reliability as the capacity to execute tasks at high speed without exceeding the maximum frequency of the processors. Diverse factors affect the reliability of an HPC system; hardware failures, high temperatures, or unknown reasons may make tasks fail to execute. In Chen et al. (2022), the authors present a robust model for scheduling tasks with reliability, considering the hardware fault rate and processor frequencies. It seems that for works focusing on reliability, processor frequencies are the essential factor.

The most common scenario in the literature is bi-objective, the makespan and energy consumption minimization, as in the following papers Gu and Budati (2020), Mack et al. (2022), Peng et al. (2022), Pirozmand et al. (2021), Santiago et al. (2021), Xie et al. (2022), Zhang et al. (2021). The work in Valentini et al. (2013) summarizes very well the approaches to energy efficiency in HPC systems, i.e., hardware opportunities where to apply optimization to improve the energy efficiency. They perform a general taxonomy of two main branches Static Power Management (SPM) and Dynamic Power Management (DPM). SPM systems use low-consumption hardware, e.g., power supplies, while DPM systems focus on allocating workloads using techniques such as the DVS. While SPM is related to the hardware design, DPM is related to the mathematical optimization of the workloads in the HPC systems, straight related to our studied scheduling optimization problem. The authors of the above work identify three DPM approaches to optimize HPC Systems: (i) Power-scalable processors (i.e., DVS), (ii) Power-scalable memory (i.e., DVS), and (iii) load balancing of the workload (i.e., switching on/off nodes in the HPC system.) Perhaps the most recent review on energy-aware scheduling-related problems for HPC systems can be found in Kocot et al. (2023). The authors from Kocot et al. (2023) conclude that most works focus on DPM with DVS as in Valentini et al. (2013), use applications represented by DAGs or independent tasks, consider systems heterogeneous, homogeneous, or a mixture of them, and the goals usually are energy and makespan. The authors review various techniques used for optimization: machine learning, dynamic programming, fuzzy logic, integer programming, randomized algorithms, evolutionary algorithms, and constraint programming, among others.

Cloud computing task scheduling is an extension of the scheduling in HPC systems with high interest in the literature. In cloud computing, other factors intervene as cloud clients that send tasks request, the intermediary (broker) between customers and cloud providers, and cloud virtual machines, among others. In Imene et al. (2022), the Non-dominated Sorted Genetic Algorithm III (NSGA-III) algorithm solves the Cloud computing scheduling as a three-objective problem, minimizing the runtime, cost, and energy consumption. Following, we mention other works in Cloud computing scheduling. The work in Mandal and Acharyya (2015) presents an empirical study of three different metaheuristics for task scheduling on virtual machines for the cloud; The studied metaheuristics were a Simulated Annealing, FireFly Algorithm, and Cuckoo Search. According to the authors, the best-performing algorithm was the Firefly Algorithm. We found examples of other emerging

Table 5
Medians and IQR of the six algorithms over 30 independent runs.

Problema	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
Fpppp-8-334-0.1-0.1	$1.02E2_{0.0E0}$▲	$9.84E1_{3.9E0}$▲	$9.40E1_{7.9E-1}$▲	$9.36E1_{8.7E-1}$▲	$9.32E1_{0.0E0}$▲	$8.81E1_{1.7E0}$
Fpppp-8-334-0.1-0.5	$6.63E2_{0.0E0}$▲	$6.38E2_{3.3E1}$▲	$6.03E2_{1.3E1}$▲	$6.01E2_{9.0E0}$▲	$6.09E2_{0.0E0}$▲	$5.62E2_{6.3E0}$
Fpppp-8-334-0.1-1	$1.68E3_{0.0E0}$▲	$1.56E3_{7.5E1}$▲	$1.49E3_{2.3E1}$▲	$1.48E3_{4.3E1}$▲	$1.50E3_{0.0E0}$▲	$1.37E3_{1.6E1}$
Fpppp-8-334-0.1-5	$2.73E3_{0.0E0}$▲	$2.77E3_{4.3E2}$▲	$2.67E3_{8.9E1}$▲	$2.59E3_{9.9E1}$▲	$2.36E3_{0.0E0}$▲	$2.12E3_{4.5E1}$
Fpppp-8-334-0.1-10	$2.39E3_{0.0E0}$▲	$1.96E3_{2.4E2}$▲	$2.12E3_{1.0E2}$▲	$2.03E3_{9.9E1}$▲	$1.65E3_{0.0E0}$▲	$1.52E3_{2.7E1}$
Fpppp-8-334-0.25-0.1	$8.88E2_{0.0E0}$▲	$8.91E2_{1.3E1}$▲	$8.68E2_{8.6E0}$▲	$8.63E2_{1.3E1}$▲	$8.55E2_{0.0E0}$▲	$8.16E2_{6.4E0}$
Fpppp-8-334-0.25-0.5	$2.73E3_{0.0E0}$▲	$2.77E3_{6.8E1}$▲	$2.65E3_{4.1E1}$▲	$2.63E3_{3.3E1}$▲	$2.57E3_{0.0E0}$▲	$2.45E3_{2.9E1}$
Fpppp-8-334-0.25-1	$2.08E3_{0.0E0}$▲	$2.04E3_{1.2E2}$▲	$1.95E3_{2.9E1}$▲	$1.93E3_{3.8E1}$▲	$1.90E3_{0.0E0}$▲	$1.78E3_{2.0E1}$
Fpppp-8-334-0.25-5	$5.96E2_{0.0E0}$▲	$6.13E2_{6.9E1}$▲	$5.98E2_{2.2E1}$▲	$5.78E2_{2.4E1}$▲	$4.80E2_{0.0E0}$▲	$4.56E2_{5.2E0}$
Fpppp-8-334-0.25-10	$2.77E3_{0.0E0}$▲	$2.60E3_{3.6E2}$▲	$2.82E3_{1.6E2}$▲	$2.68E3_{1.5E2}$▲	$2.30E3_{0.0E0}$▲	$2.10E3_{1.1E2}$
Fpppp-8-334-0.5-0.1	$3.97E2_{0.0E0}$▲	$3.94E2_{1.3E1}$▲	$3.89E2_{7.5E0}$▲	$3.83E2_{2.9E0}$▲	$3.85E2_{0.0E0}$▲	$3.61E2_{4.1E0}$
Fpppp-8-334-0.5-0.5	$2.53E3_{0.0E0}$▲	$2.34E3_{1.3E2}$▲	$2.24E3_{4.5E1}$▲	$2.25E3_{2.6E1}$▲	$2.28E3_{0.0E0}$▲	$2.11E3_{2.6E1}$
Fpppp-8-334-0.5-1	$2.25E3_{0.0E0}$▲	$2.13E3_{7.9E1}$▲	$2.07E3_{3.6E1}$▲	$2.05E3_{2.7E1}$▲	$1.99E3_{0.0E0}$▲	$1.87E3_{2.5E1}$
Fpppp-8-334-0.5-5	$4.60E3_{0.0E0}$▲	$4.18E3_{4.0E2}$▲	$4.23E3_{9.6E1}$▲	$4.02E3_{1.9E2}$▲	$3.73E3_{0.0E0}$▲	$3.13E3_{5.2E1}$
Fpppp-8-334-0.5-10	$2.69E3_{0.0E0}$▲	$2.64E3_{3.8E2}$▲	$2.70E3_{1.3E2}$▲	$2.61E3_{1.6E2}$▲	$1.99E3_{0.0E0}$▲	$1.87E3_{3.3E1}$
Fpppp-8-334-0.75-0.1	$3.21E2_{0.0E0}$▲	$3.10E2_{8.9E0}$▲	$3.00E2_{5.8E0}$▲	$2.97E2_{5.2E0}$▲	$2.96E2_{0.0E0}$▲	$2.77E2_{3.8E0}$
Fpppp-8-334-0.75-0.5	$4.06E2_{0.0E0}$▲	$4.10E2_{1.6E1}$▲	$3.93E2_{8.4E0}$▲	$3.90E2_{8.4E0}$▲	$3.74E2_{0.0E0}$▲	$3.57E2_{3.7E0}$
Fpppp-8-334-0.75-1	$7.81E2_{0.0E0}$▲	$8.12E2_{3.8E1}$▲	$7.60E2_{1.9E1}$▲	$7.48E2_{2.0E1}$▲	$7.25E2_{0.0E0}$▲	$6.80E2_{1.4E1}$
Fpppp-8-334-0.75-5	$3.07E3_{0.0E0}$▲	$2.95E3_{2.8E2}$▲	$2.89E3_{6.8E1}$▲	$2.78E3_{1.0E2}$▲	$2.47E3_{0.0E0}$▲	$2.25E3_{4.9E1}$
Fpppp-8-334-0.75-10	$5.27E3_{0.0E0}$▲	$4.34E3_{4.2E2}$▲	$4.84E3_{2.4E2}$▲	$4.73E3_{2.0E2}$▲	$4.10E3_{0.0E0}$▲	$3.57E3_{1.3E2}$
Fpppp-8-334-1-0.1	$8.12E2_{0.0E0}$▲	$8.13E2_{2.2E1}$▲	$7.96E2_{1.5E1}$▲	$7.84E2_{1.6E1}$▲	$7.77E2_{0.0E0}$▲	$7.51E2_{5.8E0}$
Fpppp-8-334-1-0.5	$1.45E3_{0.0E0}$▲	$1.38E3_{1.1E2}$▲	$1.32E3_{2.2E1}$▲	$1.30E3_{3.1E1}$▲	$1.28E3_{0.0E0}$▲	$1.21E3_{1.0E1}$
Fpppp-8-334-1-1	$3.71E2_{0.0E0}$▲	$3.43E2_{3.0E1}$▲	$3.21E2_{8.9E0}$▲	$3.18E2_{8.6E0}$▲	$3.08E2_{0.0E0}$▲	$2.88E2_{7.3E0}$
Fpppp-8-334-1-5	$4.17E3_{0.0E0}$▲	$4.85E3_{9.0E2}$▲	$4.66E3_{1.8E2}$▲	$4.52E3_{2.0E2}$▲	$3.61E3_{0.0E0}$▲	$3.27E3_{9.7E1}$
Fpppp-8-334-1-10	$3.52E3_{0.0E0}$▲	$2.99E3_{3.7E2}$▲	$3.41E3_{1.2E2}$▲	$3.21E3_{1.5E2}$▲	$2.64E3_{0.0E0}$▲	$2.40E3_{9.8E1}$
Fpppp-16-334-0.1-0.1	$4.21E2_{0.0E0}$▲	$3.96E2_{2.3E1}$▲	$3.71E2_{9.5E0}$▲	$3.71E2_{8.5E0}$▲	$3.61E2_{0.0E0}$▲	$3.44E2_{4.0E0}$
Fpppp-16-334-0.1-0.5	$2.19E3_{0.0E0}$▲	$2.15E3_{2.1E2}$▲	$1.92E3_{6.0E1}$▲	$1.92E3_{6.6E1}$▲	$1.85E3_{0.0E0}$▲	$1.76E3_{1.5E1}$
Fpppp-16-334-0.1-1	$8.87E2_{0.0E0}$▲	$9.61E2_{7.9E1}$▲	$8.63E2_{2.2E1}$▲	$8.76E2_{2.4E1}$▲	$7.95E2_{0.0E0}$▲	$7.58E2_{5.9E0}$
Fpppp-16-334-0.1-5	$3.58E3_{0.0E0}$▲	$3.61E3_{3.3E2}$▲	$3.26E3_{1.3E2}$▲	$3.25E3_{9.9E1}$▲	$2.68E3_{0.0E0}$▲	$2.51E3_{4.0E1}$
Fpppp-16-334-0.1-10	$6.55E3_{0.0E0}$▲	$5.81E3_{7.3E2}$▲	$5.50E3_{3.2E2}$▲	$5.39E3_{3.2E2}$▲	$4.97E3_{0.0E0}$▲	$4.36E3_{1.5E2}$
Fpppp-16-334-0.25-0.1	$2.01E3_{0.0E0}$▲	$1.93E3_{1.3E2}$▲	$1.79E3_{4.9E1}$▲	$1.79E3_{4.6E1}$▲	$1.71E3_{0.0E0}$▲	$1.66E3_{1.5E1}$
Fpppp-16-334-0.25-0.5	$2.04E3_{0.0E0}$▲	$1.97E3_{2.4E2}$▲	$1.77E3_{6.2E1}$▲	$1.77E3_{5.1E1}$▲	$1.69E3_{0.0E0}$▲	$1.63E3_{2.0E1}$
Fpppp-16-334-0.25-1	$6.10E2_{0.0E0}$▲	$5.75E2_{5.2E1}$▲	$5.15E2_{2.3E1}$▲	$5.11E2_{2.0E1}$▲	$4.90E2_{0.0E0}$▲	$4.55E2_{4.8E0}$
Fpppp-16-334-0.25-5	$1.71E3_{0.0E0}$▲	$1.97E3_{2.6E2}$▲	$1.86E3_{6.9E1}$▲	$1.81E3_{1.1E2}$▲	$1.48E3_{0.0E0}$▲	$1.30E3_{4.4E0}$
Fpppp-16-334-0.25-10	$1.02E3_{0.0E0}$▲	$7.65E2_{6.4E1}$▲	$8.03E2_{2.0E1}$▲	$7.73E2_{2.8E1}$▲	$6.39E2_{0.0E0}$▲	$6.06E2_{5.4E0}$
Fpppp-16-334-0.5-0.1	$1.65E3_{0.0E0}$▲	$1.55E3_{1.1E2}$▲	$1.46E3_{3.6E1}$▲	$1.44E3_{3.2E1}$▲	$1.42E3_{0.0E0}$▲	$1.39E3_{1.2E1}$
Fpppp-16-334-0.5-0.5	$1.66E3_{0.0E0}$▲	$1.50E3_{1.0E2}$▲	$1.35E3_{4.7E1}$▲	$1.35E3_{3.8E1}$▲	$1.29E3_{0.0E0}$▲	$1.27E3_{1.0E1}$
Fpppp-16-334-0.5-1	$4.56E2_{0.0E0}$▲	$4.29E2_{5.8E1}$▲	$3.78E2_{1.0E1}$▲	$3.75E2_{1.4E1}$▲	$3.47E2_{0.0E0}$▲	$3.38E2_{5.4E0}$
Fpppp-16-334-0.5-5	$3.23E3_{0.0E0}$▲	$3.29E3_{3.4E2}$▲	$2.86E3_{1.7E2}$▲	$2.85E3_{1.9E2}$▲	$2.42E3_{0.0E0}$▲	$2.18E3_{4.5E1}$
Fpppp-16-334-0.5-10	$8.66E2_{0.0E0}$▲	$9.09E2_{7.8E1}$▲	$9.23E2_{3.4E1}$▲	$8.96E2_{2.6E1}$▲	$7.69E2_{0.0E0}$▲	$6.08E2_{1.5E1}$
Fpppp-16-334-0.75-0.1	$3.75E2_{0.0E0}$▲	$3.45E2_{1.8E1}$▲	$3.25E2_{8.3E0}$▲	$3.22E2_{5.2E0}$▲	$3.04E2_{0.0E0}$▲	$3.00E2_{1.3E0}$
Fpppp-16-334-0.75-0.5	$6.77E2_{0.0E0}$▲	$6.25E2_{5.0E1}$▲	$5.84E2_{2.0E1}$▲	$5.88E2_{1.5E1}$▲	$5.35E2_{0.0E0}$▲	$5.31E2_{2.1E0}$
Fpppp-16-334-0.75-1	$7.40E2_{0.0E0}$▲	$6.67E2_{1.0E2}$▲	$6.05E2_{1.7E1}$▲	$6.10E2_{1.7E1}$▲	$5.51E2_{0.0E0}$▲	$5.40E2_{7.0E0}$
Fpppp-16-334-0.75-5	$4.51E3_{0.0E0}$▲	$4.03E3_{3.5E2}$▲	$3.69E3_{2.0E2}$▲	$3.65E3_{2.4E2}$▲	$3.29E3_{0.0E0}$▲	$2.83E3_{5.1E1}$
Fpppp-16-334-0.75-10	$2.41E3_{0.0E0}$▲	$2.02E3_{3.7E2}$▲	$1.99E3_{9.7E1}$▲	$1.93E3_{1.0E2}$▲	$1.80E3_{0.0E0}$▲	$1.63E3_{3.1E1}$
Fpppp-16-334-1-0.1	$8.05E2_{0.0E0}$▲	$7.76E2_{4.7E1}$▲	$7.49E2_{1.6E1}$▲	$7.40E2_{2.1E1}$▲	$7.12E2_{0.0E0}$▲	$7.05E2_{1.8E0}$
Fpppp-16-334-1-0.5	$1.16E3_{0.0E0}$▲	$1.02E3_{7.7E1}$▲	$9.45E2_{2.3E1}$▲	$9.47E2_{3.0E1}$▲	$9.17E2_{0.0E0}$▲	$8.93E2_{1.6E1}$
Fpppp-16-334-1-1	$5.34E2_{0.0E0}$▲	$4.89E2_{5.4E1}$▲	$4.56E2_{1.3E1}$▲	$4.50E2_{2.0E1}$▲	$4.06E2_{0.0E0}$▲	$4.01E2_{3.0E0}$
Fpppp-16-334-1-5	$1.86E3_{0.0E0}$▲	$1.62E3_{1.5E2}$▲	$1.45E3_{7.1E1}$▲	$1.45E3_{7.2E1}$▲	$1.28E3_{0.0E0}$▲	$1.23E3_{1.6E1}$
Fpppp-16-334-1-10	$1.11E3_{0.0E0}$▲	$9.70E2_{1.1E2}$▲	$9.78E2_{3.5E1}$▲	$9.45E2_{4.3E1}$▲	$9.15E2_{0.0E0}$▲	$7.60E2_{9.7E0}$

Dark/light gray emphasizes the best/second-best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

metaheuristics for cloud scheduling as the Sunflower Optimization Algorithm (Emami, 2022), Gravitational Grey Wolf (Priyadarshini et al., 2022), Seagull Optimization Algorithm (Ghafari & Mansouri, 2022), Cooperation Search Algorithm (Hamed et al., 2023). A study of more than eight different scheduling heuristics applied to cloud computing over four real applications (Molecular, LU-like, FFT, Montage) appears in NoorianTalouki et al. (2022). A comprehensive cloud job scheduling review appears in Murad et al. (2022); the authors propose a priority-based job scheduling algorithm. Another revision on metaheuristics for the cloud is the work of Houssein et al. (2021), which reviews some metaheuristics for task scheduling in cloud computing. Other works in cloud computing focus on their challenges, as Hai et al. (2023), Menaka and Sendhil Kumar (2022).

Recent works in HPC scheduling we want to mention are an Ant Colony Optimization (Elcock & Edward, 2023), an algorithm based on Reinforcement Learning (Lin et al., 2022), scheduling based on DAG internal structures (Velarde Martinez, 2020), CPU–GPU utilization aware (Tang & Fu, 2020), limited task duplication (Guo et al., 2022), prediction using Machine Learning (Memeti & Pllana, 2021), and partially heterogeneous systems (Orr & Sinnen, 2021), and for jobs and batches the works in Heydari and Aazami (2018), Pinto and Nagano (2022). Table 1 summarizes the literature review.

Nevertheless, we still consider the relevance of the initial single objective formulation of makespan minimization for precedence-constrained applications because makespan is related to energy consumption, reliability, other objectives, and decision-maker preferences. Moreover, precedence tasks represent parallel applications, usually the most complex and powerful. The above is the primary motivation of this paper to propose a new cutting-edge makespan minimization algorithm. The high-performance algorithms for comparison in the state-of-the-art are in Section 5.1.

4. The solution proposal variable neighborhood search and composite neighborhood

This section describes our main contribution, a Variable Neighborhood Search (Hansen et al., 2017) (VNS) algorithm that uses different

Table 6
Medians and IQR of the six algorithms over 30 independent runs.

Problema	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
Fpppp-32-334-0.1-0.1	$3.00E2_{0.0E0}$▲	$2.56E2_{2.3E1}$▲	$2.30E2_{9.2E0}$▲	$2.27E2_{7.5E0}$▲	$2.10E2_{0.0E0}$▲	$2.07E2_{1.2E0}$
Fpppp-32-334-0.1-0.5	$1.18E3_{0.0E0}$▲	$9.96E2_{9.0E1}$▲	$8.84E2_{1.4E1}$▲	$9.00E2_{3.1E1}$▲	$8.46E2_{0.0E0}$▲	$8.28E2_{7.2E0}$
Fpppp-32-334-0.1-1	$1.51E3_{0.0E0}$▲	$1.25E3_{1.6E2}$▲	$1.12E3_{3.4E1}$▲	$1.11E3_{2.7E1}$▲	$1.10E3_{0.0E0}$▲	$1.06E3_{1.0E0}$
Fpppp-32-334-0.1-5	$1.40E3_{0.0E0}$▲	$1.70E3_{1.4E2}$▲	$1.56E3_{9.3E1}$▲	$1.52E3_{5.0E1}$▲	$1.26E3_{0.0E0}$▲	$1.16E3_{6.7E0}$
Fpppp-32-334-0.1-10	$6.19E3_{0.0E0}$▲	$6.10E3_{7.3E2}$▲	$5.71E3_{4.0E2}$▲	$5.53E3_{1.7E2}$▲	$6.30E3_{0.0E0}$▲	$4.38E3_{4.1E2}$
Fpppp-32-334-0.25-0.1	$9.99E2_{0.0E0}$▲	$9.58E2_{1.1E2}$▲	$8.41E2_{4.0E1}$▲	$8.40E2_{2.5E1}$▲	$7.62E2_{0.0E0}$▲	$7.53E2_{3.5E0}$
Fpppp-32-334-0.25-0.5	$8.13E2_{0.0E0}$▲	$6.67E2_{7.4E1}$▲	$5.74E2_{1.9E1}$▲	$5.76E2_{1.4E1}$▲	$5.58E2_{0.0E0}$▲	$5.53E2_{2.5E0}$
Fpppp-32-334-0.25-1	$3.81E2_{0.0E0}$▲	$2.97E2_{3.7E1}$▲	$2.64E2_{7.2E0}$▲	$2.68E2_{5.2E0}$▲	$2.60E2_{0.0E0}$▲	$2.51E2_{1.8E0}$
Fpppp-32-334-0.25-5	$3.31E3_{0.0E0}$▲	$3.07E3_{3.6E2}$▲	$2.74E3_{1.4E2}$▲	$2.76E3_{1.2E2}$▲	$2.41E3_{0.0E0}$▲	$2.25E3_{7.1E1}$
Fpppp-32-334-0.25-10	$4.61E3_{0.0E0}$▲	$4.60E3_{4.2E2}$▲	$4.45E3_{1.6E2}$▲	$4.40E3_{1.7E2}$▲	$3.73E3_{0.0E0}$▲	$3.61E3_{4.5E1}$
Fpppp-32-334-0.5-0.1	$1.18E3_{0.0E0}$▲	$1.13E3_{1.1E2}$▲	$1.04E3_{3.3E1}$▲	$1.04E3_{3.9E1}$▲	$8.99E2_{0.0E0}$▲	$8.92E2_{1.7E0}$
Fpppp-32-334-0.5-0.5	$1.56E3_{0.0E0}$▲	$1.38E3_{1.6E2}$▲	$1.24E3_{3.0E1}$▲	$1.24E3_{3.7E1}$▲	$1.17E3_{0.0E0}$▲	$1.17E3_{9.2E-3}$
Fpppp-32-334-0.5-1	$8.93E2_{0.0E0}$▲	$7.24E2_{4.9E1}$▲	$6.67E2_{8.9E0}$▲	$6.68E2_{1.3E1}$▲	$6.43E2_{0.0E0}$▲	$6.17E2_{3.4E0}$
Fpppp-32-334-0.5-5	$2.38E3_{0.0E0}$▲	$2.83E3_{2.4E2}$▲	$2.39E3_{1.2E2}$▲	$2.39E3_{1.3E2}$▲	$2.19E3_{0.0E0}$▲	$1.95E3_{0.0E0}$
Fpppp-32-334-0.5-10	$3.07E3_{0.0E0}$▲	$3.21E3_{2.4E2}$▲	$3.08E3_{1.6E2}$▲	$3.01E3_{1.1E2}$▲	$3.11E3_{0.0E0}$▲	$2.69E3_{1.3E2}$
Fpppp-32-334-0.75-0.1	$1.01E3_{0.0E0}$▲	$9.62E2_{6.9E1}$▲	$8.70E2_{3.4E1}$▲	$8.63E2_{3.1E1}$▲	$7.79E2_{0.0E0}$▲	$7.73E2_{2.7E0}$
Fpppp-32-334-0.75-0.5	$1.63E3_{0.0E0}$▲	$1.40E3_{1.1E2}$▲	$1.27E3_{2.4E1}$▲	$1.29E3_{2.4E1}$▲	$1.25E3_{0.0E0}$−	$1.25E3_{0.0E0}$
Fpppp-32-334-0.75-1	$1.03E3_{0.0E0}$▲	$8.79E2_{4.1E1}$▲	$7.95E2_{2.0E1}$▲	$8.02E2_{2.9E1}$▲	$7.93E2_{0.0E0}$▲	$7.65E2_{7.9E0}$
Fpppp-32-334-0.75-5	$1.10E3_{0.0E0}$▲	$9.15E2_{7.4E1}$▲	$8.49E2_{3.0E1}$▲	$8.37E2_{2.0E1}$▲	$8.36E2_{0.0E0}$▲	$6.89E2_{1.1E1}$
Fpppp-32-334-0.75-10	$3.13E3_{0.0E0}$▲	$2.54E3_{1.5E2}$▲	$2.54E3_{7.4E1}$▲	$2.52E3_{7.0E1}$▲	$2.59E3_{0.0E0}$▲	$2.06E3_{8.0E1}$
Fpppp-32-334-1-0.1	$9.00E2_{0.0E0}$▲	$7.35E2_{6.5E1}$▲	$6.96E2_{2.3E1}$▲	$6.89E2_{1.9E1}$▲	$5.98E2_{0.0E0}$▲	$5.94E2_{2.1E0}$
Fpppp-32-334-1-0.5	$7.41E2_{0.0E0}$▲	$7.08E2_{4.9E1}$▲	$6.26E2_{1.6E1}$▲	$6.31E2_{2.6E1}$▲	$6.01E2_{0.0E0}$▲	$5.89E2_{0.0E0}$
Fpppp-32-334-1-1	$8.24E1_{0.0E0}$▲	$7.48E1_{5.7E0}$▲	$6.76E1_{1.5E0}$▲	$6.85E1_{2.4E0}$▲	$6.70E1_{0.0E0}$▲	$6.45E1_{1.2E0}$
Fpppp-32-334-1-5	$2.34E3_{0.0E0}$▲	$2.43E3_{2.2E2}$▲	$2.16E3_{9.7E1}$▲	$2.14E3_{1.0E2}$▲	$1.90E3_{0.0E0}$▲	$1.79E3_{2.4E1}$
Fpppp-32-334-1-10	$4.50E3_{0.0E0}$▲	$4.96E3_{5.9E2}$▲	$4.45E3_{1.6E2}$▲	$4.31E3_{1.5E2}$▲	$4.16E3_{0.0E0}$▲	$3.37E3_{9.2E1}$
Fpppp-64-334-0.1-0.1	$3.28E2_{0.0E0}$▲	$2.65E2_{1.9E1}$▲	$2.59E2_{2.0E0}$▲	$2.60E2_{9.8E-1}$▲	$2.59E2_{0.0E0}$▲	$2.59E2_{0.0E0}$
Fpppp-64-334-0.1-0.5	$6.65E1_{0.0E0}$▲	$5.90E1_{2.8E0}$▲	$5.54E1_{5.4E-1}$▲	$5.54E1_{5.6E-1}$▲	$5.48E1_{0.0E0}$▲	$5.46E1_{0.0E0}$
Fpppp-64-334-0.1-1	$9.63E2_{0.0E0}$▲	$7.86E2_{4.9E1}$▲	$7.29E2_{1.1E1}$▲	$7.34E2_{1.1E1}$▲	$7.27E2_{0.0E0}$▲	$7.07E2_{2.7E0}$
Fpppp-64-334-0.1-5	$2.11E3_{0.0E0}$▲	$2.28E3_{2.4E2}$▲	$2.00E3_{7.6E1}$▲	$1.99E3_{6.0E1}$▲	$1.68E3_{0.0E0}$▲	$1.61E3_{4.5E1}$
Fpppp-64-334-0.1-10	$2.42E3_{0.0E0}$▲	$2.37E3_{3.0E2}$▲	$2.26E3_{8.2E1}$▲	$2.22E3_{9.2E1}$▲	$2.12E3_{0.0E0}$▲	$2.03E3_{5.7E1}$
Fpppp-64-334-0.25-0.1	$1.07E3_{0.0E0}$▲	$9.91E2_{9.3E1}$▲	$8.93E2_{5.7E0}$▲	$8.93E2_{7.5E0}$▲	$8.88E2_{0.0E0}$−	$8.88E2_{0.0E0}$
Fpppp-64-334-0.25-0.5	$1.46E3_{0.0E0}$▲	$1.39E3_{6.0E1}$▲	$1.33E3_{9.2E0}$▲	$1.33E3_{8.8E0}$▲	$1.33E3_{0.0E0}$▲	$1.32E3_{0.0E0}$
Fpppp-64-334-0.25-1	$1.14E3_{0.0E0}$▲	$1.02E3_{5.2E1}$▲	$9.43E2_{2.1E1}$−	$9.36E2_{2.0E1}$▽	$9.74E2_{0.0E0}$▲	$9.57E2_{3.9E1}$
Fpppp-64-334-0.25-5	$2.09E3_{0.0E0}$▲	$2.38E3_{2.5E2}$▲	$2.06E3_{1.3E2}$▲	$2.05E3_{6.7E1}$▲	$1.86E3_{0.0E0}$▲	$1.65E3_{3.0E1}$
Fpppp-64-334-0.25-10	$1.66E3_{0.0E0}$▲	$1.87E3_{2.4E2}$▲	$1.75E3_{6.4E1}$▲	$1.75E3_{5.4E1}$▲	$1.56E3_{0.0E0}$▲	$1.45E3_{6.6E1}$
Fpppp-64-334-0.5-0.1	$2.57E2_{0.0E0}$▲	$2.11E2_{1.6E1}$▲	$2.04E2_{1.1E-1}$▲	$2.04E2_{1.4E0}$▲	$2.04E2_{0.0E0}$−	$2.04E2_{0.0E0}$
Fpppp-64-334-0.5-0.5	$1.19E3_{0.0E0}$▲	$1.18E3_{4.3E1}$▲	$1.13E3_{5.5E0}$▲	$1.13E3_{9.1E0}$▲	$1.13E3_{0.0E0}$▲	$1.12E3_{0.0E0}$
Fpppp-64-334-0.5-1	$7.28E2_{0.0E0}$▲	$6.33E2_{1.8E1}$▲	$5.97E2_{1.3E1}$▲	$5.98E2_{8.0E0}$▲	$5.96E2_{0.0E0}$▲	$5.81E2_{2.5E0}$
Fpppp-64-334-0.5-5	$1.25E3_{0.0E0}$▲	$1.45E3_{1.5E2}$▲	$1.24E3_{8.3E1}$▲	$1.22E3_{5.1E1}$▲	$1.13E3_{0.0E0}$▲	$1.12E3_{2.9E1}$
Fpppp-64-334-0.5-10	$1.64E3_{0.0E0}$▲	$1.80E3_{2.2E2}$▲	$1.63E3_{5.9E1}$▲	$1.62E3_{5.1E1}$▲	$1.49E3_{0.0E0}$▲	$1.39E3_{1.9E1}$
Fpppp-64-334-0.75-0.1	$1.43E2_{0.0E0}$▲	$1.29E2_{1.3E1}$▲	$1.18E2_{9.4E-2}$▽	$1.18E2_{1.2E-1}$▽	$1.19E2_{0.0E0}$▽	$1.19E2_{2.8E-1}$
Fpppp-64-334-0.75-0.5	$5.25E2_{0.0E0}$▲	$5.04E2_{2.3E1}$▲	$4.69E2_{7.0E0}$▲	$4.71E2_{7.5E0}$▲	$4.85E2_{0.0E0}$▲	$4.59E2_{7.5E0}$
Fpppp-64-334-0.75-1	$5.98E2_{0.0E0}$▲	$5.47E2_{2.1E1}$▲	$4.97E2_{1.1E1}$▲	$4.96E2_{1.1E1}$▲	$4.96E2_{0.0E0}$▲	$4.78E2_{6.3E0}$
Fpppp-64-334-0.75-5	$8.95E2_{0.0E0}$▲	$1.04E3_{7.3E1}$▲	$8.94E2_{8.8E1}$▲	$8.86E2_{5.0E1}$▲	$9.78E2_{0.0E0}$▲	$7.35E2_{3.8E1}$
Fpppp-64-334-0.75-10	$3.75E3_{0.0E0}$▲	$3.86E3_{5.5E2}$▲	$3.45E3_{1.8E2}$▲	$3.40E3_{1.7E2}$▲	$3.06E3_{0.0E0}$▲	$2.96E3_{7.7E1}$
Fpppp-64-334-1-0.1	$1.01E2_{0.0E0}$▲	$8.99E1_{6.3E0}$▲	$8.22E1_{4.8E-1}$▲	$8.26E1_{5.0E-1}$▲	$8.15E1_{0.0E0}$−	$8.15E1_{0.0E0}$
Fpppp-64-334-1-0.5	$4.13E2_{0.0E0}$▲	$3.63E2_{9.7E0}$▲	$3.55E2_{2.3E0}$−	$3.54E2_{2.5E0}$−	$3.60E2_{0.0E0}$▲	$3.55E2_{0.0E0}$
Fpppp-64-334-1-1	$1.46E3_{0.0E0}$▲	$1.35E3_{4.0E1}$▲	$1.28E3_{1.3E1}$▲	$1.28E3_{9.2E0}$▲	$1.31E3_{0.0E0}$▲	$1.25E3_{3.0E0}$
Fpppp-64-334-1-5	$9.22E2_{0.0E0}$▲	$9.71E2_{7.5E1}$▲	$8.49E2_{4.1E1}$▲	$8.60E2_{3.5E1}$▲	$8.86E2_{0.0E0}$▲	$7.29E2_{3.0E1}$
Fpppp-64-334-1-10	$5.76E3_{0.0E0}$▲	$4.76E3_{4.3E2}$▲	$4.32E3_{1.9E2}$▲	$4.29E3_{1.5E2}$▲	$3.99E3_{0.0E0}$▲	$3.84E3_{7.1E1}$

Dark/light gray emphasizes the best/second-best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

neighborhoods and Local Searches (Hoos & Stützle, 2004) to optimize precedence-constrained parallel applications on heterogeneous machines (Santiago et al., 2020a). The core idea behind a Variable Neighborhood Search is that different neighborhood operators produce different decision variables landscapes, escaping a landscape local optima when changing the neighborhood operator. A neighborhood operator is a particular movement rule for navigating the problem decision variable space. Given a solution x, their neighbor solutions $N(x)$ are the ones that can be reachable with an application of the neighborhood operator over x. The concepts above are meaningful in heuristic optimization. Iterative examining $N(x)$ through their best neighbor's neighborhood will eventually lead a path to a local optimum. The canonical VNS and General VNS (Talbi, 2009) (GVNS) are almost identical, having three steps (i) shake/perturbation, (ii) Solution improvement, and (iii) neighborhood change. Their difference lies in the solution improvement step; while VNS uses a single neighborhood Local Search, GVNS uses Variable Neighborhood Descending (VND) as a solution improvement method. A VND has a set of different neighborhood operators, sequentially applying Local Searches using these neighborhood operators. A peculiarity of the classical VND is that it restarts the search from the initial neighborhood when an improvement occurs, leading to extensive computing times when many minor objective function improvements arise. Correspondingly, initial random solutions are usually in a distant neighborhood from the optimal value for problems with many minor improvements. To avoid a lot of minor improvements is possible to initialize the search with a constructive initial heuristic as the Earliest Finish Time (EFT) in Algorithm 1. Moreover, using a Composite Local Search (Villanueva et al., 2012) (see Algorithm 2), also known as Composited Neighborhood (Gaspero & Schaerf, 2007) or Nested VND (Hansen et al., 2017), is less computationally expensive because it does not restart the search from the first neighborhood when an improvement occurs, producing

Table 7
Medians and IQR of the six algorithms over 30 independent runs.

Problema	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
LIGO-8-76-0.1-0.1	$6.02E2_{0.0E0}$▲	$6.54E2_{3.4E1}$▲	$6.17E2_{2.1E1}$▲	$6.08E2_{1.7E1}$▲	$5.65E2_{5.9E-1}$▲	$5.64E2_{4.9E0}$
LIGO-8-76-0.1-0.5	$1.19E3_{0.0E0}$▲	$9.96E2_{6.5E1}$▲	$9.23E2_{1.9E1}$▲	$9.16E2_{2.9E1}$▲	$8.36E2_{0.0E0}$▲	$8.31E2_{3.6E0}$
LIGO-8-76-0.1-1	$1.33E3_{0.0E0}$▲	$1.16E3_{1.0E2}$▲	$1.04E3_{2.6E1}$▲	$1.03E3_{2.0E1}$▲	$9.15E2_{6.2E0}$▽	$9.20E2_{4.8E0}$
LIGO-8-76-0.1-5	$1.96E3_{0.0E0}$▲	$1.58E3_{2.0E2}$▲	$1.40E3_{8.4E1}$▲	$1.29E3_{8.4E1}$▲	$1.08E3_{3.5E1}$−	$1.06E3_{5.1E1}$
LIGO-8-76-0.1-10	$1.61E3_{0.0E0}$▲	$1.57E3_{2.4E2}$▲	$1.49E3_{1.0E2}$▲	$1.46E3_{1.1E2}$▲	$9.92E2_{2.1E1}$▽	$1.04E3_{2.2E2}$
LIGO-8-76-0.25-0.1	$6.19E2_{0.0E0}$▲	$5.34E2_{3.0E1}$▲	$4.93E2_{9.8E0}$▲	$4.83E2_{9.8E0}$▲	$4.60E2_{4.6E-1}$▲	$4.59E2_{0.0E0}$
LIGO-8-76-0.25-0.5	$8.83E2_{0.0E0}$▲	$8.37E2_{8.4E1}$▲	$7.58E2_{2.3E1}$▲	$7.46E2_{1.7E1}$▲	$6.59E2_{0.0E0}$▽	$6.67E2_{4.4E0}$
LIGO-8-76-0.25-1	$6.18E2_{0.0E0}$▲	$5.16E2_{2.3E1}$▲	$4.73E2_{1.2E1}$▲	$4.71E2_{1.2E1}$▲	$4.11E2_{4.0E0}$▽	$4.17E2_{5.6E0}$
LIGO-8-76-0.25-5	$3.33E2_{0.0E0}$▲	$2.56E2_{3.2E1}$▲	$2.15E2_{9.3E0}$▲	$2.11E2_{7.7E0}$▲	$1.71E2_{3.9E0}$▽	$1.75E2_{3.2E0}$
LIGO-8-76-0.25-10	$1.63E3_{0.0E0}$▲	$2.06E3_{3.5E2}$▲	$1.75E3_{9.2E1}$▲	$1.70E3_{9.9E1}$▲	$1.44E3_{3.2E1}$−	$1.46E3_{2.4E2}$
LIGO-8-76-0.5-0.1	$5.21E2_{0.0E0}$▲	$4.66E2_{4.2E1}$▲	$4.30E2_{1.4E1}$▲	$4.26E2_{1.1E1}$▲	$3.87E2_{0.0E0}$▲	$3.83E2_{2.6E-2}$
LIGO-8-76-0.5-0.5	$1.02E3_{0.0E0}$▲	$8.92E2_{6.1E1}$▲	$8.19E2_{1.8E1}$▲	$8.06E2_{2.8E1}$▲	$7.11E2_{0.0E0}$−	$7.11E2_{2.5E0}$
LIGO-8-76-0.5-1	$8.61E2_{0.0E0}$▲	$9.32E2_{6.7E1}$▲	$8.06E2_{3.2E1}$▲	$8.18E2_{2.6E1}$▲	$6.85E2_{0.0E0}$▲	$6.85E2_{1.6E1}$
LIGO-8-76-0.5-5	$2.63E3_{0.0E0}$▲	$2.11E3_{2.7E2}$▲	$1.78E3_{6.9E1}$▲	$1.78E3_{1.1E2}$▲	$1.36E3_{3.4E1}$▲	$1.31E3_{4.0E1}$
LIGO-8-76-0.5-10	$5.13E2_{0.0E0}$▲	$4.07E2_{1.1E2}$▲	$3.37E2_{2.2E1}$▲	$3.49E2_{1.6E1}$▲	$2.76E2_{3.1E0}$−	$2.74E2_{3.0E1}$
LIGO-8-76-0.75-0.1	$2.19E2_{0.0E0}$▲	$1.90E2_{1.0E1}$▲	$1.75E2_{4.9E0}$▲	$1.72E2_{4.4E0}$▲	$1.49E2_{0.0E0}$−	$1.49E2_{0.0E0}$
LIGO-8-76-0.75-0.5	$6.31E2_{0.0E0}$▲	$5.57E2_{3.9E1}$▲	$5.22E2_{1.3E1}$▲	$5.10E2_{9.1E0}$▲	$4.33E2_{5.1E0}$▽	$4.45E2_{1.1E1}$
LIGO-8-76-0.75-1	$8.85E2_{0.0E0}$▲	$8.56E2_{7.9E1}$▲	$7.61E2_{1.9E1}$▲	$7.64E2_{2.1E1}$▲	$6.49E2_{1.3E1}$▽	$6.59E2_{1.6E1}$
LIGO-8-76-0.75-5	$7.03E2_{0.0E0}$▲	$5.78E2_{1.0E2}$▲	$4.82E2_{2.1E1}$▲	$4.70E2_{3.4E1}$▲	$3.64E2_{8.3E0}$−	$3.64E2_{2.0E1}$
LIGO-8-76-0.75-10	$1.27E3_{0.0E0}$▲	$1.04E3_{9.6E1}$▲	$9.10E2_{7.6E1}$▲	$8.90E2_{7.0E1}$▲	$6.91E2_{2.7E1}$▲	$6.47E2_{2.4E1}$
LIGO-8-76-1-0.1	$1.61E2_{0.0E0}$▲	$1.50E2_{6.6E0}$▲	$1.42E2_{3.3E0}$▲	$1.36E2_{1.7E0}$▲	$1.19E2_{0.0E0}$▲	$1.19E2_{0.0E0}$
LIGO-8-76-1-0.5	$4.79E2_{0.0E0}$▲	$4.42E2_{3.3E1}$▲	$4.15E2_{1.5E1}$▲	$4.12E2_{1.8E1}$▲	$3.60E2_{0.0E0}$▲	$3.58E2_{3.0E0}$
LIGO-8-76-1-1	$1.10E3_{0.0E0}$▲	$8.60E2_{6.1E1}$▲	$8.05E2_{2.3E1}$▲	$8.12E2_{1.4E1}$▲	$7.39E2_{0.0E0}$▲	$6.82E2_{8.0E0}$
LIGO-8-76-1-5	$4.55E2_{0.0E0}$▲	$5.06E2_{8.9E1}$▲	$4.16E2_{3.1E1}$▲	$4.08E2_{2.4E1}$▲	$3.44E2_{9.9E0}$▲	$3.31E2_{8.6E0}$
LIGO-8-76-1-10	$4.02E2_{0.0E0}$▲	$3.23E2_{4.6E1}$▲	$2.77E2_{2.1E1}$▲	$2.68E2_{1.4E1}$▲	$2.08E2_{4.4E0}$▽	$2.23E2_{0.0E0}$
LIGO-16-76-0.1-0.1	$5.83E1_{0.0E0}$▲	$5.86E1_{2.1E0}$▲	$5.53E1_{6.1E-1}$▲	$5.54E1_{5.0E-1}$▲	$5.45E1_{0.0E0}$▲	$5.45E1_{0.0E0}$
LIGO-16-76-0.1-0.5	$7.51E2_{0.0E0}$▲	$6.21E2_{4.1E1}$▲	$5.86E2_{6.0E0}$▲	$5.88E2_{6.3E0}$▲	$5.75E2_{2.3E0}$▲	$5.70E2_{5.8E0}$
LIGO-16-76-0.1-1	$7.84E2_{0.0E0}$▲	$9.41E2_{9.4E1}$▲	$7.70E2_{3.1E1}$▲	$7.94E2_{2.9E1}$▲	$7.06E2_{8.6E-1}$▲	$6.91E2_{7.2E0}$
LIGO-16-76-0.1-5	$1.48E3_{0.0E0}$▲	$1.56E3_{2.3E2}$▲	$1.31E3_{6.8E1}$▲	$1.32E3_{7.0E1}$▲	$1.02E3_{6.3E0}$▽	$1.08E3_{1.3E2}$
LIGO-16-76-0.1-10	$3.28E3_{0.0E0}$▲	$2.69E3_{4.1E2}$▲	$2.28E3_{1.7E2}$▲	$2.22E3_{1.5E2}$▲	$1.79E3_{8.0E1}$▲	$1.66E3_{7.8E1}$
LIGO-16-76-0.25-0.1	$3.42E2_{0.0E0}$▲	$2.98E2_{2.2E0}$▲	$2.94E2_{0.0E0}$▲	$2.94E2_{4.5E-1}$▲	$2.90E2_{0.0E0}$▲	$2.89E2_{0.0E0}$
LIGO-16-76-0.25-0.5	$3.07E2_{0.0E0}$▲	$3.18E2_{1.4E1}$▲	$2.98E2_{6.1E-1}$▲	$2.98E2_{1.7E0}$▲	$2.87E2_{0.0E0}$▲	$2.86E2_{0.0E0}$
LIGO-16-76-0.25-1	$1.13E2_{0.0E0}$▲	$1.13E2_{7.2E0}$▲	$1.02E2_{1.7E0}$▲	$1.01E2_{1.8E0}$▲	$9.43E1_{0.0E0}$▽	$9.46E1_{5.2E-1}$
LIGO-16-76-0.25-5	$1.18E3_{0.0E0}$▲	$1.17E3_{1.3E2}$▲	$1.04E3_{6.1E1}$▲	$1.04E3_{3.5E1}$▲	$8.50E2_{6.3E0}$▽	$8.81E2_{2.8E1}$
LIGO-16-76-0.25-10	$7.41E2_{0.0E0}$▲	$6.52E2_{6.5E1}$▲	$5.85E2_{3.8E1}$▲	$5.63E2_{2.3E1}$▲	$4.56E2_{2.0E1}$▲	$4.42E2_{2.1E1}$
LIGO-16-76-0.5-0.1	$1.54E2_{0.0E0}$▲	$1.38E2_{5.2E0}$▲	$1.35E2_{8.6E-2}$▲	$1.35E2_{3.2E-1}$▲	$1.32E2_{0.0E0}$▲	$1.30E2_{0.0E0}$
LIGO-16-76-0.5-0.5	$3.92E2_{0.0E0}$▲	$3.85E2_{2.1E1}$▲	$3.62E2_{1.0E1}$▲	$3.62E2_{6.8E0}$▲	$3.43E2_{0.0E0}$−	$3.44E2_{2.4E0}$
LIGO-16-76-0.5-1	$9.37E2_{0.0E0}$▲	$1.08E3_{1.1E2}$▲	$9.26E2_{3.8E1}$▲	$9.27E2_{3.4E1}$▲	$8.37E2_{1.1E1}$−	$8.41E2_{5.9E0}$
LIGO-16-76-0.5-5	$3.52E2_{0.0E0}$▲	$3.10E2_{1.4E1}$▲	$2.80E2_{7.0E0}$▲	$2.82E2_{8.8E0}$▲	$2.37E2_{3.7E0}$▲	$2.27E2_{1.8E0}$
LIGO-16-76-0.5-10	$3.23E2_{0.0E0}$▲	$2.73E2_{4.8E1}$▲	$2.50E2_{1.7E1}$▲	$2.43E2_{1.2E1}$▲	$2.02E2_{2.9E0}$−	$2.00E2_{1.1E1}$
LIGO-16-76-0.75-0.1	$5.37E2_{0.0E0}$▲	$5.04E2_{1.5E1}$▲	$4.93E2_{0.0E0}$▲	$4.93E2_{0.0E0}$▲	$4.90E2_{5.8E0}$▲	$4.84E2_{0.0E0}$
LIGO-16-76-0.75-0.5	$1.94E2_{0.0E0}$▲	$1.80E2_{7.4E0}$▲	$1.65E2_{5.4E0}$▲	$1.65E2_{5.1E0}$▲	$1.54E2_{9.0E-1}$▲	$1.53E2_{2.7E0}$
LIGO-16-76-0.75-1	$3.87E2_{0.0E0}$▲	$4.33E2_{3.4E1}$▲	$3.77E2_{1.6E1}$▲	$3.73E2_{9.6E0}$▲	$3.28E2_{0.0E0}$▲	$3.26E2_{9.6E0}$
LIGO-16-76-0.75-5	$1.76E3_{0.0E0}$▲	$1.33E3_{1.6E2}$▲	$1.10E3_{6.4E1}$▲	$1.12E3_{7.1E1}$▲	$9.09E2_{2.8E1}$▲	$8.59E2_{2.3E1}$
LIGO-16-76-0.75-10	$1.17E3_{0.0E0}$▲	$8.15E2_{1.1E2}$▲	$6.76E2_{3.7E1}$▲	$6.69E2_{3.0E1}$▲	$5.40E2_{8.4E0}$−	$5.28E2_{6.1E1}$
LIGO-16-76-1-0.1	$1.24E2_{0.0E0}$▲	$1.16E2_{4.5E0}$▲	$1.11E2_{2.7E-1}$▲	$1.11E2_{9.9E-1}$▲	$1.10E2_{0.0E0}$−	$1.10E2_{0.0E0}$
LIGO-16-76-1-0.5	$1.47E2_{0.0E0}$▲	$1.39E2_{5.0E0}$▲	$1.33E2_{2.4E0}$▲	$1.34E2_{2.0E0}$▲	$1.30E2_{3.6E-1}$▲	$1.21E2_{5.3E-1}$
LIGO-16-76-1-1	$2.76E2_{0.0E0}$▲	$3.11E2_{1.9E1}$▲	$2.84E2_{6.5E0}$▲	$2.86E2_{5.6E0}$▲	$2.67E2_{0.0E0}$▲	$2.59E2_{7.8E0}$
LIGO-16-76-1-5	$1.59E2_{0.0E0}$▲	$1.53E2_{1.1E1}$▲	$1.39E2_{9.5E0}$▲	$1.37E2_{4.8E0}$▲	$1.16E2_{0.0E0}$▲	$1.09E2_{0.0E0}$
LIGO-16-76-1-10	$2.37E2_{0.0E0}$▲	$2.73E2_{4.2E1}$▲	$2.22E2_{2.0E1}$▲	$2.30E2_{1.9E1}$▲	$1.68E2_{1.3E0}$−	$1.67E2_{2.6E-3}$

Dark/light gray emphasizes the best/second-best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

fewer computing times. In our experience, many minor improvements surface for parallel applications scheduling instances (Huacuja et al., 2015; Pecero et al., 2012; Pineda, Pecero, Huacuja, Barbosa, & Bouvry, 2013b; Santiago et al., 2021, 2020a; Soto-Monterrubio, Santiago, Fraire-Huacuja, Frausto-Solís, & Terán-Villanueva, 2016).

Algorithm 1 Earliest Finish Time

Input: An order execution of the tasks $O = \{o_1, ..., o_{|T|}\}$
Output: An assignation of machines to tasks
1: **for** $i = 1$ to $|T|$ **do**
2: Assign to the task o_i the machine m_j which produce their minimum makespan.
3: **end for**

4.1. Neighborhoods

The work in Santiago et al. (2021) presents an extensive state-of-the-art Local Searches study for scheduling precedence-constrained parallel applications. The Friedman average ranking in Santiago et al. (2021) rPALS_Lex stands out as the best performer of the studied Local Searches in three of four cases. The above is interesting because rPALS_Lex uses more than one neighborhood; following, we explain how we apply the different neighborhoods in our proposal.

Algorithm 2 Composite Local Search

Input: A solution s to be improved and a set of k different neighborhoods.
1: **for** $i = 1$ to k **do**
2: $LocalSearch_i(s)$ ▷ Local search using the ith neighborhood.
3: **end for**

The first neighborhood is of swaps movements. The neighborhood operator starts by fixing a machine m_j and a task $t_i \in m_j$ for later selecting a random machine $m_{swap}|m_j \neq m_{swap}$ and a random task $t_{swap} \in m_{swap}$, then swap the machine assignation between t_i and t_{swap}.

Table 8
Medians and IQR of the six algorithms over 30 independent runs.

Problema	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
LIGO-32-76-0.1-0.1	$5.96E2_{0.0E0}$▲	$5.50E2_{7.0E0}$▲	$5.41E2_{1.9E0}$▲	$5.41E2_{1.3E0}$▲	$5.38E2_{0.0E0}$▲	$5.36E2_{0.0E0}$
LIGO-32-76-0.1-0.5	$6.51E2_{0.0E0}$▲	$6.58E2_{5.1E1}$▲	$6.02E2_{1.2E1}$▲	$6.03E2_{2.0E1}$▲	$5.78E2_{0.0E0}$▲	$5.77E2_{1.4E0}$
LIGO-32-76-0.1-1	$3.02E1_{0.0E0}$▲	$3.52E1_{3.3E0}$▲	$3.06E1_{6.0E-1}$▲	$3.03E1_{4.6E-1}$▲	$2.82E1_{0.0E0}$−	$2.83E1_{2.0E-1}$
LIGO-32-76-0.1-5	$1.79E3_{0.0E0}$▲	$1.63E3_{2.2E2}$▲	$1.30E3_{9.6E1}$▲	$1.25E3_{4.6E1}$▲	$1.12E3_{3.1E1}$−	$1.10E3_{2.0E1}$
LIGO-32-76-0.1-10	$2.67E3_{0.0E0}$▲	$2.29E3_{3.6E2}$▲	$2.01E3_{1.0E2}$▲	$1.95E3_{1.2E2}$▲	$1.63E3_{5.4E1}$−	$1.62E3_{1.6E2}$
LIGO-32-76-0.25-0.1	$1.05E2_{0.0E0}$▲	$1.06E2_{1.3E0}$▲	$1.04E2_{1.0E0}$▲	$1.03E2_{5.7E-1}$▲	$1.02E2_{1.7E-1}$▲	$1.01E2_{6.7E-1}$
LIGO-32-76-0.25-0.5	$8.26E1_{0.0E0}$▲	$9.53E1_{7.3E0}$▲	$8.63E1_{2.3E0}$▲	$8.63E1_{2.7E0}$▲	$8.11E1_{3.6E-1}$▲	$7.95E1_{8.1E-1}$
LIGO-32-76-0.25-1	$8.56E2_{0.0E0}$▲	$9.51E2_{7.6E1}$▲	$8.38E2_{2.0E1}$▲	$8.36E2_{2.3E1}$▲	$7.89E2_{4.3E0}$−	$7.87E2_{6.6E0}$
LIGO-32-76-0.25-5	$7.78E2_{0.0E0}$▲	$6.45E2_{4.1E1}$▲	$5.90E2_{2.6E1}$▲	$5.83E2_{2.8E1}$▲	$5.19E2_{1.6E0}$−	$5.22E2_{2.6E1}$
LIGO-32-76-0.25-10	$1.06E3_{0.0E0}$▲	$9.95E2_{1.4E2}$▲	$8.33E2_{8.8E1}$▲	$8.25E2_{6.0E1}$▲	$6.68E2_{1.1E1}$−	$6.77E2_{5.3E1}$
LIGO-32-76-0.5-0.1	$6.87E2_{0.0E0}$▲	$6.76E2_{1.1E1}$▲	$6.76E2_{0.0E0}$▲	$6.76E2_{0.0E0}$▲	$6.76E2_{0.0E0}$▲	$6.64E2_{2.8E0}$
LIGO-32-76-0.5-0.5	$3.75E2_{0.0E0}$▲	$4.03E2_{3.4E1}$▲	$3.78E2_{9.1E0}$▲	$3.78E2_{6.7E0}$▲	$3.61E2_{0.0E0}$▲	$3.52E2_{9.6E0}$
LIGO-32-76-0.5-1	$4.18E2_{0.0E0}$▲	$4.91E2_{4.3E1}$▲	$4.24E2_{1.5E1}$▲	$4.21E2_{1.6E1}$▲	$3.99E2_{1.1E0}$−	$3.98E2_{9.8E0}$
LIGO-32-76-0.5-5	$8.16E2_{0.0E0}$▲	$7.20E2_{5.6E1}$▲	$6.44E2_{1.6E1}$▲	$6.34E2_{3.8E1}$▲	$5.18E2_{1.6E1}$−	$5.19E2_{3.1E1}$
LIGO-32-76-0.5-10	$1.92E3_{0.0E0}$▲	$1.85E3_{1.4E2}$▲	$1.71E3_{8.2E1}$▲	$1.61E3_{6.5E1}$▲	$1.32E3_{6.6E1}$−	$1.31E3_{7.3E1}$
LIGO-32-76-0.75-0.1	$5.79E2_{0.0E0}$▲	$5.70E2_{0.0E0}$▲	$5.70E2_{0.0E0}$▲	$5.70E2_{0.0E0}$▲	$5.70E2_{0.0E0}$▲	$5.62E2_{0.0E0}$
LIGO-32-76-0.75-0.5	$3.60E2_{0.0E0}$▲	$3.41E2_{1.4E1}$▲	$3.19E2_{8.1E0}$▲	$3.19E2_{1.2E0}$▲	$3.16E2_{0.0E0}$▲	$3.04E2_{1.3E0}$
LIGO-32-76-0.75-1	$6.63E1_{0.0E0}$▲	$7.38E1_{6.7E0}$▲	$6.60E1_{1.8E0}$▲	$6.63E1_{9.0E-1}$▲	$6.00E1_{1.5E-1}$▲	$5.95E1_{2.4E0}$
LIGO-32-76-0.75-5	$2.05E3_{0.0E0}$▲	$1.58E3_{1.2E2}$▲	$1.41E3_{4.3E1}$▲	$1.38E3_{1.3E2}$▲	$1.14E3_{3.6E1}$▲	$1.08E3_{4.1E1}$
LIGO-32-76-0.75-10	$2.58E3_{0.0E0}$▲	$1.72E3_{5.4E2}$▲	$1.44E3_{1.2E2}$▲	$1.45E3_{9.8E1}$▲	$1.30E3_{6.4E0}$−	$1.29E3_{3.1E1}$
LIGO-32-76-1-0.1	$4.45E2_{0.0E0}$▲	$4.24E2_{8.4E0}$▲	$4.23E2_{0.0E0}$▲	$4.23E2_{0.0E0}$▲	$4.23E2_{0.0E0}$▲	$4.14E2_{0.0E0}$
LIGO-32-76-1-0.5	$1.11E2_{0.0E0}$▲	$1.16E2_{3.3E0}$▲	$1.09E2_{3.0E0}$▲	$1.09E2_{2.9E0}$▲	$1.03E2_{2.7E-1}$▲	$1.00E2_{3.6E0}$
LIGO-32-76-1-1	$1.68E2_{0.0E0}$▲	$1.77E2_{1.8E1}$▲	$1.59E2_{2.9E0}$▲	$1.59E2_{6.4E0}$▲	$1.48E2_{0.0E0}$▲	$1.45E2_{2.3E0}$
LIGO-32-76-1-5	$7.44E2_{0.0E0}$▲	$6.83E2_{9.7E1}$▲	$5.39E2_{3.8E1}$▲	$5.45E2_{3.7E1}$▲	$4.75E2_{1.4E0}$▲	$4.40E2_{2.0E1}$
LIGO-32-76-1-10	$2.83E3_{0.0E0}$▲	$2.49E3_{3.4E2}$▲	$2.28E3_{1.9E2}$▲	$2.13E3_{1.4E2}$▲	$1.74E3_{4.6E1}$−	$1.75E3_{1.1E2}$
LIGO-64-76-0.1-0.1	$6.04E2_{0.0E0}$▲	$6.07E2_{3.5E0}$▲	$6.02E2_{1.1E0}$▲	$6.02E2_{1.1E0}$▲	$6.02E2_{0.0E0}$▲	$5.92E2_{3.0E0}$
LIGO-64-76-0.1-0.5	$3.97E2_{0.0E0}$▲	$4.80E2_{4.7E1}$▲	$4.14E2_{1.4E1}$▲	$4.22E2_{9.7E0}$▲	$3.97E2_{0.0E0}$▲	$3.96E2_{1.7E0}$
LIGO-64-76-0.1-1	$1.67E2_{0.0E0}$▲	$2.02E2_{2.3E1}$▲	$1.74E2_{4.6E0}$▲	$1.73E2_{4.4E0}$▲	$1.56E2_{0.0E0}$▽	$1.60E2_{4.3E0}$
LIGO-64-76-0.1-5	$1.52E3_{0.0E0}$▲	$1.34E3_{2.2E2}$▲	$1.11E3_{4.6E1}$▲	$1.09E3_{4.8E1}$▲	$9.93E2_{2.0E1}$▲	$9.64E2_{3.5E1}$
LIGO-64-76-0.1-10	$1.50E3_{0.0E0}$▲	$1.47E3_{1.6E2}$▲	$1.34E3_{7.2E1}$▲	$1.28E3_{5.6E1}$▲	$1.15E3_{2.2E1}$▲	$1.08E3_{3.0E1}$
LIGO-64-76-0.25-0.1	$6.32E1_{0.0E0}$▲	$6.09E1_{1.0E0}$▲	$5.92E1_{6.8E-1}$▲	$5.93E1_{1.2E0}$▲	$5.87E1_{0.0E0}$▲	$5.77E1_{5.1E-1}$
LIGO-64-76-0.25-0.5	$1.69E2_{0.0E0}$▲	$1.79E2_{4.9E0}$▲	$1.69E2_{2.0E0}$▲	$1.69E2_{2.2E0}$▲	$1.63E2_{0.0E0}$▲	$1.63E2_{9.1E-1}$
LIGO-64-76-0.25-1	$4.10E2_{0.0E0}$▲	$4.91E2_{5.0E1}$▲	$4.28E2_{1.8E1}$▲	$4.31E2_{1.1E1}$▲	$3.99E2_{1.4E0}$▲	$3.87E2_{2.7E0}$
LIGO-64-76-0.25-5	$1.88E3_{0.0E0}$▲	$2.02E3_{2.9E2}$▲	$1.68E3_{5.3E1}$▲	$1.67E3_{6.4E1}$▲	$1.52E3_{3.3E1}$▲	$1.38E3_{4.8E1}$
LIGO-64-76-0.25-10	$2.22E3_{0.0E0}$▲	$2.13E3_{3.5E2}$▲	$1.84E3_{1.1E2}$▲	$1.79E3_{9.1E1}$▲	$1.59E3_{4.6E1}$−	$1.57E3_{1.8E2}$
LIGO-64-76-0.5-0.1	$6.67E2_{0.0E0}$▲	$6.27E2_{1.4E0}$▲	$6.26E2_{0.0E0}$▲	$6.26E2_{0.0E0}$▲	$6.26E2_{0.0E0}$▲	$6.21E2_{2.4E-1}$
LIGO-64-76-0.5-0.5	$6.65E2_{0.0E0}$▲	$7.24E2_{2.8E1}$▲	$6.23E2_{6.8E0}$▲	$6.22E2_{2.7E0}$▲	$5.97E2_{0.0E0}$▲	$5.96E2_{0.0E0}$
LIGO-64-76-0.5-1	$3.66E2_{0.0E0}$▲	$3.62E2_{2.2E1}$▲	$3.25E2_{1.2E1}$▲	$3.21E2_{1.5E1}$▲	$3.06E2_{1.5E0}$▲	$3.05E2_{2.9E0}$
LIGO-64-76-0.5-5	$1.66E3_{0.0E0}$▲	$1.55E3_{1.7E2}$▲	$1.29E3_{8.9E1}$▲	$1.26E3_{6.6E1}$▲	$1.04E3_{4.2E1}$▲	$9.39E2_{5.6E1}$
LIGO-64-76-0.5-10	$1.22E3_{0.0E0}$▲	$1.14E3_{9.7E1}$▲	$9.84E2_{6.6E1}$▲	$9.44E2_{5.3E1}$▲	$8.68E2_{1.9E1}$▲	$8.53E2_{1.0E2}$
LIGO-64-76-0.75-0.1	$2.72E2_{0.0E0}$▲	$2.72E2_{6.5E-2}$▲	$2.72E2_{0.0E0}$▲	$2.72E2_{0.0E0}$▲	$2.72E2_{0.0E0}$▲	$2.68E2_{0.0E0}$
LIGO-64-76-0.75-0.5	$3.01E2_{0.0E0}$▲	$2.91E2_{1.0E1}$▲	$2.79E2_{0.0E0}$▲	$2.79E2_{0.0E0}$▲	$2.79E2_{0.0E0}$▲	$2.61E2_{7.4E0}$
LIGO-64-76-0.75-1	$3.31E2_{0.0E0}$▲	$4.01E2_{3.2E1}$▲	$3.25E2_{1.8E1}$▲	$3.37E2_{1.5E1}$▲	$3.11E2_{0.0E0}$▲	$3.08E2_{5.3E0}$
LIGO-64-76-0.75-5	$1.07E3_{0.0E0}$▲	$1.01E3_{1.5E2}$▲	$8.31E2_{4.1E1}$▲	$8.21E2_{4.8E1}$▲	$7.57E2_{1.1E1}$−	$7.48E2_{5.5E1}$
LIGO-64-76-0.75-10	$2.82E3_{0.0E0}$▲	$2.70E3_{3.8E2}$▲	$2.40E3_{1.2E2}$▲	$2.34E3_{1.3E2}$▲	$2.04E3_{5.7E1}$▲	$1.86E3_{1.8E2}$
LIGO-64-76-1-0.1	$3.01E2_{0.0E0}$▲	$3.01E2_{0.0E0}$▲	$3.01E2_{0.0E0}$▲	$3.01E2_{0.0E0}$▲	$3.01E2_{0.0E0}$▲	$2.90E2_{0.0E0}$
LIGO-64-76-1-0.5	$2.55E2_{0.0E0}$▲	$2.58E2_{2.9E0}$▲	$2.55E2_{0.0E0}$▲	$2.55E2_{0.0E0}$▲	$2.55E2_{0.0E0}$▲	$2.27E2_{8.7E0}$
LIGO-64-76-1-1	$1.72E2_{0.0E0}$▲	$1.79E2_{1.7E1}$▲	$1.63E2_{2.1E0}$▲	$1.63E2_{2.4E0}$▲	$1.56E2_{1.3E0}$▲	$1.51E2_{3.1E0}$
LIGO-64-76-1-5	$1.41E3_{0.0E0}$▲	$1.75E3_{2.2E2}$▲	$1.47E3_{9.2E1}$▲	$1.51E3_{7.1E1}$▲	$1.35E3_{0.0E0}$▲	$1.27E3_{7.3E1}$
LIGO-64-76-1-10	$1.50E2_{0.0E0}$▲	$1.46E2_{2.7E1}$▲	$1.20E2_{6.2E0}$▲	$1.17E2_{6.9E0}$▲	$1.03E2_{2.6E0}$−	$1.02E2_{1.4E1}$

Dark/light gray emphasizes the best/second-best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

The operator ensures to swap machine assignation only for tasks in different machines, avoiding unnecessary swaps between tasks in the same machine.

The second neighborhood is of "move" movements. The neighborhood operator starts by fixing a machine m_j and selecting a random machine $m_{move}|m_j \neq m_{move}$ and a random task $t_{move} \in m_{move}$, finally setting to the task t_{move} to the machine m_j. The operator ensures not to assign tasks to their original assigned machine.

According to Santiago et al. (2021, 2020a), the priority of tasks (order of execution) significantly impacts the algorithms' performance. As a thought experiment, imagine a valid topological order of the DAG where the tasks with greater weights in their precedence edges have priority. The above is counter-intuitive if we pursue to find their minimum execution time. In most cases, the optimal value will probably not be in this order of the task's execution.

Given that the execution order is a permutation, we explore the neighborhood of insertion movements over the order of tasks as a third neighborhood. The neighborhood starts by fixing a position in the priority tasks to be inserted in a second random position, producing a new permutation (order) as in Schiavinotto and Stützle (2004).

Since we are changing the priorities with the insertion neighborhood, it is hard to expect quality solutions based only on the new tasks' execution orders without modifying their original task–machine assignment. Hence, we complement using a modified version of the constructive Earliest Finish Time Heuristic (EFT) with diversity. The implemented version EFT-α uses a parameter α for a probability to assign the regular EFT procedure, usually a greedy machine decision; otherwise, the previous machine assigned in the solution stands. A quality solution is produced similar to the original EFT heuristic but with diversity.

4.2. Composite local search

The sequence of application of the neighborhoods in the Composite Local Search (see Algorithm 3) is as follows: (i) The insertion neighborhood to explore possible orders with improvements, (ii) The

EFT heuristic to quickly generate a quality solution, (iii) The move neighborhood, to explore different task–machine assignations, and (iv) The neighborhood of swaps, to explore savings in the communication times swapping tasks between machines.

Algorithm 3 First improvement Composite Local Search

Input: A solution s with the task-machine assignments an order of execution $O = \{o_1, ..., o_{|T|}\}$.
for $i = 1$ to $|T|$ **do**
  $s' \leftarrow Insert(s, o_i)$ ▷ Generate a neighbor of s with a new order (see Section 4.1)
  if $f(s') < f(s)$ **then**
    $s \leftarrow s'$
    $i \leftarrow 1$
  end if
end for
$s' \leftarrow EFT\text{-}\alpha(O, \alpha)$ ▷ Generate a neighbor applying EFT-α over O (see Section 4.1)
if $f(s') < f(s)$ **then**
  $s \leftarrow s'$
end if
for $i = 1$ to $|T|$ **do**
  $m_j \leftarrow$ The machine assigned to o_i
  $s' \leftarrow Move(s, m_j)$ ▷ Generate a move neighbor fixing m_j (see Section 4.1)
  if $f(s') < f(s)$ **then**
    $s \leftarrow s'$
    $i \leftarrow 1$
  end if
end for
for $i = 1$ to $|T|$ **do**
  $s' \leftarrow Swap(s, o_i)$ ▷ Generate a swap neighbor fixing the task o_i (see Section 4.1)
  if $f(s') < f(s)$ **then**
    $s \leftarrow s'$
    $i \leftarrow 1$
  end if
end for

It is essential to highlight that Composite Local Search (CLS) is less computationally expensive than Variable Neighborhood Descending (VND). Although the Local Searches in our CLS reinitialize the search when an improvement surfaces, it only reinitializes from the current neighborhood (Local Search); it does not reinitialize the entire search from the first neighborhood as in VND. Therefore Composite Local Search is less computationally expensive.

4.3. EFT-GVNS

Based on the discussion mentioned above from Section 4, in this work, we take two major design decisions (i) Initialize our proposed algorithm using the Earliest Finish Time (EFT) heuristic in synergy with the b-level (Santiago et al., 2021) priority, and (ii) The use of a Composite Local Search, instead of the classical Variable Neighborhood Descending (VND).

Algorithm 4 and Fig. 1 summarizes our proposal for scheduling parallel applications on heterogeneous machines with accuracy.

5. Experimental setup

This section gives all the necessary details to reproduce our experiments.

5.1. Algorithms for the comparison

This subsection briefly describes the algorithms considered to compare the performance of EFT-GVNS. Table 2 shows the Big O complexities of the algorithms in time and memory.

5.1.1. HEFT

In Topcuoglu, Hariri, and Wu (2002), the Heterogeneous Earliest Finish Time (HEFT) first appears. HEFT is a deterministic heuristic that follows the principle of Earliest Finish Time (Santiago et al., 2020a) (EFT) for the task/machine scheduling, followed by optimizing idle times in the machines looking for makespan improvements. The HEFT is a well-known reference algorithm in many studies; thus, we included it in the experiments.

Algorithm 4 Proposed EFT-GVNS

Input: A DAG G for an application and their computational times P.
Output: s_{best}
$O \leftarrow b\text{-}level(G, P)$ ▷ Compute b-level task priority
$s_{best} \leftarrow EFT(O)$
while Stopping criterion not reached **do**
  for $i = 1$ to k **do**
    $s' \leftarrow Perturbation_i(s_{best})$ ▷ Shake using the ith neighborhood.
    $CompositeLocalSearch(s')$ ▷ (See Algorithm 2)
    if $f(s') < f(s_{best})$ **then**
      $s_{best} \leftarrow s'$
    end if
  end for
end while

5.1.2. EDA enhanced with heuristics and a two neighborhood path-relinking

The Estimation Distribution Algorithms (Hauschild & Pelikan, 2011) (EDAs) use a probability model of the population to generate new individuals, unlike other Evolutionary Algorithms (EAs) that use recombination or mutation operators to generate new solutions. An EDA for precedence-constraint tasks scheduling appears in ge Wu, Wang, and jing Wang (2021), where individuals represent the tasks' priorities (execution orders). In ge Wu et al. (2021), the EDA is improved by the constructive heuristics EFT (Santiago et al., 2020a), HEFT (Topcuoglu et al., 2002), OEFT (ge Wu et al., 2021), OHEFT (ge Wu et al., 2021), and a path-relinking of two neighborhoods (swap and insert) is proposed. In the path-relinking, the insert operator builds temporal solutions to be later guided by the swap operator.

5.1.3. EFT-ILS

The Earliest Finish Time - Iterated Local Search (EFT-ILS) was introduced in Pineda et al. (2013b). The algorithm starts by generating several feasible random tasks' execution order and evaluating it through the produced Earliest Finish Time(EFT) makespan. After the orders' generation, the best-found is the fixed priority, while EFT produces the initial task/machine assignation. Next, an Iterated Local Search (ILS) improves the makespan using a first improvement Local Search.

5.1.4. GRASP-CPA

Cellular Processing Algorithms (Terán-Villanueva et al., 2019, 2013) (CPAs) are ensemble heuristics that focus on stagnation detection and communication mechanisms between the Processing Cells (Heuristics). In Santiago et al. (2020a), a CPA formed by three Processing Cells (PCs) of Iterated Local Searches (ILSs) is proposed; the PCs initialized with a solution generated by a Greedy Randomized Search Procedure (Feo & Resende, 1995) (GRASP), the ILS uses the first improvement Local Search as in EFT-ILS, and the communication mechanism is a single-point crossover.

5.1.5. MPQGA

The Multiple Priority Queues Genetic Algorithm (MPQGA) was introducing in Xu, Li, Hu, and Li (2014). MPQGA chromosomes represent the tasks' execution priority (order). It starts by seeding the population with three tasks' priority heuristics: t-level (Santiago et al., 2021), b-level (Santiago et al., 2021), and l-level (Arabnejad & Barbosa, 2014).

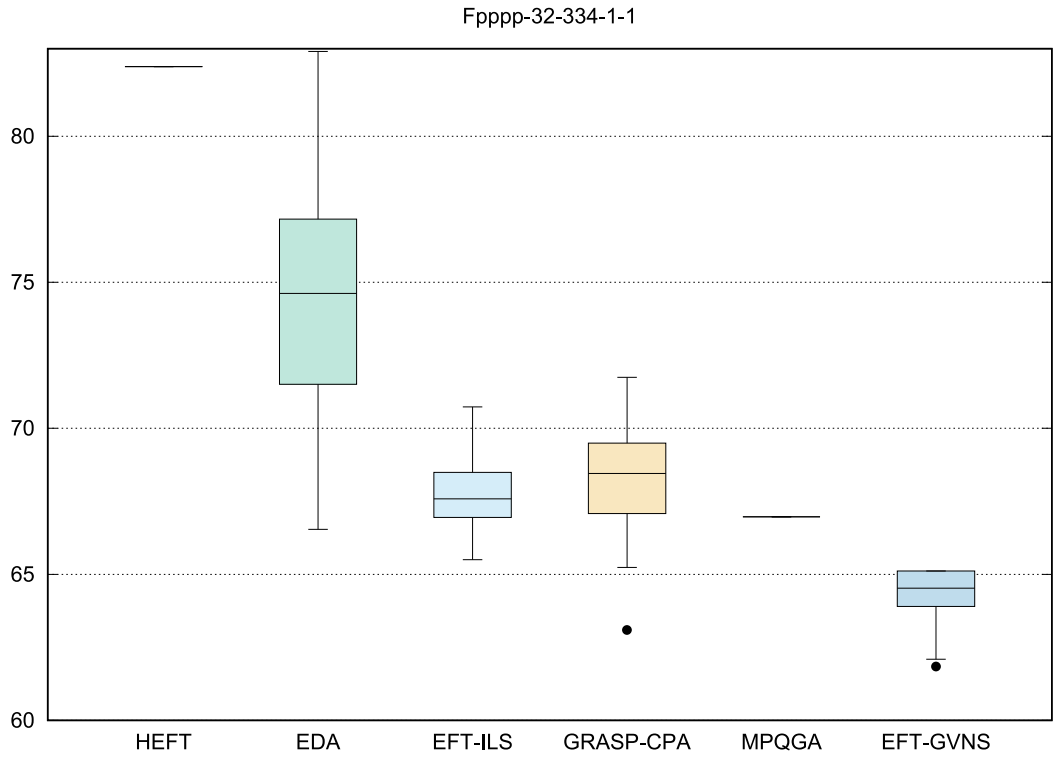


Fig. 2. Boxplots for the six studied algorithms on the problem instance Fpppp-34-334-1-1, the one for which EFT-GVNS offers the best performance.

Later the population is fulfilled based on insertions of the three original individuals. Once the population is full, generations of the algorithm produce new individuals based on a Tabu list and a distance metric to ensure diversity. Individuals' fitness is evaluated based on the produced EFT makespan.

5.2. Benchmark instances

We perform an extensive experimental comparison over real-world applications and a synthetic benchmark from the literature (Ahmad, Dhodhi, & Ul-Mustafa, 1998; Arabnejad, 2013; Arabnejad & Barbosa, 2011; Daoud & Kharma, 2011; Eswari & Nickolas, 2010; Hsu, Hsieh, & Yang, 2007; Ilavarasan, Thambidurai, & Mahilmannan, 2005; Kang & Lin, 2011; Kang, Zhang, & Chen, 2011; Lai & Yang, 2008; Lee, Chen, Chang, Tang, & Pan, 2009; Lee & Zomaya, 2008). The real-world applications have four categories: Fpppp (Double precision floating point FORTRAN benchmark), LIGO (Laser Interferometer Gravitational-Wave Observatory), Robot (Robot control application), and Sparse (Sparse matrix solver). Two features characterize the real-world applications; the cost-to-communication ratio (CCR (Xu et al., 2014)) and the heterogeneity factor (β (Topcuoglu et al., 2002)), both influencing the precedence-constraint tasks scheduling.

$$CCR = \frac{\frac{1}{|C|}\sum_{i=1}^{|T|}\overline{C(t_i,t_j)}}{\frac{1}{|T|}\sum_{i=1}^{|T|}\overline{p}_i} \tag{5}$$

$$\overline{p}_i \cdot \left(1-\frac{\beta}{2}\right) \leq p_{i,j} \leq \overline{p}_i \cdot \left(1+\frac{\beta}{2}\right) \tag{6}$$

The Communication to Computation Ratio (CCR) in Eq. (5) compute two terms: (i) the sum of the average communication costs $\overline{C(t_i,t_j)}$ of the t_i tasks divided by the number of edges in the Directed Acyclic Graph (DAG) $|C|$, and (ii) the sum of the average computing tasks $\overline{p_i}$ of the t_i tasks divided by the number of tasks $|T|$. High values of CCR indicate an application is communication-intensive, and low values of CCR indicate an application is computation-intensive. The heterogeneity factor β in Eq. (6) modules the range of allow computation times $p_{i,j}$ for the task t_i on the machine m_j, a greater value allows more differences among the machines computing capabilities, $p_{i,j}$ is generally uniform randomly selected between the allowed range. We consider the values of CCRs= $\{0.1, 0.5, 1, 5, 10\}$, $\beta = \{0.1, 0.25, 0.5, 0.75, 1\}$ and the number of machines $|M| = \{8, 16, 32, 64\}$ for the four real-world applications, giving a total of 400 scheduling problem instances, the same as in Santiago et al. (2021, 2020a). The problem nomenclature is Application-Machines-Tasks-β-CCR. Another 14 small scheduling problems from the literature, as in Santiago et al. (2020a), are studied. The synthetic benchmark nomenclature is Author-Machines-Tasks. The complete set of instances can be downloaded from Santiago et al. (2020b).

5.3. Experimental settings

The algorithms HEFT, EDA, EFT-ILS, GRASP-CPA, and MPQGA configurations are as their original authors suggest in their original publications. Their parameter settings for the experimentation are in Table 3. The parameter α for EFT in EFT-GVNS was previously tuned, testing the configurations $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. Table 3 includes the best performing α for EFT-GVNS on the studied problems. Every algorithm executes 30 independent runs to evaluate its performance statistically. The stopping criterion for the algorithms is a maximum CPU time of 5 min for the real-world applications set and 100,000 objective function evaluations for the small synthetic benchmark from the literature. The exception is HEFT; due to its constructive nature, there are no time limitations or objective function evaluations to compute it. All the experiments were under the same conditions on the same computing system: Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60 GHz, 16 GB of RAM, and CentOS Linux release 7.7.1908.

5.4. Statistical tests

We present the experimental results as the median value from the independent runs and the Inter Quartile Range (IQR) as central tendency and dispersion measures, respectively. Result tables follow the format

Table 9
Medians and IQR of the six algorithms over 30 independent runs.

Problema	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
Robot-8-88-0.1-0.1	$1.85E3_{0.0E0}$▲	$1.97E3_{7.0E1}$▲	$1.88E3_{2.1E1}$▲	$1.88E3_{2.3E1}$▲	$1.85E3_{1.5E0}$▲	$1.84E3_{2.7E0}$
Robot-8-88-0.1-0.5	$7.75E1_{0.0E0}$▲	$7.62E1_{3.5E0}$▲	$7.24E1_{1.1E0}$▲	$7.20E1_{1.4E0}$▲	$6.67E1_{3.2E-2}$▽	$6.69E1_{1.0E-1}$
Robot-8-88-0.1-1	$2.50E3_{0.0E0}$▲	$2.11E3_{8.9E1}$▲	$2.01E3_{4.7E1}$▲	$1.98E3_{4.0E1}$▲	$1.88E3_{1.2E1}$▲	$1.87E3_{5.5E0}$
Robot-8-88-0.1-5	$3.99E3_{0.0E0}$▲	$3.53E3_{3.5E2}$▲	$3.05E3_{1.0E2}$▲	$3.03E3_{8.6E1}$▲	$2.59E3_{3.2E1}$▲	$2.54E3_{3.7E1}$
Robot-8-88-0.1-10	$3.13E3_{0.0E0}$▲	$2.66E3_{2.6E2}$▲	$2.39E3_{1.1E2}$▲	$2.25E3_{7.8E1}$▲	$1.96E3_{2.5E1}$▲	$1.88E3_{2.5E1}$
Robot-8-88-0.25-0.1	$7.11E2_{0.0E0}$▲	$6.81E2_{4.2E1}$▲	$6.47E2_{9.0E0}$▲	$6.41E2_{1.0E1}$▲	$6.24E2_{0.0E0}$▲	$6.23E2_{4.3E-1}$
Robot-8-88-0.25-0.5	$6.71E2_{0.0E0}$▲	$6.82E2_{3.2E1}$▲	$6.50E2_{9.0E0}$▲	$6.49E2_{1.0E1}$▲	$6.26E2_{0.0E0}$▽	$6.31E2_{3.3E0}$
Robot-8-88-0.25-1	$5.19E2_{0.0E0}$▲	$4.91E2_{2.0E1}$▲	$4.73E2_{7.1E0}$▲	$4.71E2_{6.7E0}$▲	$4.26E2_{2.1E0}$▽	$4.30E2_{1.5E0}$
Robot-8-88-0.25-5	$3.20E3_{0.0E0}$▲	$1.76E3_{1.8E2}$▲	$1.55E3_{4.8E1}$▲	$1.55E3_{4.8E1}$▲	$1.31E3_{3.3E1}$▲	$1.26E3_{2.5E1}$
Robot-8-88-0.25-10	$4.31E3_{0.0E0}$▲	$2.88E3_{2.3E2}$▲	$2.72E3_{8.6E1}$▲	$2.69E3_{6.2E1}$▲	$2.37E3_{1.6E1}$–	$2.36E3_{1.2E2}$
Robot-8-88-0.5-0.1	$1.19E3_{0.0E0}$▲	$1.08E3_{4.7E1}$▲	$1.03E3_{2.0E1}$▲	$1.02E3_{1.2E1}$▲	$9.63E2_{3.9E0}$▲	$9.54E2_{6.2E-2}$
Robot-8-88-0.5-0.5	$9.62E2_{0.0E0}$▲	$8.42E2_{4.3E1}$▲	$7.97E2_{1.9E1}$▲	$7.89E2_{1.1E1}$▲	$7.23E2_{2.9E0}$▲	$6.99E2_{3.1E-1}$
Robot-8-88-0.5-1	$1.03E3_{0.0E0}$▲	$7.82E2_{3.2E1}$▲	$7.45E2_{1.6E1}$▲	$7.42E2_{1.2E1}$▲	$6.91E2_{7.9E0}$–	$6.88E2_{1.2E1}$
Robot-8-88-0.5-5	$1.97E3_{0.0E0}$▲	$1.47E3_{8.9E1}$▲	$1.35E3_{3.4E1}$▲	$1.34E3_{4.0E1}$▲	$1.17E3_{2.2E1}$▲	$1.06E3_{2.2E1}$
Robot-8-88-0.5-10	$5.83E2_{0.0E0}$▲	$4.10E2_{2.6E1}$▲	$3.60E2_{1.3E1}$▲	$3.53E2_{1.7E1}$▲	$2.74E2_{3.6E0}$–	$2.76E2_{4.5E0}$
Robot-8-88-0.75-0.1	$6.36E2_{0.0E0}$▲	$5.92E2_{2.6E1}$▲	$5.62E2_{9.6E0}$▲	$5.62E2_{1.2E1}$▲	$5.13E2_{8.6E-1}$▲	$5.13E2_{0.0E0}$
Robot-8-88-0.75-0.5	$1.89E3_{0.0E0}$▲	$1.66E3_{4.9E1}$▲	$1.59E3_{2.2E1}$▲	$1.58E3_{1.9E1}$▲	$1.50E3_{9.5E0}$▲	$1.48E3_{5.2E-1}$
Robot-8-88-0.75-1	$1.86E3_{0.0E0}$▲	$1.51E3_{7.9E1}$▲	$1.41E3_{3.3E1}$▲	$1.40E3_{2.3E1}$▲	$1.25E3_{0.0E0}$–	$1.25E3_{4.7E0}$
Robot-8-88-0.75-5	$3.43E3_{0.0E0}$▲	$2.84E3_{1.5E2}$▲	$2.53E3_{7.4E1}$▲	$2.45E3_{7.6E1}$▲	$2.04E3_{2.5E1}$–	$2.04E3_{2.2E1}$
Robot-8-88-0.75-10	$1.59E3_{0.0E0}$▲	$1.16E3_{9.8E1}$▲	$1.08E3_{4.0E1}$▲	$1.03E3_{3.4E1}$▲	$9.18E2_{2.7E1}$▲	$8.33E2_{6.1E1}$
Robot-8-88-1-0.1	$2.99E2_{0.0E0}$▲	$3.22E2_{1.7E1}$▲	$3.05E2_{5.4E0}$▲	$2.97E2_{6.9E0}$▲	$2.75E2_{1.6E-1}$▲	$2.73E2_{0.0E0}$
Robot-8-88-1-0.5	$4.56E2_{0.0E0}$▲	$3.98E2_{2.2E1}$▲	$3.75E2_{8.7E0}$▲	$3.69E2_{8.4E0}$▲	$3.30E2_{3.9E0}$▲	$3.20E2_{6.7E0}$
Robot-8-88-1-1	$1.86E3_{0.0E0}$▲	$1.53E3_{1.3E2}$▲	$1.39E3_{3.5E1}$▲	$1.37E3_{4.7E1}$▲	$1.22E3_{1.4E1}$▲	$1.20E3_{1.7E1}$
Robot-8-88-1-5	$1.48E3_{0.0E0}$▲	$1.07E3_{6.4E1}$▲	$9.94E2_{3.2E1}$▲	$9.93E2_{3.5E1}$▲	$8.20E2_{1.4E1}$▲	$7.96E2_{4.8E1}$
Robot-8-88-1-10	$3.29E3_{0.0E0}$▲	$2.23E3_{1.1E2}$▲	$2.02E3_{9.0E1}$▲	$1.92E3_{5.8E1}$▲	$1.69E3_{2.6E1}$▲	$1.67E3_{6.0E1}$
Robot-16-88-0.1-0.1	$7.09E2_{0.0E0}$▲	$6.89E2_{5.9E0}$▲	$6.78E2_{3.3E0}$▲	$6.79E2_{1.7E0}$▲	$6.72E2_{2.7E-1}$–	$6.73E2_{3.0E0}$
Robot-16-88-0.1-0.5	$1.87E3_{0.0E0}$▲	$1.99E3_{4.4E1}$▲	$1.91E3_{1.5E1}$▲	$1.93E3_{1.5E1}$▲	$1.86E3_{5.1E0}$–	$1.86E3_{2.7E0}$
Robot-16-88-0.1-1	$1.30E3_{0.0E0}$▲	$1.22E3_{6.8E1}$▲	$1.15E3_{1.9E1}$▲	$1.14E3_{2.0E1}$▲	$1.05E3_{1.8E-1}$▲	$1.05E3_{4.7E-1}$
Robot-16-88-0.1-5	$6.11E2_{0.0E0}$▲	$5.72E2_{3.1E1}$▲	$5.13E2_{1.9E1}$▲	$5.18E2_{2.2E1}$▲	$4.33E2_{6.5E0}$▲	$4.01E2_{7.7E0}$
Robot-16-88-0.1-10	$3.49E3_{0.0E0}$▲	$2.69E3_{1.7E2}$▲	$2.53E3_{5.4E1}$▲	$2.47E3_{6.2E1}$▲	$2.22E3_{5.5E1}$▲	$2.09E3_{7.8E1}$
Robot-16-88-0.25-0.1	$6.10E1_{0.0E0}$▲	$6.15E1_{1.4E0}$▲	$6.00E1_{5.1E-1}$▲	$5.99E1_{6.0E-1}$▲	$5.85E1_{0.0E0}$▽	$5.94E1_{6.3E-1}$
Robot-16-88-0.25-0.5	$3.65E2_{0.0E0}$▲	$3.79E2_{8.6E0}$▲	$3.66E2_{2.5E0}$▲	$3.67E2_{1.6E0}$▲	$3.57E2_{8.0E-1}$▽	$3.60E2_{8.0E-1}$
Robot-16-88-0.25-1	$7.62E2_{0.0E0}$▲	$5.83E2_{1.5E1}$▲	$5.48E2_{7.8E0}$▲	$5.46E2_{9.7E0}$▲	$5.13E2_{1.1E0}$–	$5.14E2_{8.4E0}$
Robot-16-88-0.25-5	$3.64E3_{0.0E0}$▲	$4.32E3_{2.1E2}$▲	$3.77E3_{1.7E2}$▲	$3.78E3_{1.3E2}$▲	$3.13E3_{4.1E1}$▲	$2.96E3_{1.4E2}$
Robot-16-88-0.25-10	$5.02E3_{0.0E0}$▲	$3.92E3_{3.1E2}$▲	$3.52E3_{1.4E2}$▲	$3.46E3_{1.5E2}$▲	$2.97E3_{5.7E1}$▲	$2.82E3_{2.1E2}$
Robot-16-88-0.5-0.1	$1.03E3_{0.0E0}$▲	$9.46E2_{8.8E0}$▲	$9.34E2_{1.9E0}$▲	$9.36E2_{9.3E-1}$▲	$9.30E2_{3.1E0}$▲	$9.17E2_{2.6E0}$
Robot-16-88-0.5-0.5	$1.51E3_{0.0E0}$▲	$1.45E3_{4.6E1}$▲	$1.38E3_{1.6E1}$▲	$1.37E3_{1.7E1}$▲	$1.31E3_{6.4E0}$–	$1.31E3_{1.8E1}$
Robot-16-88-0.5-1	$2.77E2_{0.0E0}$▲	$2.43E2_{7.2E0}$▲	$2.34E2_{2.3E0}$▲	$2.35E2_{3.8E0}$▲	$2.20E2_{1.3E0}$▽	$2.21E2_{2.0E0}$
Robot-16-88-0.5-5	$2.33E3_{0.0E0}$▲	$2.07E3_{1.4E2}$▲	$1.89E3_{4.4E1}$▲	$1.88E3_{7.3E1}$▲	$1.60E3_{4.2E1}$▲	$1.55E3_{1.3E2}$
Robot-16-88-0.5-10	$3.47E3_{0.0E0}$▲	$3.57E3_{1.7E2}$▲	$3.33E3_{8.8E1}$▲	$3.23E3_{8.9E1}$▲	$2.88E3_{6.2E1}$▲	$2.62E3_{3.6E2}$
Robot-16-88-0.75-0.1	$3.60E2_{0.0E0}$▲	$3.53E2_{3.1E0}$▲	$3.48E2_{2.5E-1}$▲	$3.48E2_{7.5E-1}$▲	$3.47E2_{0.0E0}$▲	$3.43E2_{8.1E-1}$
Robot-16-88-0.75-0.5	$1.93E2_{0.0E0}$▲	$1.68E2_{3.0E0}$▲	$1.63E2_{1.6E0}$▲	$1.62E2_{1.8E0}$▲	$1.55E2_{0.0E0}$▲	$1.52E2_{3.5E0}$
Robot-16-88-0.75-1	$7.03E2_{0.0E0}$▲	$7.86E2_{4.4E1}$▲	$7.17E2_{1.0E1}$▲	$7.10E2_{2.1E1}$▲	$6.72E2_{0.0E0}$▲	$6.49E2_{1.2E1}$
Robot-16-88-0.75-5	$3.74E2_{0.0E0}$▲	$2.59E2_{2.0E1}$▲	$2.38E2_{6.9E0}$▲	$2.30E2_{9.2E0}$▲	$2.11E2_{1.9E0}$▲	$1.99E2_{6.2E0}$
Robot-16-88-0.75-10	$1.31E3_{0.0E0}$▲	$1.01E3_{3.6E1}$▲	$8.98E2_{4.2E1}$▲	$8.94E2_{2.5E1}$▲	$7.48E2_{1.2E1}$–	$7.55E2_{3.2E1}$
Robot-16-88-1-0.1	$3.32E2_{0.0E0}$▲	$2.91E2_{1.2E1}$▲	$2.82E2_{1.5E0}$▲	$2.80E2_{1.2E0}$▲	$2.79E2_{0.0E0}$▲	$2.76E2_{0.0E0}$
Robot-16-88-1-0.5	$5.97E2_{0.0E0}$▲	$5.24E2_{8.5E0}$▲	$5.14E2_{1.3E0}$▲	$5.16E2_{3.0E0}$▲	$5.06E2_{5.3E-1}$▲	$4.69E2_{3.3E0}$
Robot-16-88-1-1	$2.01E3_{0.0E0}$▲	$1.90E3_{1.0E2}$▲	$1.76E3_{3.4E1}$▲	$1.77E3_{2.1E1}$▲	$1.69E3_{3.0E1}$▲	$1.56E3_{2.0E1}$
Robot-16-88-1-5	$1.59E3_{0.0E0}$▲	$1.28E3_{1.2E2}$▲	$1.15E3_{5.8E1}$▲	$1.14E3_{6.2E1}$▲	$9.45E2_{1.3E1}$▲	$9.15E2_{4.3E1}$
Robot-16-88-1-10	$1.39E3_{0.0E0}$▲	$9.56E2_{1.2E2}$▲	$8.19E2_{3.8E1}$▲	$8.19E2_{2.9E1}$▲	$7.19E2_{1.1E1}$▲	$6.37E2_{8.4E1}$

Dark/light gray emphasizes the best/second-best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

$MEDIAN_{IQR}$. Tables also emphasize the best and second-best results with gray and gray-light backgrounds. Following the recommendations in Derrac, García, Molina, and Herrera (2011), García, Molina, Lozano, and Herrera (2008) to confirm the statistical significance of the results, we apply the non-parametric Wilcoxon signed ranks test on the results to assess statistical differences in a pairwise comparison for every instance, at 95% confidence level (Corder & Foreman, 2011). In the results tables, a symbol ▲ indicates that EFT-GVNS outperforms with statistical significance according to the Wilcoxon signed ranks test; we use ▽ otherwise. We marked '–' the cases with no statistical differences. In addition, we compute the multi-comparison non-parametric Friedman test examining for a 95% of confidence level and their respective Friedman average rankings to evaluate the global performance of the algorithms as in Santiago et al. (2021).

6. Results

Firstly we describe the achieved results over a large set of real-world application instances. For the FORTRAN double-precision floating-point instances (Fpppp), EFT-GVNS achieves an overwhelming performance, achieving the best-median value in all the instances considering eight and sixteen machines, a total of 50 scheduling problems, outperforming HEFT, EDA, EFT-ILS, GRASP-CPA, and MPQGA in every problem with statistical significance, results in Table 5. Similar behavior for the Fpppp instances with 32 and 64 machines, with 47 best-median values from a total of 50 scheduling problems achieved by EFT-GVNS, results in Table 6. Only in two cases is EFT-GVNS worse with statistical significance, the problem Fpppp-64-334-0.25-1 beaten by GRASP-CPA and the problem Fpppp-64-334-0.75-0.1 beaten by EFT-ILS, GRASP-CPA, and MPQGA, with no numeric difference

Table 10
Medians and IQR of the six algorithms over 30 independent runs.

Problema	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
Robot-32-88-0.1-0.1	$1.07E3_{0.0E0}$▲	$1.09E3_{9.3E0}$▲	$1.07E3_{1.9E0}$▲	$1.07E3_{4.5E-2}$▲	$1.07E3_{0.0E0}$▲	$1.07E3_{7.8E-1}$
Robot-32-88-0.1-0.5	$1.04E3_{0.0E0}$▲	$1.11E3_{3.8E1}$▲	$1.07E3_{7.1E0}$▲	$1.07E3_{1.1E1}$▲	$1.04E3_{0.0E0}$▲	$1.04E3_{0.0E0}$
Robot-32-88-0.1-1	$1.57E3_{0.0E0}$▲	$1.39E3_{4.4E1}$▲	$1.32E3_{2.0E1}$▲	$1.33E3_{1.0E1}$▲	$1.28E3_{1.1E1}$▲	$1.27E3_{2.8E0}$
Robot-32-88-0.1-5	$2.61E3_{0.0E0}$▲	$2.82E3_{2.4E2}$▲	$2.48E3_{1.1E2}$▲	$2.34E3_{1.2E2}$▲	$2.15E3_{5.1E1}$▲	$1.92E3_{1.4E1}$
Robot-32-88-0.1-10	$3.64E2_{0.0E0}$▲	$3.27E2_{1.9E1}$▲	$2.98E2_{1.2E1}$▲	$2.72E2_{1.4E1}$▲	$2.57E2_{3.4E0}$▲	$2.52E2_{8.8E0}$
Robot-32-88-0.25-0.1	$1.26E3_{0.0E0}$▲	$1.27E3_{7.6E0}$▲	$1.26E3_{2.3E0}$▲	$1.26E3_{9.7E-1}$▲	$1.26E3_{1.5E-1}$▲	$1.24E3_{4.9E0}$
Robot-32-88-0.25-0.5	$8.88E1_{0.0E0}$▲	$9.07E1_{3.4E0}$▲	$8.72E1_{5.0E-1}$▲	$8.71E1_{7.2E-1}$▲	$8.35E1_{0.0E0}$▽	$8.48E1_{4.3E-1}$
Robot-32-88-0.25-1	$1.88E3_{0.0E0}$▲	$2.01E3_{5.0E1}$▲	$1.91E3_{2.1E1}$▲	$1.91E3_{3.4E1}$▲	$1.76E3_{8.0E0}$▽	$1.80E3_{2.9E1}$
Robot-32-88-0.25-5	$3.78E3_{0.0E0}$▲	$3.42E3_{2.1E2}$▲	$3.06E3_{9.3E1}$▲	$3.07E3_{1.1E2}$▲	$2.62E3_{2.8E1}$▲	$2.46E3_{3.8E1}$
Robot-32-88-0.25-10	$2.54E3_{0.0E0}$▲	$2.05E3_{3.0E2}$▲	$1.83E3_{7.9E1}$▲	$1.69E3_{6.5E1}$▲	$1.53E3_{4.6E1}$▲	$1.40E3_{5.8E1}$
Robot-32-88-0.5-0.1	$3.80E2_{0.0E0}$▲	$3.82E2_{2.7E0}$▲	$3.75E2_{3.2E0}$▲	$3.74E2_{7.7E-1}$▲	$3.74E2_{1.7E0}$▲	$3.69E2_{2.5E0}$
Robot-32-88-0.5-0.5	$3.26E2_{0.0E0}$▲	$3.34E2_{6.5E0}$▲	$3.24E2_{3.5E0}$▲	$3.20E2_{3.5E0}$▲	$3.10E2_{0.0E0}$▲	$3.08E2_{3.4E0}$
Robot-32-88-0.5-1	$8.60E2_{0.0E0}$▲	$9.16E2_{2.5E1}$▲	$8.58E2_{1.4E1}$▲	$8.64E2_{1.1E1}$▲	$8.21E2_{5.8E0}$▲	$8.12E2_{1.5E1}$
Robot-32-88-0.5-5	$7.42E2_{0.0E0}$▲	$7.88E2_{5.7E1}$▲	$7.31E2_{2.4E1}$▲	$7.20E2_{2.5E1}$▲	$6.50E2_{1.5E1}$▲	$5.92E2_{2.9E1}$
Robot-32-88-0.5-10	$3.34E3_{0.0E0}$▲	$2.16E3_{1.7E2}$▲	$1.99E3_{7.9E1}$▲	$1.99E3_{5.4E1}$▲	$1.81E3_{2.5E1}$▲	$1.74E3_{0.0E0}$
Robot-32-88-0.75-0.1	$9.62E2_{0.0E0}$▲	$9.46E2_{4.7E0}$▲	$9.42E2_{0.0E0}$▲	$9.42E2_{1.5E0}$▲	$9.42E2_{0.0E0}$▲	$9.30E2_{0.0E0}$
Robot-32-88-0.75-0.5	$1.27E3_{0.0E0}$▲	$1.34E3_{3.4E1}$▲	$1.28E3_{9.0E0}$▲	$1.28E3_{1.0E1}$▲	$1.23E3_{0.0E0}$▲	$1.18E3_{3.7E1}$
Robot-32-88-0.75-1	$1.44E3_{0.0E0}$▲	$1.45E3_{2.4E1}$▲	$1.39E3_{1.3E1}$▲	$1.39E3_{1.8E1}$▲	$1.35E3_{1.2E1}$▲	$1.29E3_{2.5E1}$
Robot-32-88-0.75-5	$1.79E3_{0.0E0}$▲	$1.64E3_{8.9E1}$▲	$1.49E3_{4.5E1}$▲	$1.52E3_{4.3E1}$▲	$1.36E3_{2.6E1}$▲	$1.25E3_{3.3E1}$
Robot-32-88-0.75-10	$6.81E2_{0.0E0}$▲	$5.11E2_{3.6E1}$▲	$4.73E2_{2.0E1}$▲	$4.58E2_{1.4E1}$▲	$4.15E2_{1.1E1}$▲	$3.63E2_{1.6E1}$
Robot-32-88-1-0.1	$1.08E3_{0.0E0}$▲	$1.03E3_{1.5E1}$▲	$1.02E3_{3.7E0}$▲	$1.02E3_{2.3E0}$▲	$1.02E3_{0.0E0}$▲	$1.01E3_{5.9E0}$
Robot-32-88-1-0.5	$4.96E2_{0.0E0}$▲	$4.98E2_{2.7E1}$▲	$4.67E2_{4.1E-1}$▲	$4.67E2_{4.1E-1}$▲	$4.59E2_{0.0E0}$▲	$4.28E2_{1.5E1}$
Robot-32-88-1-1	$9.59E2_{0.0E0}$▲	$8.49E2_{3.1E1}$▲	$7.99E2_{2.2E1}$▲	$7.78E2_{3.5E1}$▲	$7.56E2_{3.2E-1}$▲	$7.36E2_{1.4E1}$
Robot-32-88-1-5	$1.57E3_{0.0E0}$▲	$1.24E3_{7.8E1}$▲	$1.14E3_{3.4E1}$▲	$1.14E3_{3.8E1}$▲	$1.07E3_{1.4E1}$▲	$1.04E3_{5.1E1}$
Robot-32-88-1-10	$1.92E3_{0.0E0}$▲	$1.51E3_{1.5E2}$▲	$1.40E3_{5.5E1}$▲	$1.38E3_{5.0E1}$▲	$1.26E3_{2.9E1}$▲	$1.15E3_{4.6E1}$
Robot-64-88-0.1-0.1	$2.15E2_{0.0E0}$▲	$2.17E2_{8.9E-1}$▲	$2.13E2_{6.6E-1}$−	$2.13E2_{1.0E0}$−	$2.12E2_{0.0E0}$▽	$2.13E2_{2.3E0}$
Robot-64-88-0.1-0.5	$8.76E2_{0.0E0}$▲	$9.28E2_{2.5E1}$▲	$8.72E2_{1.2E1}$▲	$8.86E2_{1.6E1}$▲	$8.50E2_{0.0E0}$▽	$8.54E2_{6.7E0}$
Robot-64-88-0.1-1	$9.26E2_{0.0E0}$▲	$9.86E2_{1.9E1}$▲	$9.51E2_{1.8E1}$▲	$9.45E2_{1.3E1}$▲	$9.13E2_{4.1E0}$▲	$8.98E2_{1.4E0}$
Robot-64-88-0.1-5	$2.47E3_{0.0E0}$▲	$2.86E3_{3.2E2}$▲	$2.56E3_{9.2E1}$▲	$2.49E3_{1.3E2}$▲	$2.31E3_{6.5E1}$▲	$2.09E3_{1.8E1}$
Robot-64-88-0.1-10	$4.27E3_{0.0E0}$▲	$3.67E3_{2.6E2}$▲	$3.28E3_{6.4E1}$▲	$3.18E3_{1.5E2}$▲	$2.85E3_{0.0E0}$▲	$2.63E3_{4.5E1}$
Robot-64-88-0.25-0.1	$1.36E3_{0.0E0}$▲	$1.37E3_{1.5E1}$▲	$1.35E3_{5.4E-1}$▲	$1.35E3_{3.1E0}$▲	$1.35E3_{0.0E0}$▲	$1.34E3_{8.5E0}$
Robot-64-88-0.25-0.5	$1.65E3_{0.0E0}$▲	$1.70E3_{5.1E1}$▲	$1.64E3_{1.2E1}$▲	$1.64E3_{1.2E1}$▲	$1.60E3_{5.3E0}$▲	$1.59E3_{1.1E1}$
Robot-64-88-0.25-1	$1.37E3_{0.0E0}$▲	$1.15E3_{3.5E1}$▲	$1.09E3_{1.1E1}$▲	$1.09E3_{1.0E1}$▲	$1.06E3_{2.3E0}$−	$1.06E3_{4.4E0}$
Robot-64-88-0.25-5	$2.55E3_{0.0E0}$▲	$2.50E3_{1.7E2}$▲	$2.25E3_{1.0E2}$▲	$2.23E3_{8.8E1}$▲	$2.01E3_{6.9E1}$▲	$1.90E3_{1.0E2}$
Robot-64-88-0.25-10	$3.01E2_{0.0E0}$▲	$1.93E2_{2.2E1}$▲	$1.75E2_{7.2E0}$−	$1.69E2_{6.6E0}$▽	$1.60E2_{1.9E0}$▽	$1.73E2_{8.5E0}$
Robot-64-88-0.5-0.1	$3.60E2_{0.0E0}$▲	$3.60E2_{4.8E-2}$▲	$3.57E2_{0.0E0}$▲	$3.57E2_{0.0E0}$▲	$3.56E2_{0.0E0}$▲	$3.53E2_{3.0E0}$
Robot-64-88-0.5-0.5	$1.09E3_{0.0E0}$▲	$1.13E3_{2.2E1}$▲	$1.10E3_{1.4E1}$▲	$1.11E3_{1.3E1}$▲	$1.09E3_{0.0E0}$▲	$1.07E3_{1.3E1}$
Robot-64-88-0.5-1	$4.74E2_{0.0E0}$▲	$4.72E2_{6.9E0}$▲	$4.51E2_{7.9E0}$▲	$4.49E2_{5.7E0}$▲	$4.33E2_{0.0E0}$−	$4.33E2_{6.8E0}$
Robot-64-88-0.5-5	$2.63E3_{0.0E0}$▲	$1.99E3_{2.5E2}$▲	$1.80E3_{7.7E1}$▲	$1.81E3_{7.4E1}$▲	$1.62E3_{3.9E1}$▲	$1.52E3_{6.6E1}$
Robot-64-88-0.5-10	$4.13E3_{0.0E0}$▲	$3.51E3_{3.8E2}$▲	$3.19E3_{1.6E2}$▲	$3.15E3_{6.2E1}$▲	$2.95E3_{5.4E1}$▲	$2.85E3_{5.1E1}$
Robot-64-88-0.75-0.1	$8.27E2_{0.0E0}$▲	$8.16E2_{1.4E0}$▲	$8.15E2_{9.8E-1}$▲	$8.15E2_{0.0E0}$▲	$8.12E2_{1.4E0}$▲	$8.01E2_{9.2E0}$
Robot-64-88-0.75-0.5	$1.58E3_{0.0E0}$▲	$1.63E3_{4.5E1}$▲	$1.58E3_{1.2E1}$▲	$1.58E3_{9.4E0}$▲	$1.55E3_{8.0E0}$▲	$1.53E3_{1.3E1}$
Robot-64-88-0.75-1	$2.20E2_{0.0E0}$▲	$2.39E2_{9.8E0}$▲	$2.25E2_{5.9E0}$▲	$2.19E2_{3.5E0}$▲	$2.13E2_{3.4E0}$▲	$2.10E2_{6.8E0}$
Robot-64-88-0.75-5	$2.88E3_{0.0E0}$▲	$2.81E3_{1.7E2}$▲	$2.54E3_{6.4E1}$▲	$2.46E3_{7.0E1}$▲	$2.28E3_{4.9E1}$▲	$2.21E3_{3.3E1}$
Robot-64-88-0.75-10	$6.04E3_{0.0E0}$▲	$5.31E3_{3.4E2}$▲	$4.96E3_{1.7E2}$▲	$4.87E3_{1.7E2}$▲	$4.34E3_{7.3E1}$▲	$4.03E3_{3.0E2}$
Robot-64-88-1-0.1	$8.45E2_{0.0E0}$▲	$8.45E2_{0.0E0}$▲	$8.45E2_{0.0E0}$▲	$8.45E2_{0.0E0}$▲	$8.45E2_{0.0E0}$▲	$8.27E2_{1.1E1}$
Robot-64-88-1-0.5	$1.01E3_{0.0E0}$▲	$1.01E3_{4.2E1}$▲	$9.59E2_{3.1E0}$▲	$9.59E2_{3.1E0}$▲	$9.58E2_{0.0E0}$▲	$8.88E2_{2.1E1}$
Robot-64-88-1-1	$2.47E2_{0.0E0}$▲	$2.57E2_{4.6E0}$▲	$2.47E2_{1.8E0}$▲	$2.45E2_{2.1E0}$▲	$2.41E2_{3.5E0}$▲	$2.30E2_{6.1E0}$
Robot-64-88-1-5	$1.95E3_{0.0E0}$▲	$1.88E3_{1.4E2}$▲	$1.72E3_{7.0E1}$▲	$1.71E3_{5.9E1}$▲	$1.58E3_{1.6E1}$▲	$1.41E3_{3.0E1}$
Robot-64-88-1-10	$3.83E3_{0.0E0}$▲	$3.57E3_{5.8E2}$▲	$3.08E3_{1.4E2}$▲	$3.01E3_{1.1E2}$▲	$2.81E3_{6.9E1}$▲	$2.50E3_{2.4E1}$

Dark/light gray emphasizes the best/second-best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

in the achieved median value by MPQGA, MPQGA standing out as the best competitor for EFT-GVNS. About The Laser Interferometer Gravitational-Wave Observatory (LIGO) results appear in Tables 7 and 8, MPQGA achieved 20 best-median values versus 80 best-median values found by EFT-GVNS. According to Wilcoxon's test, MPQGA outperforms EFT-GVNS with statistical significance in eleven cases, while EFT-GVNS outperforms MPQGA with statistical significance in 64 cases and 25 cases with no statistical difference. The other study algorithms (HEFT, EDA, EFT-ILS, GRASP-CPA) do not outperform with statistical significance EFT-GVNS in any instance of the LIGO application. Regarding the Robot control application instances, EFT-GVNS stands as the best algorithm with 83 best-median values, followed by MPQGA with 17 best-median values, results reported in Tables 9 and 10. According to Wilcoxon's test, MPQGA outperforms EFT-GVNS in 10 robot instances with statistical significance. In contrast, EFT-GVNS outperforms MPQGA in 77 cases with statistical significance, with no statistical difference found for 13 instances between them. Regarding the Sparse Matrix solver application (Sparse), EFT-GVNS achieves 73 best-median values, MPQGA achieves 27 best-median values, results on Tables 11 and 12. According to Wilcoxon's test, MPQGA outperforms EFT-GVNS in 21 sparse cases with statistical significance. EFT-GVNS outperforms MPQGA in 60 cases and, for 19 cases, found no statistical significance between MPQGA and EFT-GVNS.

To validate the overall performance of EFT-GVNS on the set of real-world applications, we compute the Friedman non-parametric test, the test outcome a p-value ≤ 0.05, ensuring a statistical significance of 95%. According to their respective Friedman average ranking in Table 4, the best performer algorithm is our proposed EFT-GVNS, followed by MPQGA, GRASP-CPA, EFT-ILS, EDA, and HEFT, respectively.

To give a graphical insight into the performance of EFT-GVNS, we boxplot their best performer instance of the studied real-world applications. Fig. 2 shows the boxplot for the six algorithms over the Fpppp-34-334-1-1 scheduling instance (the best result for EFT-GVNS). The result of HEFT is due is deterministic nature, with an interquartile range (IQR) of zero, and as the reference of minimum performance. EDA is the second-worst and least stable algorithm with a broader IQR.

Table 11
Medians and IQR of the six algorithms over 30 independent runs.

Problema	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
Sparse-8-96-0.1-0.1	$2.39E1_{0.0E0}$▲	$2.57E1_{1.3E0}$▲	$2.63E1_{6.2E-1}$▲	$2.61E1_{4.0E-1}$▲	$2.38E1_{5.6E-2}$−	$2.38E1_{2.4E-2}$
Sparse-8-96-0.1-0.5	$2.47E2_{0.0E0}$▲	$2.55E2_{1.9E1}$▲	$2.66E2_{7.8E0}$▲	$2.69E2_{1.2E1}$▲	$2.30E2_{5.9E-1}$▲	$2.30E2_{3.8E-1}$
Sparse-8-96-0.1-1	$4.25E2_{0.0E0}$▲	$4.35E2_{5.9E1}$▲	$4.47E2_{2.1E1}$▲	$4.44E2_{1.5E1}$▲	$3.66E2_{5.7E-1}$▲	$3.64E2_{6.7E-1}$
Sparse-8-96-0.1-5	$9.79E2_{0.0E0}$▲	$7.41E2_{4.8E1}$▲	$7.72E2_{2.6E1}$▲	$7.60E2_{3.5E1}$▲	$5.58E2_{9.0E0}$▲	$5.37E2_{6.6E0}$
Sparse-8-96-0.1-10	$1.18E3_{0.0E0}$▲	$8.18E2_{1.1E2}$▲	$8.85E2_{4.9E1}$▲	$9.06E2_{4.0E1}$▲	$7.08E2_{2.8E1}$▲	$5.13E2_{2.9E1}$
Sparse-8-96-0.25-0.1	$5.10E2_{0.0E0}$▲	$4.94E2_{2.6E1}$▲	$5.08E2_{1.1E1}$▲	$5.11E2_{1.0E1}$▲	$4.61E2_{1.7E0}$▲	$4.59E2_{2.1E0}$
Sparse-8-96-0.25-0.5	$2.05E1_{0.0E0}$▲	$2.13E1_{1.0E0}$▲	$2.18E1_{9.2E-1}$▲	$2.20E1_{6.0E-1}$▲	$1.89E1_{9.7E-2}$▲	$1.88E1_{9.4E-2}$
Sparse-8-96-0.25-1	$3.47E2_{0.0E0}$▲	$3.18E2_{2.1E1}$▲	$3.30E2_{1.2E1}$▲	$3.35E2_{2.3E1}$▲	$2.74E2_{1.2E0}$▲	$2.72E2_{1.6E0}$
Sparse-8-96-0.25-5	$2.86E2_{0.0E0}$▲	$1.94E2_{2.2E1}$▲	$2.08E2_{5.8E0}$▲	$2.06E2_{1.2E1}$▲	$1.51E2_{3.7E0}$▲	$1.46E2_{3.0E0}$
Sparse-8-96-0.25-10	$1.79E3_{0.0E0}$▲	$1.40E3_{8.8E1}$▲	$1.46E3_{4.7E1}$▲	$1.45E3_{4.2E1}$▲	$1.14E3_{2.7E-1}$▲	$1.12E3_{6.4E0}$
Sparse-8-96-0.5-0.1	$8.49E2_{0.0E0}$▲	$8.09E2_{4.7E1}$▲	$8.48E2_{2.5E1}$▲	$8.45E2_{3.9E1}$▲	$7.26E2_{4.2E0}$▲	$7.20E2_{4.7E0}$
Sparse-8-96-0.5-0.5	$7.93E2_{0.0E0}$▲	$8.21E2_{6.2E1}$▲	$8.41E2_{2.6E1}$▲	$8.40E2_{2.5E1}$▲	$7.37E2_{3.9E0}$−	$7.36E2_{4.1E0}$
Sparse-8-96-0.5-1	$5.58E2_{0.0E0}$▲	$5.80E2_{4.9E1}$▲	$5.98E2_{2.0E1}$▲	$6.05E2_{2.5E1}$▲	$4.98E2_{3.3E0}$−	$4.97E2_{3.2E0}$
Sparse-8-96-0.5-5	$5.49E1_{0.0E0}$▲	$3.34E1_{2.2E0}$▲	$3.58E1_{1.3E0}$▲	$3.55E1_{1.2E0}$▲	$2.73E1_{8.7E-1}$▲	$2.38E1_{6.4E-1}$
Sparse-8-96-0.5-10	$8.52E2_{0.0E0}$▲	$5.29E2_{7.0E1}$▲	$5.67E2_{3.2E1}$▲	$5.59E2_{3.0E1}$▲	$4.51E2_{1.8E1}$▲	$4.08E2_{1.8E1}$
Sparse-8-96-0.75-0.1	$6.05E2_{0.0E0}$▲	$6.18E2_{1.6E1}$▲	$6.50E2_{1.8E1}$▲	$6.43E2_{1.6E1}$▲	$5.61E2_{2.1E0}$−	$5.60E2_{5.7E0}$
Sparse-8-96-0.75-0.5	$2.56E2_{0.0E0}$▲	$2.56E2_{1.2E1}$▲	$2.67E2_{6.8E0}$▲	$2.66E2_{8.5E0}$▲	$2.21E2_{1.7E0}$▽	$2.23E2_{2.0E0}$
Sparse-8-96-0.75-1	$5.93E2_{0.0E0}$▲	$6.13E2_{5.0E1}$▲	$6.51E2_{2.4E1}$▲	$6.53E2_{1.5E1}$▲	$5.26E2_{4.3E0}$▽	$5.39E2_{5.5E0}$
Sparse-8-96-0.75-5	$1.30E3_{0.0E0}$▲	$8.64E2_{1.0E2}$▲	$9.43E2_{8.7E1}$▲	$9.51E2_{6.9E1}$▲	$7.13E2_{1.8E1}$▲	$6.26E2_{3.9E1}$
Sparse-8-96-0.75-10	$1.26E3_{0.0E0}$▲	$9.25E2_{1.2E2}$▲	$1.03E3_{1.1E2}$▲	$9.20E2_{5.1E1}$▲	$7.80E2_{2.6E1}$▲	$6.80E2_{3.9E1}$
Sparse-8-96-1-0.1	$6.89E2_{0.0E0}$▲	$6.64E2_{2.1E1}$▲	$6.98E2_{1.1E1}$▲	$6.84E2_{1.1E1}$▲	$6.09E2_{3.4E0}$▽	$6.14E2_{5.9E0}$
Sparse-8-96-1-0.5	$6.84E2_{0.0E0}$▲	$6.07E2_{4.9E1}$▲	$6.38E2_{2.8E1}$▲	$6.39E2_{2.6E1}$▲	$5.04E2_{3.7E0}$▽	$5.13E2_{1.0E1}$
Sparse-8-96-1-1	$2.57E2_{0.0E0}$▲	$2.37E2_{1.4E1}$▲	$2.49E2_{7.3E0}$▲	$2.45E2_{1.1E1}$▲	$1.99E2_{2.4E0}$▽	$2.05E2_{3.2E0}$
Sparse-8-96-1-5	$7.40E2_{0.0E0}$▲	$5.15E2_{5.2E1}$▲	$5.43E2_{2.1E1}$▲	$5.29E2_{2.3E1}$▲	$4.16E2_{1.2E1}$▲	$3.43E2_{1.6E1}$
Sparse-8-96-1-10	$2.65E2_{0.0E0}$▲	$1.83E2_{2.1E1}$▲	$1.97E2_{1.3E1}$▲	$1.95E2_{1.1E1}$▲	$1.64E2_{6.6E0}$▲	$1.52E2_{8.9E0}$
Sparse-16-96-0.1-0.1	$4.08E2_{0.0E0}$▲	$5.87E2_{1.6E2}$▲	$5.48E2_{2.1E1}$▲	$5.45E2_{2.2E1}$▲	$4.00E2_{1.6E-1}$▲	$4.00E2_{0.0E0}$
Sparse-16-96-0.1-0.5	$3.00E2_{0.0E0}$▲	$3.46E2_{8.7E1}$▲	$3.56E2_{1.9E1}$▲	$3.60E2_{1.5E1}$▲	$2.55E2_{5.6E-1}$▲	$2.55E2_{0.0E0}$
Sparse-16-96-0.1-1	$4.17E2_{0.0E0}$▲	$4.68E2_{1.1E2}$▲	$4.64E2_{1.8E1}$▲	$4.53E2_{2.3E1}$▲	$3.59E2_{3.1E0}$▲	$3.58E2_{6.2E-1}$
Sparse-16-96-0.1-5	$2.21E2_{0.0E0}$▲	$1.83E2_{1.8E1}$▲	$1.91E2_{5.2E0}$▲	$1.90E2_{6.4E0}$▲	$1.68E2_{4.8E-1}$▲	$1.51E2_{1.3E1}$
Sparse-16-96-0.1-10	$6.18E2_{0.0E0}$▲	$5.52E2_{7.3E1}$▲	$5.56E2_{7.2E0}$▲	$5.53E2_{8.1E0}$▲	$4.63E2_{6.2E1}$▲	$3.93E2_{1.4E0}$
Sparse-16-96-0.25-0.1	$2.62E2_{0.0E0}$▲	$3.05E2_{6.7E1}$▲	$3.05E2_{1.4E1}$▲	$3.03E2_{1.1E1}$▲	$2.26E2_{1.3E0}$▽	$2.28E2_{2.7E0}$
Sparse-16-96-0.25-0.5	$2.62E2_{0.0E0}$▲	$2.22E2_{4.7E1}$▲	$2.28E2_{8.3E0}$▲	$2.26E2_{8.5E0}$▲	$1.74E2_{0.0E0}$−	$1.74E2_{1.1E0}$
Sparse-16-96-0.25-1	$2.34E2_{0.0E0}$▲	$2.92E2_{6.1E1}$▲	$2.79E2_{1.0E1}$▲	$2.74E2_{1.2E1}$▲	$2.19E2_{0.0E0}$▲	$2.16E2_{1.4E0}$
Sparse-16-96-0.25-5	$9.29E2_{0.0E0}$▲	$7.86E2_{8.9E1}$▲	$8.08E2_{2.5E1}$▲	$8.04E2_{1.7E1}$▲	$7.26E2_{5.6E0}$▲	$6.04E2_{3.6E0}$
Sparse-16-96-0.25-10	$1.28E3_{0.0E0}$▲	$1.05E3_{7.4E1}$▲	$1.10E3_{3.7E1}$▲	$1.07E3_{3.5E1}$▲	$9.84E2_{3.9E1}$▲	$7.54E2_{0.0E0}$
Sparse-16-96-0.5-0.1	$1.68E2_{0.0E0}$▲	$1.69E2_{2.8E1}$▲	$1.91E2_{9.2E0}$▲	$1.93E2_{6.1E0}$▲	$1.40E2_{1.5E-1}$−	$1.41E2_{1.3E0}$
Sparse-16-96-0.5-0.5	$3.13E2_{0.0E0}$▲	$2.74E2_{6.5E1}$▲	$2.79E2_{1.3E1}$▲	$2.81E2_{8.9E0}$▲	$2.14E2_{1.3E0}$▲	$2.09E2_{4.8E0}$
Sparse-16-96-0.5-1	$2.15E2_{0.0E0}$▲	$1.96E2_{5.3E1}$▲	$2.05E2_{7.2E0}$▲	$2.04E2_{8.8E0}$▲	$1.72E2_{2.7E0}$▲	$1.62E2_{6.6E0}$
Sparse-16-96-0.5-5	$7.83E2_{0.0E0}$▲	$7.26E2_{1.1E2}$▲	$7.59E2_{3.9E1}$▲	$7.69E2_{7.6E1}$▲	$6.81E2_{2.1E1}$▲	$6.21E2_{3.0E1}$
Sparse-16-96-0.5-10	$2.46E3_{0.0E0}$▲	$1.79E3_{9.9E1}$▲	$1.81E3_{2.1E1}$▲	$1.79E3_{4.5E1}$▲	$1.72E3_{1.0E1}$▲	$1.44E3_{1.4E2}$
Sparse-16-96-0.75-0.1	$3.80E2_{0.0E0}$▲	$3.52E2_{5.2E1}$▲	$3.55E2_{1.0E1}$▲	$3.47E2_{1.3E1}$▲	$2.67E2_{2.6E0}$▽	$2.78E2_{5.3E0}$
Sparse-16-96-0.75-0.5	$3.00E2_{0.0E0}$▲	$3.04E2_{6.7E1}$▲	$2.92E2_{1.5E1}$▲	$2.92E2_{1.1E1}$▲	$2.25E2_{1.4E0}$▲	$2.22E2_{6.7E0}$
Sparse-16-96-0.75-1	$3.69E2_{0.0E0}$▲	$3.34E2_{7.6E1}$▲	$3.46E2_{1.2E1}$▲	$3.39E2_{1.1E1}$▲	$2.77E2_{5.1E0}$▲	$2.74E2_{1.1E1}$
Sparse-16-96-0.75-5	$7.19E2_{0.0E0}$▲	$5.80E2_{1.1E2}$▲	$5.95E2_{3.9E1}$▲	$5.97E2_{2.7E1}$▲	$5.29E2_{2.5E1}$▲	$4.97E2_{5.0E1}$
Sparse-16-96-0.75-10	$1.79E3_{0.0E0}$▲	$1.54E3_{2.4E2}$▲	$1.65E3_{7.1E1}$▲	$1.64E3_{7.9E1}$▲	$1.42E3_{2.0E1}$▲	$1.30E3_{1.8E1}$
Sparse-16-96-1-0.1	$1.75E2_{0.0E0}$▲	$1.97E2_{2.1E1}$▲	$2.01E2_{7.5E0}$▲	$2.02E2_{8.5E0}$▲	$1.49E2_{1.5E0}$▽	$1.52E2_{2.2E0}$
Sparse-16-96-1-0.5	$2.77E2_{0.0E0}$▲	$2.33E2_{5.4E1}$▲	$2.50E2_{1.8E1}$▲	$2.47E2_{1.4E1}$▲	$1.86E2_{1.7E0}$▲	$1.86E2_{0.0E0}$
Sparse-16-96-1-1	$1.64E2_{0.0E0}$▲	$1.72E2_{5.1E1}$▲	$1.81E2_{8.4E0}$▲	$1.79E2_{7.0E0}$▲	$1.37E2_{1.4E0}$−	$1.38E2_{9.7E0}$
Sparse-16-96-1-5	$3.92E2_{0.0E0}$▲	$3.66E2_{4.8E1}$▲	$3.72E2_{1.9E1}$▲	$3.71E2_{2.2E1}$▲	$3.23E2_{9.6E0}$▲	$3.05E2_{4.6E0}$
Sparse-16-96-1-10	$2.11E3_{0.0E0}$▲	$1.70E3_{5.4E1}$▲	$1.77E3_{5.3E1}$▲	$1.73E3_{6.6E1}$▲	$1.63E3_{3.8E0}$▲	$1.52E3_{1.5E2}$

Dark/light gray emphasizes the best/second-best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

The performance of EDA could be because a random priority task order initializes it without guarantees of sampling by probability a quality priority for tasks. Following EDA in performance order are GRASP-CPA and EFT-ILS, producing similar median values and IQR; EFT-ILS is more stable but with outliers. The second-best algorithm is MPQGA, with an IQR of zero, confirmed in Table 6. The achieved median value by EFT-GVNS is significantly superior to the other algorithms. Also, graphically, the achieved minimum value by EFT-GVNS is relevant without considering the produced outlier. The boxplot's graphical results are consistent with the Friedman ranking from Table 4. After analyzing the results with the median values, IQRs, Friedman ranking, and Boxplots, we can confirm that EFT-GVNS outperforms the algorithms in the comparison meaningful in the studied set of real-world applications.

Next, we analyze the performance of EFT-GVNS compared to the considered state-of-the-art algorithms over a set of synthetic instances from the literature, the same as in Santiago et al. (2020a). Unlike the real-world application's problem set with a five-minute stop criterion, the synthetic benchmark has 100,000 objective function evaluations per independent run. Once again, we compute the non-parametric Friedman test, giving a statistical significance of 95% for differences in the synthetic benchmark. The computed Friedman average rankings for the small benchmark are EFT-GVNS 1.99, MPQGA 2.64, GRASP-CPA 2.69, EFT-ILS 3.12, EDA 4.92, and HEFT 5.64, being consistent with the results of the real-world applications set. Table 13 shows the achieved median values and IQR over the small benchmark of 14 synthetic scheduling problems and their respective citations. EFT-GVNS achieves the 14 best-median values, from 14 cases, with an IQR of zero in every case, followed by MPQGA with ten best-median values, GRASP-CPA with seven, EFT-ILS with 6, and EDA with one. EFT-GVNS outperforms HEFT and EDA in every case, EFT-ILS in ten cases, GRASP-CPA in 8 cases, and MPQGA in four. None of the algorithms in the comparison outperforms EFT-GVNS with statistical significance.

Table 14 shows the optimum values for the small benchmark of 14 synthetic scheduling problems and the best values achieved by the algorithms; HEFT is not in the comparison. EFT-GVNS stands as the best performer achieving 14 optimum values, followed by GRASP-CPA with

Table 12
Medians and IQR of the six algorithms over 30 independent runs.

Problema	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
Sparse-32-96-0.1-0.1	$3.73E2_{0.0E0}$▲	$4.19E2_{3.6E1}$▲	$3.70E2_{2.1E-1}$▲	$3.70E2_{4.5E-1}$▲	$3.69E2_{1.3E0}$▲	$3.66E2_{0.0E0}$
Sparse-32-96-0.1-0.5	$5.47E2_{0.0E0}$▲	$5.67E2_{6.1E1}$▲	$4.94E2_{1.6E1}$▲	$4.93E2_{1.2E1}$▲	$4.85E2_{2.3E0}$▽	$4.89E2_{0.0E0}$
Sparse-32-96-0.1-1	$5.39E2_{0.0E0}$▲	$6.05E2_{7.2E1}$▲	$5.33E2_{9.9E0}$▲	$5.33E2_{1.0E1}$▲	$4.76E2_{5.2E-1}$▽	$4.81E2_{5.7E0}$
Sparse-32-96-0.1-5	$2.35E2_{0.0E0}$▲	$2.62E2_{1.5E1}$▲	$2.37E2_{6.5E0}$▲	$2.37E2_{5.2E0}$▲	$1.92E2_{3.3E0}$▲	$1.79E2_{8.8E-1}$
Sparse-32-96-0.1-10	$1.47E3_{0.0E0}$▲	$1.48E3_{9.0E1}$▲	$1.41E3_{7.7E0}$▲	$1.41E3_{3.5E1}$▲	$1.34E3_{1.2E1}$▲	$1.30E3_{2.0E1}$
Sparse-32-96-0.25-0.1	$2.50E2_{0.0E0}$▲	$2.23E2_{2.1E1}$▲	$1.97E2_{9.4E0}$▲	$1.94E2_{1.0E1}$▲	$1.81E2_{2.0E0}$▽	$1.83E2_{7.1E0}$
Sparse-32-96-0.25-0.5	$1.74E2_{0.0E0}$▲	$1.59E2_{1.1E1}$▲	$1.41E2_{2.4E0}$▲	$1.41E2_{4.2E0}$▲	$1.34E2_{6.2E-1}$−	$1.34E2_{1.6E0}$
Sparse-32-96-0.25-1	$4.31E2_{0.0E0}$▲	$4.84E2_{4.0E1}$▲	$4.34E2_{1.6E1}$▲	$4.30E2_{9.5E0}$▲	$4.13E2_{1.8E0}$−	$4.13E2_{5.9E0}$
Sparse-32-96-0.25-5	$7.81E2_{0.0E0}$▲	$6.91E2_{4.8E1}$▲	$6.09E2_{3.3E1}$▲	$5.91E2_{4.7E1}$▲	$5.58E2_{1.3E1}$▲	$4.99E2_{2.0E0}$
Sparse-32-96-0.25-10	$1.23E3_{0.0E0}$▲	$7.84E2_{3.2E1}$▲	$7.48E2_{4.5E0}$▲	$7.50E2_{4.6E0}$▲	$5.26E2_{6.3E0}$−	$5.30E2_{1.4E1}$
Sparse-32-96-0.5-0.1	$4.57E2_{0.0E0}$▲	$4.22E2_{3.7E1}$▲	$3.68E2_{1.9E1}$▲	$3.65E2_{1.6E1}$▲	$3.40E2_{0.0E0}$▽	$3.42E2_{1.7E0}$
Sparse-32-96-0.5-0.5	$3.89E1_{0.0E0}$▲	$3.91E1_{3.1E0}$▲	$3.48E1_{1.1E0}$▲	$3.45E1_{1.6E0}$▲	$3.21E1_{2.4E-3}$▲	$3.12E1_{4.3E-1}$
Sparse-32-96-0.5-1	$5.01E2_{0.0E0}$▲	$6.15E2_{4.8E1}$▲	$5.33E2_{1.1E1}$▲	$5.29E2_{8.1E0}$▲	$4.68E2_{8.3E0}$−	$4.68E2_{7.6E0}$
Sparse-32-96-0.5-5	$3.98E2_{0.0E0}$▲	$3.93E2_{3.8E1}$▲	$3.57E2_{1.8E1}$▲	$3.52E2_{2.2E1}$▲	$3.21E2_{1.0E1}$▲	$3.08E2_{2.4E1}$
Sparse-32-96-0.5-10	$4.38E2_{0.0E0}$▲	$4.54E2_{2.0E1}$▲	$4.28E2_{3.2E0}$▲	$4.27E2_{3.3E0}$▲	$4.17E2_{7.9E0}$▲	$3.34E2_{0.0E0}$
Sparse-32-96-0.75-0.1	$1.42E2_{0.0E0}$▲	$1.39E2_{1.1E1}$▲	$1.20E2_{5.0E0}$▲	$1.20E2_{5.1E0}$▲	$1.18E2_{0.0E0}$▲	$1.16E2_{1.4E0}$
Sparse-32-96-0.75-0.5	$2.12E2_{0.0E0}$▲	$2.45E2_{2.6E1}$▲	$2.17E2_{6.3E0}$▲	$2.15E2_{4.0E0}$▲	$2.03E2_{2.2E0}$−	$2.01E2_{5.6E0}$
Sparse-32-96-0.75-1	$2.53E2_{0.0E0}$▲	$2.50E2_{1.6E1}$▲	$2.27E2_{7.8E0}$▲	$2.27E2_{8.4E0}$▲	$2.02E2_{4.1E0}$▽	$2.10E2_{7.2E0}$
Sparse-32-96-0.75-5	$4.06E1_{0.0E0}$▲	$3.98E1_{2.7E0}$▲	$3.63E1_{1.2E0}$▲	$3.60E1_{9.4E-1}$▲	$3.32E1_{7.8E-1}$▲	$2.97E1_{0.0E0}$
Sparse-32-96-0.75-10	$8.51E2_{0.0E0}$▲	$7.01E2_{1.6E1}$▲	$6.66E2_{2.4E1}$▲	$6.54E2_{3.3E1}$▲	$5.54E2_{1.1E1}$▲	$5.42E2_{2.0E0}$
Sparse-32-96-1-0.1	$3.36E1_{0.0E0}$▲	$3.37E1_{3.3E0}$▲	$3.00E1_{5.3E-1}$▲	$3.01E1_{5.3E-1}$▲	$2.91E1_{2.6E-1}$▲	$2.83E1_{0.0E0}$
Sparse-32-96-1-0.5	$2.86E2_{0.0E0}$▲	$3.23E2_{2.1E1}$▲	$2.87E2_{6.7E0}$▲	$2.89E2_{5.6E0}$▲	$2.65E2_{8.1E0}$▲	$2.61E2_{4.1E0}$
Sparse-32-96-1-1	$2.23E2_{0.0E0}$▲	$2.33E2_{1.6E1}$▲	$2.06E2_{4.9E0}$▲	$2.08E2_{7.2E0}$▲	$1.88E2_{2.2E0}$▽	$1.99E2_{6.5E0}$
Sparse-32-96-1-5	$6.61E2_{0.0E0}$▲	$6.95E2_{4.9E1}$▲	$6.00E2_{2.7E1}$▲	$5.84E2_{3.3E1}$▲	$5.66E2_{1.3E1}$−	$5.68E2_{5.8E1}$
Sparse-32-96-1-10	$1.20E3_{0.0E0}$▲	$1.08E3_{3.4E1}$▲	$9.55E2_{8.5E1}$▲	$9.72E2_{1.1E2}$▲	$8.91E2_{1.3E1}$▲	$8.64E2_{1.5E1}$
Sparse-64-96-0.1-0.1	$1.37E2_{0.0E0}$▲	$1.39E2_{2.5E0}$▲	$1.36E2_{4.2E-1}$▲	$1.37E2_{3.7E-2}$▲	$1.36E2_{0.0E0}$▲	$1.36E2_{7.9E-1}$
Sparse-64-96-0.1-0.5	$4.82E2_{0.0E0}$▲	$5.10E2_{3.2E1}$▲	$4.74E2_{0.0E0}$▽	$4.75E2_{3.3E0}$▽	$4.74E2_{1.1E-1}$▽	$4.78E2_{2.9E0}$
Sparse-64-96-0.1-1	$3.31E2_{0.0E0}$▲	$3.28E2_{1.0E1}$▲	$3.12E2_{3.4E0}$▲	$3.12E2_{2.2E0}$▲	$3.10E2_{1.8E0}$▽	$3.11E2_{0.0E0}$
Sparse-64-96-0.1-5	$3.59E2_{0.0E0}$▲	$3.51E2_{3.0E1}$▲	$3.29E2_{1.1E0}$▽	$3.29E2_{1.1E0}$▽	$3.27E2_{8.4E-2}$▽	$3.30E2_{0.0E0}$
Sparse-64-96-0.1-10	$1.52E2_{0.0E0}$▲	$1.52E2_{1.7E1}$▲	$1.39E2_{6.3E-1}$▲	$1.38E2_{1.1E-1}$▲	$1.38E2_{2.8E-14}$▲	$1.29E2_{2.4E-1}$
Sparse-64-96-0.25-0.1	$3.66E2_{0.0E0}$▲	$2.78E2_{3.4E0}$▲	$2.73E2_{2.2E-1}$▽	$2.73E2_{2.2E-1}$▽	$2.73E2_{0.0E0}$▽	$2.76E2_{3.1E0}$
Sparse-64-96-0.25-0.5	$4.05E2_{0.0E0}$▲	$4.13E2_{7.2E0}$▲	$4.05E2_{2.2E0}$▲	$4.05E2_{4.4E0}$▲	$3.98E2_{4.4E0}$−	$3.98E2_{5.9E0}$
Sparse-64-96-0.25-1	$1.00E2_{0.0E0}$▲	$1.07E2_{4.4E0}$▲	$9.32E1_{6.1E-1}$▲	$9.32E1_{0.0E0}$▲	$9.31E1_{4.8E-2}$−	$9.31E1_{0.0E0}$
Sparse-64-96-0.25-5	$3.57E2_{0.0E0}$▲	$4.73E2_{3.1E1}$▲	$3.77E2_{1.1E1}$▲	$3.57E2_{9.4E0}$▲	$3.51E2_{6.5E-1}$▲	$3.07E2_{0.0E0}$
Sparse-64-96-0.25-10	$5.63E2_{0.0E0}$▲	$5.63E2_{1.8E1}$▲	$5.47E2_{4.6E-1}$▲	$5.47E2_{5.7E-1}$▲	$5.45E2_{3.9E-1}$▲	$3.83E2_{0.0E0}$
Sparse-64-96-0.5-0.1	$4.52E2_{0.0E0}$▲	$4.53E2_{1.1E1}$▲	$4.52E2_{5.8E-1}$▲	$4.52E2_{5.8E-1}$▲	$4.50E2_{0.0E0}$▲	$4.41E2_{0.0E0}$
Sparse-64-96-0.5-0.5	$1.21E2_{0.0E0}$−	$1.44E2_{8.4E0}$▲	$1.26E2_{6.7E0}$▲	$1.21E2_{6.7E0}$▲	$1.21E2_{0.0E0}$−	$1.21E2_{0.0E0}$
Sparse-64-96-0.5-1	$3.27E2_{0.0E0}$▲	$2.93E2_{1.2E0}$▲	$2.74E2_{2.5E0}$▲	$2.75E2_{3.2E0}$▲	$2.69E2_{0.0E0}$▲	$2.38E2_{0.0E0}$
Sparse-64-96-0.5-5	$7.82E2_{0.0E0}$▲	$7.76E2_{2.4E1}$▲	$7.03E2_{1.7E1}$−	$6.99E2_{1.1E1}$−	$6.79E2_{3.5E0}$▽	$7.12E2_{3.4E1}$
Sparse-64-96-0.5-10	$1.85E3_{0.0E0}$▲	$1.79E3_{1.0E2}$▲	$1.74E3_{0.0E0}$−	$1.74E3_{0.0E0}$−	$1.74E3_{0.0E0}$−	$1.74E3_{6.4E1}$
Sparse-64-96-0.75-0.1	$2.91E2_{0.0E0}$▲	$2.94E2_{3.0E0}$▲	$2.90E2_{5.2E0}$▲	$2.86E2_{5.2E0}$▲	$2.86E2_{0.0E0}$▲	$2.84E2_{0.0E0}$
Sparse-64-96-0.75-0.5	$3.79E2_{0.0E0}$▲	$3.84E2_{1.5E1}$▲	$3.57E2_{0.0E0}$▲	$3.57E2_{3.9E0}$▲	$3.50E2_{1.8E0}$▲	$3.46E2_{4.0E0}$
Sparse-64-96-0.75-1	$4.53E2_{0.0E0}$▲	$4.02E2_{1.0E1}$▲	$3.71E2_{3.6E0}$▲	$3.71E2_{1.3E1}$▲	$3.48E2_{6.6E0}$▽	$3.53E2_{1.2E1}$
Sparse-64-96-0.75-5	$3.33E2_{0.0E0}$▲	$3.28E2_{7.3E0}$▲	$3.13E2_{3.0E0}$▲	$3.13E2_{1.8E0}$▲	$3.04E2_{0.0E0}$▲	$2.88E2_{1.8E0}$
Sparse-64-96-0.75-10	$4.72E2_{0.0E0}$▲	$4.89E2_{1.6E1}$▲	$4.64E2_{3.0E0}$▲	$4.64E2_{4.8E0}$▲	$4.59E2_{1.7E0}$▲	$4.53E2_{5.8E1}$
Sparse-64-96-1-0.1	$2.56E2_{0.0E0}$▲	$2.32E2_{7.7E0}$▲	$2.32E2_{0.0E0}$−	$2.32E2_{0.0E0}$−	$2.32E2_{0.0E0}$−	$2.32E2_{0.0E0}$
Sparse-64-96-1-0.5	$1.90E2_{0.0E0}$▲	$1.88E2_{7.3E0}$▲	$1.72E2_{2.9E0}$▲	$1.72E2_{2.5E0}$▲	$1.53E2_{8.1E0}$−	$1.57E2_{6.2E0}$
Sparse-64-96-1-1	$8.61E1_{0.0E0}$▲	$9.24E1_{1.3E1}$▲	$8.33E1_{0.0E0}$▽	$8.33E1_{0.0E0}$▽	$8.33E1_{0.0E0}$▽	$8.47E1_{1.0E0}$
Sparse-64-96-1-5	$4.15E2_{0.0E0}$▲	$4.66E2_{3.5E1}$▲	$4.09E2_{6.8E0}$▲	$4.06E2_{7.0E0}$▲	$3.95E2_{3.2E0}$▲	$3.75E2_{1.7E0}$
Sparse-64-96-1-10	$1.84E3_{0.0E0}$▲	$1.66E3_{3.8E0}$▲	$1.65E3_{1.1E0}$▲	$1.65E3_{0.0E0}$▲	$1.64E3_{6.8E0}$▲	$1.41E3_{0.0E0}$

Dark/light gray emphasizes the best/second-best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

Table 13
Medians and IQR of the six algorithms over 30 independent runs.

Problem	HEFT	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
Ahmad-3-9 (Ahmad et al., 1998)	$38_{0.0}$▲	$27_{1.0}$▲	$27_{0.0}$−	$27_{0.0}$−	$27_{0.0}$−	$27_{0.0}$
Daoud-2-11 (Daoud & Kharma, 2011)	$78.5_{0.0}$▲	70.5_{11}▲	$60_{1.5}$▲	$58_{4.0}$▲	$56_{0.0}$−	$56_{0.0}$
Eswari-2-11 (Eswari & Nickolas, 2010)	$78.5_{0.0}$▲	70_{10}▲	$60_{1.5}$▲	$58.5_{4.0}$▲	$56_{0.0}$−	$56_{0.0}$
Hamid-3-10 (Arabnejad, 2013)	$110_{0.0}$▲	108_{11}▲	$100_{0.0}$−	$100_{0.0}$−	$100_{0.0}$−	$100_{0.0}$
Heteropar-4-12 (Arabnejad & Barbosa, 2011)	$150_{0.0}$▲	$136_{0.0}$▲	$136_{0.0}$▲	133_{12}▲	$136_{0.0}$▲	$124_{0.0}$
Hsu-3-10 (Hsu et al., 2007)	$92_{0.0}$▲	$88_{8.0}$▲	$80_{0.0}$▲	$80_{0.0}$−	$80_{0.0}$−	$80_{0.0}$
Ilavarasan-3-10 (Ilavarasan et al., 2005)	$80_{0.0}$▲	$80_{9.0}$▲	$73_{3.0}$▲	$73_{0.0}$▲	$73_{0.0}$−	$73_{0.0}$
Kang1-3-10 (Kang et al., 2011)	$80_{0.0}$▲	80_{11}▲	$76_{0.0}$▲	$76_{3.0}$▲	$76_{0.0}$▲	$73_{0.0}$
Kang2-3-10 (Kang & Lin, 2011)	$109_{0.0}$▲	102_{16}▲	$83_{0.0}$▲	$83_{0.0}$▲	$83_{0.0}$▲	$81_{2.0}$
Kuan-3-10 (Lai & Yang, 2008)	$30_{0.0}$▲	$30_{2.0}$▲	$27_{1.0}$▲	$27_{1.0}$▲	$26_{0.0}$−	$26_{0.0}$
Liang-3-10 (Lee et al., 2009)	$80_{0.0}$▲	$81_{9.0}$▲	$76_{3.0}$▲	$73_{0.0}$−	$73_{0.0}$−	$73_{0.0}$
Sample-3-8 (Pineda et al., 2013a)	$84_{0.0}$▲	$89_{7.0}$▲	$83_{2.0}$▲	$82_{3.0}$▲	$84_{0.0}$▲	$81_{0.0}$
SampleFig-3-8 (Pineda et al., 2013a)	$89_{0.0}$▲	81_{12}▲	$66_{0.0}$−	$66_{0.0}$−	$66_{0.0}$−	$66_{0.0}$
YCLee-3-8 (Lee & Zomaya, 2008)	$88_{0.0}$▲	80_{12}▲	$66_{0.0}$−	$66_{0.0}$−	$66_{0.0}$−	$66_{0.0}$

Dark gray emphasizes the best results. ▲ EFT-GVNS outperforms with statistical significance; otherwise ▽, - for no statistical difference.

Table 14
Best results found by the five algorithms over 30 independent runs.

Problem	Optimum	EDA	EFT-ILS	GRASP-CPA	MPQGA	EFT-GVNS
Ahmad-3-9 (Ahmad et al., 1998)	27	27	27	27	27	27
Daoud-2-11 (Daoud & Kharma, 2011)	56	56	56	56	56	56
Eswari-2-11 (Eswari & Nickolas, 2010)	56	60	56	56	56	56
Hamid-3-10 (Arabnejad, 2013)	100	100	100	100	100	100
Heteropar-4-12 (Arabnejad & Barbosa, 2011)	124	136	136	124	136	124
Hsu-3-10 (Hsu et al., 2007)	80	80	80	80	80	80
Ilavarasan-3-10 (Ilavarasan et al., 2005)	73	73	73	73	73	73
Kang1-3-10 (Kang et al., 2011)	73	76	76	73	$76_{0.0}$	73
Kang2-3-10 (Kang & Lin, 2011)	79	83	83	83	83	79
Kuan-3-10 (Lai & Yang, 2008)	26	27	26	26	26	26
Liang-3-10 (Lee et al., 2009)	73	76	73	73	73	73
Sample-3-8 (Pineda et al., 2013a)	81	84	81	81	84	81
SampleFig-3-8 (Pineda et al., 2013a)	66	66	66	66	66	66
YCLee-3-8 (Lee & Zomaya, 2008)	66	66	66	66	66	66

Dark gray emphasizes the best results.

12, EFT-ILS with 11, MPQGA with 10, and EDA with seven optimum values.

7. Sensitivity analysis of EFT-GVNS

This section is devoted to analyzing the parameter sensitivity of EFT-GVNS. One advantage of EFG-GVNS is their low parameter number. It only requires one parameter, the α value for the modified greedy Earliest Finish Time (EFT) heuristic. In Section 5, we mention examining the values of $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.9\}$, the value of 1.0 was ignoring because of their equivalence to the regular EFT heuristic. We graphically analyze the median objective function changes (on 100 independent runs) for the best configuration, $\alpha = 0.6$, and four configurations near the best one, i.e., the configurations of $\alpha = \{0.4, 0.5, 0.7, 0.8\}$ over the 14 synthetic instances from the small literature benchmark (Santiago et al., 2020a). Fig. 3 shows the evolution of the fitness changes over every instance of the small benchmark. For the instances Ahmad-3-9, Hamid-3-10, HSU-3-10, Kang1-3-10, Kuan-3-10, SampleFig. 3-8, and YCLee-3-8 Fig. 3 shows an outstanding performance for all the analyzed configurations, converging to the same value about the 5% of the objective function evaluations, these cases denote an insignificant sensitivity to the parameter α.

A few sensitivity is present for the instances of Heteropar-4-12, Ilvarasan-3-10, and Liang-3-10, where in all cases, all configurations converge further in the search, about the 10% of the objective function evaluations. Only in four of the studied instances was the value of the parameter α influential, namely Daoud-2-11, Eswari-2-11, Sample3-9, and Kang2-3-10. For the above first three cases, all the values of α converge to the same best value at about 40% of the search. The only case where the different parameters α do not converge to the same best value is the instance Kang2-3-10, in which, in any case, our tune parameter $\alpha = 0.6$ from the experimental setup achieves the best objective value at about 70% of the search. Therefore, the graphical results show a low sensitivity of parameters for our proposed EFT-GVNS.

8. Conclusions and future work

Examining the four real-world applications and their 400 scheduling instances, results, and Friedman rankings, we conclude that EFT-GVNS outperforms four high-performance algorithms from state-of-the-art (MPQGA, GRASP-CPA, EFT-ILS, EDA) and one reference heuristic (HEFT). The first place in the Friedman ranking was EFT-GVNS, followed by MPQGA, GRASP-CPA, EFT-ILS, EDA, and HEFT. A consistent behavior when studying a small representative testbed from diverse literature authors produces the same places in the Friedman ranking. Using a Composite Local Search instead of a VND in a GVNS effectively schedules precedence-constraint tasks on heterogeneous machines, producing quality results in a reasonable time, five minutes for large instances and 100,000 objective function evaluations in the small ones. The results are also consistent with the reported ones in the literature. The proposed EFT-GVNS overcomes the limitation of exploring a few orders of tasks' execution, increasing the possibility of visiting the optimal permutation and consequently finding the optimum value, as the small literature testbed results show. Summarizing the work contributions, we propose a new GVNS using Composite Local Search instead of Variable Neighborhood Descending for scheduling parallel applications (EFT-GVNS), we perform an experimental comparison over 400 parallel applications, EFT-GVNS achieves a global improvement of 37.6%, 27.4%, 17.8%, 6.1%, 2.2% to HEFT, EDA, EFT-ILS, GRASP-CPA, and MPQGA, respectively, at least EFT-GVNS achieves the 14 optimal values of the studied small synthetic benchmark. Now we have developed EFT-GVNS, an accurate precedence tasks scheduling algorithm; for future research, we want to explore other scheduling variants, such as cloud computing or independent tasks, and other objectives or preferences, such as energy consumption, reliability, and security. In addition, we want to explore particular problem knowledge, such as restricting the neighborhoods, for example, to elements from the critical path of the DAG application (the longest precedences and the ones that directly impact the makespan), moving elements directly to idle time in the machines, using idle times to improve energy efficiency, or enable machines on/off, to improve the EFT-GVNS results and to adapt it to other scenarios.

CRediT authorship contribution statement

Alejandro Humberto García Ruiz: Writing – original draft, Investigation, Methodology, Formal analysis, Software. **Aurelio Alejandro Santiago Pineda:** Conceptualization, Writing – original draft, Investigation, Methodology, Software, Validation, Formal analysis, Writing – review & editing, Supervision. **José Antonio Castán Rocha:** Writing – original draft, Writing – review & editing. **Salvador Ibarra Martínez:** Writing – original draft, Investigation, Methodology, Writing – review & editing. **Jesús David Terán Villanueva:** Writing – original draft, Investigation, Software.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

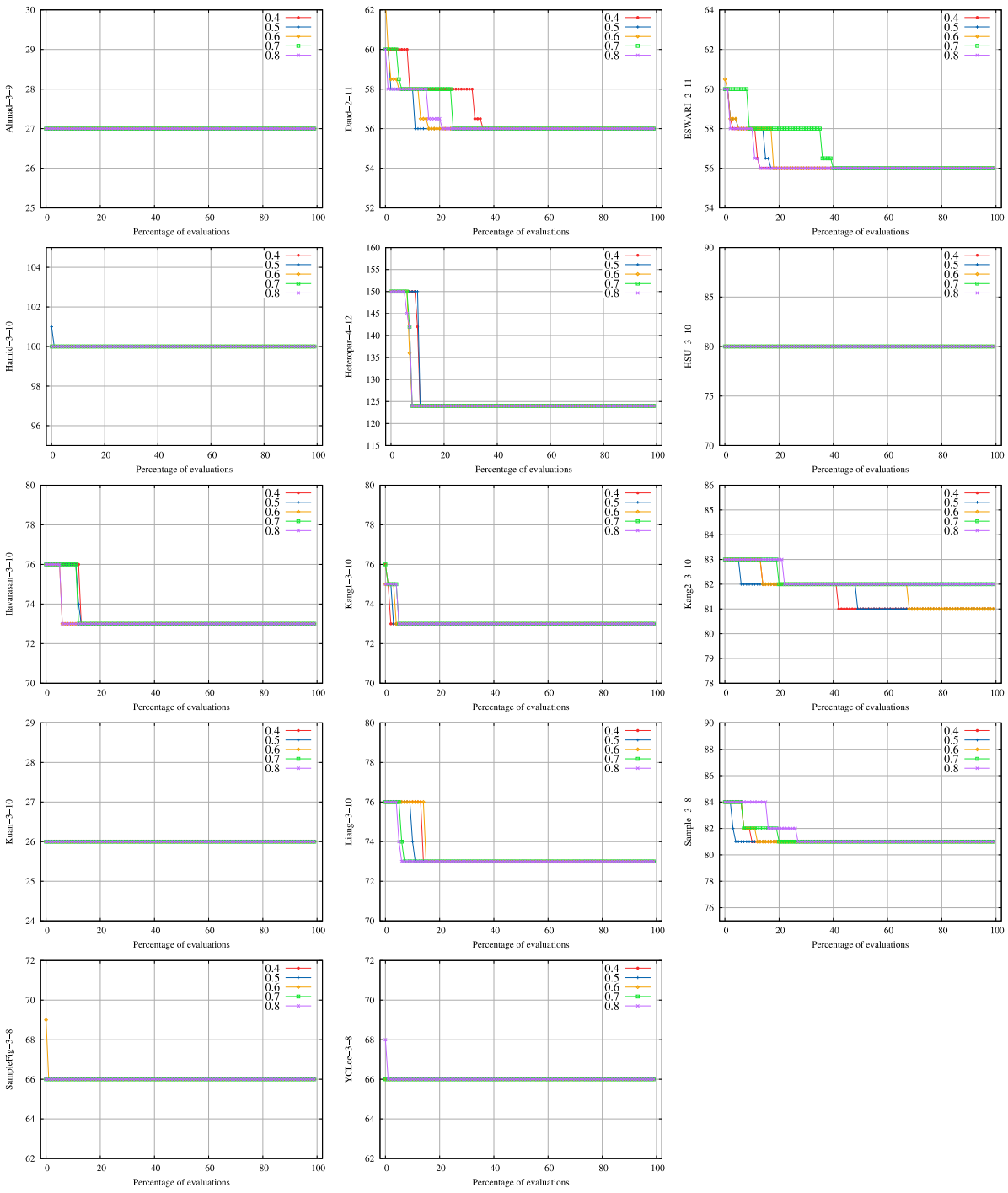


Fig. 3. Graphical convergence of EFT-GVNS for different α parameters.

Acknowledgment

Aurelio Alejandro Santiago Pineda wants to acknowledge the Consejo Nacional de Humanidades Ciencias y Tecnologías (CONAHCyT), Mexico for their SNI salary award.

References

Abd Elaziz, M., Abualigah, L., & Attiya, I. (2021). Advanced optimization technique for scheduling IoT tasks in cloud-fog computing environments. *Future Generation Computer Systems*, *124*, 142–154. http://dx.doi.org/10.1016/j.future.2021.05.026, https://www.sciencedirect.com/science/article/pii/S0167739X2100176X.

Ahmad, I., Dhodhi, M. K., & Ul-Mustafa, R. (1998). DPS: dynamic priority scheduling heuristic for heterogeneous computing systems. *IEE Proceedings - Computers and Digital Techniques*, *145*(6), 411–418.

Ait Aba, M., Zaourar, L., & Munier, A. (2020). Efficient algorithm for scheduling parallel applications on hybrid multicore machines with communications delays and energy constraint. *Concurrency Computations: Practice and Experience*, *32*(15), Article e5573.

Alsina-Pages, R. M., Navarro, J., Alias, F., & Hervas, M. (2017). Homesound: Real-time audio event detection based on high performance computing for behaviour and surveillance remote monitoring. *Sensors*, *17*(4), http://dx.doi.org/10.3390/s17040854, https://www.mdpi.com/1424-8220/17/4/854.

Arabnejad, H. (2013). List based task scheduling algorithms on heterogeneous systems-an overview. In *Doctoral symposium in informatics engineering* (p. 93).

Arabnejad, H., & Barbosa, G. J. (2011). Performance evaluation of list based scheduling on heterogeneous systems. http://icl.cs.utk.edu/workshops/heteropar2011/slides/heteropar_JorgeBarbosa.pdf. (Accessed 17 August 2020).

Arabnejad, H., & Barbosa, G. J. (2014). List scheduling algorithm for heterogeneous systems by an optimistic cost table. *IEEE Transactions on Parallel and Distributed Systems*, *25*(3), 682–694. http://dx.doi.org/10.1109/TPDS.2013.57.

Arunarani, A., Manjula, D., & Sugumaran, V. (2019). Task scheduling techniques in cloud computing: A literature survey. *Future Generation Computer Systems*, *91*, 407–415. http://dx.doi.org/10.1016/j.future.2018.09.014, https://www.sciencedirect.com/science/article/pii/S0167739X17321519.

Bastrakov, S., Meyerov, I., Gergel, V., Gonoskov, A., Gorshkov, A., Efimenko, E., ... Vildemanov, A. (2013). High performance computing in biomedical applications. *Procedia Computer Science*, *18*, 10–19. http://dx.doi.org/10.1016/j.procs.2013.05.164, https://www.sciencedirect.com/science/article/pii/S1877050913003074, 2013 International Conference on Computational Science.

Chen, Q., Han, Y., Wu, J., & Gan, Y. (2022). Energy-saving task scheduling based on hard reliability requirements: A novel approach with low energy consumption and high reliability. *Sustainability*, *14*, 11. http://dx.doi.org/10.3390/su14116591, https://www.mdpi.com/2071-1050/14/11/6591.

Chen, J., Impagliazzo, J., & Shen, L. (2020). High-performance computing and engineering educational development and practice. In *2020 IEEE frontiers in education conference* FIE, (pp. 1–8). http://dx.doi.org/10.1109/FIE44824.2020.9274100.

Cho, H., Kim, C., Sun, J., Easwaran, A., Park, J. D., & Choi, B. C. (2020). Scheduling parallel real-time tasks on the minimum number of processors. *IEEE Transactions on Parallel and Distributed Systems*, *31*(1), 171–186. http://dx.doi.org/10.1109/TPDS.2019.2929048.

Corder, G. W., & Foreman, I. D. (2011). *Nonparametric statistics for non-statisticians*. John Wiley & Sons, Inc.

Correa-Baena, J. P., Hippalgaonkar, K., van Duren, J., Jaffer, S., Chandrasekhar, V. R., Stevanovic, V., ... Buonassisi, T. (2018). Accelerating materials development via automation, machine learning, and high-performance computing. *Joule*, *2*(8), 1410–1420. http://dx.doi.org/10.1016/j.joule.2018.05.009, https://www.sciencedirect.com/science/article/pii/S2542435118302289.

Dan Mironescu, I., & Vintan, L. (2017). A task scheduling algorithm for HPC applications using colored stochastic Petri net models. In *2017 13th IEEE international conference on intelligent computer communication and processing* (pp. 479–486). http://dx.doi.org/10.1109/ICCP.2017.8117051.

Daoud, M. I., & Kharma, N. (2011). A hybrid heuristic-genetic algorithm for task scheduling in heterogeneous processor networks. *Journal of Parallel and Distributed Computing*, *71*(11), 1518–1531. http://dx.doi.org/10.1016/j.jpdc.2011.05.005, http://www.sciencedirect.com/science/article/pii/S0743731511001018.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, *1*(1), 3–18. http://dx.doi.org/10.1016/j.swevo.2011.02.002, https://www.sciencedirect.com/science/article/pii/S2210650211000034.

Elcock, J., & Edward, N. (2023). An efficient ACO-based algorithm for task scheduling in heterogeneous multiprocessing environments. *Array*, *17*, Article 100280. http://dx.doi.org/10.1016/j.array.2023.100280, https://www.sciencedirect.com/science/article/pii/S259000562300005X.

Emami, H. (2022). Cloud task scheduling using enhanced sunflower optimization algorithm. *ICT Express*, *8*(1), 97–100. http://dx.doi.org/10.1016/j.icte.2021.08.001, https://www.sciencedirect.com/science/article/pii/S2405959521000898.

Eswari, R., & Nickolas, S. (2010). Path-based heuristic task scheduling algorithm for heterogeneous distributed computing systems. In *2010 International conference on advances in recent technologies in communication and computing* (pp. 30–34).

Feo, T. A., & Resende, G. M. (1995). Greedy randomized adaptive search procedures. *Journal of global optimization*, *6*(2), 109–133.

García, S., Molina, D., Lozano, M., & Herrera, F. (2008). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 special session on real parameter optimization'2005 special session on real parameter optimization. *Journal of Heuristics*, *15*(6), 617–644.

Gaspero, L. D., & Schaerf, A. (2007). A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics*, *13*(2), 189–207.

ge Wu, C., Wang, L., & jing Wang, J. (2021). A path relinking enhanced estimation of distribution algorithm for direct acyclic graph task scheduling problem. *Knowledge-Based Systems*, *228*(107255), http://dx.doi.org/10.1016/j.knosys.2021.107255, https://www.sciencedirect.com/science/article/pii/S0950705121005177.

Ghafari, R., & Mansouri, N. (2022). An efficient task scheduling based on seagull optimization algorithm for heterogeneous cloud computing platforms. *International Journal of Engineering*, *35*(2), 433–450. http://dx.doi.org/10.5829/ije.2022.35.02b.20, https://www.ije.ir/article_140950.html.

Green, R. C., Wang, L., & Alam, M. (2013). Applications and trends of high performance computing for electric power systems: Focusing on smart grid. *IEEE Transactions on Smart Grid*, *4*(2), 922–931. http://dx.doi.org/10.1109/TSG.2012.2225646.

Gu, Y., & Budati, C. (2020). Energy-aware workflow scheduling and optimization in clouds using bat algorithm. *Future Generation Computer Systems*, *113*, 106–112. http://dx.doi.org/10.1016/j.future.2020.06.031, https://www.sciencedirect.com/science/article/pii/S0167739X19317066.

Guo, H., Zhou, J., & Gu, H. (2022). Limited duplication-based list scheduling algorithm for heterogeneous computing system. *Micromachines*, *13*, 7. http://dx.doi.org/10.3390/mi13071067, https://www.mdpi.com/2072-666X/13/7/1067.

Hai, T., Zhou, J., Jawawi, D., Wang, D., Oduah, U., Biamba, C., & Jain, S. K. (2023). Task scheduling in cloud environment: optimization, security prioritization and processor selection schemes. *Journal of Cloud Computing*, *12*(1), 15.

Hamed, A. Y., Elnahary, M. K., Alsubaei, F. S., & El-Sayed, H. H. (2023). Optimization task scheduling using cooperation search algorithm for heterogeneous cloud computing systems. *Computers, Materials & Continua*, *74*(1), 2133–2148. http://dx.doi.org/10.32604/cmc.2023.032215, http://www.techscience.com/cmc/v74n1/49845.

Hansen, P., Mladenović, N., Todosijević, R., & Hanafi, S. (2017). Variable neighborhood search: basics and variants. *EURO Journal on Computational Optimization*, *5*(3), 423–454.

Hauschild, M., & Pelikan, M. (2011). An introduction and survey of estimation of distribution algorithms. *Swarm and Evolutionary Computation*, *1*(3), 111–128. http://dx.doi.org/10.1016/j.swevo.2011.08.003, https://www.sciencedirect.com/science/article/pii/S2210650211000435.

Heydari, M., & Aazami, A. (2018). Minimizing the maximum tardiness and makespan criteria in a job shop scheduling problem with sequence dependent setup times. *Journal of Industrial and Systems Engineering*, *11*(2), 134–150.

Hoos, H. H., & Stützle, T. (2004). *Stochastic local search: foundations and applications*. Elsevier.

Houssein, E. H., Gad, A. G., Wazery, Y. M., & Suganthan, P. N. (2021). Task scheduling in cloud computing based on meta-heuristics: Review, taxonomy, open challenges, and future trends. *Swarm and Evolutionary Computation*, *62*, Article 100841. http://dx.doi.org/10.1016/j.swevo.2021.100841, https://www.sciencedirect.com/science/article/pii/S221065022100002X.

Hsu, C. H., Hsieh, C. W., & Yang, C. T. (2007). A generalized critical task anticipation technique for dag scheduling. In *International conference on algorithms and architectures for parallel processing* (pp. 493–505).

Huacuja, H. J. F., Santiago, A., Pecero, J. E., Dorronsoro, B., Bouvry, P., Monterrubio, J. C. S., ... Santillan, C. G. (2015). A comparison between memetic algorithm and seeded genetic algorithm for multi-objective independent task scheduling on heterogeneous machines. In Melin, P., Castillo, O., & Kacprzyk, J. (Eds.), *Design of Intelligent Systems Based on Fuzzy Logic, Neural Networks and Nature-Inspired Optimization* (pp. 377–389). Cham: Springer International Publishing, http://dx.doi.org/10.1007/978-3-319-17747-2_29.

Ilavarasan, E., Thambidurai, P., & Mahilmannan, R. (2005). Performance effective task scheduling algorithm for heterogeneous computing system. In *The 4th international symposium on parallel and distributed computing* (pp. 28–38).

Imene, L., Sihem, S., Okba, K., & Mohamed, B. (2022). A third generation genetic algorithm NSGAIII for task scheduling in cloud computing. *Journal of King Saud University - Computer and Information Sciences*, *34*(9), 7515–7529. http://dx.doi.org/10.1016/j.jksuci.2022.03.017, https://www.sciencedirect.com/science/article/pii/S131915782200101X.

Jelovac, D., Ljubojević, Č., & Ljubojević, L. (2022). HPC in business: the impact of corporate digital responsibility on building digital trust and responsible corporate digital governance. *Digital Policy, Regulation and Governance*, *24*(6), 485–497.

Kang, Y., & Lin, Y. (2011). A recursive algorithm for scheduling of tasks in a heterogeneous distributed environment. In *2011 4th International conference on biomedical engineering and informatics, vol. 4* (pp. 2099–2103).

Kang, Y., Zhang, Z., & Chen, P. (2011). An activity-based genetic algorithm approach to multiprocessor scheduling. In *2011 Seventh international conference on natural computation, vol. 2* (pp. 1048–1052).

Kocot, B., Czarnul, P., & Proficz, J. (2023). Energy-aware scheduling for high-performance computing systems: A survey. *Energies*, *16*(2), http://dx.doi.org/10.3390/en16020890, https://www.mdpi.com/1996-1073/16/2/890.

Kodiyalam, S., Yang, R., Gu, L., & Tho, C. H. (2004). Multidisciplinary design optimization of a vehicle system in a scalable, high performance computing environment. *Structural and Multidisciplinary Optimization*, *26*, 256–263.

Kołodziej, J., & Khan, S. U. (2012). Multi-level hierarchic genetic-based scheduling of independent jobs in dynamic heterogeneous grid environment. *Information Sciences*, *214*, 1–19. http://dx.doi.org/10.1016/j.ins.2012.05.016, https://www.sciencedirect.com/science/article/pii/S002002551200360X.

Lai, K. C., & Yang, T. C. (2008). A dominant predecessor duplication scheduling algorithm for heterogeneous systems. *The Journal of Supercomputing*, *44*(2), 126–145.

Lee, L., Chen, C., Chang, H., Tang, C., & Pan, K. (2009). A non-critical path earliest-finish algorithm for inter-dependent tasks in heterogeneous computing environments. In *2009 11th IEEE international conference on high performance computing and communications* (pp. 603–608).

Lee, C. A., Gasster, S. D., Plaza, A., Chang, C. I., & Huang, B. (2011). Recent developments in high performance computing for remote sensing: A review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *4*(3), 508–527. http://dx.doi.org/10.1109/JSTARS.2011.2162643.

Lee, Y., & Zomaya, A. (2008). A novel state transition method for metaheuristic-based scheduling in heterogeneous computing systems. *IEEE Transactions on Parallel and Distributed Systems*, *19*(9), 1215–1223.

Li, C., Tang, J., Tang, H., & Luo, Y. (2019). Collaborative cache allocation and task scheduling for data-intensive applications in edge computing environment. *Future Generation Computer Systems*, *95*, 249–264. http://dx.doi.org/10.1016/j.future.2019.01.007, https://www.sciencedirect.com/science/article/pii/S0167739X18309439.

Li, H., Wang, D., Cañizares Abreu, J. R, Zhao, Q., & Bonilla Pineda, O. (2021). PSO+ LOA: hybrid constrained optimization for scheduling scientific workflows in the cloud. *The Journal of Supercomputing*, *77*(11), 13139–13165.

Lin, Z., Li, C., & Tian, B. (2022). A scheduling algorithm based on reinforcement learning for heterogeneous environments. *Applied Soft Computing*, *130*, Article 109707. http://dx.doi.org/10.1016/j.asoc.2022.109707, https://www.sciencedirect.com/science/article/pii/S1568494622007566.

Mack, J., Arda, S. E., Ogras, U. Y., & Akoglu, A. (2022). Performant, multi-objective scheduling of highly interleaved task graphs on heterogeneous system on chip devices. *IEEE Transactions on Parallel and Distributed Systems*, *33*(09), 2148–2162. http://dx.doi.org/10.1109/TPDS.2021.3135876.

Majeed, A., & Lee, S. (2021). Applications of machine learning and high-performance computing in the era of COVID-19. *Applied System Innovation*, *4*(3), http://dx.doi.org/10.3390/asi4030040, https://www.mdpi.com/2571-5577/4/3/40.

Mandal, T., & Acharyya, S. (2015). Optimal task scheduling in cloud computing environment: Meta heuristic approaches. In *2015 2nd International conference on electrical information and communication technologies* EICT, (pp. 24–28). http://dx.doi.org/10.1109/EICT.2015.7391916.

Memeti, S., & Pllana, S. (2021). Optimization of heterogeneous systems with AI planning heuristics and machine learning: a performance and energy aware approach. *Computing*, *103*(12), 2943–2966.

Menaka, M., & Sendhil Kumar, K. (2022). Workflow scheduling in cloud environment – challenges, tools, limitations & methodologies: A review. Measurement. *Sensors*, *24*, Article 100436. http://dx.doi.org/10.1016/j.measen.2022.100436, https://www.sciencedirect.com/science/article/pii/S2665917422000708.

Mohammad Hasani Zade, B., Mansouri, N., & Javidi, M. M. (2021). SAEA: A security-aware and energy-aware task scheduling strategy by parallel squirrel search algorithm in cloud environment. *Expert Systems with Applications*, *176*, Article 114915. http://dx.doi.org/10.1016/j.eswa.2021.114915, https://www.sciencedirect.com/science/article/pii/S0957417421003560.

Murad, S. A., Muzahid, A. J. M., Azmi, Z. R. M., Hoque, M. I., & Kowsher, M. (2022). A review on job scheduling technique in cloud computing and priority rule based intelligent framework. *Journal of King Saud University - Computer and Information Sciences*, *34*(6, Part A), 2309–2331. http://dx.doi.org/10.1016/j.jksuci.2022.03.027, https://www.sciencedirect.com/science/article/pii/S1319157822001112.

Nayak, S. K., Padhy, S. K., & Panigrahi, S. P. (2012). A novel algorithm for dynamic task scheduling. *Future Generation Computer Systems*, *28*(5), 709–717. http://dx.doi.org/10.1016/j.future.2011.12.001, https://www.sciencedirect.com/science/article/pii/S0167739X11002354, Special Section: Energy efficiency in large-scale distributed systems.

Niculescu, V. (2020). On the impact of high performance computing in big data analytics for medicine. *Applied Medical Informatics*, *42*(1), 9–18, https://ami.info.umfcluj.ro/index.php/AMI/article/view/766.

NoorianTalouki, R., Hosseini Shirvani, M., & Motameni, H. (2022). A heuristic-based task scheduling algorithm for scientific workflows in heterogeneous cloud computing platforms. *Journal of King Saud University - Computer and Information Sciences*, *34*(8, Part A), 4902–4913. http://dx.doi.org/10.1016/j.jksuci.2021.05.011, https://www.sciencedirect.com/science/article/pii/S1319157821001269.

Orr, M., & Sinnen, O. (2021). Optimal task scheduling for partially heterogeneous systems. *Parallel Computing*, *107*, Article 102815. http://dx.doi.org/10.1016/j.parco.2021.102815, https://www.sciencedirect.com/science/article/pii/S0167819121000636.

Pecero, J. E., Huacuja, H. J. F., Bouvry, P., Pineda, A. A. S., Locés, M. C. L., & Barbosa, J. J. G. (2012). On the energy optimization for precedence constrained applications using local search algorithms. In *2012 International conference on high performance computing simulation* HPCS, (pp. 133–139). http://dx.doi.org/10.1109/HPCSim.2012.6266902.

Peng, J., Li, K., Chen, J., & Li, K. (2022). HEA-PAS: A hybrid energy allocation strategy for parallel applications scheduling on heterogeneous computing systems. *Journal of Systems Architecture*, *122*, Article 102329. http://dx.doi.org/10.1016/j.sysarc.2021.102329, https://www.sciencedirect.com/science/article/pii/S1383762121002265.

Pineda, A. A. S., Pecero, J., Huacuja, H., Barbosa, J., & Bouvry, P. (2013a). An iterative local search algorithm for scheduling precedence-constrained applications on heterogeneous machines. In *6th Multidisciplinary international conference on scheduling: theory and applications* (pp. 472–485).

Pineda, A. A. S., Pecero, J., Huacuja, H., Barbosa, J., & Bouvry, P. (2013b). An iterative local search algorithm for scheduling precedence-constrained applications on heterogeneous machines. In *6th Multidisciplinary international conference on scheduling: theory and applications* (pp. 472–485).

Pinto, A. R. F., & Nagano, M. S. (2022). A comprehensive review of batching problems in low-level picker-to-parts systems with order due dates: Main gaps, trade-offs, and prospects for future research. *Journal of Manufacturing Systems*, *65*, 1–18. http://dx.doi.org/10.1016/j.jmsy.2022.08.006, https://www.sciencedirect.com/science/article/pii/S0278612522001339.

Pirozmand, P., Hosseinabadi, A. A. R., Farrokhzad, M., Sadeghilalimi, M., Mirkamali, S., & Slowik, A. (2021). Multi-objective hybrid genetic algorithm for task scheduling problem in cloud computing. *Neural computing and applications*, *33*(19), 13075–13088.

Priyadarshini, R., Alagirisamy, M., Rajendran, N., Marandi, A. K., Patil, V. V., & Vivek (2022). Minimization of makespan and energy consumption in task scheduling in heterogeneous cloud environment. *International Journal of Intelligent Systems and Applications in Engineering*, *10*, 276–280, https://ijisae.org/index.php/IJISAE/article/view/2400.

Qin, S., Pi, D., & Shao, Z. (2022). AILS: A budget-constrained adaptive iterated local search for workflow scheduling in cloud environment. *Expert Systems with Applications*, *198*, Article 116824. http://dx.doi.org/10.1016/j.eswa.2022.116824, https://www.sciencedirect.com/science/article/pii/S0957417422002809.

Sanbonmatsu, K., & Tung, S. C. (2007). High performance computing in biology: Multimillion atom simulations of nanoscale systems. *Journal of Structural Biology*, *157*(3), 470–480. http://dx.doi.org/10.1016/j.jsb.2006.10.023, https://www.sciencedirect.com/science/article/pii/S104784770600308X, Advances in Molecular Dynamics Simulations.

(2021). Energy idle aware stochastic lexicographic local searches for precedence-constraint task list scheduling on heterogeneous systems. *Energies*, *14*, 12. http://dx.doi.org/10.3390/en14123473, https://www.mdpi.com/1996-1073/14/12/3473.

Santiago, A., Terán-Villanueva, J. D., Martínez, S. I., Rocha, J. A. C., Menchaca, J. L., Berrones, M. G. T., & Ponce-Flores, M. (2020a). GRASP and iterated local search-based cellular processing algorithm for precedence-constraint task list scheduling on heterogeneous systems. *Applied Sciences*, *10*(21), http://dx.doi.org/10.3390/app10217500, https://www.mdpi.com/2076-3417/10/21/7500.

Santiago, A., Terán-Villanueva, J. D., Martínez, S. I., Rocha, J. A. C., Menchaca, J. L., Borreones, M. G. T., & Ponce-Flores, M. (2020b). Instance set for: GRASP and iterated local search based cellular processing algorithm for precedence-constraint task list scheduling on heterogeneous systems. https://github.com/AASantiago/SchedulingInstances. (Accessed 27 August 2020).

Schiavinotto, T., & Stützle, T. (2004). The linear ordering problem: Instances, search space analysis and algorithms. *Journal of Mathematical Modelling and Algorithms*, *3*(4), 367–402.

Schryen, G., Kliewer, N., & Fink, A. (2020). *High performance business computing vol. 62*. Springer.

Shu, W., Cai, K., & Xiong, N. N. (2021). Research on strong agile response task scheduling optimization enhancement with optimal resource usage in green cloud computing. *Future Generation Computer Systems*, *124*, 12–20. http://dx.doi.org/10.1016/j.future.2021.05.012, https://www.sciencedirect.com/science/article/pii/S0167739X2100162X.

Sinnen, O. (2007). *Task scheduling for parallel systems, vol. 60*. John Wiley & Sons.

Soto-Monterrubio, J. C., Santiago, A., Fraire-Huacuja, H. J., Frausto-Solís, J., & Terán-Villanueva, D. (2016). Branch and bound algorithm for the heterogeneous computing scheduling multi-objective problem. *International Journal of Combinatorial Optimization Problems and Informatics*, *7*(3), 7–19, https://www.ijcopi.org/ojs/article/view/23.

Talbi, E. G. (2009). *Metaheuristics: from design to implementation, vol. 74*. John Wiley & Sons.

Tang, X., & Fu, Z. (2020). CPU–GPU utilization aware energy-efficient scheduling algorithm on heterogeneous computing systems. *IEEE Access*, *8*, 58948–58958. http://dx.doi.org/10.1109/ACCESS.2020.2982956.

Terán-Villanueva, J. D., Fraire-Huacuja, H. J., Ibarra Martínez, S., Cruz-Reyes, L., Castán Rocha, J. A, Gómez Santillán, C., & Menchaca, J. L. (2019). Cellular processing algorithm for the vertex bisection problem: Detailed analysis and new component design. *Information Sciences*, *478*, 62–82. http://dx.doi.org/10.1016/j.ins.2018.11.020, https://www.sciencedirect.com/science/article/pii/S0020025518309101.

Terán-Villanueva, J. D., Huacuja, H. J. F., Valadez, J. M. C., Pazos Rangel, R. A., Soberanes, H. J. P., & Flores, J. A. M. (2013). Cellular processing algorithms. In P. Melin, & O. Castillo (Eds.), *Soft computing applications in optimization, control, and recognition* (pp. 53–74). Berlin, Heidelberg: Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-642-35323-9_3.

Topcuoglu, H., Hariri, S., & Wu, M. Y. (2002). Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Transactions on Parallel and Distributed Systems*, *13*(3), 260–274. http://dx.doi.org/10.1109/71.993206.

Ullman, J. (1975). NP-complete scheduling problems. *Journal of Computer and System Sciences*, *10*(3), 384–393. http://dx.doi.org/10.1016/S0022-0000(75)80008-0, https://www.sciencedirect.com/science/article/pii/S0022000075800080.

Valentini, G. L., Lassonde, W., Khan, S. U., Min-Allah, N., Madani, S. A., Li, J., ... others (2013). An overview of energy efficiency techniques in cluster computing systems. *Cluster Computing*, *16*, 3–15.

Velarde Martinez, A. (2020). Scheduling in heterogeneous distributed computing systems based on internal structure of parallel tasks graphs with meta-heuristics. *Applied Sciences*, *10*(18), http://dx.doi.org/10.3390/app10186611, https://www.mdpi.com/2076-3417/10/18/6611.

Villanueva, D. T., Flores, J. A. M., López-Locés, M. C., Escobar, D. E. Z., Pineda, A. S., et al. (2012). Hybrid grasp with composite local search and path-relinking for the linear ordering problem with cumulative costs. *International Journal of Combinatorial Optimization Problems and Informatics*, *3*(1), 21–30.

Wang, L., Khan, S. U., Chen, D., Kołodziej, J., Ranjan, R., zhong Xu, C., & Zomaya, A. (2013). Energy-aware parallel task scheduling in a cluster. *Future Generation Computer Systems*, *29*(7), 1661–1670. http://dx.doi.org/10.1016/j.future.2013.02.010, https://www.sciencedirect.com/science/article/pii/S0167739X13000484, Including

Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications - Big Data, Scalable Analytics, and Beyond.

Wu, Y., Xiang, Y., Ge, J., & Muller, P. (2018). High-performance computing for big data processing. *Future Generation Computer Systems*, *88*, 693–695. http://dx.doi.org/10.1016/j.future.2018.07.054, https://www.sciencedirect.com/science/article/pii/S0167739X18317679.

Xie, G., Xiao, X., Peng, H., & Li, K. (2022). A survey of low-energy parallel scheduling algorithms. *IEEE Transactions on Sustainable Computing*, *7*(1), 27–46. http://dx.doi.org/10.1109/TSUSC.2021.3057983.

Xing, L., Zhang, M., Li, H., Gong, M., Yang, J., & Wang, K. (2022). Local search driven periodic scheduling for workflows with random task runtime in clouds. *Computers & Industrial Engineering*, *168*, Article 108033. http://dx.doi.org/10.1016/j.cie.2022.108033, https://www.sciencedirect.com/science/article/pii/S0360835222001036.

Xu, Y., Li, K., Hu, J., & Li, K. (2014). A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues. *Information Sciences*, *270*, 255–287. http://dx.doi.org/10.1016/j.ins.2014.02.122, https://www.sciencedirect.com/science/article/pii/S002002551400228X.

Yi, G., & Loia, V. (2019). High-performance computing systems and applications for AI. *The Journal of Supercomputing*, *75*, 4248–4251.

Yi, N., Xu, J., Yan, L., & Huang, L. (2020). Task optimization and scheduling of distributed cyber–physical system based on improved ant colony algorithm. *Future Generation Computer Systems*, *109*, 134–148. http://dx.doi.org/10.1016/j.future.2020.03.051, https://www.sciencedirect.com/science/article/pii/S0167739X19327608.

Zenios, S. A. (1999). High-performance computing in finance: The last 10 years and the next. *Parallel Computing*, *25*(13), 2149–2175. http://dx.doi.org/10.1016/S0167-8191(99)00083-6, https://www.sciencedirect.com/science/article/pii/S0167819199000836.

Zhang, L., Li, K., Li, C., & Li, K. (2017). Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems. *Information Sciences*, *379*, 241–256. http://dx.doi.org/10.1016/j.ins.2016.08.003, https://www.sciencedirect.com/science/article/pii/S0020025516305722.

Zhang, L., Li, K., Xu, Y., Mei, J., Zhang, F., & Li, K. (2015). Maximizing reliability with energy conservation for parallel task scheduling in a heterogeneous cluster. *Information Sciences*, *319*, 113–131. http://dx.doi.org/10.1016/j.ins.2015.02.023, https://www.sciencedirect.com/science/article/pii/S0020025515001231, Energy Efficient Data, Services and Memory Management in Big Data Information Systems.

Zhang, Y., Tong, F., Li, C., & Xu, Y. (2021). Bi-objective workflow scheduling on heterogeneous computing systems using a memetic algorithm. *Electronics*, *10*(2), http://dx.doi.org/10.3390/electronics10020209, https://www.mdpi.com/2079-9292/10/2/209.

Zhang, L., Zhou, L., & Salah, A. (2020). Efficient scientific workflow scheduling for deadline-constrained parallel tasks in cloud computing environments. *Information Sciences*, *531*, 31–46. http://dx.doi.org/10.1016/j.ins.2020.04.039, https://www.sciencedirect.com/science/article/pii/S0020025520303479.